Digital Image Processing

Lecture # 5

Image Enhancement in Spatial Domain- I

ALI JAVED

Lecturer

SOFTWARE ENGINEERING DEPARTMENT

U.E.T TAXILA

Email:: ali.javed@uettaxila.edu.pk

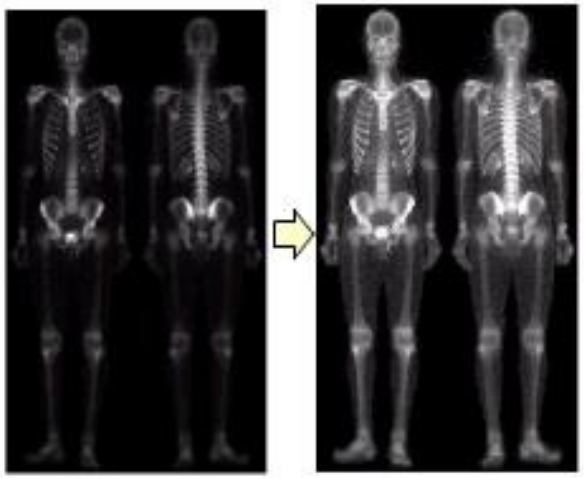
Office Room #:: 7

Presentation Outline

- Image Enhancement
- Basic Operations of Image Enhancement
  - Point Operations
  - Local Operations
  - Global Operations
- Gray Level Transformation Functions
  - Identity Function
  - Image Negation
  - Power Law transform
  - Log Transform
  - Piece Wise Linear Transform
    - Contrast Stretching
    - Gray level Slicing
    - Bit Plane Slicing
- Arithmetic/Logical operations on Images

Image Enhancement

- Process an image to make the result more suitable than the original image for a specific application
- The reasons for doing this include:
  - Highlighting interesting details in the image
  - Removing noise from images
  - Making images visually more appealing
- *Image enhancement is subjective (problem /application oriented)*



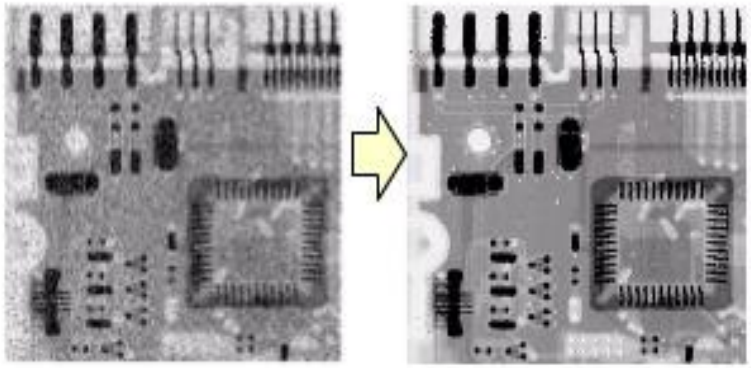


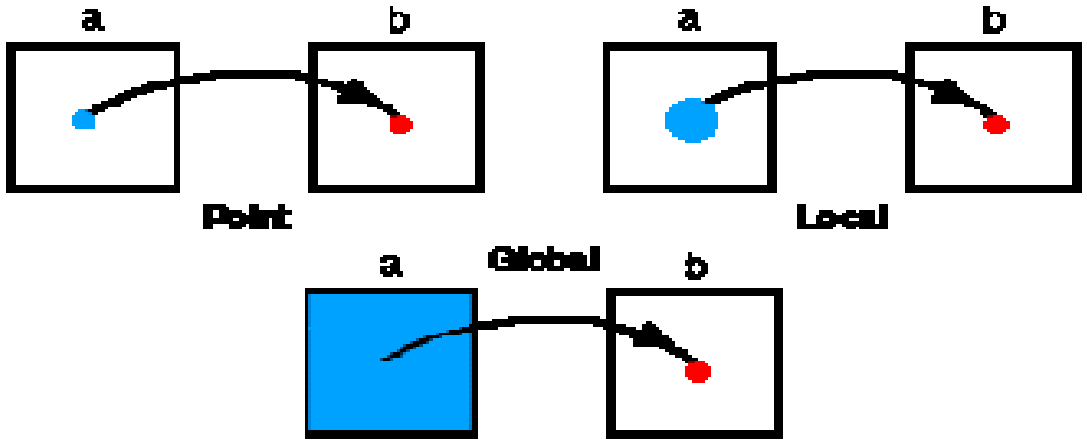
Image Enhancement

- There are two broad categories of Image enhancement techniques:
  - Spatial domain: Direct manipulation of pixel in an image (on the image plane)
  - Frequency domain: Processing the image based on modifying the Fourier transform of an image
- Many techniques are based on various combinations of methods from these two categories

Image Enhancement

Types of image enhancement operations

Point/pixel operations	Output value at specific coordinates (x,y) is dependent only on the input value at (x,y)
Local operations	The output value at (x,y) is dependent on the input values in the *neighborhood* of (x,y)
Global operations	The output value at (x,y) is dependent on all the values in the input image



Basic Concepts

- Spatial domain enhancement methods can be generalized as
  - $g(x,y) = T[f(x,y)]$

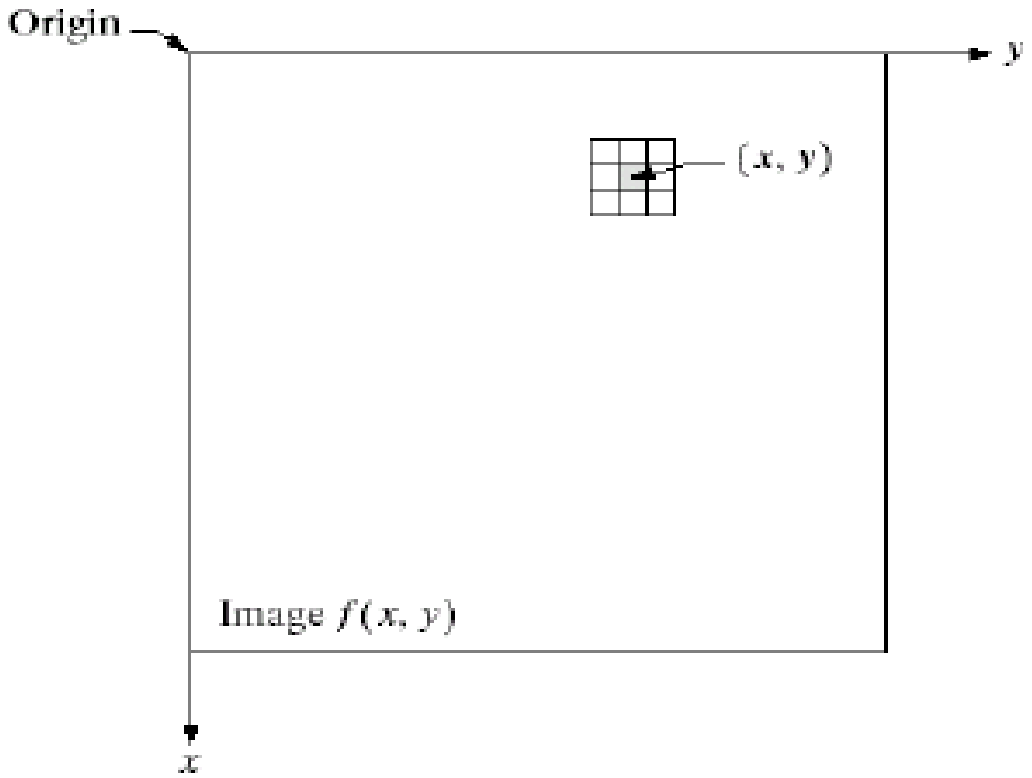
  $f(x,y)$: input image

  $g(x,y)$: processed (output) image

  $T[*]$: an operator on f (or a set of input images), defined over neighborhood of (x,y)

- *Neighborhood about* (x,y): a square or rectangular sub-image area centered at (x,y)

Basic Concepts

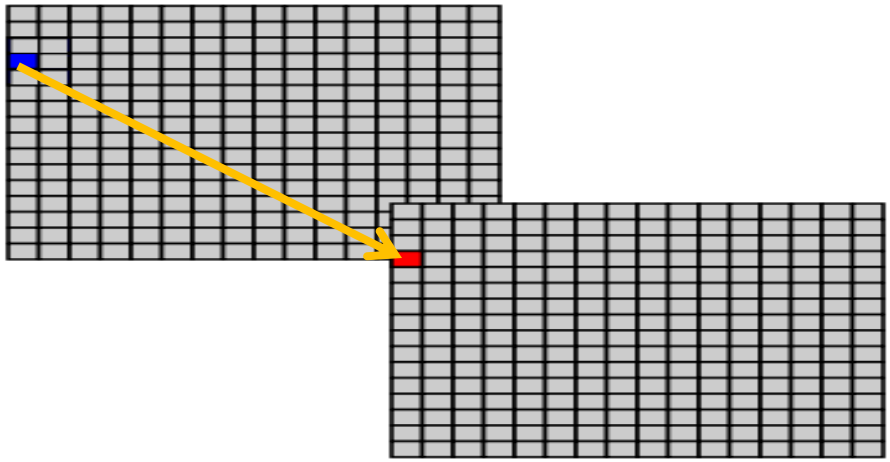


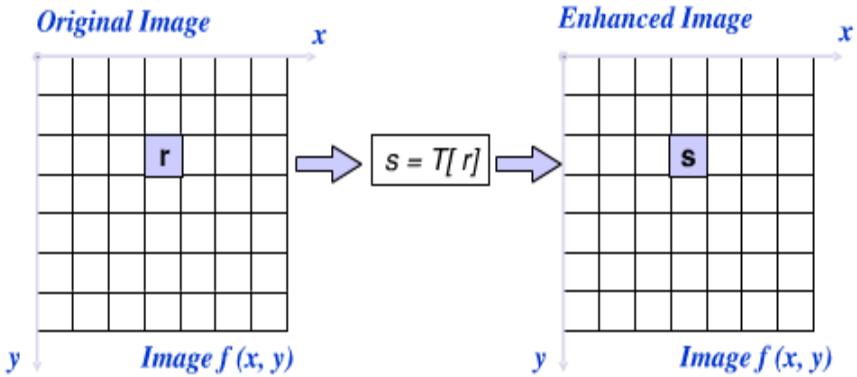
3x3 neighborhood about (x,y)

Pixel Operations

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

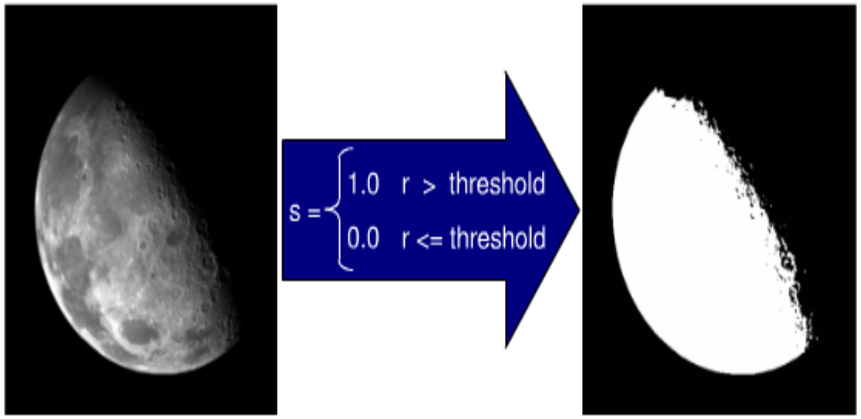
- Pixel/point operation: The simplest operation in the image processing occurs when the neighborhood is simply the pixel itself
  - Neighborhood of size 1x1: g depends only on f at (x,y)
  - T: a gray-level/intensity transformation/mapping function
  - Let $r = f(x,y)$ $\quad s = g(x,y)$
  - r and s represent gray levels of f and g at (x,y)
  - Then $s = T(r)$





Pixel Operations

- Example
  - Image negation or invert
  - Power Law Transform
  - Log Transform
  - Piece wise linear Transform
  - Thresholding



Thresholding

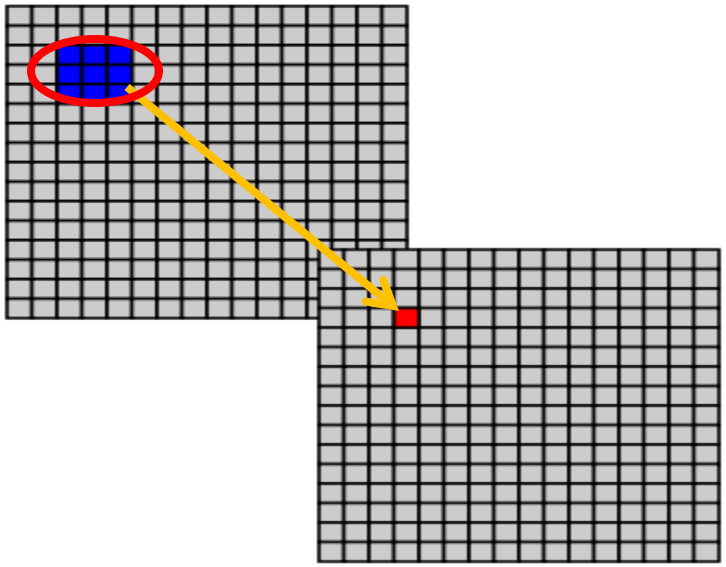
Image negation

Power Law Transform

Local Operations

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

- Local operations:
  - g depends on the predefined number of neighbors of f at (x,y)
  - Implemented by using mask processing or filtering
  - Masks (filters, windows, kernels, templates) :
  - a small (e.g. 3×3) 2-D array, in which the values of the coefficients determine the nature of the process



Local Operations

- Example
  - Image Smoothing (Noise Removal)
  - Image Sharpening (Edge Detection)

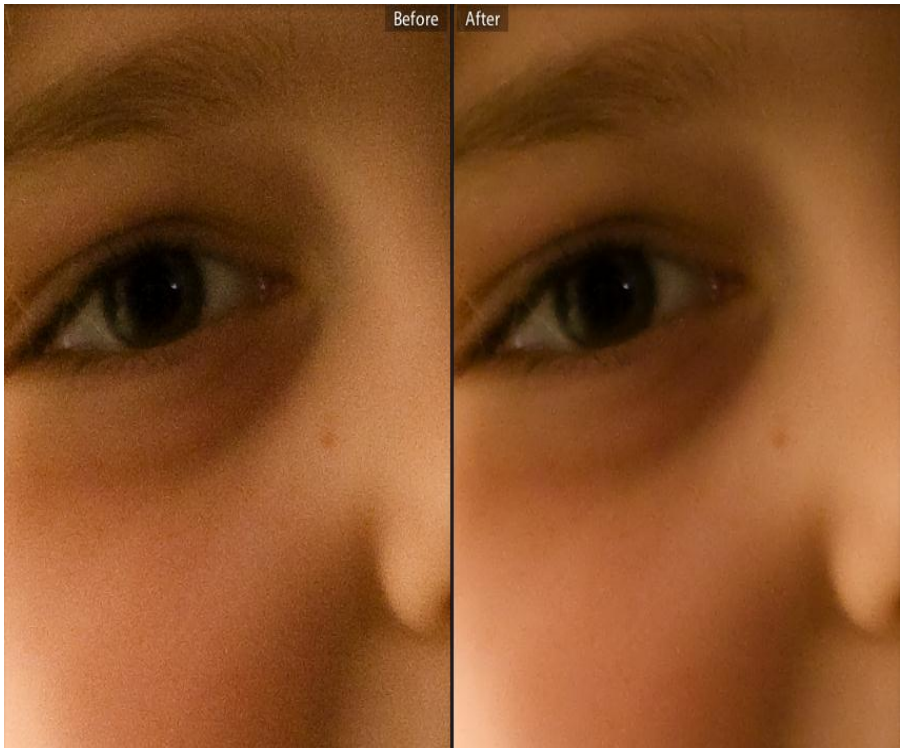


Image Smoothing

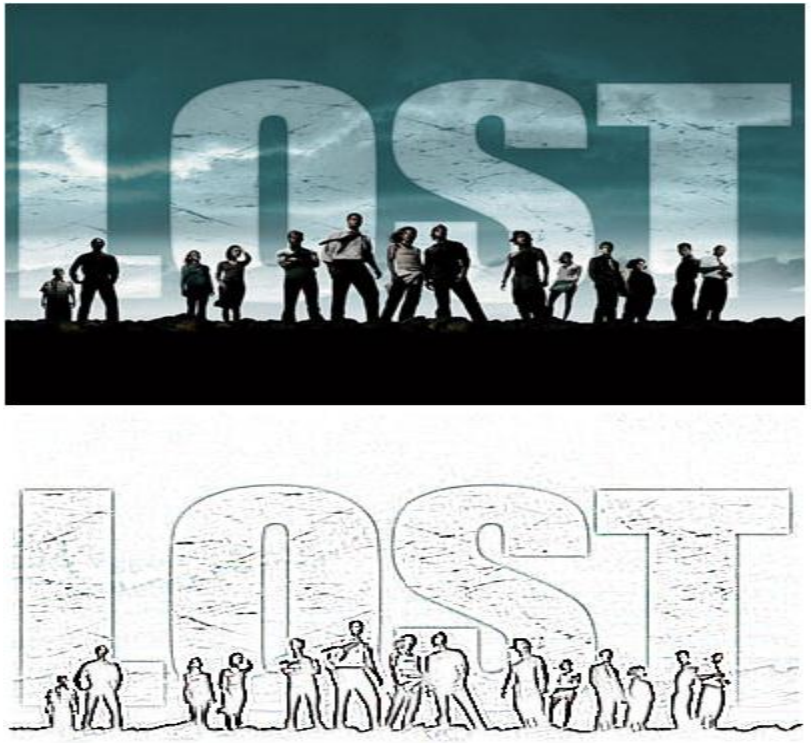
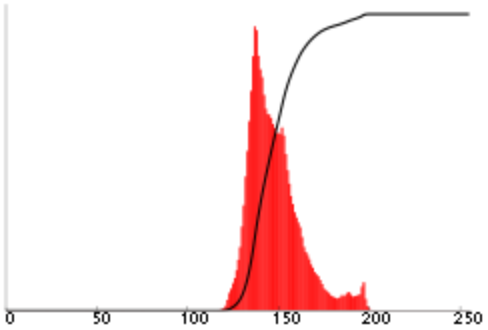


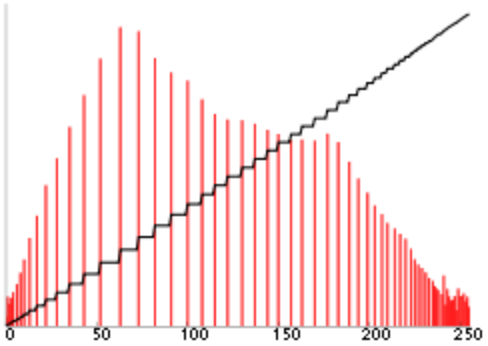
Image Sharpening

Global Operations

- Global operations:

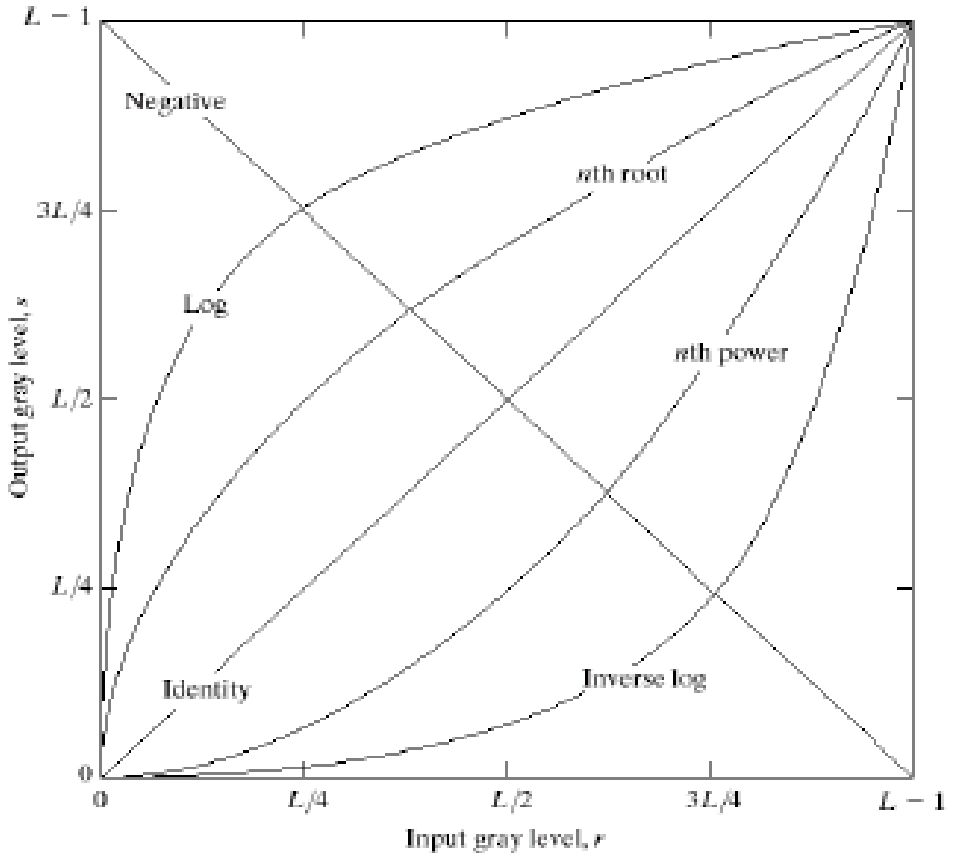
  ❑ An operation on an image that will manipulate the images as a whole
  ❑ Example:: Histogram Equalization





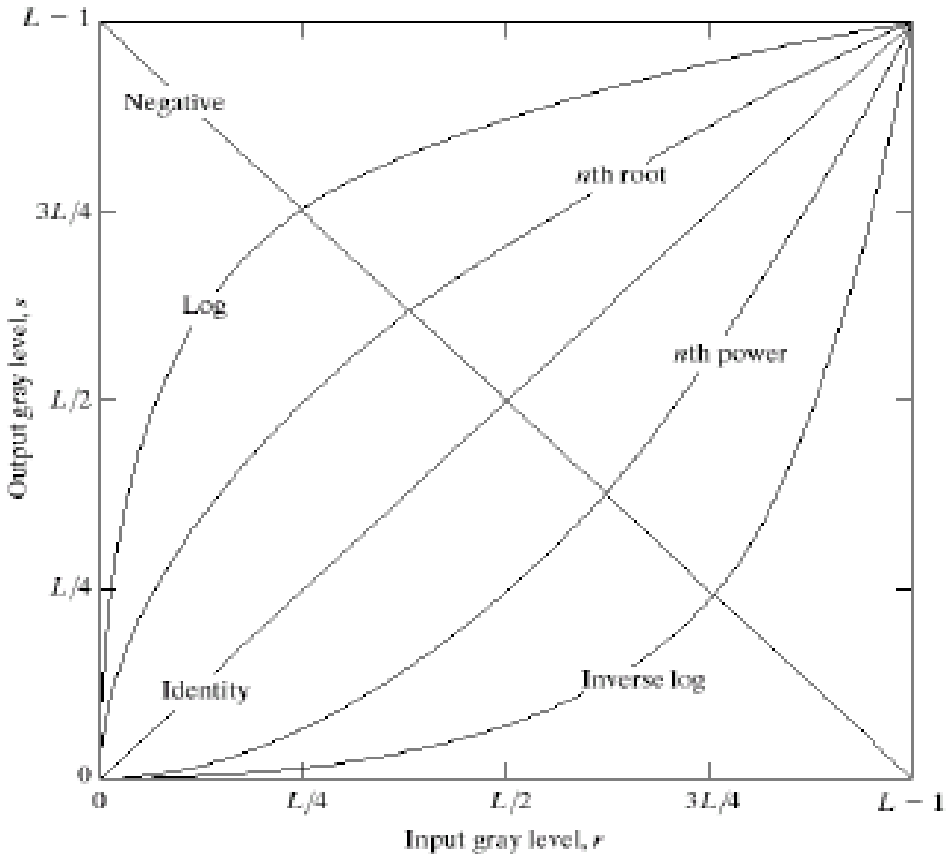
3 basic gray-level transformation functions

- Linear function
  - Negative and identity transformations
- Logarithm function
  - Log and inverse-log transformation
- Power-law function
  - n^{th} power and n^{th} root transformations



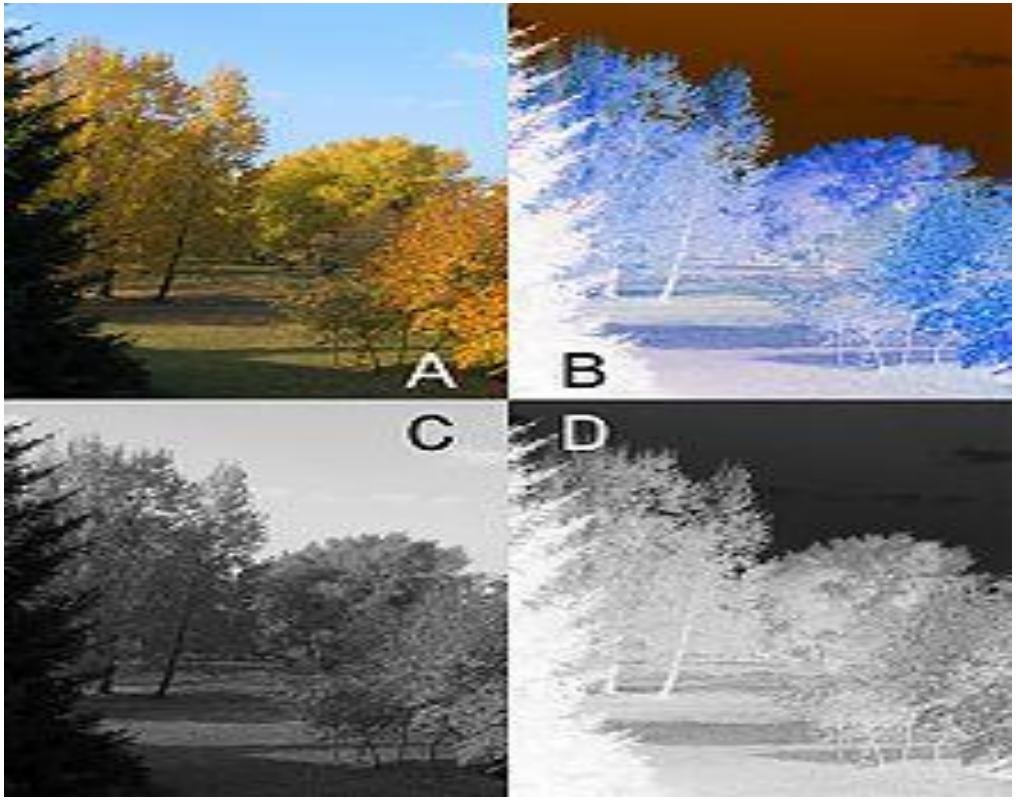
Identity Function

- Output intensities are identical to input intensities.
- Is included in the graph only for completeness.



Negative Image

- A negative image is a total inversion of a positive image, in which light areas appear dark and vice versa. A negative color image is additionally color reversed, with red areas appearing cyan, greens appearing magenta and blues appearing yellow.



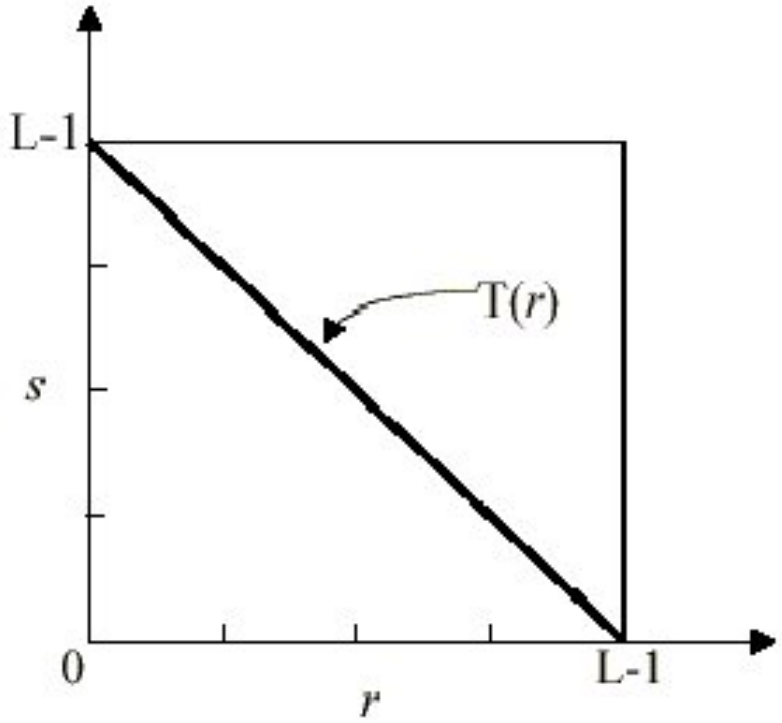
Color, positive picture (A) and negative (B), monochrome positive picture (C) and negative (D)

Negative Image

- Image Negative is a typical grey scale transformations that does not depend on the position of the pixel in the image.

- The output grey value s is related to the input grey value as follows:

s=T(r)



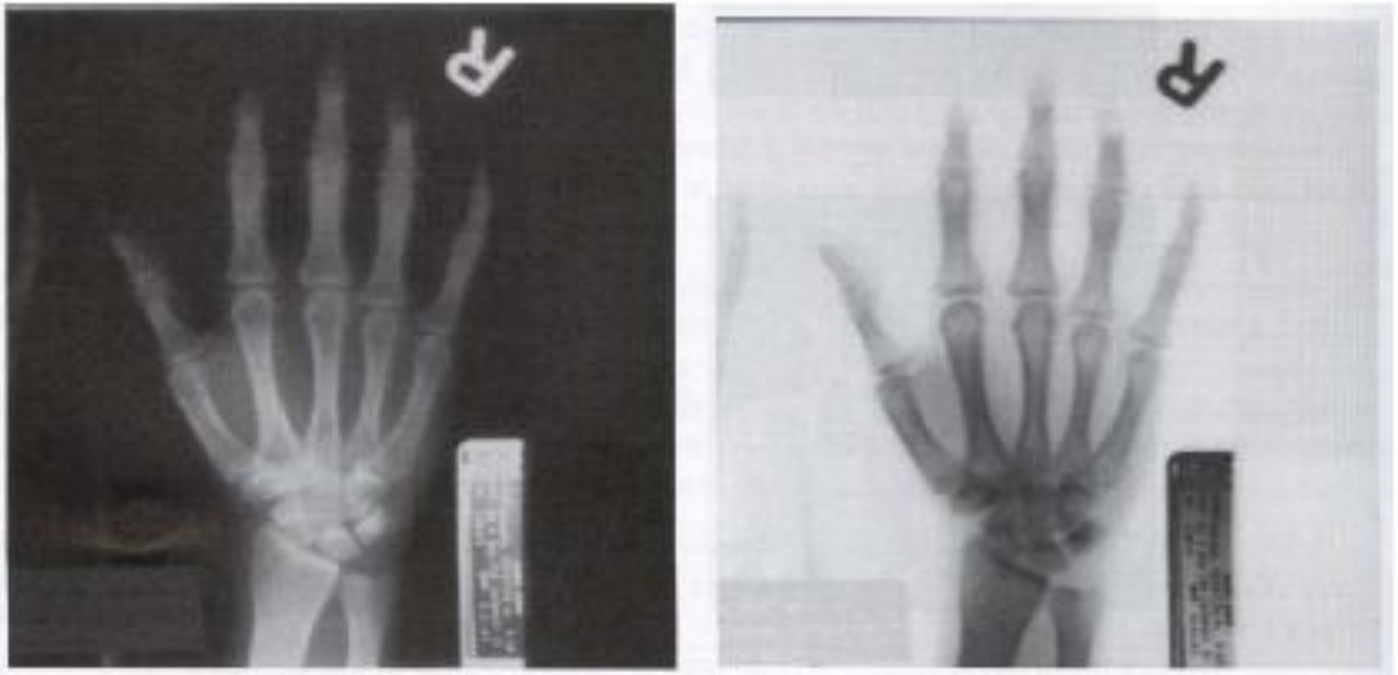
Gray level transformation function for obtaining the image negative of an image

Negative Image

- Reverses the gray level order

- For L gray levels the transformation function is

$$s = T(r) = (L - 1) - r$$

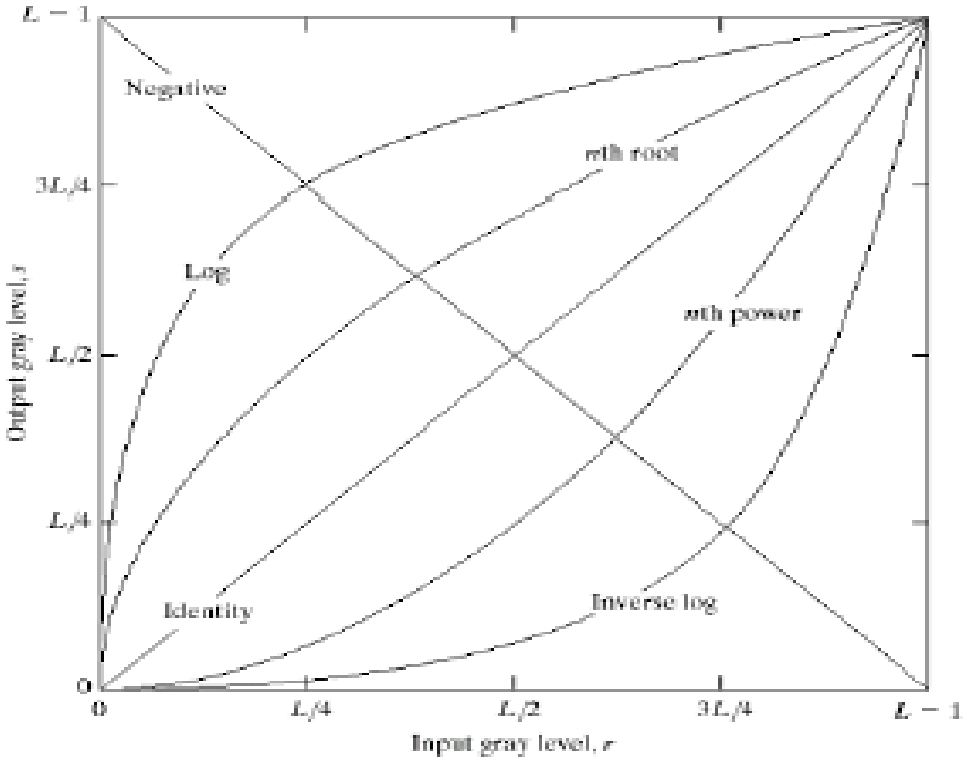


Input image (X-ray image) Output image (negative)

Logarithmic Transformations

- Log Transformation is particularly useful when the input gray level values may have an extremely large range of values

- **Function of Log Transform , $s = c * \text{Log}(1+r)$**
  - **r= Input Pixel Values**
  - **s= Output Pixel values**

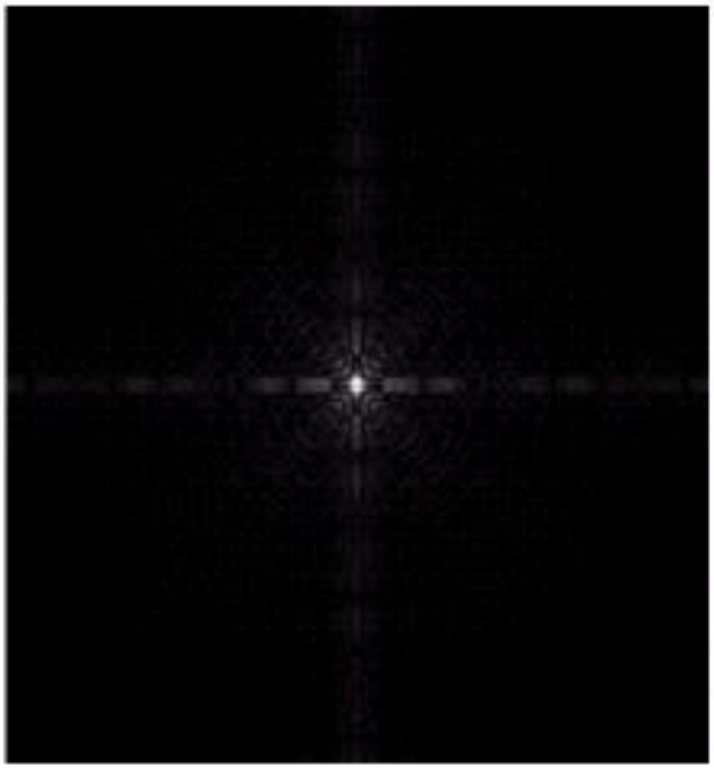


Logarithmic Transformations

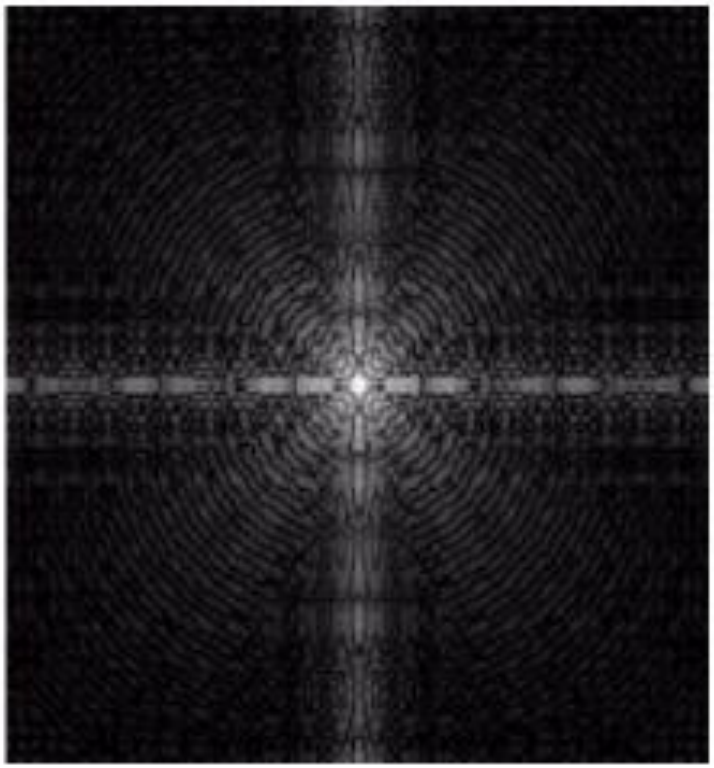
- **Properties of log transformations**
  - *For lower amplitudes of input image the range of gray levels is expanded*
  - *For higher amplitudes of input image the range of gray levels is compressed*
- **Application:**
  - *This transformation is suitable for the case when the dynamic range of a processed image far exceeds the capability of the display device (e.g. display of the Fourier spectrum of an image)*
  - *Also called "dynamic-range compression / expansion"*

Logarithmic Transformations

- In the following example the Fourier transform of the image is put through a Log Transformation to reveal more detail



Fourier spectrum with values of range 0 to 1.5×10^6 scaled linearly

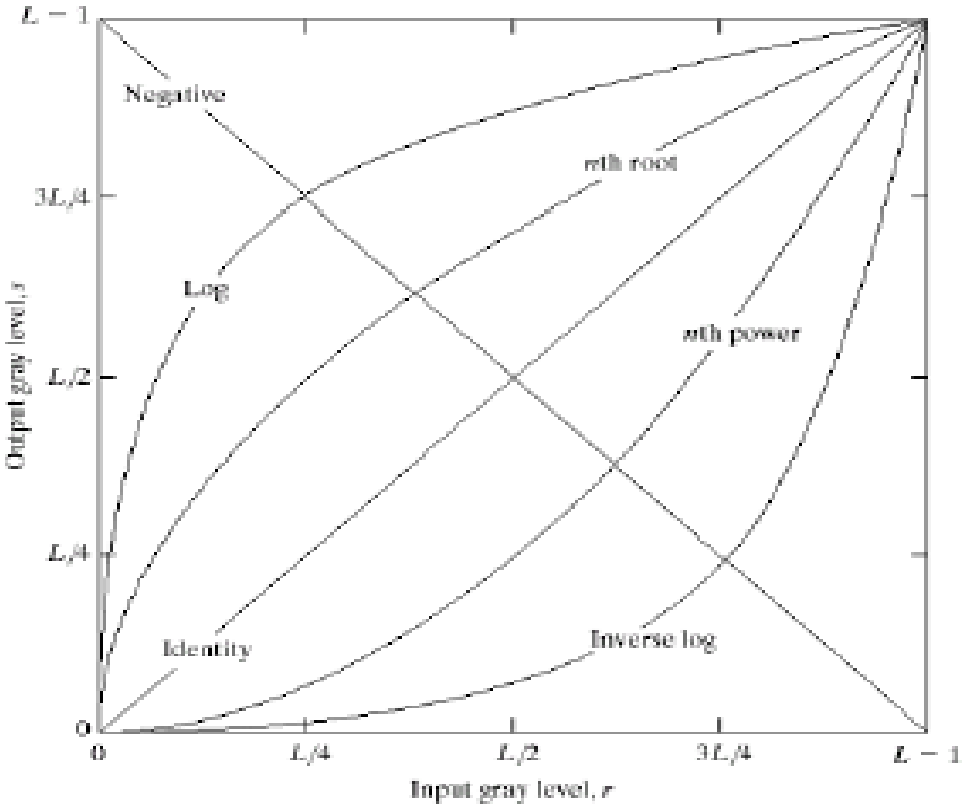


The result applying log transformation, $c = 1$

Inverse Logarithmic Transformation

- Do opposite to the Log Transformations

- Used to expand the values of high pixels in an image while compressing the darker-level values.



Power Law Transformation

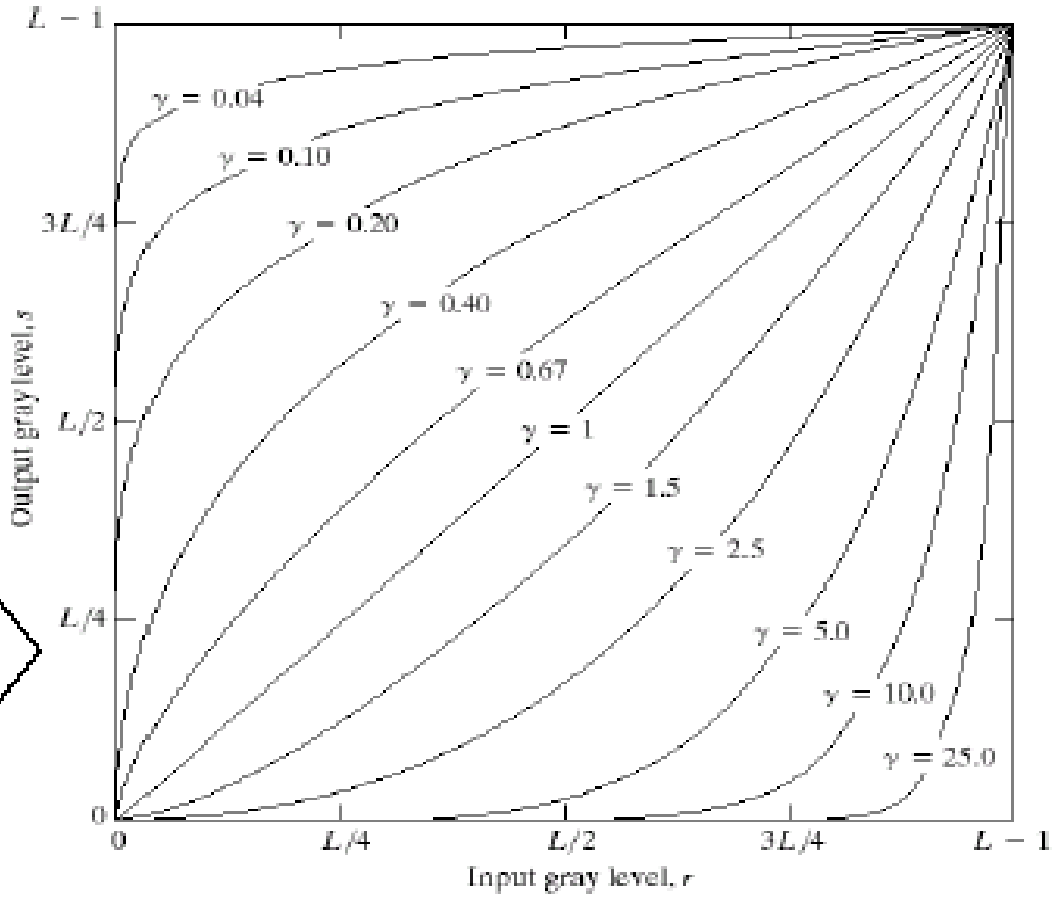
- Map a narrow range of dark input values into a wider range of output values and vice versa

- Varying Gamma gives a whole family of curves

Basic form:

$$s = cr^{\gamma},$$

where c & γ are positive



Plots of equation $s = cr^{\gamma}$, For various values of γ (c = 1)

Power Law Transformation

- For $\gamma < 1$: Expands values of dark pixels, compress values of brighter pixels

- For $\gamma > 1$: Compresses values of dark pixels, expand values of brighter pixels

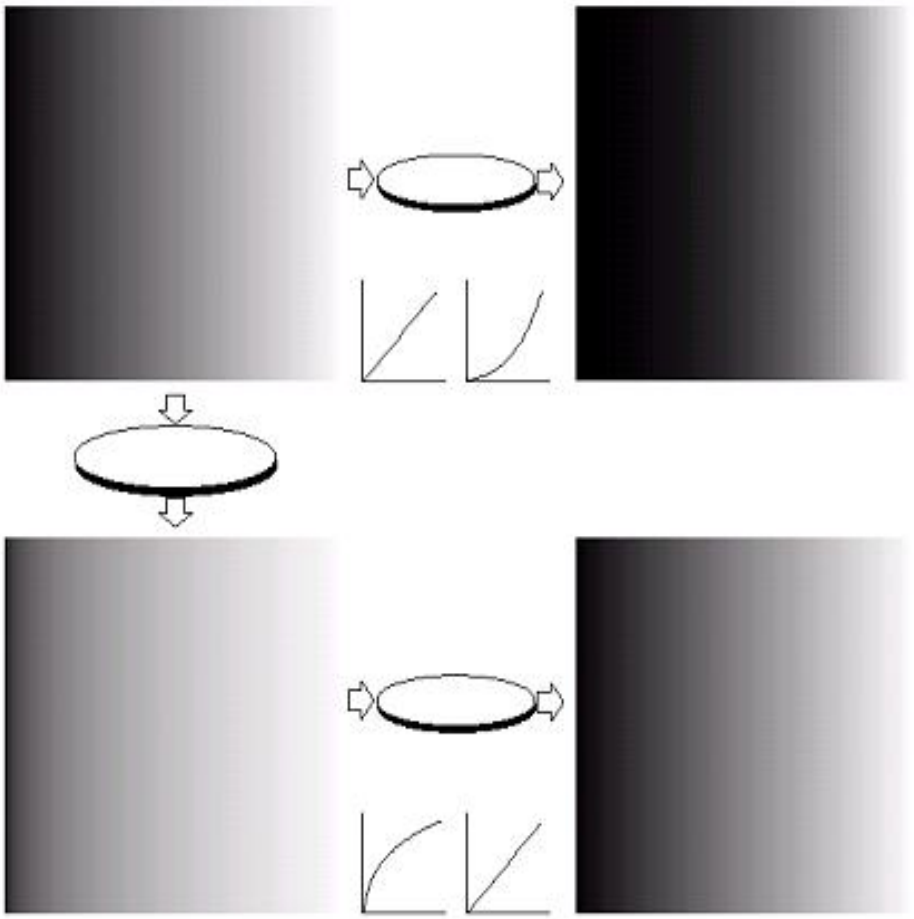
- If $\gamma=1$ & $c=1$: Identity transformation ($s = r$)

  - ❑ A variety of devices (image capture, printing, display) respond according to power law and need to be corrected

- Gamma (γ) correction
  - ❑ The process used to correct the power-law response phenomena

Gamma Correction

- Cathode ray tube (CRT) devices have an intensity-to-voltage response that is a power function, with γ varying from **1.8 to 2.5**
- The picture will become darker.
- Gamma correction is done by preprocessing the image before inputting it to the monitor with $s = cr^{1/\gamma}$



Power Law Transformation: Example

- The images shows the Magnetic Resonance image of a fractured human spine
- Different curves highlight different details

MRI image of fractured human spine

Result of applying power-law transformation

$c = 1, \gamma = 0.6$

Result of applying power-law transformation

$c = 1, \gamma = 0.4$

Result of applying power-law transformation

$c = 1, \gamma = 0.3$

Power Law Transformation: Example

- An aerial photo of a runway is shown
- This time Power Law Transform is used to darken the image
- Different curves highlight different details

Piece Wise Linear Transformation

Contrast Stretching

- **Goal:**

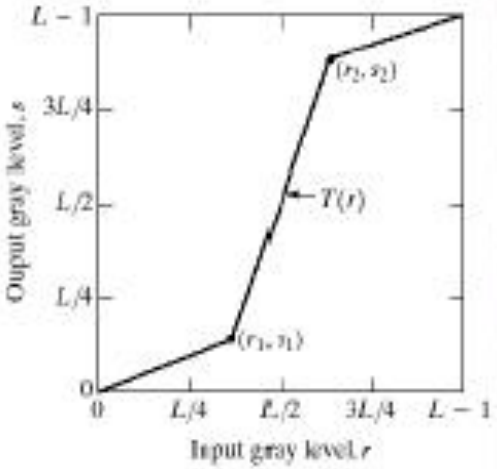
  ❑ Increase the dynamic range of the gray levels for low contrast images

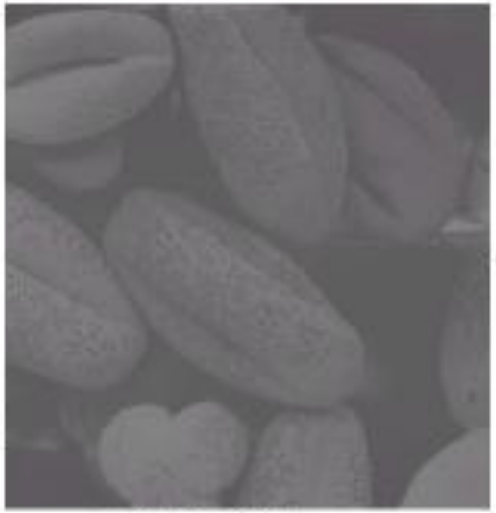
- **Low-contrast images can result from**

  ❑ poor illumination
  ❑ lack of dynamic range in the imaging sensor
  ❑ wrong setting of a lens during image acquisition

Contrast Stretching: Example

Form of Transformation function





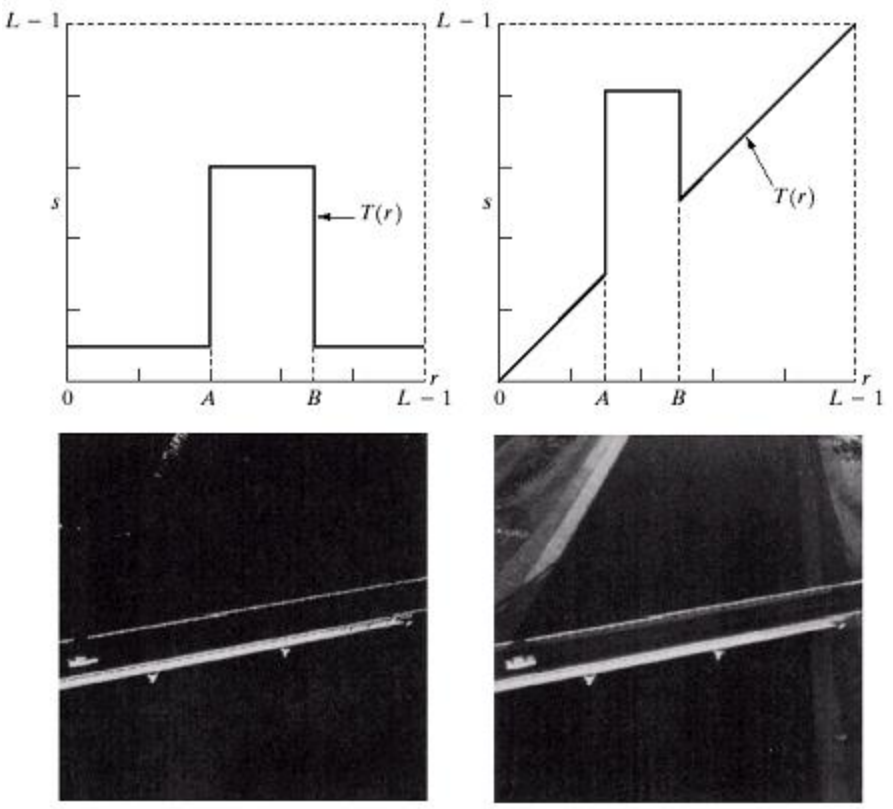
Original low-contrast image

Result of contrast stretching

Result of thresholding

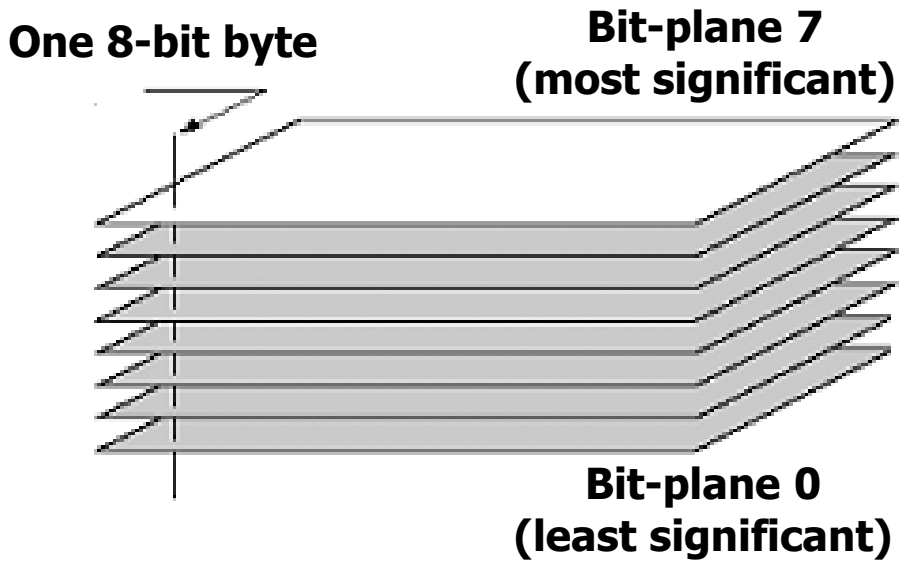
Piecewise-Linear Transformation: Gray-level slicing

- Highlighting a specific range of gray levels in an image
  - Display a high value of all gray levels in the range of interest and a low value for all other gray levels
- (a) transformation highlights range [A,B] of gray level and reduces all others to a constant level
- (b) transformation highlights range [A,B] but preserves all other levels

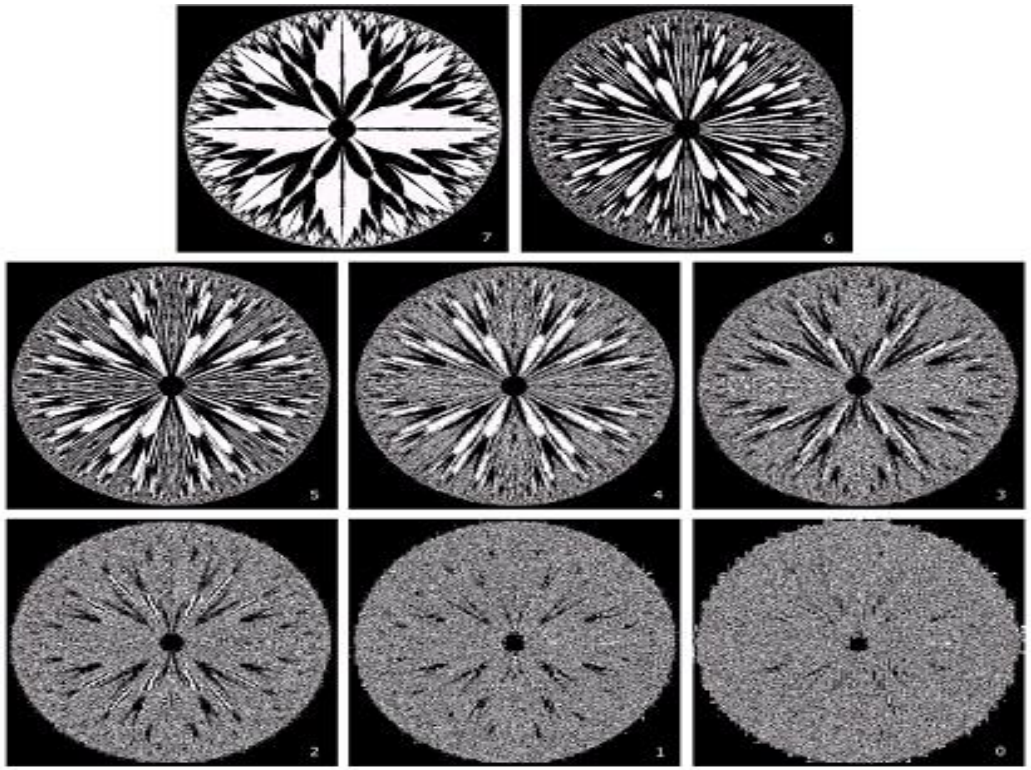


Piecewise-Linear Transformation: Bit Plane slicing

- Highlighting the contribution made to total image appearance by specific bits
- Suppose each pixel is represented by 8 bits
- Higher-order bits contain the majority of the visually significant data
- Useful for analyzing the relative importance played by each bit of the image



8- bit Planes



Bit-plane 7	Bit-plane 6	
Bit-plane 5	Bit-plane 4	Bit-plane 3
Bit-plane 2	Bit-plane 1	Bit-plane 0

Arithematic/Logical Operations on Images

- **Addition**
  - *Averaging images for noise removal*
  - Add edge image into blurred image to get the sharper image
- **Subtraction**
  - *Removal of background from images*
  - *Image matching*
  - *Moving/displaced object tracking*
- **Multiplication/Division**
  - *Scaling*
  - *Shading*
  - *Convolution*
- **AND / OR operations**
  - *To remove the unnecessary area of an image through mask operations*

Image Addition

- In its most straightforward implementation, this operator takes as input two identically sized images and produces as output a third image of the same size as the first two, in which each pixel value is the sum of the values of the corresponding pixel from each of the two input images. More sophisticated versions allow more than two images to be combined with a single operation.

- A common variant of the operator simply allows a specified constant to be added to every pixel.

How It Works

- The addition of two images is performed straightforwardly in a single pass. The output pixel values are given by:

$$Q(i,j) = P_1(i,j) + P_2(i,j)$$

- Or if it is simply desired to add a constant value *C* to a single image then:

$$Q(i,j) = P_1(i,j) + C$$

- If the pixel values in the input images are actually vectors rather than scalar values (*e.g.* for color images) then the individual components (*e.g.* red, blue and green components) are simply added separately to produce the output value.

Image Averaging for Noise Reduction

A noisy image can be represented by

$$g(x, y) = f(x, y) + \eta(x, y),$$

where $\eta(x, y)$ denotes the noise in the image

Since the noise is random and the content $f(x, y)$ is fixed,

The noise can be removed by taking more noisy images of the same object and averaging them out

$$\bar{g}(x, y) = \frac{1}{K} \sum_{i=1}^{K} g_i(x, y),$$

Image Averaging for Noise Reduction

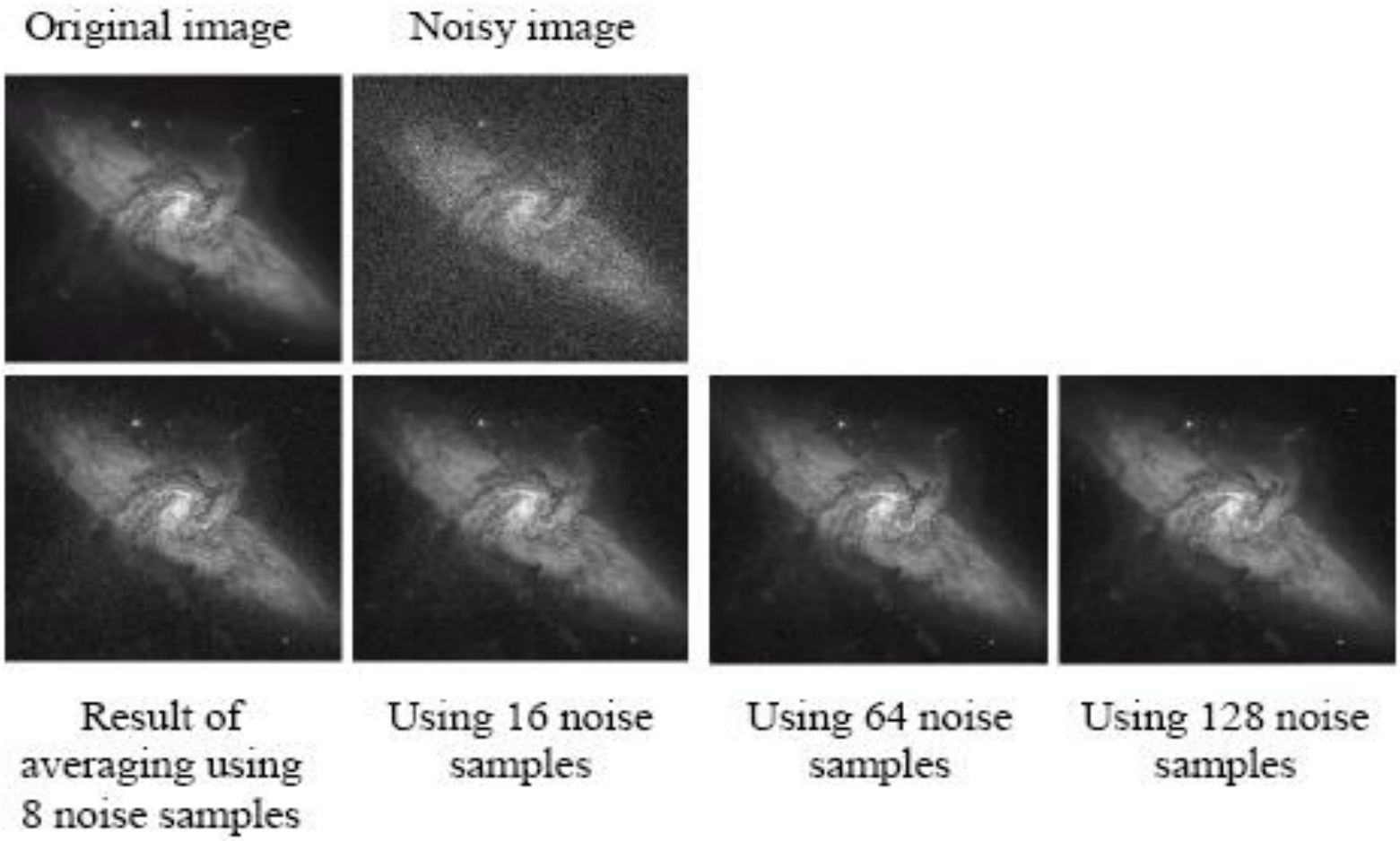
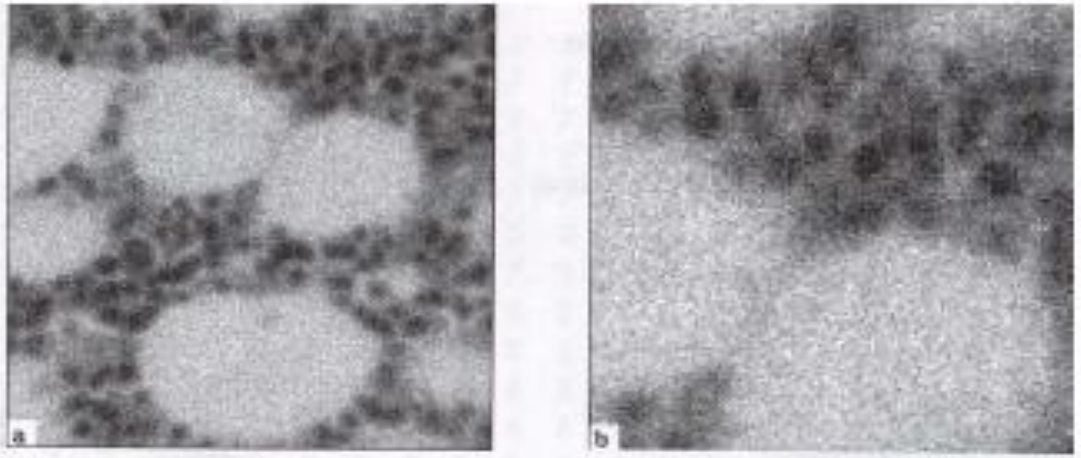


Image Averaging for Noise Reduction

Noisy image



Noise reduction by averaging 256 samples

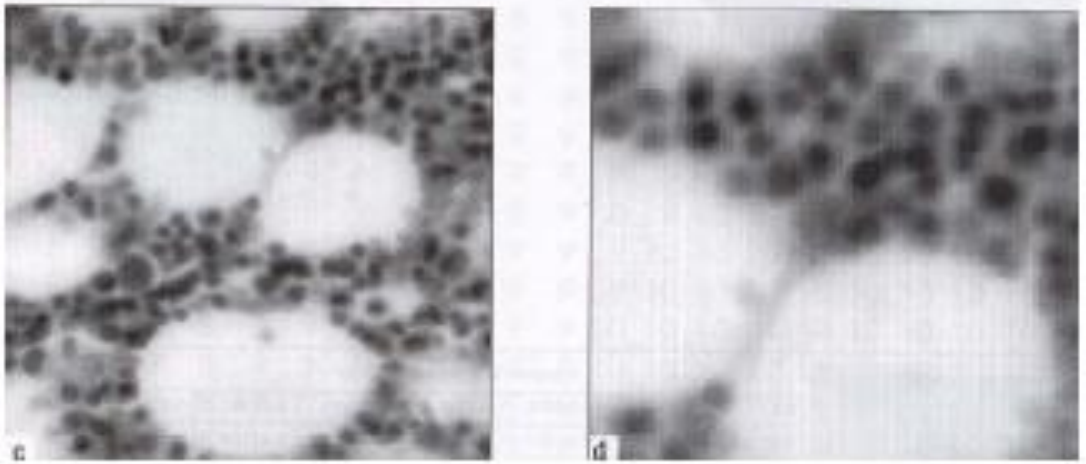


Image Subtraction

- Takes two images as input and produces a third image whose pixel values are those of the first image minus the corresponding pixel values from the second image

- Variants

  - It is also often possible to just use a single image as input and subtract a constant value from all the pixels

  - Just output the absolute difference between pixel values, rather than the straightforward signed output.

Image Subtraction

- The subtraction of two images is performed in a single pass

$$Q(i,j) = P_1(i,j) - P_2(i,j)$$

- If the operator computes absolute differences between the two input images then:

$$Q = |P_1(i,j) - P_2(i,j)|$$

- If it is simply desired to subtract a constant value C from a single image then:

$$Q = P_1(i,j) - C$$

Image Subtraction

- If the operator calculates absolute differences, then it is impossible for the output pixel values to be outside the range
- In rest of the two cases the pixel value may become negative
- This is one good reason for using absolute differences.
- How to solve problem of negative pixels?

Image Subtraction

How to solve problem of negative pixels?

- 1st Method
  - Let we have an 8 bit Gray scale image (Value Range= 0 t0 255)
  - The result of image subtraction may come in the range of -255 to +255
  - One scheme can be to add 255 to every pixel and then divide by 2
  - Method is easy and fast
  - Limitations
    - **Truncation errors can cause loss of accuracy**
    - **Full range of display may not be utilized**

Image Subtraction

How to solve problem of negative pixels?

- 2nd Method
  - first, find the minimum gray value of the subtracted image
  - second, find the maximum gray value of the subtracted image
  - set the minimum value to be zero and the maximum to be 255
  - while the rest are adjusted according to the interval [0, 255], by timing each value with 255/max

Example:: Image Subtraction

Original image

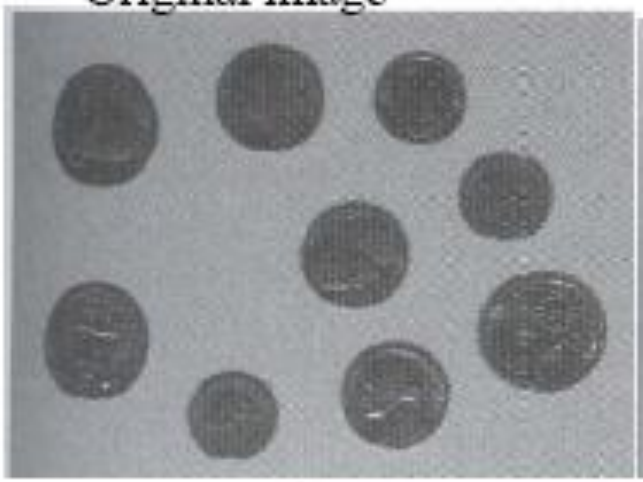
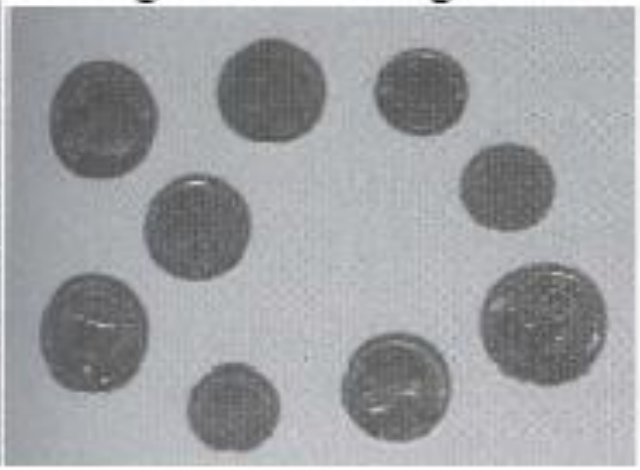


Image after moving one coin



Difference image after pixel by pixel subtraction of second image from first image

Example: Background Removal Using Image Subtraction



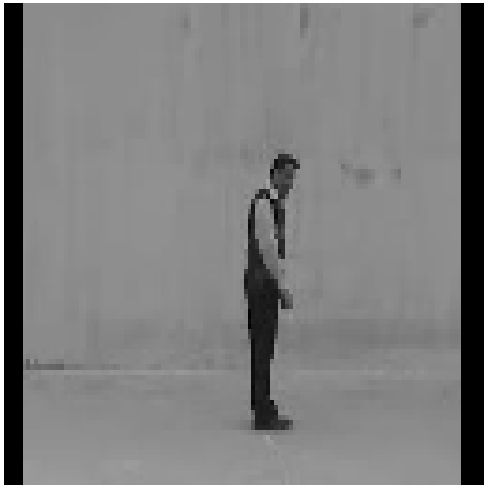
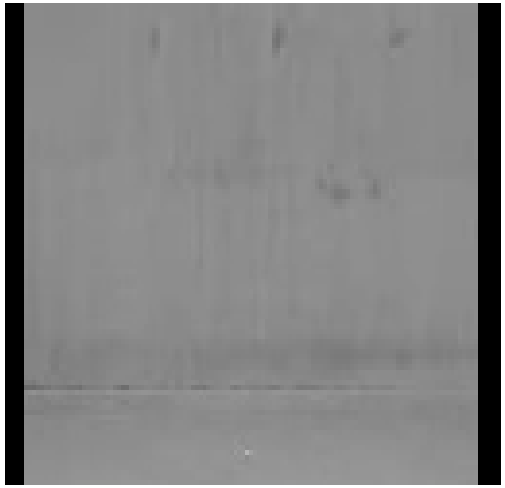
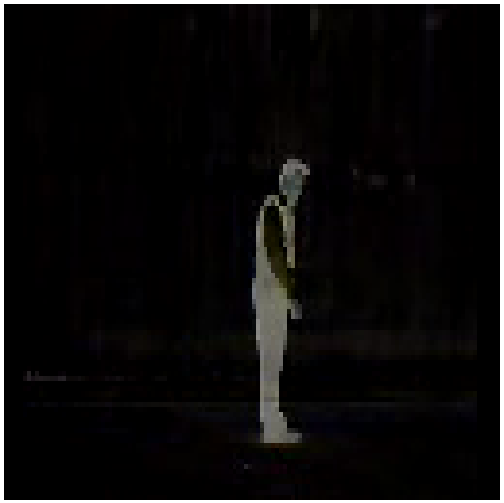





Image Multiplication

- Like other image arithmetic operators, multiplication comes in two main forms.
  - The first form takes two input images and produces an output image in which the pixel values are just those of the first image, multiplied by the values of the corresponding values in the second image.
  - The second form takes a single input image and produces output in which each pixel value is multiplied by a specified constant.
- **How It Works**
  - The multiplication of two images is performed in the obvious way in a single pass using the formula:

$$Q(i,j) = P_1(i,j) \times P_2(i,j)$$

  - Scaling by a constant is performed using:

$$Q(i,j) = P_1(i,j) \times C$$

Image Multiplication

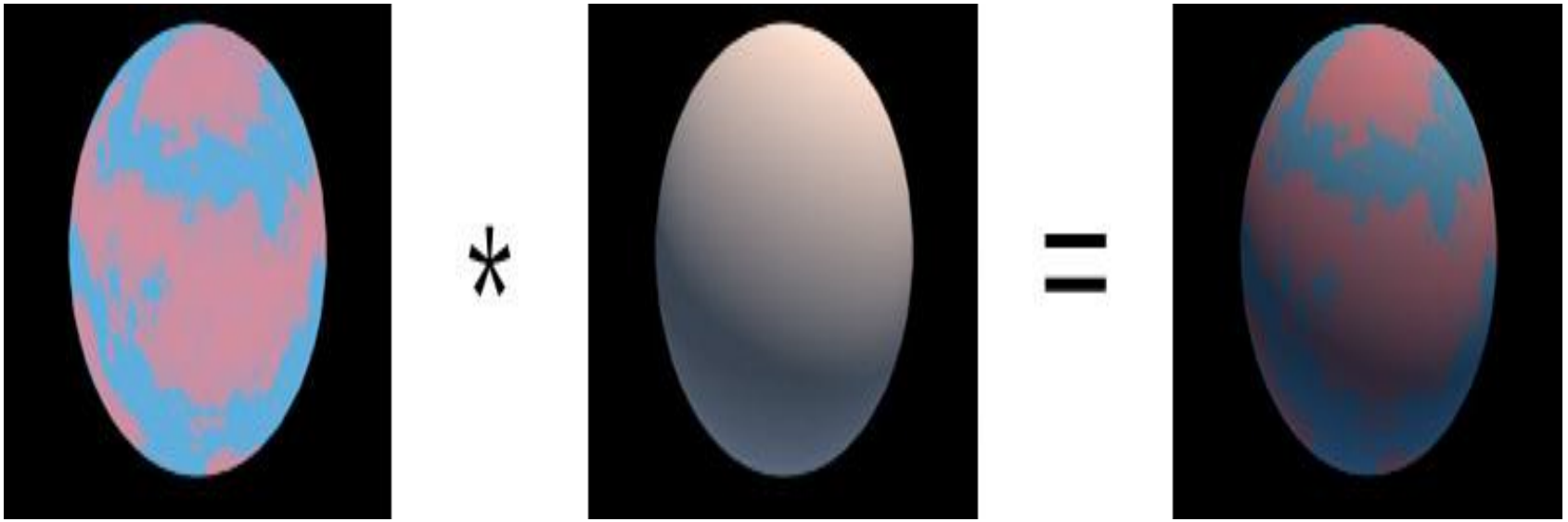
- **Guidelines for Use**

  - There are many specialist uses for scaling. In general though, given a scaling factor greater than one, scaling will brighten an image. Given a factor less than one, it will darken the image.

  - For instance, shows a picture of model robot that was taken under low lighting conditions. Simply scaling every pixel by a factor of 3, we obtain the one shown in the middle which is much clearer. However, when using pixel multiplication, we should make sure that the calculated pixel values don't exceed the maximum possible value. If we, for example, scale the above image by a factor of 5 using a 8-bit representation, we obtain the one shown in last. All the pixels which, in the original image, have a value greater than 51 exceed the maximum value and are (in this implementation) wrapped around from *255* back to *0*.

Example:: Image Multiplication

- **Multiplication** also provides a good way of "shading" artwork. You can use it to introduce a sense of diffuse lighting into your painting.



Example:: Image Multiplication

- **Multiplication** provides a good way to color line drawings. Here you can really see the "black times anything is black, white times anything is that thing unchanged" rule in action.

Logic Operations

AND / OR / NOT Operations

To remove the unnecessary area of an image through mask operations (AND/OR)

To invert the image same as image negative (NOT)

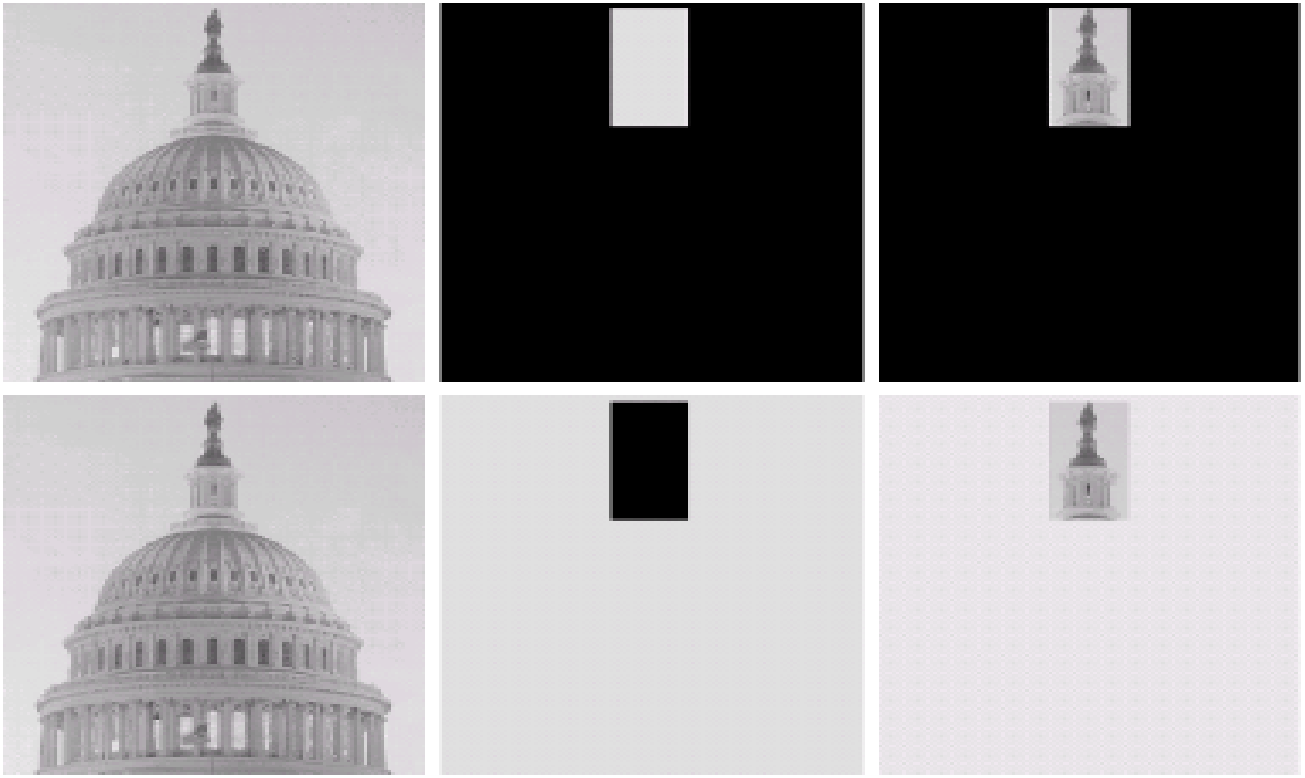
Logical Operations

- Logical operators operate on a pixel by pixel basis
- When Logical operation performs on gray-level images, the pixel values are processed as string of binary numbers
  - E.g performing the NOT operation on a black, 8 bit pixel (a string of 8 0's) produces a white pixel (a string of 8 1's)
  - Intermediate values are processed the same way changing all 1's to 0's and vice versa
- NOT operation = negative transformation

Logical Operations

- The AND and OR operations are used for masking; that is for selecting sub-images in an image as mentioned in the figure 3.27

- Light represents a binary 1, and dark represents a binary 0



a b c
d e f

FIGURE 3.27 (a) Original image. (b) AND image mask. (c) Result of the AND operation on images (a) and (b). (d) Original image. (e) OR image mask. (f) Result of operation OR on images (d) and (e).

Any question