ELIC 629

Digital Image Processing

Winter 2005

Image Enhancement in the Spatial Domain:

Basic Gray Level Transformations: image negatives, log transformations, power and piecewise-linear transformations, histogram processing

Bill Kapralos

Monday, October 3 2005

ELIC 629, Fall 2005, Bill Kapralos

Overview (1):

- **Before We Begin**
  - Administrative details
  - Review → some questions to consider
- **Linear and Non-Linear Operators**
  - Linear operators
  - Non-linear operators
- **Image Enhancement in the Spatial Domain**
  - Introduction → what is image enhancement ?
  - Background to processing in the spatial domain

Overview (2):

- **Basic Gray-Level Transformations**
  - Introduction
  - Image negatives
  - Log transformations
  - Power law transformations
  - Piece-wise linear transformations
- **Histogram Processing**
  - Introduction
  - Examples

Before We Begin

Administrative Details (1):

- **Miscellaneous Notes**
  - Assignments and lab reports submitted late will result in a penalty
  - No lecture next week (Oct. 10)
    - Thanksgiving

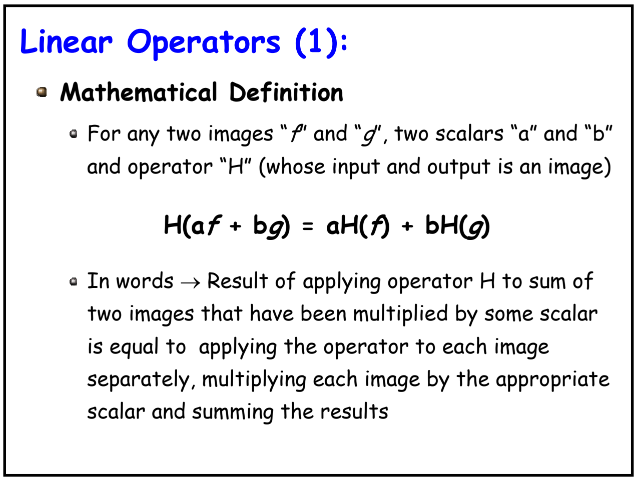
Administrative Details (2):

- **Lab Three**
  - Will involve the use of a digital camera which I will distribute (along with appropriate equipment) during the lab
  - No lab report required for this lab but there is a lab assignment due October 17 (two weeks from now)

Some Questions to Consider (1):

- What is spatial sub-sampling (shrinking) ?
- What is spatial up-sampling (zooming) ?
- What is aliasing ?
- What is interpolation ?
- Define a pixel’s 4 & 8-neighbours
- What is adjacency ?
- Define connected, region and boundary
- List and define the three distance measures discussed

Linear & Non-Linear Operators

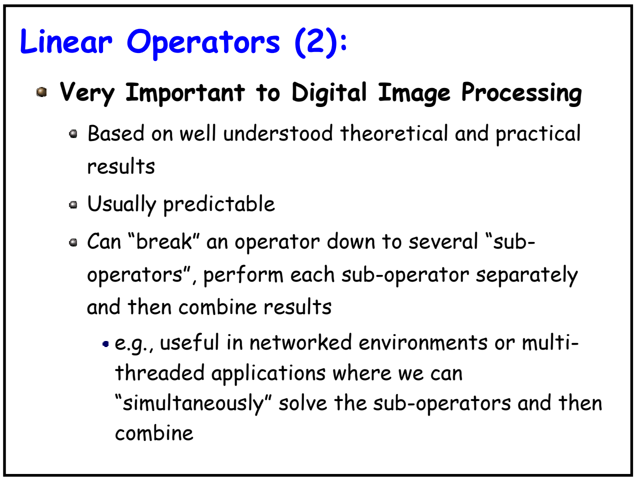


Linear Operators (1):

- **Mathematical Definition**
  - For any two images "*f*" and "*g*", two scalars "a" and "b" and operator "H" (whose input and output is an image)

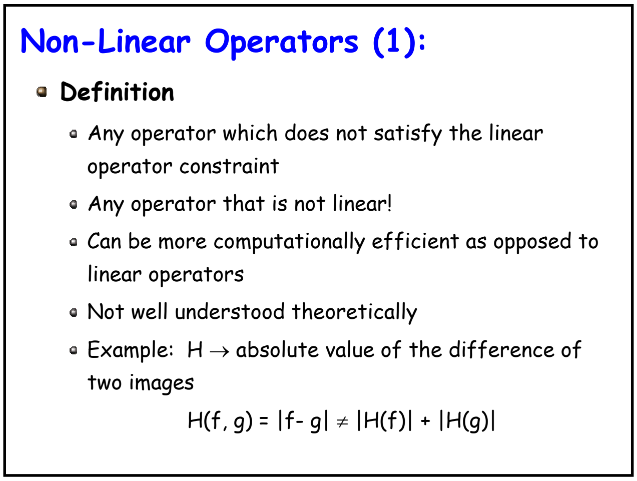
$$H(a f + b g) = aH(f) + bH(g)$$

  - In words → Result of applying operator H to sum of two images that have been multiplied by some scalar is equal to applying the operator to each image separately, multiplying each image by the appropriate scalar and summing the results



Linear Operators (2):

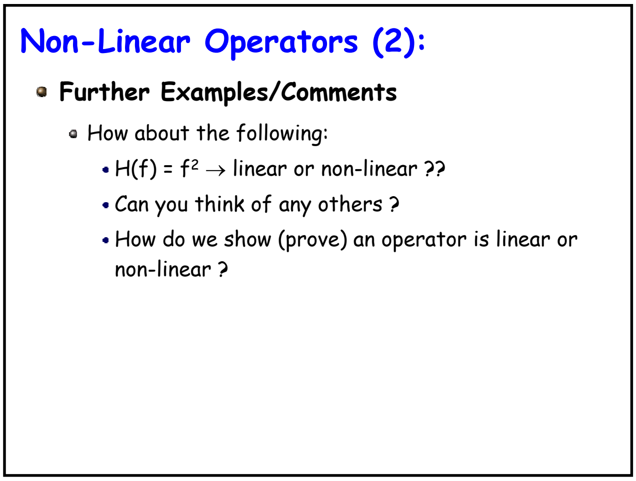
- **Very Important to Digital Image Processing**
  - Based on well understood theoretical and practical results
  - Usually predictable
  - Can "break" an operator down to several "sub-operators", perform each sub-operator separately and then combine results
    - e.g., useful in networked environments or multi-threaded applications where we can "simultaneously" solve the sub-operators and then combine



Non-Linear Operators (1):

- **Definition**
  - Any operator which does not satisfy the linear operator constraint
  - Any operator that is not linear!
  - Can be more computationally efficient as opposed to linear operators
  - Not well understood theoretically
  - Example: H → absolute value of the difference of two images

$$H(f, g) = |f - g| \neq |H(f)| + |H(g)|$$



Non-Linear Operators (2):

- **Further Examples/Comments**
  - How about the following:
    - $H(f) = f^2$ → linear or non-linear ??
    - Can you think of any others ?
    - How do we show (prove) an operator is linear or non-linear ?

Introduction: Image Enhancement in the Spatial Domain

What is Image Enhancement ? (1):

- **Purpose**
  - Process an image so that the resulting image is more suitable than the original for a specific application
  - Important → application specific
    - Application (needs) determine the process applied to the image (or portion of an image)
  - One of the most interesting and visually appealing areas of image processing
    - You get to see the results, typically immediately!

What is Image Enhancement ? (2):

- **Two Broad Categories of Image Enhancement**
  - Spatial domain processing
    - Processing applied to the image plane itself e.g., direct manipulation of the pixel intensity values
  - Frequency domain processing
    - Processing applied to the Fourier spectrum of the image
    - Need to "compute" the Fourier transform of spatial sdomain representation first
    - Once processing done, convert Fourier representation back to spatial domain

What is Image Enhancement ? (3):

- **Application (Viewer) Based**
  - No general method (steps to follow) to image enhancement
    - Dependant on the application
    - Dependant on the viewer who is ultimately the end "user" of the processed image → subjective and once again, my view of a good image may differ from yours!
    - Makes it very difficult to compare/judge image enhancement methods although one thing to consider is processing time

Spatial Processing – Background (1):

- **Denoting a Spatial Domain Process**
  - Denoted by the expression

$$g(x,y) = T[f(x,y)]$$

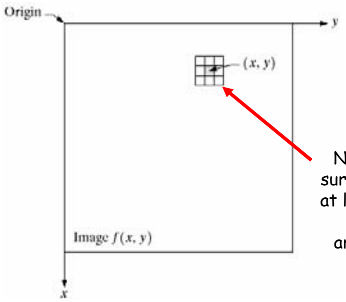
  - f(x,y) → input (original) image
  - g(x,y) → output (processed) image
  - T → a process (operator) on f defined over some neighborhood of spatial position (x,y) but can also operate on a set of images

Spatial Processing – Background (2):

- **Denoting a Spatial Domain Process (cont…)**
  - How do we define a neighborhood over (x,y) ?
    - Most popular approach → use a square or rectangular "sub-image" area (neighborhood) centered about pixel (x,y)
    - Process T is applied to pixels in sub-image only → e.g., g(x,y) is calculated based on the values of the pixels in this sub-image centered about (x,y)
    - Of course, not restricted to square or rectangular neighborhood → can use circular, elliptical etc. but much more complicated!

Spatial Processing – Background (3):

- **Denoting a Spatial Domain Process (cont…)**
  - Graphical illustration of neighborhood



Neighborhood surrounding pixel at location (x,y) – can be of arbitrary size

Spatial Processing – Background (4):

- **Kernels or Masks or Templates or Windows**
  - Neighborhoods greater than 1x1
  - A small 2D array (matrix, sub-image, grid etc.)
  - The individual values of the mask (kernel etc.) are known as the mask coefficients
    - Determine the "nature" of the process (e.g., the process performed on the output is based on the value of the coefficients)
    - Mask processing or mask filtering → enhancement techniques based on operators that use a mask, kernel etc. (discussed later…)

Spatial Processing – Background (5):

- **Further Details Regarding Operator "T"**
  - Simplest form of operator T is one which uses a neighborhood of size 1 x 1 (single pixel) - in other words, output (intensity) at g(x,y) is determined by using the intensity at f(x,y) only
    - Known as a gray-level (or intensity or mapping) transformation function denoted by:

$$s = T(r)$$

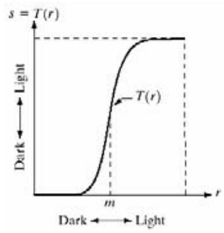
    - r → intensity at input pixel location (x,y)
    - s → intensity at output pixel location (x,y)

Spatial Processing – Background (6):

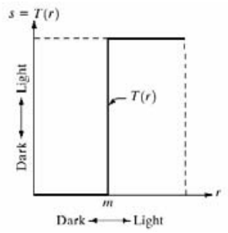
- **Example of a 1x1 Operator**
  - Contrast Stretching → output image with higher contrast intensity
    - Intensities of input image below "m" are compressed into narrow range of "s" toward black (e.g., darkened)
    - Intensities above "m" are brightened
    - Thresholding → In the limiting case, binary image results

Spatial Processing – Background (7):

- **Contrast Stretching**



Values below m are "compressed" into small region of output & values above m are stretched



Thresholding in the limiting case

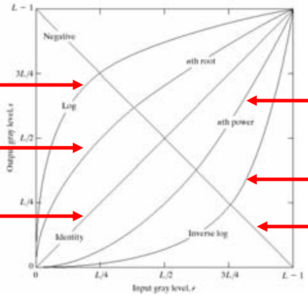
Basic Gray-Level Transformations

Introduction (1):

- **Simplest Image Enhancement Techniques**
  - Recall → values of pixels before processing known as “r”, after processing “s” and related by $s = T(r)$
  - Three common types of gray-level transformations
    1. Linear (negative and identity transformations)
    2. Logarithmic (log and inverse-log transformations)
    3. Power-law (n^{th} power & n^{th} root transformations)

Introduction (2):

- **Linear, Logarithmic & power Law Operators**



Log

n^{th} root

Identity

n^{th} power

Inverse log

Negative

Image Negatives (1):

- **Equivalent to a Photographic Negative**

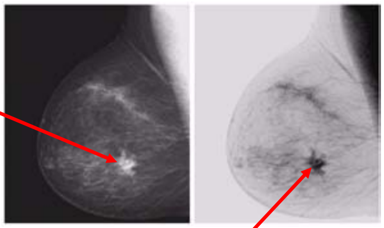
$$s = L - 1 - r$$

  - L → maximum intensity value (e.g., 255)
  - Reversing the intensity levels of an image
  - Primary use → enhancing white or gray detail embedded in dark regions of image especially when these dark regions are largest
    - Medical imaging → can detect problems (lesions etc.)

Image Negatives (2):

- **Mammogram Example**
  - Small legion in original easier to analyze in negative!

Legion in original image



Legion in enhanced image is more evident and easier to analyze

Log Transformations (1):

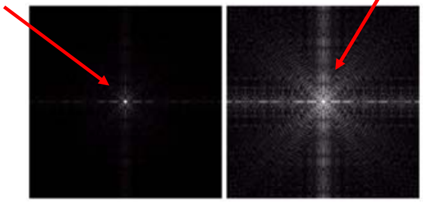
- **Expand Values of Dark Pixels in Image, Compress Higher Level Values**
  - Map narrow range of low gray-levels into wider range of output values
  - Map higher range of high gray-levels into narrow region of output values
  - Particularly useful for compressing dynamic range of images with large variations in pixel values
    - Fourier transform → high dynamic range (0 – 10^6) can't be represented on display device → devices cannot faithfully reproduce such a wide range

Log Transformations (2):

- **Graphical Illustration**

Fourier spectrum with intensity values in range 0 – 1.5 x 10^6 – no detail evident!

After log transformation – detail now evident



- Most Fourier spectra we look at will be log transformed

Power Law Transformations (1):

- **Mathematically**

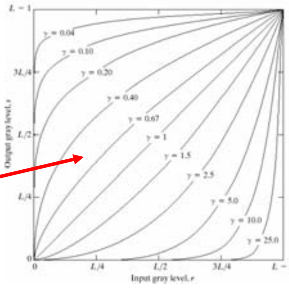
$$(1) \qquad s = cr^{\gamma} \text{ or}$$

$$(2) \qquad s = c(r + \varepsilon)^{\gamma}$$

- $c > 0$, $\gamma > 0$ (γ → "gamma")
- (2) is used as an "offset" when input is zero (e.g., when input is zero, output will be not zero!)
- As with log transform, fractional values of γ (e.g., $\gamma < 1$) map narrow range of dark input values to wider range of output values and higher input levels to narrow range of output values

Power Law Transformations (2):

- **Mathematically (cont…)**
  - When $\gamma > 1$ → opposite effect!



Power law transform for various values of γ

Power Law Transformations (3):

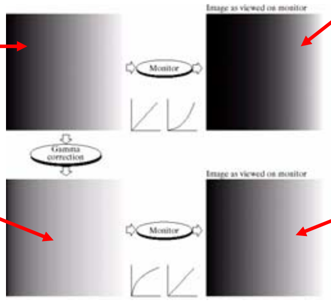
- **Power Law is Everywhere!**
  - Although you may not be aware of it, a large number of devices including digital cameras, printers, monitors etc. respond to their input via a power law
  - Automatically "adjust" their input so that it is scaled following a power law
  - We typically have to account for this scaling and apply inverse power law to output of device
    - This is known as gamma correction

Power Law Transformations (4):

- **Gamma Correction**
  - The process used to account (correct) for the power law scaling
  - Example → cathode ray tubes (CRTs) intensity-to-voltage response is a power function with gamma values ranging from 1.8 to 2.5
    - Produces images that are darker than we would expect!
    - Can easily account for this by applying an inverse power law process → $s = r^{1/\gamma}$ where γ is original gamma value

Power Law Transformations (5):

- **Gamma Correction (cont…)**
  - Graphical illustration



Gray-scale image viewed on monitor

Original gray-scale image

Gamma corrected image viewed on monitor

Gamma correction via inverse power law transformation

Power Law Transformations (6):

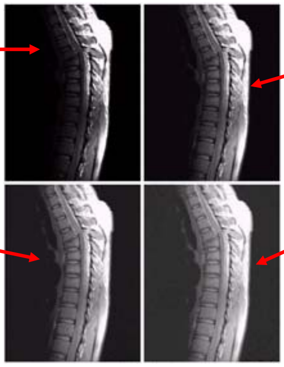
- **Gamma Correction (cont...)**
  - Important if you need to display image accurately on a computer screen e.g., medical imaging or color based applications
  - If not applied properly then
    - Image can appear too dark or "washed out"
  - Particularly relevant over the last several years given images on the internet with the many types of displays used to view them
    - Typically use an average gamma to process them

Power Law Transformations (7):

- **Gamma Correction (cont...)**
  - In addition to gamma correction, also used for modifying contrast in general
    - Useful in medical imaging → allows for details to be seen which might not have otherwise been visible

Power Law Transformations (8):

- **Gamma Correction (gamma < 1)**



Original MRI image

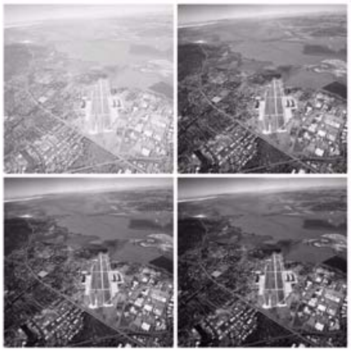
c = 1, γ = 0.6

c = 1, γ = 0.4

c = 1, γ = 0.3

Power Law Transformations (9):

- **Gamma Correction (gamma > 1)**



Original aerial image

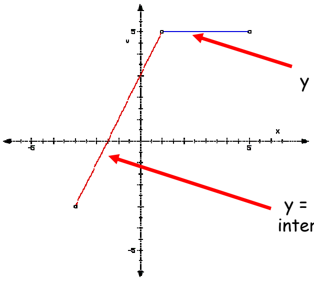
c = 1, γ = 3.0

c = 1, γ = 4.0

c = 1, γ = 5.0

Piecewise-Linear Transformations (1):

- **Function Composed of Two or More Pieces**
  - Typically non-continuous and may not necessarily be defined at all points especially where pieces "meet"



y = 5 on interval (1, 5)

y = 2x + 3 on interval (-3, 1)

Piecewise-Linear Transformations (2):

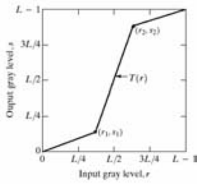
- ***Advantages***
  - The form of piecewise functions can be arbitrarily complex
  - Some important transformations can only be formulated as piecewise linear functions
- **Disadvantages**
  - Specifying them requires more user input
    - More variables to deal with etc.
    - Handling the "boundary" conditions can sometimes be problematic

Piecewise-Linear Transformations (3):

- ***Contrast Stretching***
  - Increase the contrast of an image
    - Low contrast may result from → poor illumination, lack of dynamic range in sensor etc.
  - Simplest piecewise linear function → increase the dynamic range of gray-levels in image

Piecewise-Linear Transformations (4):

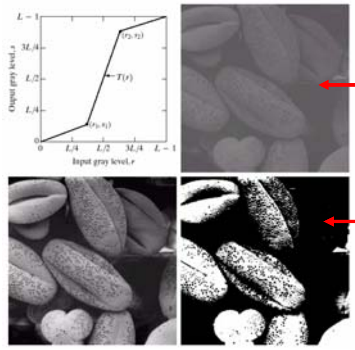
- ***Contrast Stretching (cont…)***
  - Example: (r_1, s_1) and (r_2, s_2)
  - If $r_1 = s_1$ and $r_2 = s_2$ → no change
  - If $r_1 = r_2$ and $s_1 = 0$ $s_2 = L-1$ → thresholding function – binary image
  - Generally assume $r_1 \leq r_2$ and $s_1 \leq s_2$ → ensures function is monotonically increasing avoiding introduction of un-wanted artifacts



Piecewise-Linear Transformations (5):

- ***Contrast Stretching (cont…)***

Transform function



Low contrast image

Thresholding

$(r_1, s_1) = r_{min}, 0)$ &
$(r_2, s_2) = r_{max}, L-1)$

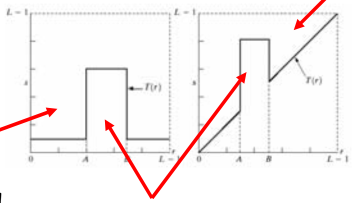
Piecewise-Linear Transformations (6):

- **Gray-Level Slicing**
  - Used to highlight a specific range of gray levels
    - Enhancing masses of water in satellite images, X-rays etc.
  - Several ways to accomplish this
    1. Display high gray level value for all gray levels in specific region and low value for everything else
    2. Same as 1) but preserve background

Piecewise-Linear Transformations (7):

- **Gray-Level Slicing (cont…)**
  - Graphical illustration

Preserve background



Background not preserved

Gray-level range of interest

Histogram Processing

Histograms - Introduction (1):

- **What is a Histogram (H_f) of an Image ?**
  - A plot or graph of the frequency of occurrence of each gray-level in the image
    - 1D function with domain [0, … L-1] and range from 0 to total number of pixels in image
    - Basically, for each gray level k_i of an image, the histogram gives you a count of how many pixels have this particular gray level k_i
    - Mathematically:

$$H_f(k_i) = J$$

where H_f → histogram, k_i → gray level "i", J → number of occurrences of gray level "k_i" in image

Histograms - Introduction (2):

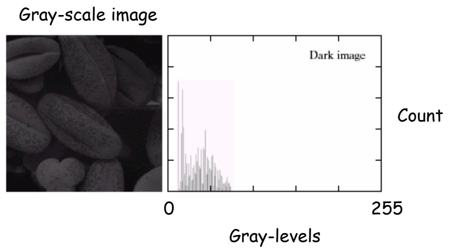
- **Histogram Normalization**
  - Common to normalize a histogram such that its return value is between 0 to 1.
    - Accomplished by dividing each of the histogram's values by total number of pixels in image
    - Basically, the normalized output gives the probability of the occurrence of the particular gray-level "k"
    - Sum of all histogram values is equal to 1
    - Think of it as providing a distribution of gray-levels in the image

Histograms - Introduction (3):

- **Histogram Properties**
  - Image histograms are a fundamental construct in image processing and widely used!
  - Simple to use and can accomplish good results, quickly, including for real-time applications
  - Important: histogram results in a reduction of dimensionality → cannot deduce image f from the values of the histogram – it does not provide us spatial information of the gray-levels
    - Provides only a count of the occurrence of a particular gray level in the image

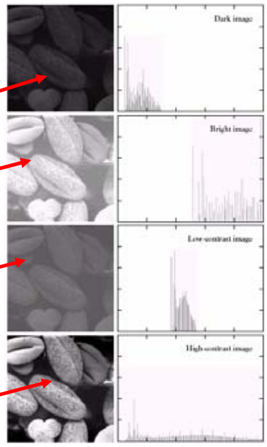
Histograms - Introduction (4):

- **Histogram Illustration**



Histograms – Intro. (5):

- **Histogram Illustration**



Dark image

Bright image

Low contrast image

High contrast image