CSC 751

Image Processing Notes

Document I

September 3, 2003

"Anyone who cannot cope with mathematics
is not fully human. At best he is a tolerable
subhuman who has learned to wear shoes,
bathe, and not make messes in the house."

Robert Anson Heinlein
Time Enough for Love

To the reader: The author has **no interest** in anything "GUI" or the menial details of "making an image look "good"". What the author does enjoy doing is being able to programmatically decipher an image by segmentation and applying various algorithms for understanding and describing a region. Then, machine learning is used and this makes image processing interesting. I suppose, you could say my interest is not in Image Processing, but more of Computer Vision. I'm more interested in getting my hands dirty, than using a pre-built "System.Util.DoEverythingForMe" approach…no offense to the I.T. workers of this World.

September 3, 2003, Wednesday:

What is image processing?

Computer Graphics ←-→ Image Processing ←-→ Computer Vision
"Image Synthesis" ←-→ ←-→ Image Analysis

Computer graphics is the inverse of Computer Vision.

What are images? 2-D optical Images
It is an intensity function f(x,y), where x and y are spatial variables.

Digital Images vs. Continuous Images

- What are digital images?
- What are continuous images?

Digitizing Images

Continuous image is described by $0 \le f(x,y) \le \infty$.

September 5, 2003, Friday

Reading assignments
- Read Chapter 1
- Start Chapter 2

Digital Image Processing

Important Areas of Image Processing
(1) Digitization
(2) Display
(3) Image Enhancement
(4) Restoration
(5) Encoding (compression)
(6) Segmentation
(7) Feature Analysis & Image Analysis

Digitization	
Display	
Image Enhancement	Bringing out details that are obscured, or highlighting certain features of interest in an image.
Restoration	Improving the appearance of an image using objective techniques, in a sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation.
Encoding (compression)	Deals with the techniques for reducing the storage required to save an image, or the bandwidth required to transmit it.
Segmentation	Segmentation is the process of partitioning an image into its constituent parts or objects.
Feature Analysis & Image Analysis	

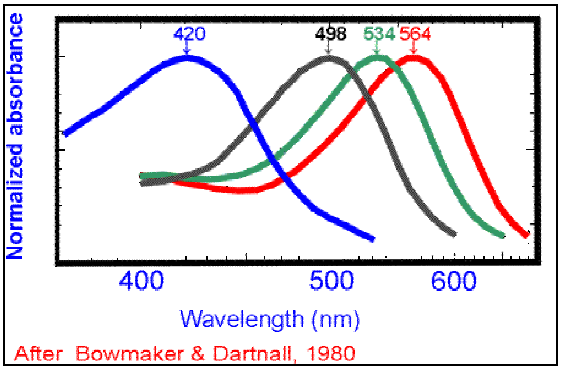
Digitization from Sources including, but not limited to:
* X-rays, gamma, microwave, sound/ultrasound

We will concentrate mainly on optical imagery.

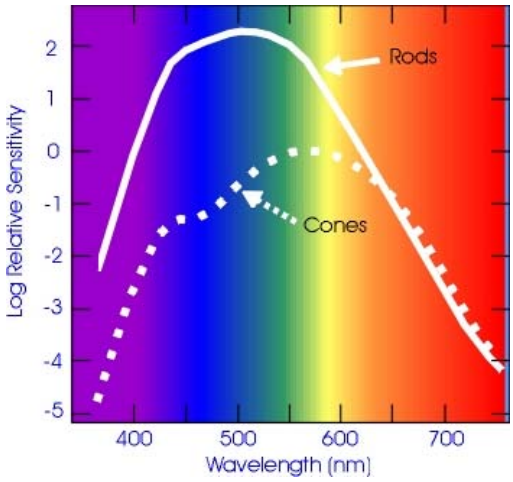
3D Scene ---------->2D Image

Image Formation

Spatial Resolution: Rows and columns, the actual # of pixels in an image. This value is dependent on the medium on which the image is to be displayed.
Intensity Resolution: The number of intensity levels. How man colors are enough? How many discrete intensity levels are enough??



Trichromatic theory of color vision:
Four classes of photopigments as shown in the graph. The colors of the curves do not represent the colors of the photopigments. The wavelength of maximum absorbance is indicated at the top of each curve. The 420 curve is for the short wavelength cones, the 498 curve is for the rods, and the 534 and 564 curves are for the middle and long wavelength sensitive cones respectively.



Rods vs. Cones
There are three types of cones (*at least, according to the lecture there is*):

Cone vision is **phototopic** or bright-light vision.

This was "uttered" at the beginning of the hardware discussion: Using a color map to exchange gray to color can bring out certain *features* (gray→color)

Display Hardware

Frame Buffer: "Holds the current display" (size can be 256x256, 512x512, 1024x… or any other combination – e.g. 512x1024)
Look-up Table: sits "logically" between the frame buffer and the display.

Given intensity in the frame buffer, how do we display?

In 8-bit, grayscale, the LUT looks something like this:

Pixel Intensity (an actual physical location, like an array index)	**Display Intensity**
0	0
1	1
…	…
255	255

In RGB (24 -bit), one would describe an LUT "conceptually" as:

Pixel Intensity	Display Intensity (Red)	Display Intensity (Green)	Display Intensity (Blue)
0	0	0	0
1	1	1	1
…			
16 million or so	255	255	255

Where 0 is black and RGB = 255 x 255 x 255 is completely white.

It is important to note that the values in the lookup table can be changed so that we can "highlight" a certain intensity value. A pixel with intensity value 1 could actually map to pure red by changing the LUT to have 255, 0, and 0 in the RGB components. By doing so, we have implemented the idea of "pseudocolor".

Pseudocolor: is a color mapping of a monochrome image array, which is intended to enhance the detectability of detail within the image. **Pseudocolor** functions fall under the techniques of *"color image enhancements".*

Demonstrations:

- Loaded 256 intensity level (grayscale) image of cat
- Demonstrated the reduction on the number of intensity levels (256-> 64->4->2).
- Demonstrated Binary Thresholding.
- Loaded a brain image and demonstrated using pseudocolor techniques on it. It was also demonstrated on the cat face image. However, the brain image was more interesting in that some important features of the brain image can be brought out using pseudocolor, which is of use to medical professionals.

September 8, 2003, Monday

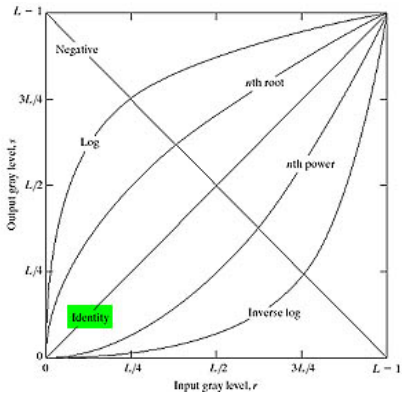
Reading Assignment
Chapters 1+2, start on Ch. 3.

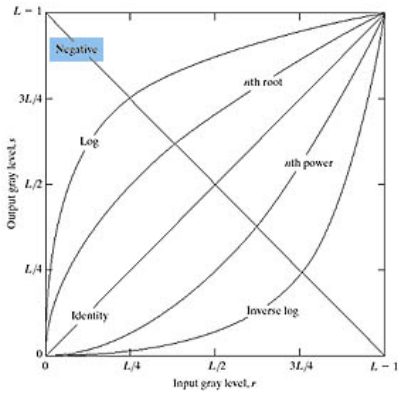
Course Webpage(s)

http://www.hpcnet.org/sdsmt/directory/courses/2003fa/csc464001
http://www.hpcnet.org/sdsmt/directory/courses/2003fa/csc751001

Recap for previous class:

- Intro to DIP
- Hardware used
  - Frame Buffer
  - Lookup Tables
    - Can use LUTs in memory (instead of hardware). This allows for changes to intensity values instead of the pixels themselves. This saves time for functions where we perform multiple operations on pixels (256 elements * 2 operations is much cheaper than 1,000,000 pixels * 2 operations).
  - Color Vision

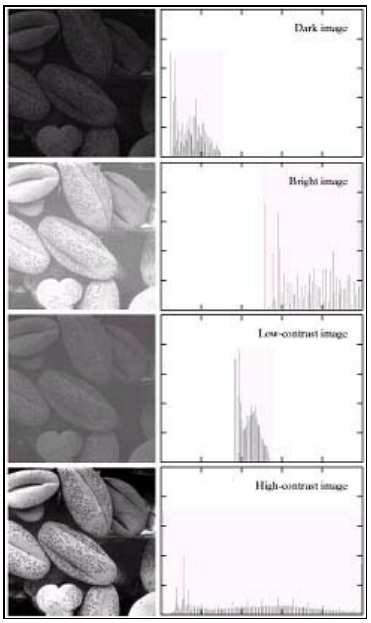




Diagrams representing the Intensity value (x) vs display value (y)

For the LUT, we can assign pixels of an image in the following way:

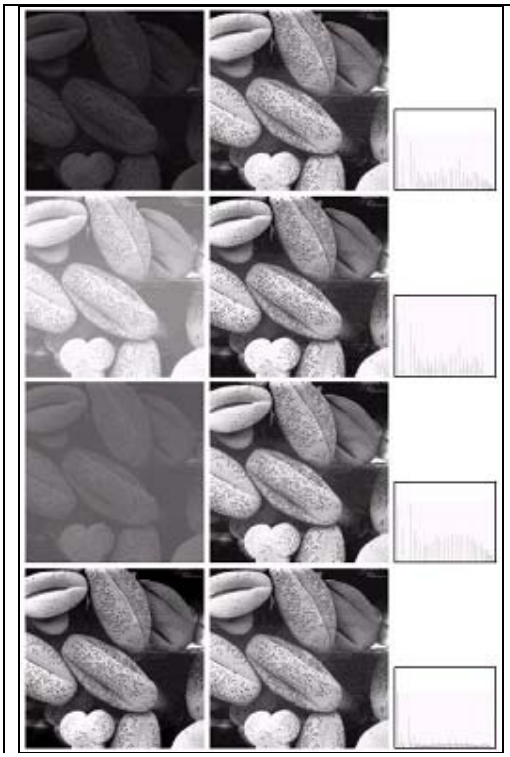
```
For(row=0;row<nrows;row++) {
   For(col=0;col<ncols;col++) {
         Image[row][col]=LUT[(image[row][col])];
   }
}
```



a b

FIGURE 3.15 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

It was demonstrated that the histograms on low contrast, high contrast images appeared something like the plots on the left.



Effect of Histogram Equalization on these images (on the right)

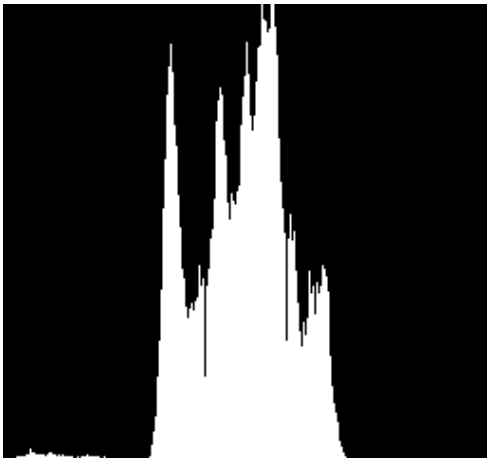
Point processes: Depends only on the image intensity at a single point.
- Contrast Stretching
- Histogram Equalization

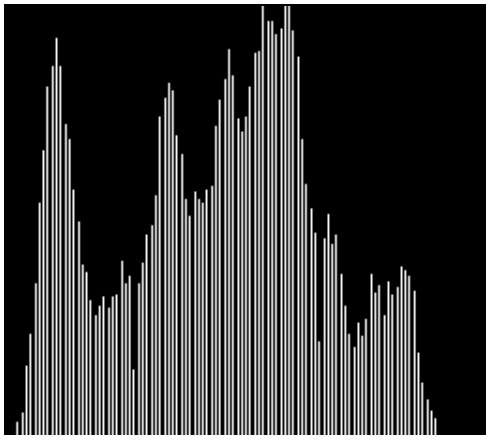
Neighborhood processes: Depends also on values of neighboring pixel intensities.

For enhancing contrast (i.e. adjusting low contrast or high contrast images), one can use **contrast stretching** or **histogram equalization**.

Original Image

Contrast Stretching





Histogram Contrast manipulation using Contrast Stretch

HSI model

Hue: color, shade
Saturation: how much white is mixed in.
Intensity: the measure of the shade. Essentially, this is the brightness

Allocating space using the predefined byte datatype:

```
Image=new byte*[nrows];
For (row=0;row<nrows;rows++)
       Image [row]=new byte [ncols];
```

How would the contrast-stretching algorithm be implemented?

September 10, 2003, Wednesday

Finish chapter 1+ 2

Read Chapter 3

What we have covered:
Point Processes: Modification of brightness, contrast
Neighborhood processes: filtering, sharpening, edge detection.

Probability Distribution Function -→ Histogram

We can scale the histogram by making the sum of the values in the histogram equal to 1. This can be done using x/(number of pixels), where x is the number of values per intensity bin of the histogram.

Histogram-based modifications
- Contrast stretching: Stretching the histogram
- Equalization

Posterization: This is the process of thresholding, but using a set number of thresholds.

- Discussed "Walk along the Color Cube, a NASA Langley algorithm"
- Demonstrated Color Sawtooth.

Neighborhood Operations
- Sharpening
- Unsharp Masking
- Edge Detection
  - Problem inherent with Edge Detection algorithms? Noise
  - How to handle: Perform an amount of "smoothing" first.

Median filter: gets rid of salt and pepper noise.

Uses of **Standard Deviation**: This operation looks like edge detection because the varying amounts of intensities are greater in neighborhoods.

Morphological operations
- Erosion: this is an operation on the min of a neighborhood
- Dilation: This is an operation using the max of a neighborhood.

Point Processes: Trying to define a mapping from input to intensity R to output intensity S

$$S=T(r).$$

In general, our mappings (transformations) should be
(1) Single Valued
(2) Monotonically increasing
(3) Retain consistent intensity range

Gamma: s=c*r^gamma

Log: s=c*log r ➔ s=c*log(r+1) because log(0) is undefined

Histogram Modification

- Contrast stretching
  - Transformation is based on Min(i) and Max(i).

$T(r)= \{$ 0 for r<rmin

255/rmax-rmin * (r-rmin) for rmin<=r<=rmax
255 for r>rmax

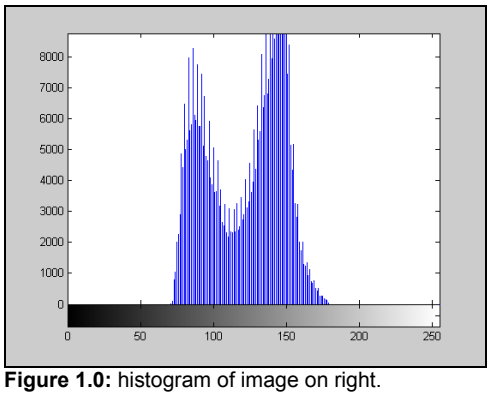


Figure 1.0: histogram of image on right.

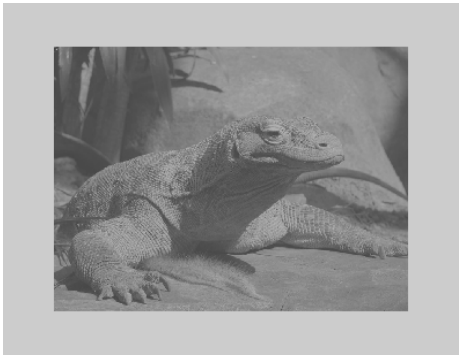


Figure 1.1: low contrast lizard.

In my code in matlab, why is the image too bright after the contrast stretch?

My calculated **Min** and **max** are **70** and **156**, respectively. If you look at figure 1.0, you can see these are "correct" measurements. However, it has to be that these pixels that are bright are being set to 256 (Matlab is 1 based)

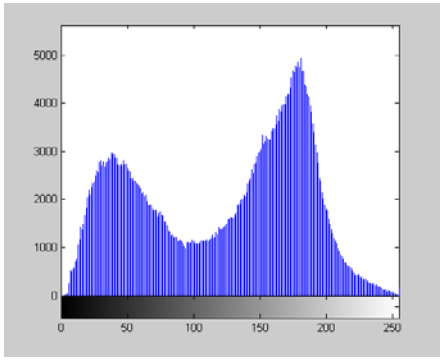


Figure 1.2: Histogram of good lizard (right)

Figure 1.3: good lizard picture.

I found the reason: Its because the maximum is being computed as ignoring the top 5 percent of pixels before computing the max index. So, now I just use the first non-zero entry. Being that this process doesn't work for every case, a proposed algorithm needs to be discussed. Incidentally, the new min and max are 70 and 181.

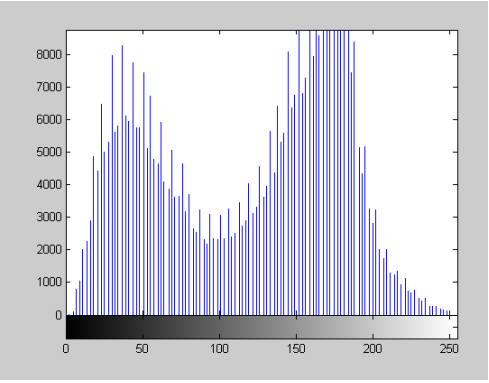


Figure: Contrast-stretched Histogram

Figure: Adjusted image.

Matlab Algorithm (after computed max and min), A represents the 2D representation of the lizard image.

```
for dx=1:size(A,1)
        for dy=1:size(A,2)
            if(A(dx,dy)<=mind)
                A(dx,dy)=0;
            else if(A(dx,dy)>=maxd)
                A(dx,dy)=255;
            else
                A(dx,dy)=(255.0/(maxd-mind)) *(double(A(dx,dy))-mind);
            end
        end
    end
```

Use a LUT for this for speed-up

- Histogram Modification
  - Produces the maximum global contrast based on cumulative distribution of pixel intensities.
  - The level in the histogram ->average(N^2/255)=➔ CDF(k) = summation (h(i)) over I=0 to k.

End of Notes for Sep. 10

September 12, 2003, Friday

Read Ch. 3
Start Chapter 6

Point Processes: e.g. binary thresholding, contrast, brightness, pseudocolor, gamma, log, posterization, negation, solarization, etc. f➔ g

Histogram modification: e.g. contrast stretching, equalization

All of these can be implemented using LUTs

Negation is not monotonically increasing!

Equalization

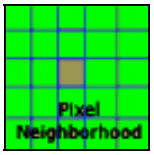
Histogram Transforms (mappings)

CDF(k) = summation

Clipped Histogram Equalization

"Adaptive Histogram Equalization", "Local", P 103 – Section 3.3.3

For each pixel
Perform Histogram Equalization
For just that pixel, use a small neighborhood.



- What is the effect of having a smaller neighborhood (compared to using the whole image)?

  Contrast will be brighter

- How can we speedup the neighborhood operation of Histogram equalization?

Remove one column of pixels from the histogram, and add the new one as we move across in a left-to-right fashion.

September 15, 2003, Monday

I am/was absent
Sept. 15, 2003, Colorado Springs, Co: Looks like assignment was given

Notes from Dave Steffan:

I won't be in Tuesday like I normally would be.
Here's a brief overview of notes for Digital Image Processing.
For most of the class period he went over the new programming assignment, which is posted on the course website.
the course directory on the Linux boxes at tech has the ImageLib files along with example programs he wrote to help learn how to use them.
Weiss's libraries take care of most of the GUI stuff we don't need to deal with, and gives a simpler way to handle that, since this isn't a GUI course.
example1.c on the linuxboxes has the basic setup for a basic GUI program using ImageLib.
Note that the data, when taken from the file, is stored in a struct, so you can technically access all the data fields in the struct yourself, Weiss suggest you use the functions in the struct to access.
You must write an InputEventHandler to deal with input (look in example program for mor! e details).
Weiss recommends you use a makefile, which tells the system how to compile things, put dependencies and things in there, and type "make" on the command line. He has an example makefile in the course directory on the linuxboxes.
I think all you need to alter in the makefile to compile a different source file is to change the source file name near the top of the makefile (Not sure,

he did that really fast). May want to add -g option in makefile for debugging purposes.
In the makefile, the dependencies are listed, for example when compiling example1, will look for example1.o first (or something like that).
example3 shows use of dialog boxes and some use of ImageLib.
example4 shows use of menus.
He has documentation for ImageLib on the website, note it is a work in progress, so parts may be missing or incomplete right now.
Some things in ImageLib are not implemented yet (I think he's a little be! hind from working for Realtronics this summer), he will get them up and running soon.
Weiss recommends linking files instead of copying them to your directory, that way if/when he updates them with newer versions, you will be using the newer versions automatically.
AddToMenu is used to create menus.
MenuSelection is the routine to deal with menu selections chosen by the user, the value passed to MenuSelection is an integer representing the menu item.

on to a little color image processing stuff.
note that it's harder to equalize an RGB image than a monochrome, if you do histogram equalization on each of the three colors, then you may end up with unwanted color shifts. Because the three histograms tend not to look the same, the effect of the function on each is different, which is why you may get unwanted color shifts. A way around this is to convert everything into Hue/Saturation/Intensity (which Weiss sometimes calls IC for Image/Chromaticity), then the Intensity can be changed direc! tly without affecting the color.

back to the program for a bit.
example5 shows how to set up keyboard shortcuts if you want to.
GetRows gets the number of rows in image
GetCols gets the number of columns in image
GetData gets the image (I think in a 1-D array)
GetData2D gets the image in 2-D array
ICtoRGB converts IC represented image into RGB represented image
RGBtoIC does the opposite
He also said of course we have to write our own functions for the actual image operations even though i think they're in the library.
He showed some of the examples in class, you should go try them out and look at the documentation.

September 17, 2003, Wednesday

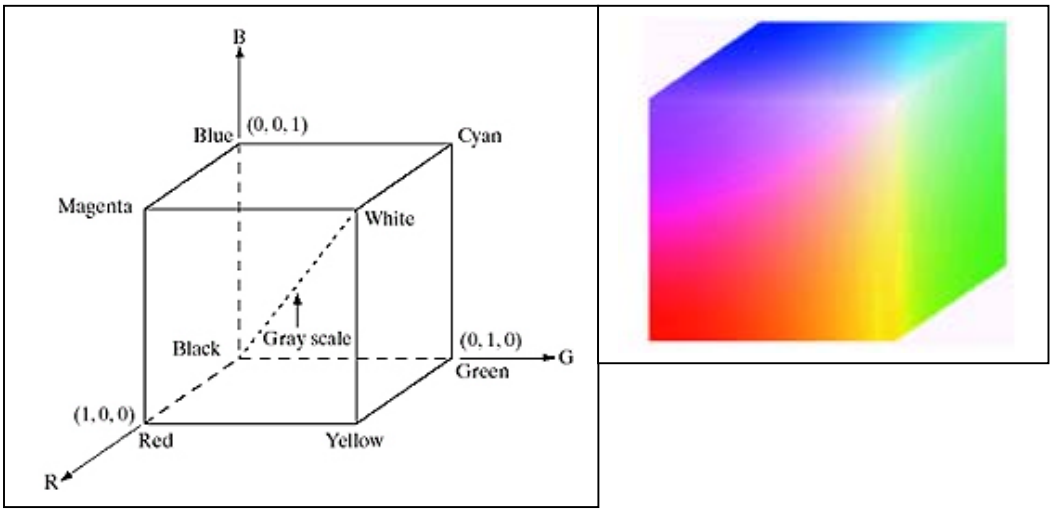
Read Chapters 3,6
PA #1 Due Mon. October 6, 2003

- ImageLib
- Color I.P.
- LUGNUTs meeting today @ 4 for installing Linux
- Color Models: RGB, CMY(K), YIQ, HIS, etc.
    - We will be working with models other than H.S.I.

RGB => is used by most color hardware
But for image processing, we want to separate intensity from chromaticity.

Definitions:
Lumanence:
Chromaticity:
Hue: Color
Saturation: purity of an intensity.

RGB: An "Additive" Color Model
Drew out the Color Cube and expressed what each part of the cube means.



Finding RGB Percentage
R=1/(r + g + b)
G=1/(r + g + b)
B=1/(r + g + b)
Where 0<R or G or B<1

CMY: A "subtractive" color model. Typically, it's of little use in image processing.

Cyan: **1-R**
Magenta: **1-G**
Yellow: **1-B**

C+M+Y (1,1,1)= 0 (Black) and C+M+Y (0,0,0)= 1 (white)

YIQ: Color T.V. Standard
Luminance = Y = Brightness/intensity
Inphase = I = Chromaticity
Quadrature = Q = Chromaticity

RGB→YIQ

Y=0.299R + 0.587G + 0.114B (This converts a color to grayscale by itself)
I =0.596R - 0.275G – 0.321B
Q=0.212R –0.523G + 0.311B

YIQ→RGB
R= Y+0.956I + 0.621Q
G= Y – 0.272I – 0.647Q
B= Y – 1.106I + 1.703Q

H.S.I.

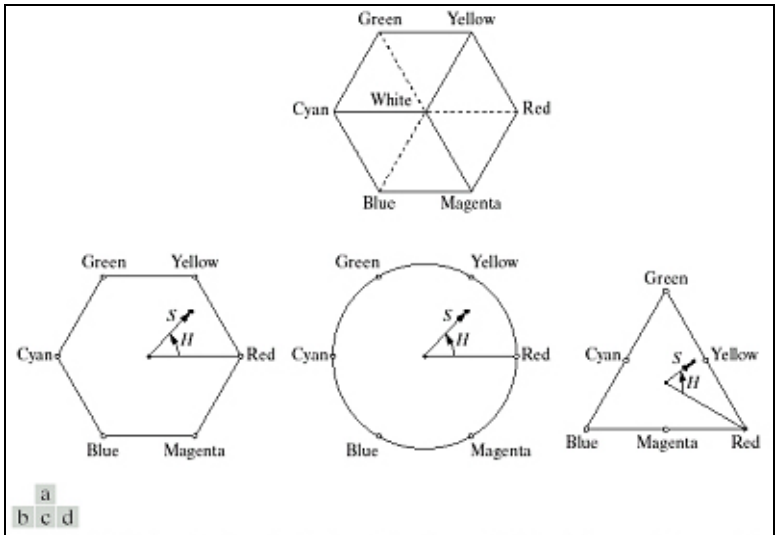


FIGURE 6.13 Hue and saturation in the HSI color model. The dot is an arbitrary color point. The angle from the red axis gives the hue, and the length of the vector is the saturation. The intensity of all colors in any of these planes is given by the position of the plane on the vertical intensity axis.

HIS Is similar to Human vision
The hue and saturation components make up the chromaticity.
Discussed Geometric interpretation

I=1/3(r+g+b)
S=1-(3/(r+g+b)) =min(rgb)=(1-min(rgb))/I
H=invcos(1/2(r-g)+(r-b)/sqrt((r-g)^2+ (r-B) (G-b))

What if our I value is 0? This is undefined

Discussed the use of Pseudocolor in remote sensing. For example, the thermal band would be green, the visible band would be red channel, etc.

September 19, 2003, Friday

Color Models
RGB
CMY(K)
YIQ
H.S.I

False color images

Functions that can be used for PA #1: GetData, GetRGBData, display functions, load and save functions.

ImageDescriptor structure

- rgbdata has data in [rgb][rgb] format …each color is a byte.
- Intensity holds the intensity data (the y component)

- Histogram holds the image histogram (0-255)
- Palette holds the color palette. [r][g][b] where r represents 0-255, g represents 0-255, b represents 0-255.

Neighborhood processes

- smoothing/blurring, sharpening, noise reduction, edge detection, filtering (spatial domain).
- Morphological operations (erosion, dilation, etc): "mathematical morphology"

Filtering:

$\sum W_i f_i$ where Wi is the weight (mask or filter) and fi is the pixel intensity.

Convolution➔ $$\sum_{i=-m}^{+m} \sum_{j=-n}^{n} W(i,j) * f(x+i, y+j)$$

Averaging: performs smoothing, denoising or noise reduction, blurring.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

A basic averaging filter looks like such.

Averaging works well with Gaussian noise (u=0, sigma=N)

Edges are discontinuities in an image. Essentially, an area where pixels are correlated has an area.

3x3 Weighted average

1	2	1
2	4	2
1	2	1

x 1/16 (for scaling to 1)

Sharpening (p. 125):

0	-1	0
-1	5	-1
0	-1	0

Characteristic of sharpening: Some weights have negative values.

Prewitt Operator (edge detector):

0	-1	0
-1	4	-1
0	-1	0

This works because equal pixels (surrounding) make a pixel go to 0 and some other value when they are not equal.

Unsharp Masking

Process:

- get original image
- subtract blurred copy from it.

$\alpha f(x,y) - \beta g(x,y)$ alpha=0.8, beta=0.2 for scaling (because we want to see many dark pixels).

In analog photographs,
Is it the same as applying edge detection first and then subtracting it from the original?

Smoothing operator that is best is the Gaussian.

September 22, 2003, Monday

Read Ch. 3, 6, Start Chap. 4 (Frequency Domain)
Neighborhood processes

- Image Filtering
- G(x,y) = f(x,y) * h(x,y)
- This is convolution (the convolution operator: *)

Rank Order Filters (not based on convolution)

-**Median filter**: Reduces Impulse Noise, i.e. salt and pepper noise.
-**Neighborhood minimum**: find the smallest intensity ← corresponds to erosion
-**Neighborhood Maximum**: Find the largest intensity ← corresponds to dilation
-**Range:**
-**Variance/standard deviation:**

What sort routine would work best for the median filter? Selection sort. **Why?** Only have to go through half the list.

What happens when we have white noise and we apply the neighborhood minimum? The white dots are removed, but we see that the image has darkened. We can restore the image to almost "original" quality by using the neighborhood maximum.

Range: This is the Maximum-miminum. What is the effect of applying this filter? We get an edge detected image.

Variance: a measure of the dispersion. Applying it produces an edge detection effect to the image.

"Noise Cleaning"

A "noisy" pixel is one that doesn't have correlation with surrounding pixels.

The noisy cleaning algorithm is as such:

$$P_0 = \begin{cases} P_0, if \left| P_0 - \frac{1}{8}\sum_{i=1}^{8} P_i \right| < \varepsilon \\ \frac{1}{8}\sum_{i=1}^{8} P_i, otherwise \end{cases}$$

P_1	P_2	P_3
P_8	P_0	P_4
P_7	P_6	P_5

So, if P0-average of its neighbors is less than the epsilon, then we leave P0 alone, otherwise we set it to the average of its neighbors.

Embossing: this operation's filter matrix looks like this:

0	0	0
0	1	0
0	0	-1

What is the problem with this filter matrix? There will be some negative and positive components in the output (range is –255 to 255). How to handle? There are several ways; one way is to add 255, divide by 2.

Edge Detection

An edge occurs at a sharp transition between dark and light pixel intensities.

Types of Edges
Step function
i.e. step edges
Ramp edges
sigmoidal edges

Some edge detection filters are based on the first derivative, i.e. the gradient.

September 24, 2003, Wednesday

- PA #1, due Oct 6, Monday (a week from the coming Monday)
- Read Chapter 3, 6, Start Chapter 4
- Rank Order Filtering
  - Median
  - Min
  - Max
  - Range |
  - SD/Variance | These two measure dispersion in the neighborhood.

- **Noise Cleaning**: See Previous class notes. This is to handle noisy pixels, i.e. pixels that are not closely correlated with it's neighboring pixels. Noise cleaning gives a way of handling this.
- **Edge Detection**

- ToUpper doesn't work in linux with ImageLib. Why?

- A new release of ImageLib in the next day or so to fix some issues, such as the scrollbars not working.

People are having problems following John Weiss's documentation on ImageLib. We discussed the functions, in more detail, today. Things described were; the palette functions, the RGB functions, the Intensity functions (only getting the I value of the HIS or YIQ model).

Palette is stored in RGB format, i.e. 256 reds, followed by 256 greens, followed by 256 blues.

How could one display only red for the intensity?

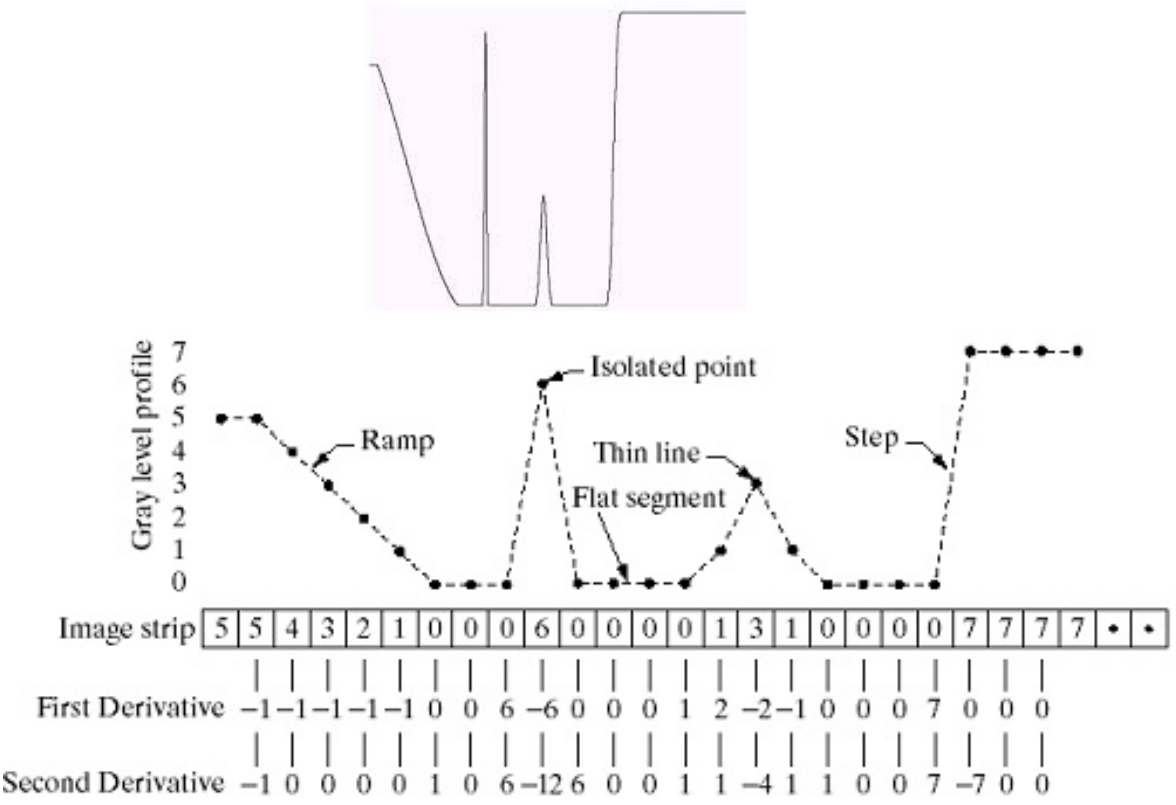
```
Int red=*rgbdata++;
*rgbdata++=0; //green
*rgbdata++=0; //blue
```

How do we do a simple check and setting of binary thresholding?

```
*data = *data <128?0:255;
```

Edge Detection

(1) First Derivative (gradient)
(2) Second Derivative
(3) Template Matching



Zero crossings in the second derivative

September 26, 2003, Friday

No Class, M-Day. Classes>=12 noon are let out.

September 29, 2003, Monday

Guest Speaker today: Karl Lamonthe
Discussion: Image Processing in Remote Sensing

Infrared Band

Absorption: Water absorbs infared, thus water shows up as black.

Operations: Band Math

- Red is typically used with Band Math.
- Iron is the operation of red/blue bands.
- Clay Index: Band 7/ Band 1.

Principal Component Analysis

Neural Networks

Aspens vs. pine

Terrain rendering in VR (here at SDSMT)

Image Generation: Augmented Paleontology

Examples for these two (below) are “Brain aneuryisms”

- 3D Medical Imaging
- Voxels Processing

October 1, 2003, Wednesday

- Finish 3, 6, Read Chapter 4
- Program #1 due Mon 10/6 (Midnight)
- Note:
- (a) New version of enhance in /home/classes/csc751
    - Saves the modified file (instead of the original).
    - Histograms displayed in different Windows
- (b) Contrast stretch specify inconsistency.

Gamma (power) s=c*r^gamma,
Try doing like (s/255^2) * 255 or r/255.00

Edge Detection: Chapter 10

Three major categories in Edge Detection

(1) First Derivative (gradient): local max/minima
(2) Second Derivative: (laplacian): Zero Crossings
(3) Template Matching

$$\vec{\nabla} f = \begin{bmatrix} \partial/\partial x \\ \partial/\partial y \end{bmatrix} = \vec{i}\,\frac{\partial}{\partial_x} + j\frac{\partial}{\partial_y}$$

$\vec{\nabla} f(x,y) = \frac{\partial f}{\partial x}\cos(\theta) + \frac{\partial f}{\partial y}\sin(\theta) = f_x * \cos\theta + f_y * \sin\theta$ (Gradient in the direction of θ).

Gradient Magnitude: $|\vec{\nabla} f| = \sqrt{f_x^{\,2} + f_y^{\,2}} = |f_x| + |f_y|$

Gradient Direction: $\theta = \tan^{-1}(f_y / f_x)$

Discrete Approximation to gradient: Use Differences

First Derivative Operators

- **Prewitt Edge Operator**

-1	0	1
-1	0	1
-1	0	1

-1	-1	-1
0	0	0
1	1	1

- **Sobel Edge Operator**

-1	0	1
-2	0	2
-1	0	1

-1	-1	-1
0	0	0
1	1	1

- **Roberts Edge Operator**

0	0	0
0	0	-1
0	1	0

0	0	0
0	-1	0
0	0	1

- **Frei Chen Edge Operator**

-1	0	1
$-\sqrt{2}$	0	$+\sqrt{2}$
-1	1	1

-1	$-\sqrt{2}$	-1
0	0	-1
1	$\sqrt{2}$	1

Why are the $-\sqrt{2}, \sqrt{2}$ used in these equations? It is because of the distance to diagonal elements are considered.

Apply for magnitude

(1) Vertical

(2) Horizontal
(3) Check Magnitudes
(4) Scale and/or clip as needed

Edge Linking can use the direction

Canny Operator: John Canny

- Gradient
- Hysteresis (using direction of the Gradient)
- **hysteresis** is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold
- **Hysteresis:** You will see that the output of this detector contains weak edges that we would like to eliminate. Simple thresholding has the problem that some weak edges are worth keeping. A clever trick is *hysteresis thresholding*, which eliminates weak edges below some low threshold, but not if they are connected to strong edges above some high threshold through some chain of pixels all above the low threshold.

- Called a DOG (Derivative of Gausssian)

Second derivative or "Laplacian"

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$ It is a scalar quantity (vs. a vector quantity)

Discrete Approximations

0	-1	0
-1	4	-1
0	-1	0

or

-1	-1	-1
-1	8	-1
-1	-1	-1

LOG: Laplacian of Gaussian
DOG: Derivative of Gaussian (Canny)

October 3, 2003, Friday

Finish 3,6 start/read ch. 4, also some materials in Chapters 5 and 10
Chapter 5: *Image Restoration*
Chapter 10: *Image Segmentation*

PA #1, extension to homework due date, 10/8

New version of ImageLib as of Wednesday afternoon.
ImageLib doc: fixed omission of FreeRGBData2D

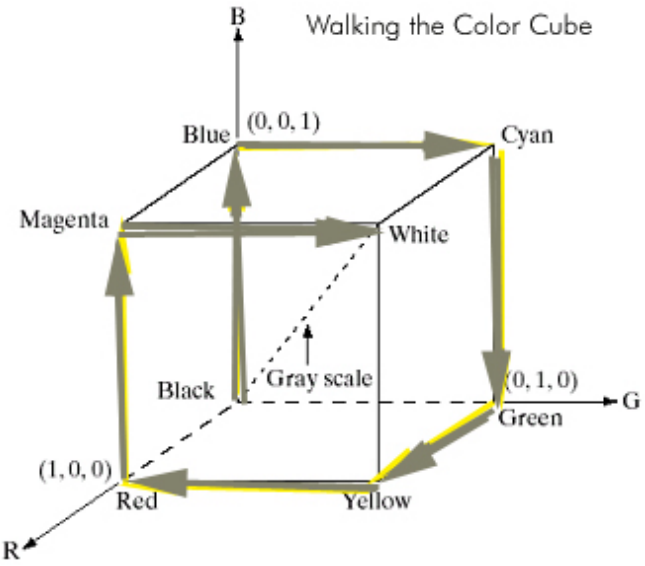
Enhance demo was updated
Binary threshold, for example, has been changed.

Midterm up to chapter 3, 6 scheduled for Oct. 15th

Contrast Enhancement Questions

What is the difference between “increasing contrast” and “contrast histogram specification”?

Pseudocolor techniques for Assignment



Walking the Cube Algorithm

1) Start at black, and move to blue. Hold all values steady except for blue.
    a. Blue: 0-255
2) Move to Cyan (0,255,255), keep all values “steady”, while increasing green.
    a. Green: 0-255
3) Move to Green, change the blue value to move slowly to pure green.
    a. Blue: 255-0
4) Move to yellow (255,255): Increase the red value slowly.
    a. Red: 0-255
5) Move to Red, reduce green.
    a. Green:255->0.
6) Move to Magenta(255,0,255), increase blue
    a. Blue: 0->255
7) Move to white, increase green.
    a. Green: 0-255

Edge Detection Revisited

(1) gradient (or 1st derivative)

a. e.g., Sobel

What is the largest magnitude we can see from applying the edge detection algorithm?

It will be outside of the max intensity. Clip or scale as needed.

(2) Laplacian (or 2nd derivative): Marr-Hildreth edge op.

- non-directional
- looking for zero crossings

How to perform the LoG:
(1) Gaussian smoothing
(2) Apply laplacian

When the two are combined, we have the famous shape the "Mexican sombrero".

Canny: 1st derivative of the Gaussian.

(3) Template Matching (correlation)

Rotational 3-level Edge Operator

-1	-1	-1
0	0	0
1	1	1

270 degrees

-1	0	1
-1	0	1
-1	0	1

0 Degrees

1	1	1
0	0	0
-1	-1	-1

90 degrees

1	0	-1
1	0	-1
1	0	-1

180 degrees

Magnitude: $R(x, y = \max_i(R_i \oplus f(x, y))$

$\oplus$: Correlation operator

Direction: Given by Ri with max response.

The resolution on the filters are such that we've got an error of +-45 degrees. We should attempt to increase our resolution. How can this be done?

By adding more masks

-1	-1	0
-1	0	1
0	1	1

45 degrees

Kirsch Operator

-3	-3	-3
-3	0	-3
5	5	5

There are 8 different orientations for this.

Prewitt Operator

-1	-1	-1
1	-2	1
1	1	1

Separability
What is separability and how can we use it?

Separability is the ability to implement a 2D filter as two successive 1D filters.

1	1	1
1	1	1
1	1	1

x 1/9 ==>

1
1
1

1	**1**	**1**

How is it useful to us? It allows us to speed things up because the number of operations is smaller.

An mxm filter takes m^2 multiplies and m^2 additions w/o separability.
M multiplies and m adds with.

References

Pratt, William K., *"Digital Image Processing", John Wiley and Sons, Inc,* 2nd Edition, 1991
Gonzalez, Woods, *"Digital Image Processing"*, 2nd Edition, Prentice Hall, 2002

Cool Links

Morphological Operations

Morphological image analysis applied to crop field mapping

http://ams.jrc.it/soille/ivc2000/