Digital Image Processing

Image Enhancement

(Point Processing)

Contents

In this lecture we will look at image enhancement point processing techniques:

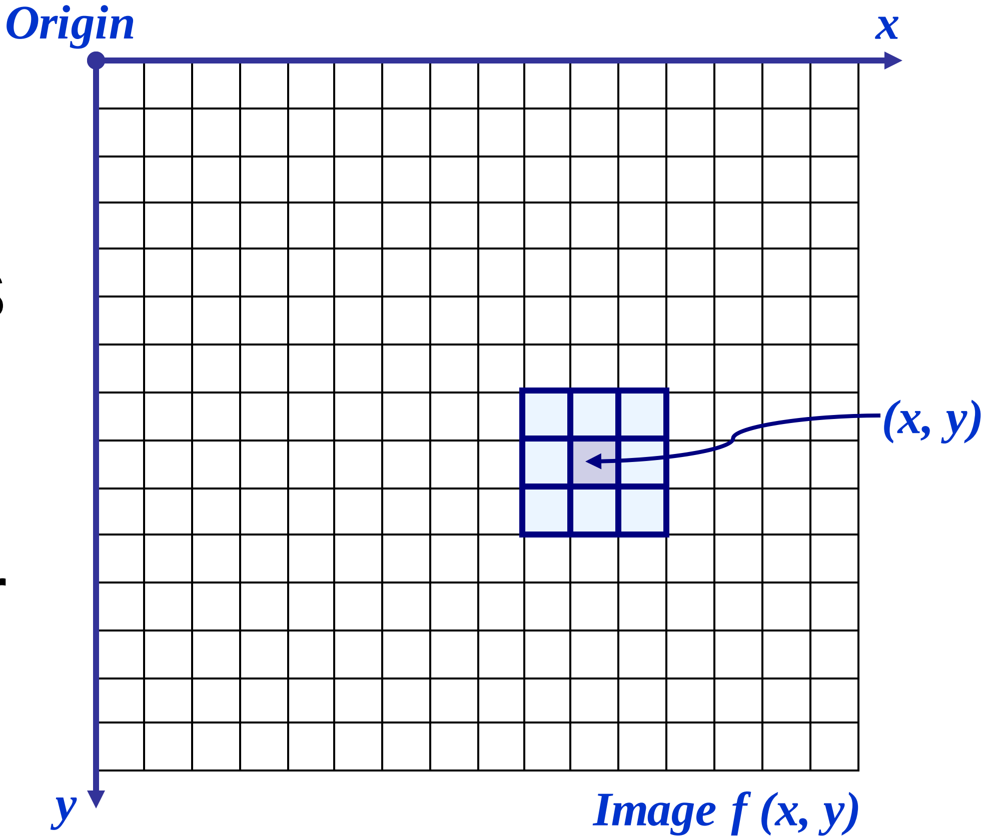
- What is point processing?
- Negative images
- Thresholding
- Logarithmic transformation
- Power law transforms
- Grey level slicing
- Bit plane slicing

Basic Spatial Domain Image Enhancement

Most spatial domain enhancement operations can be reduced to the form

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

where $f(x, y)$ is the input image, $g(x, y)$ is the processed image and T is some operator defined over some neighbourhood of (x, y)



Point Processing

The simplest spatial domain operations occur when the neighbourhood is simply the pixel itself

In this case T is referred to as a *grey level transformation function* or a *point processing operation*

Point processing operations take the form

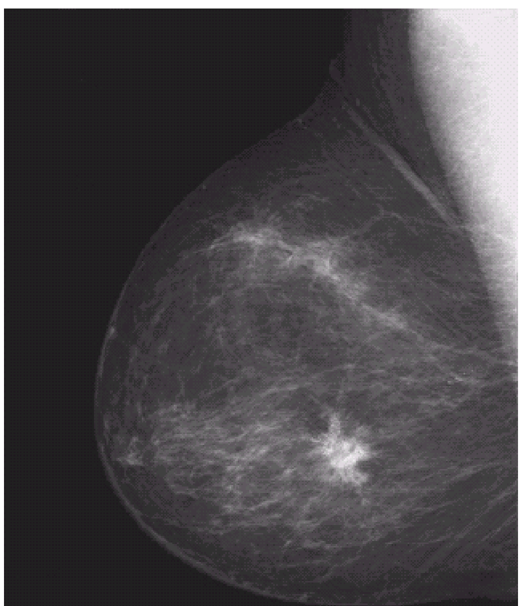
$$s = T (r)$$

where s refers to the processed image pixel value and r refers to the original image pixel value

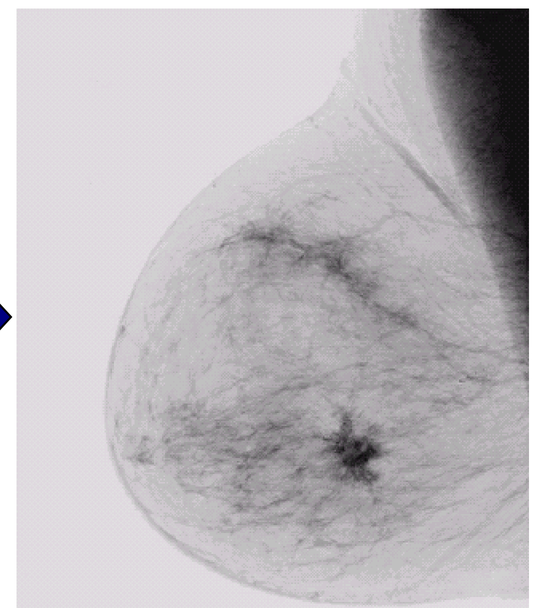
Point Processing Example: Negative Images

- Negative images are useful for enhancing white or grey detail embedded in dark regions of an image
  - Note how much clearer the tissue is in the negative image of the mammogram below

Original Image

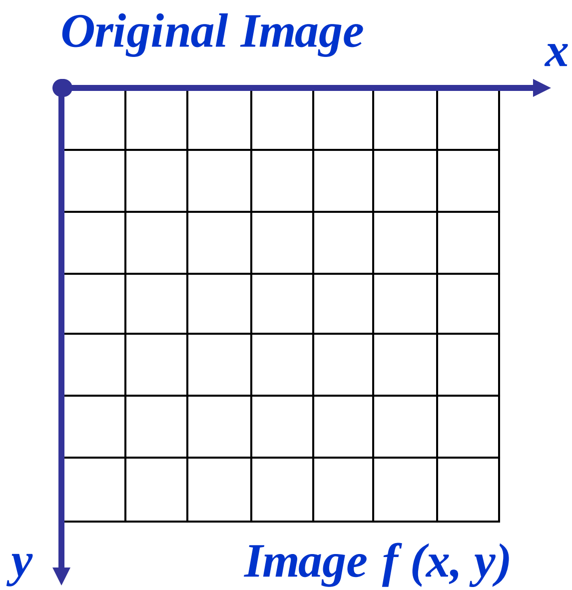


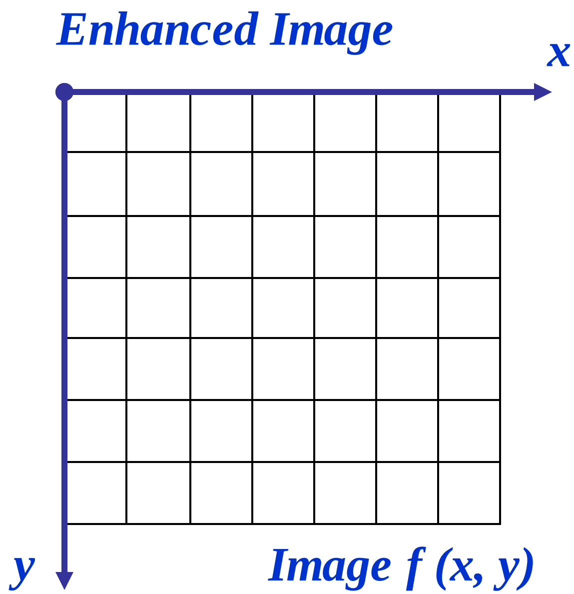
$s = 1.0 - r$



Negative Image

Point Processing Example: Negative Images (cont…)



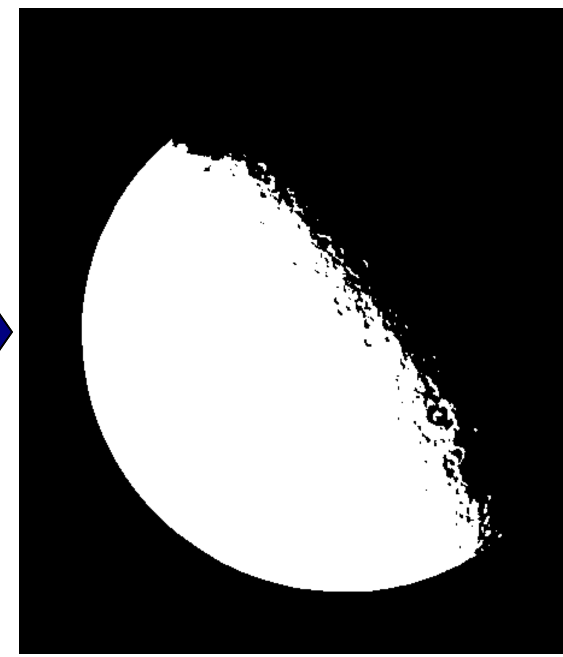


$$s = intensity_{max} - r$$

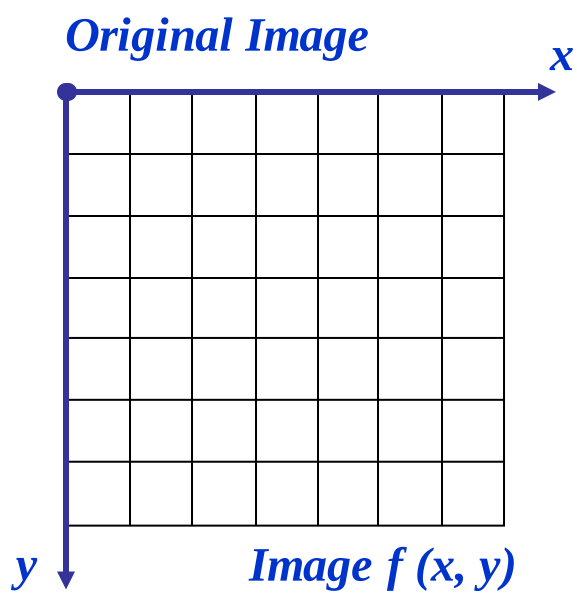
Point Processing Example: Thresholding

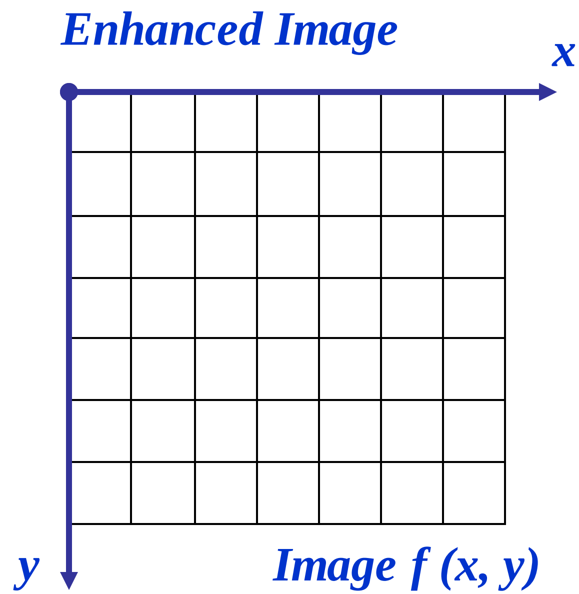
Thresholding transformations are particularly useful for segmentation in which we want to isolate an object of interest from a background

$$s = \begin{cases} 1.0 & r > \text{threshold} \\ 0.0 & r <= \text{threshold} \end{cases}$$



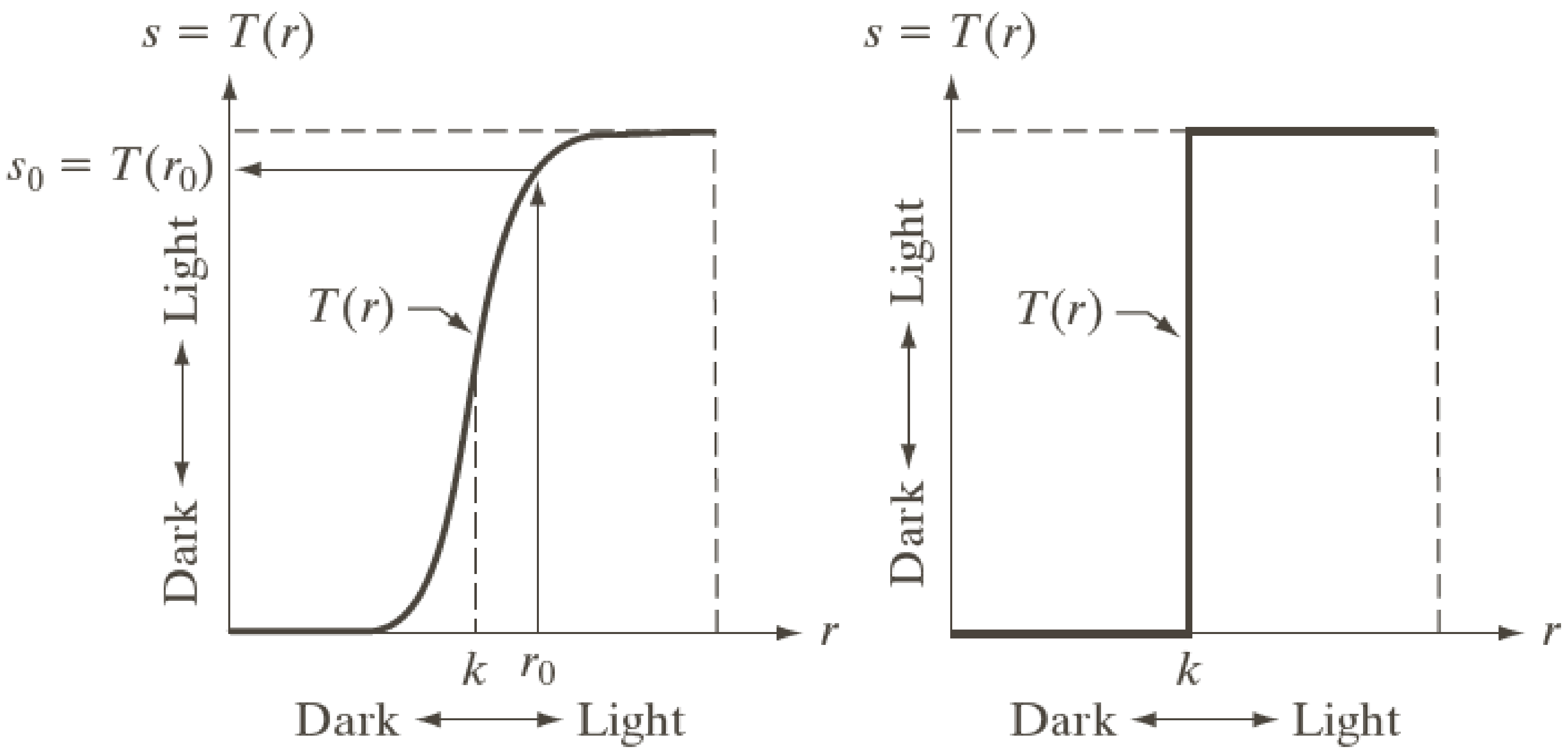
Point Processing Example: Thresholding (cont…)





$$s = \begin{cases} 1.0 & r > threshold \\ 0.0 & r <= threshold \end{cases}$$

Intensity Transformations

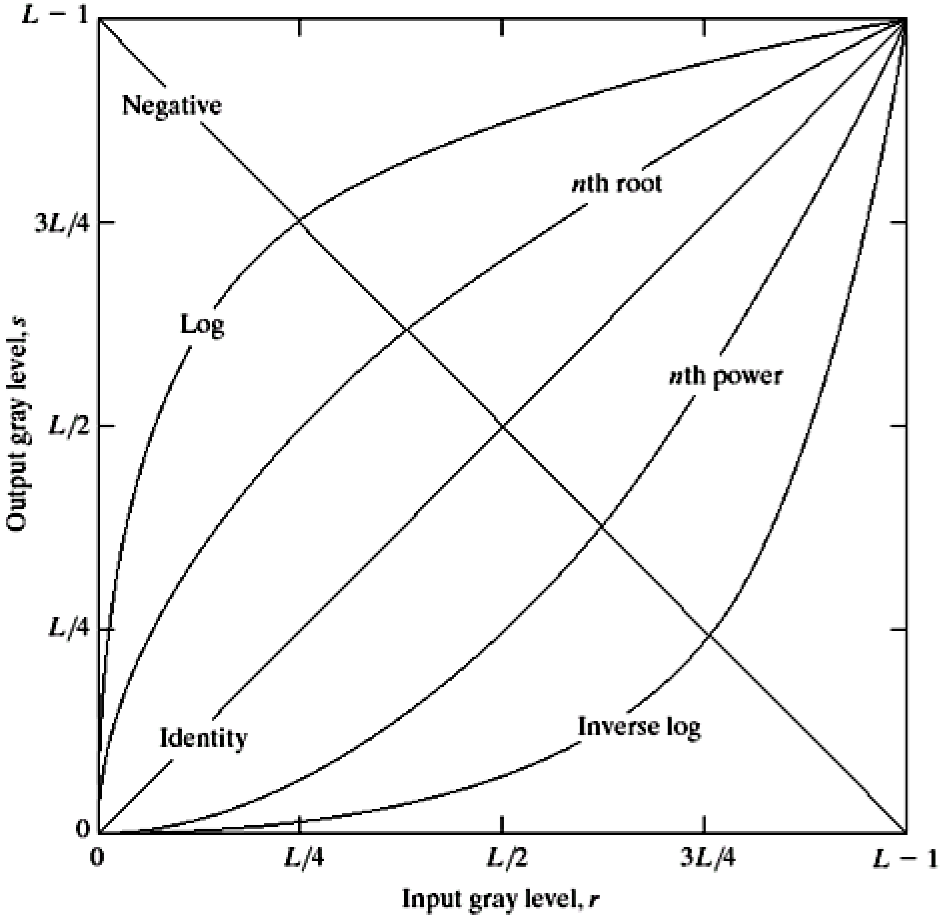


Basic Grey Level Transformations

There are many different kinds of grey level transformations

Three of the most common are shown here

- Linear
  - Negative/Identity
- Logarithmic
  - Log/Inverse log
- Power law
  - n^{th} power/n^{th} root



Logarithmic Transformations

The general form of the log transformation is

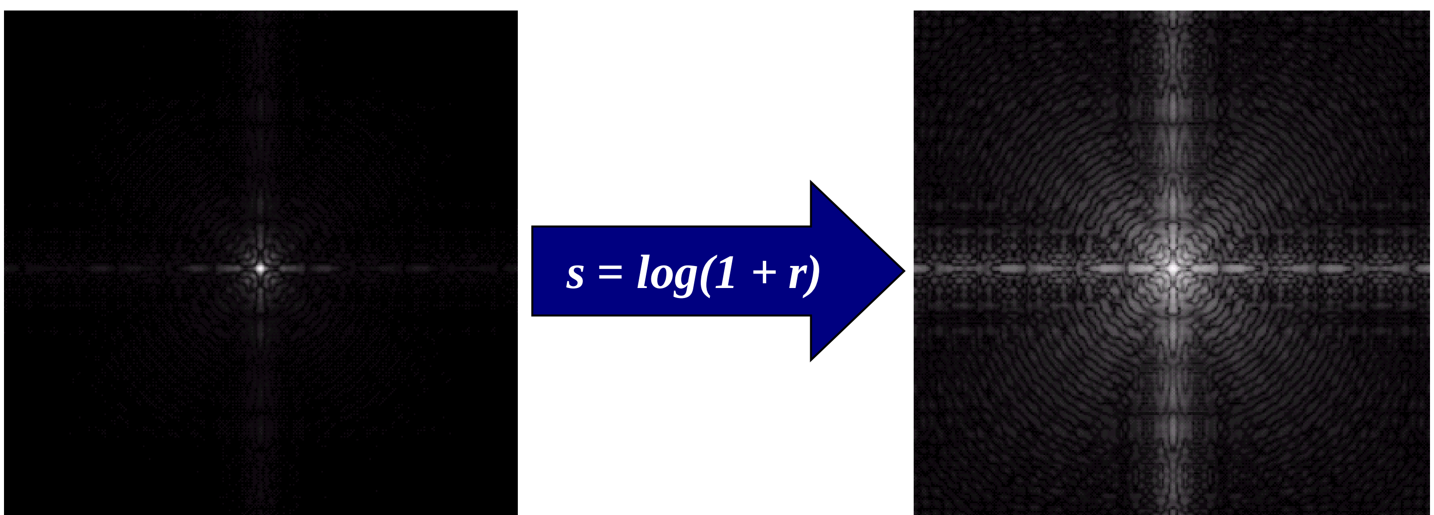
$$s = c * log(1 + r)$$

The log transformation maps a narrow range of low input grey level values into a wider range of output values

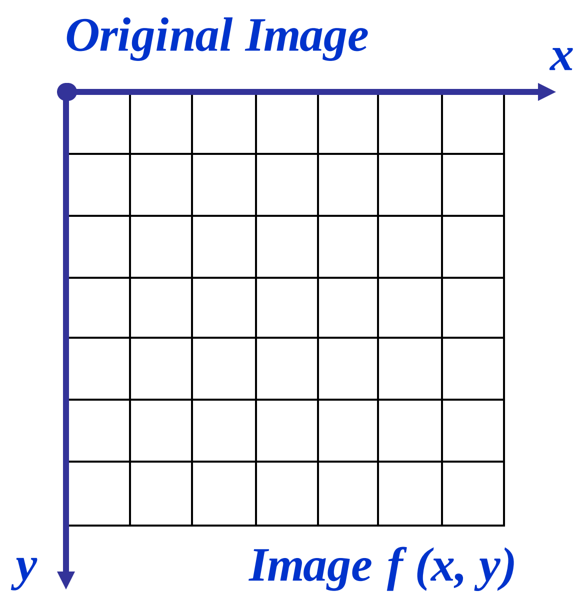
The inverse log transformation performs the opposite transformation

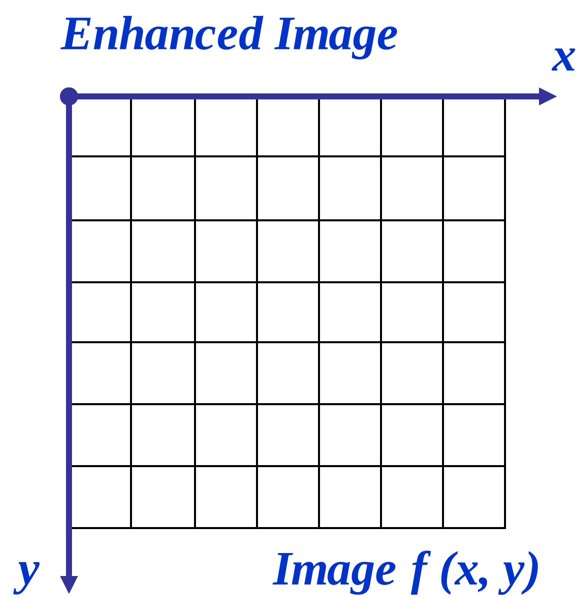
Logarithmic Transformations (cont…)

- Log functions are particularly useful when the input grey level values may have an extremely large range of values
- In the following example the Fourier transform of an image is put through a log transform to reveal more detail



Logarithmic Transformations (cont…)





$$s = log(1 + r)$$

We usually set c to 1

Grey levels must be in the range [0.0, 1.0]

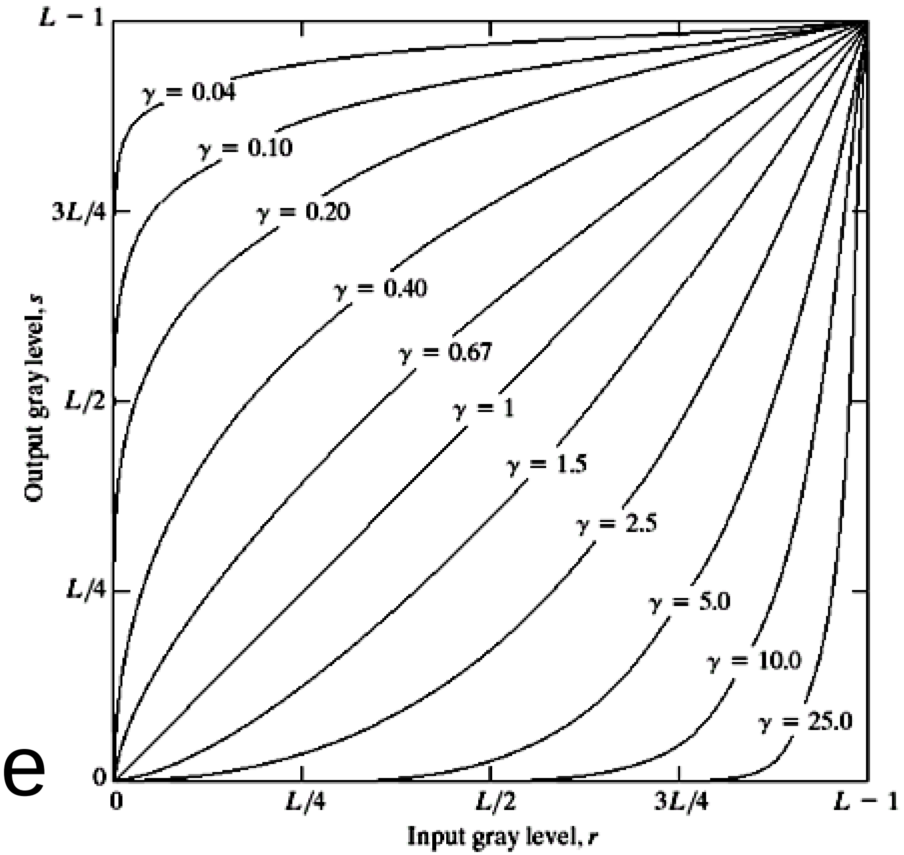
Power Law Transformations

Power law transformations have the following form

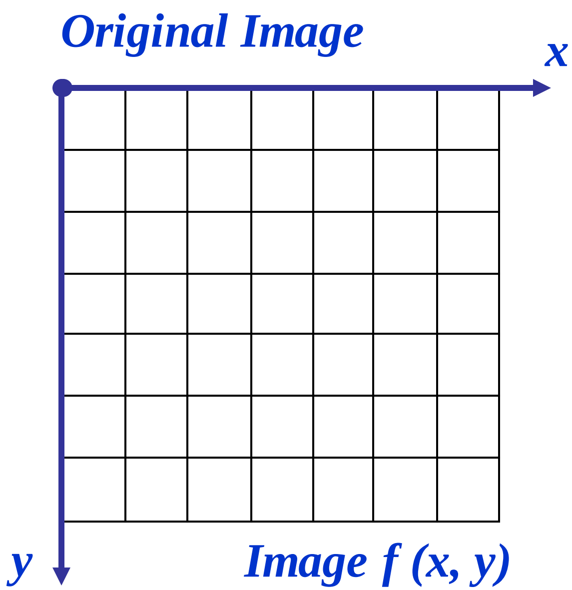
$$s = c * r^{\gamma}$$

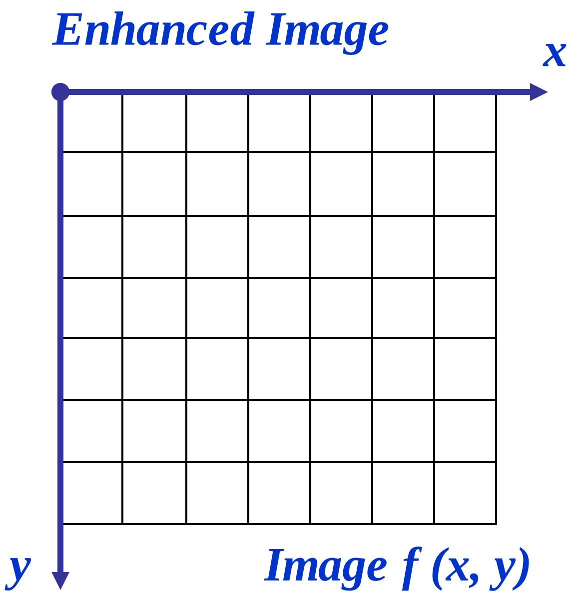
Map a narrow range of dark input values into a wider range of output values or vice versa

Varying γ gives a whole family of curves



Power Law Transformations (cont…)





$$s = r^{\gamma}$$

We usually set c to 1

Grey levels must be in the range [0.0, 1.0]

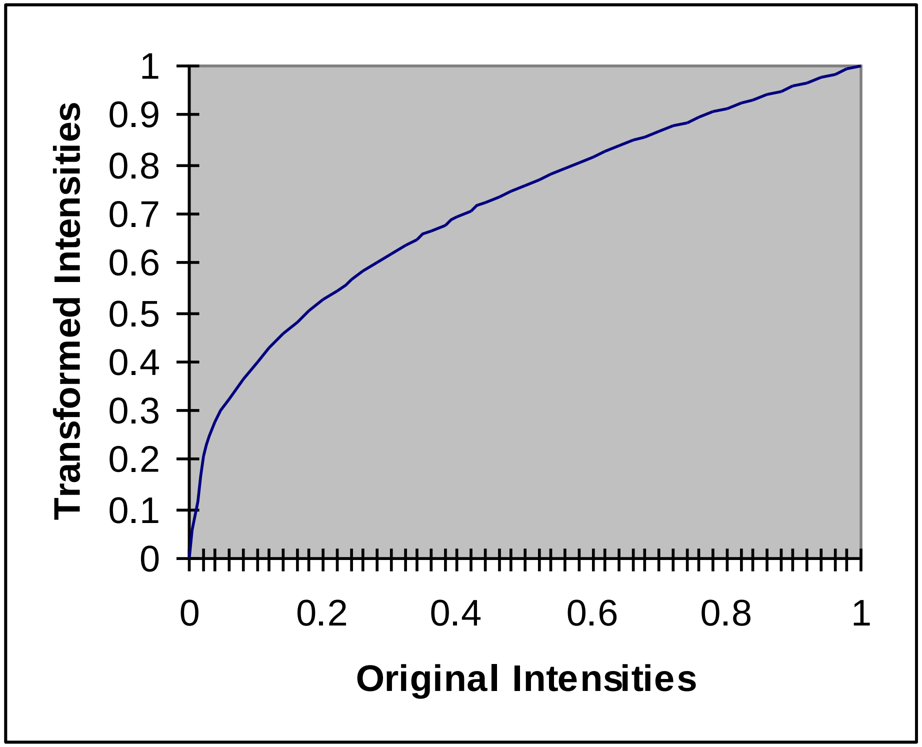
Power Law Example

Power Law Example (cont…)

$$\gamma = 0.6$$

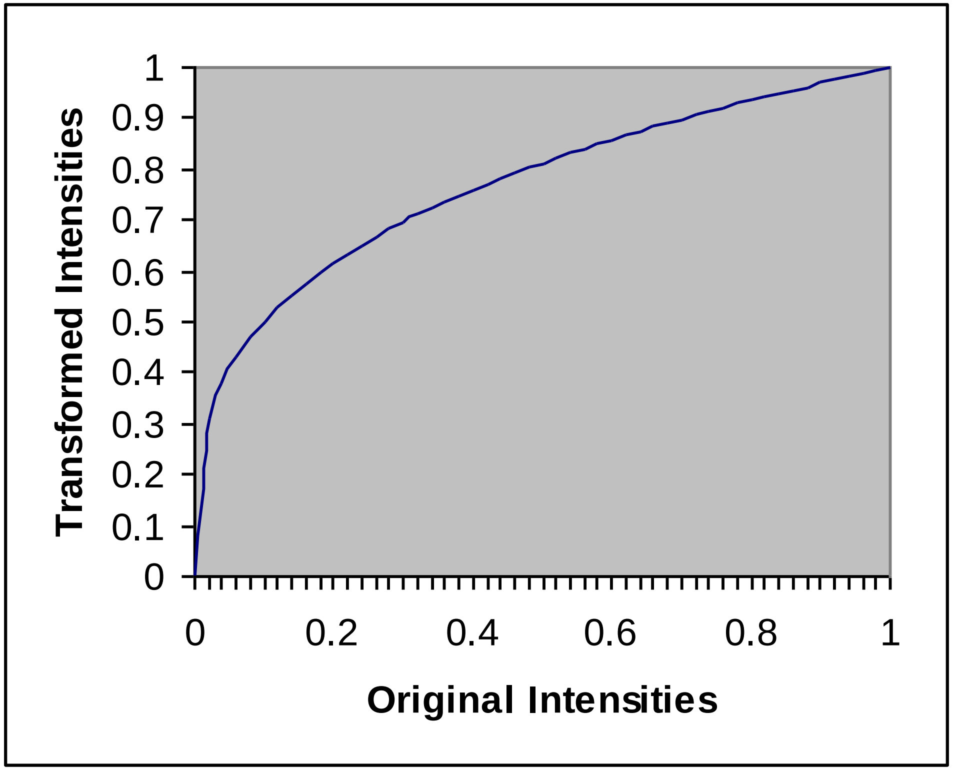
Power Law Example (cont…)

$$\gamma = 0.4$$



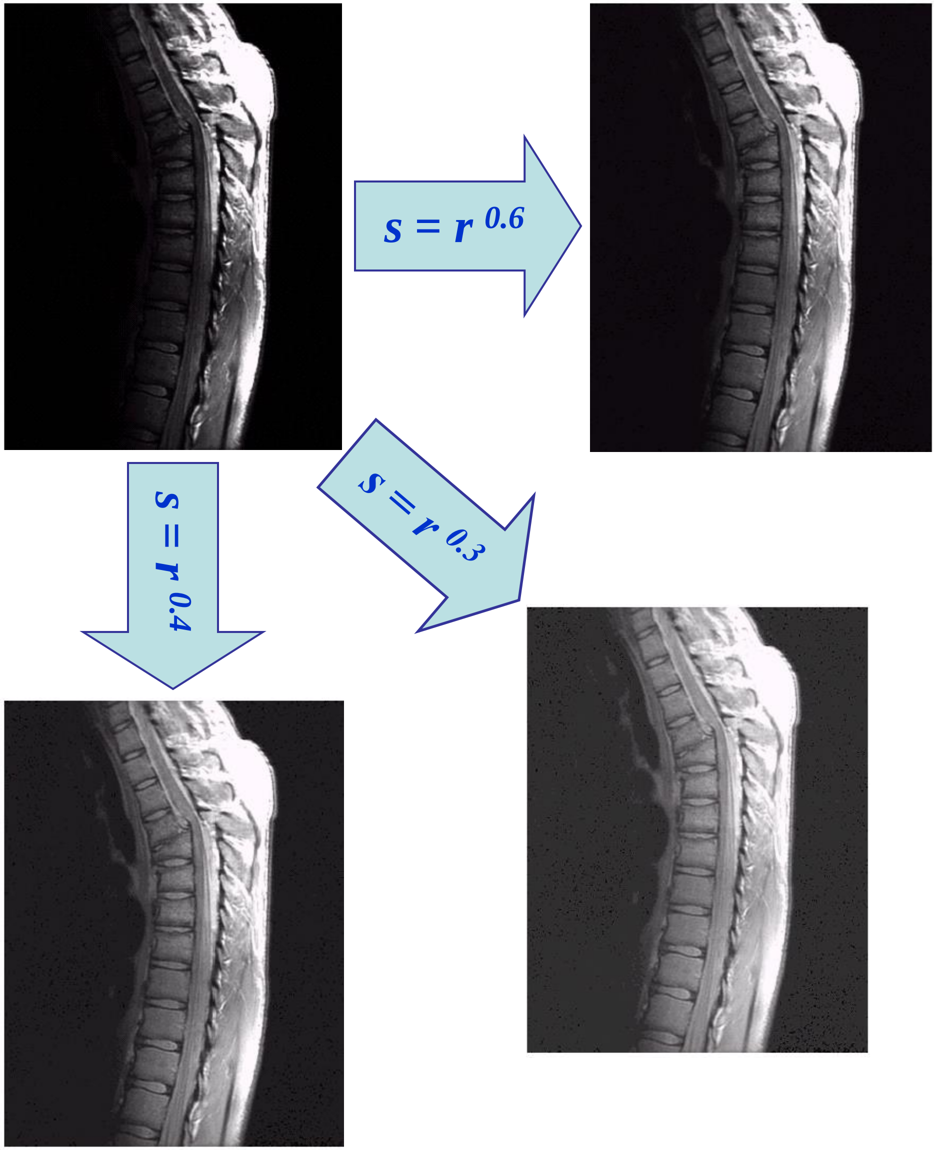
Power Law Example (cont…)

$$\gamma = 0.3$$



Power Law Example (cont…)

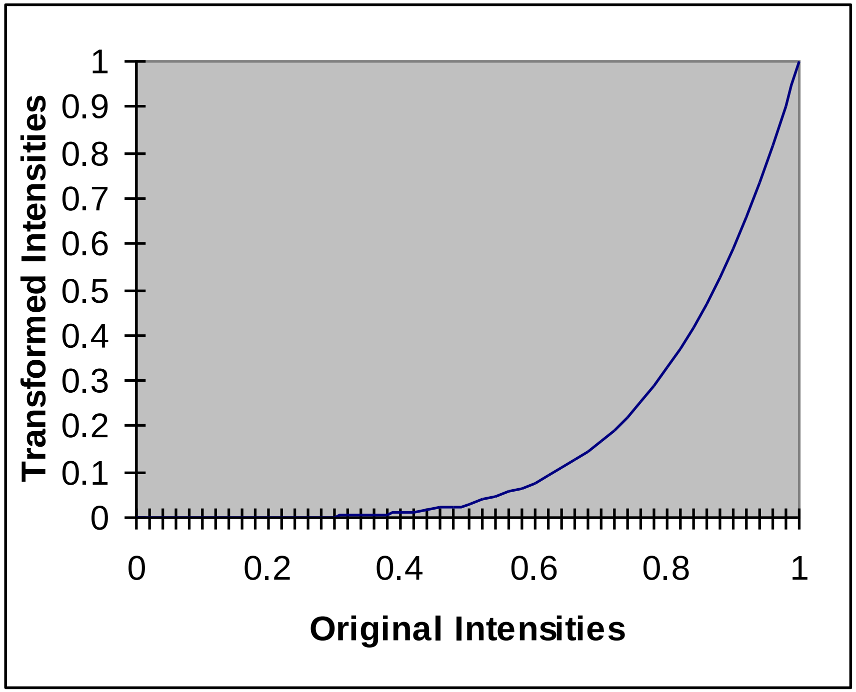
- The images to the right show a magnetic resonance (MR) image of a fractured human spine
- Different curves highlight different detail



Power Law Example

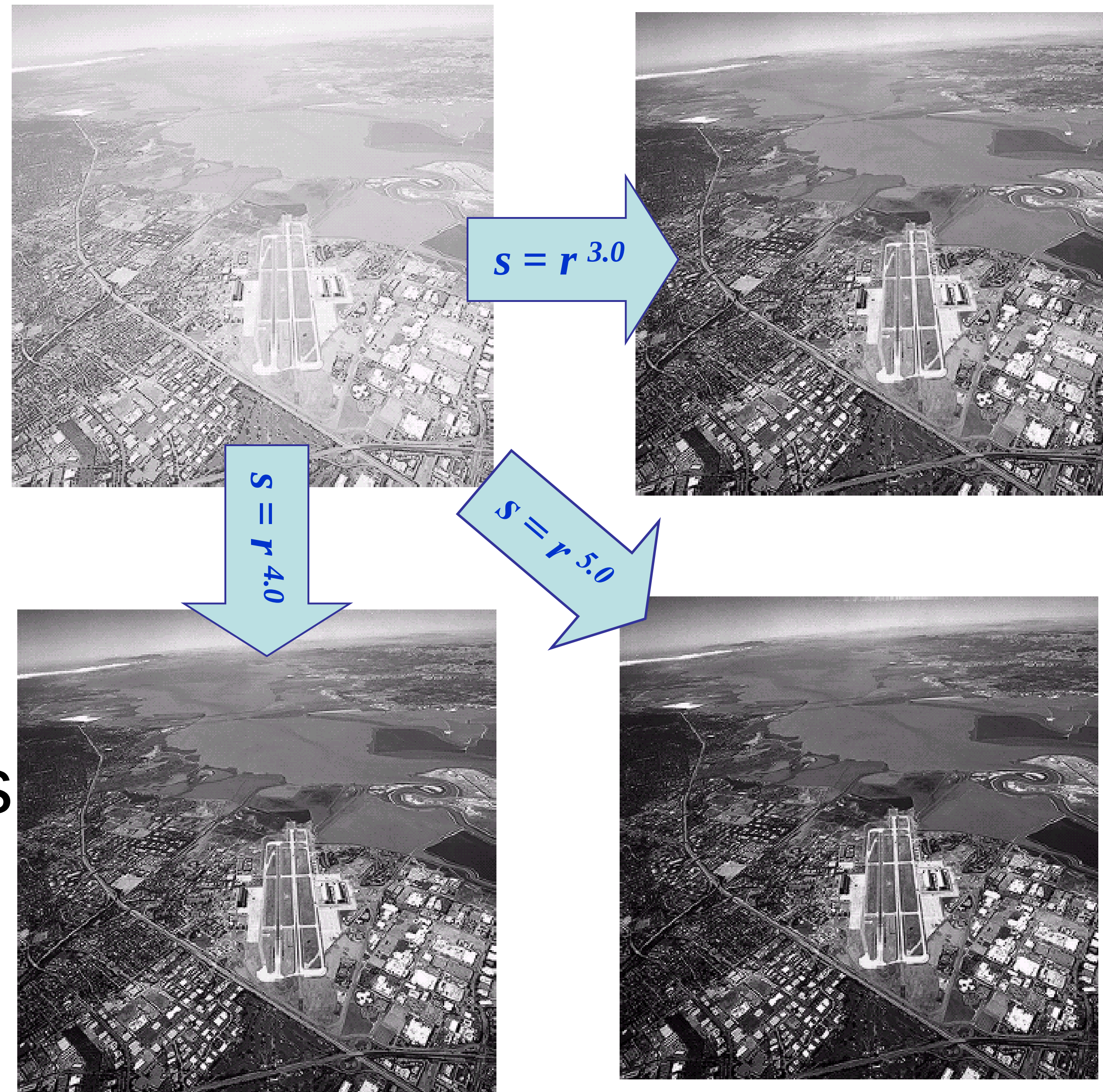
Power Law Example (cont…)

$$\gamma = 5.0$$



Power Law Transformations (cont…)

- An aerial photo of a runway is shown
- This time power law transforms are used to darken the image
- Different curves highlight different detail

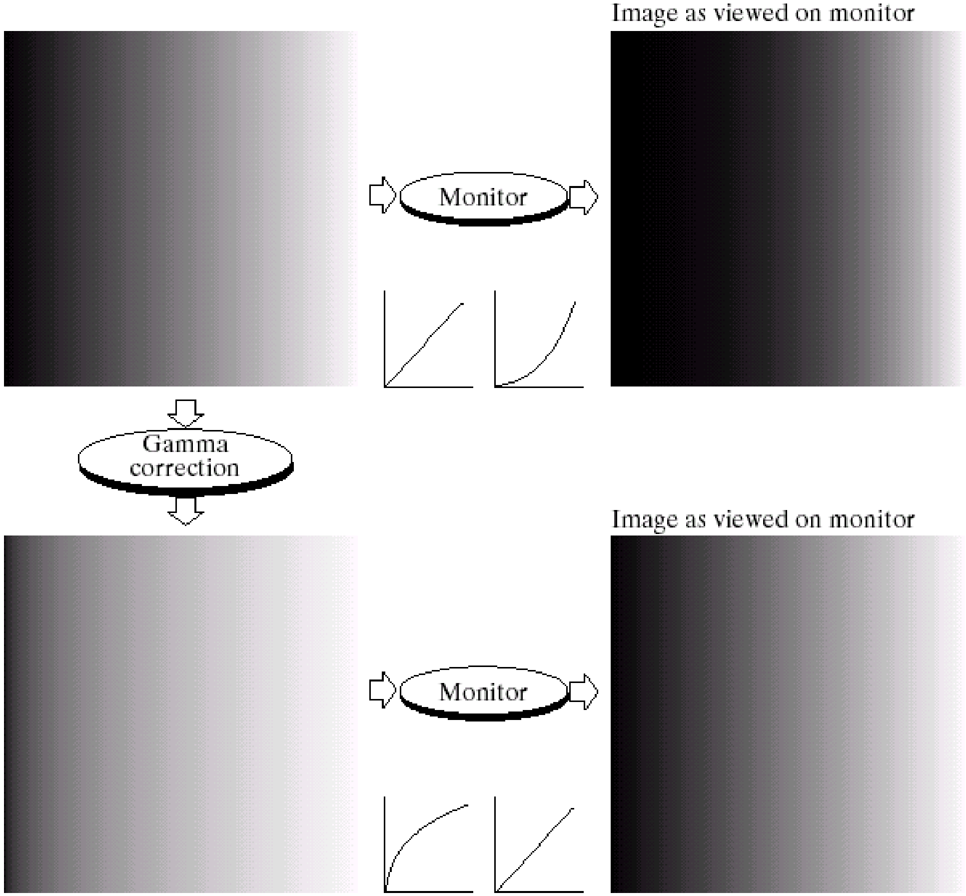


Gamma Correction

Many of you might be familiar with gamma correction of computer monitors

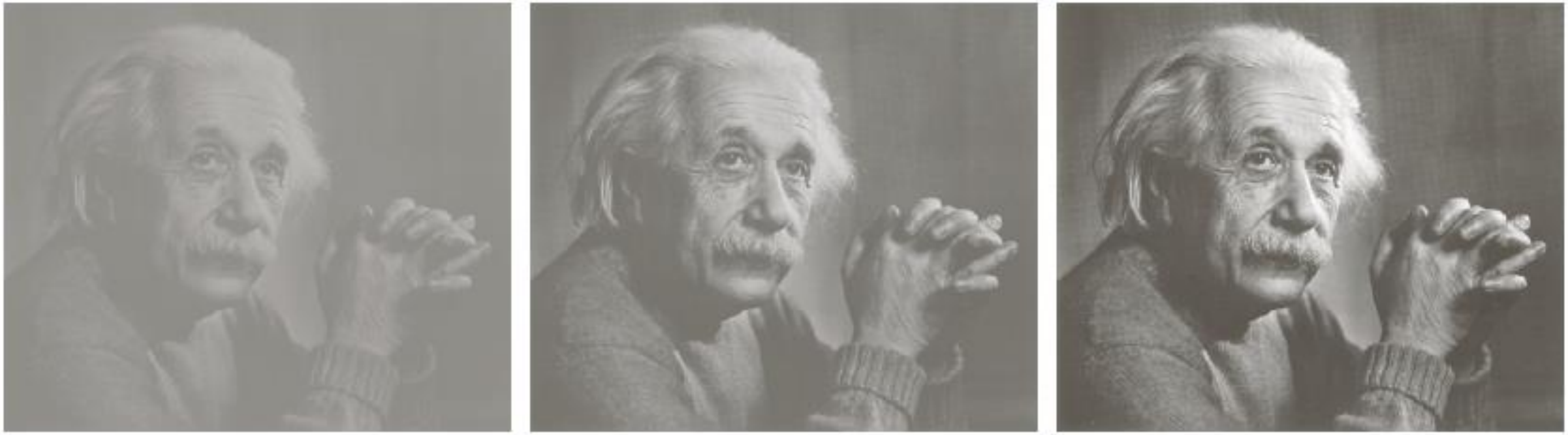
Problem is that display devices do not respond linearly to different intensities

Can be corrected using a log transform



More Contrast Issues

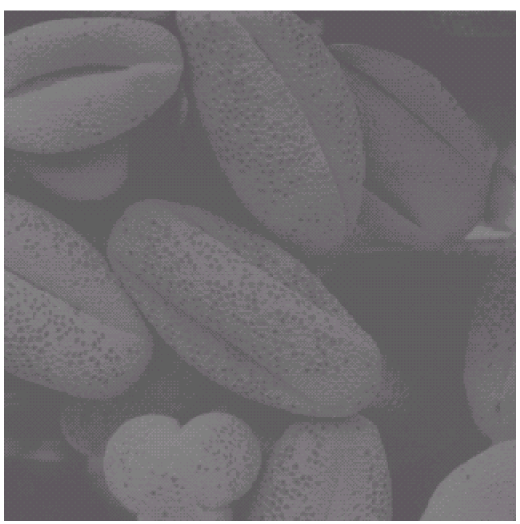
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

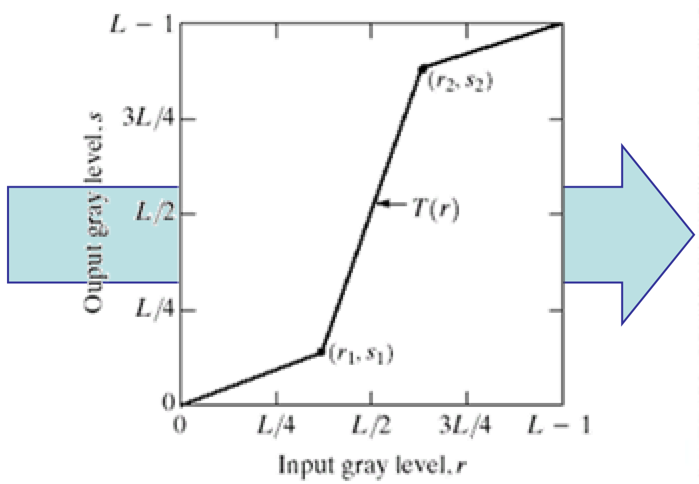


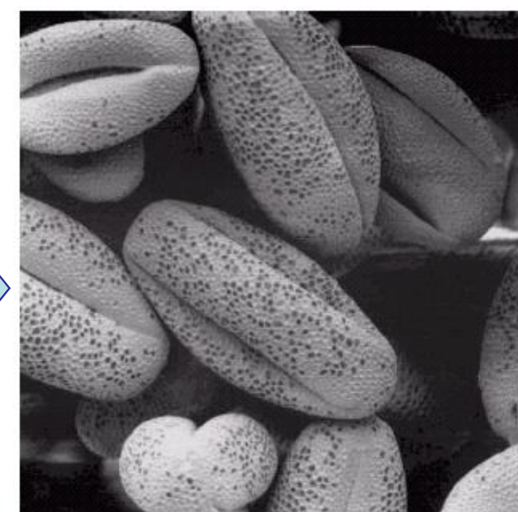
Piecewise Linear Transformation Functions

Rather than using a well defined mathematical function we can use arbitrary user-defined transforms

The images below show a contrast stretching linear transform to add contrast to a poor quality image



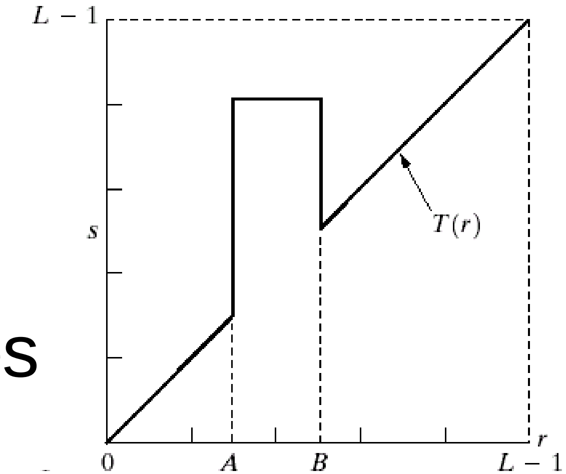


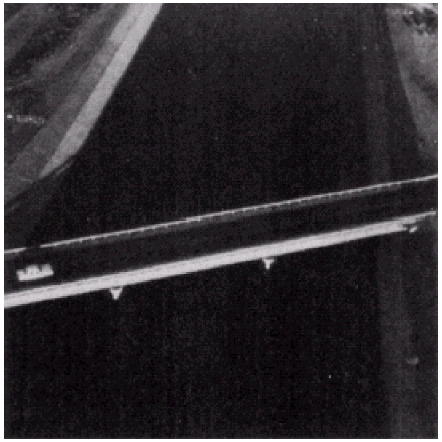


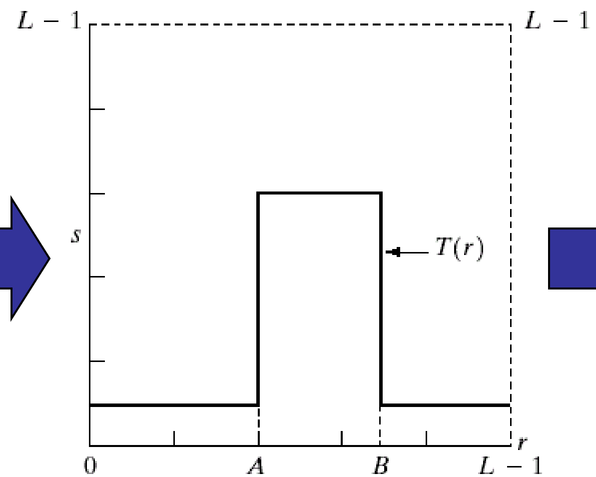
Gray Level Slicing

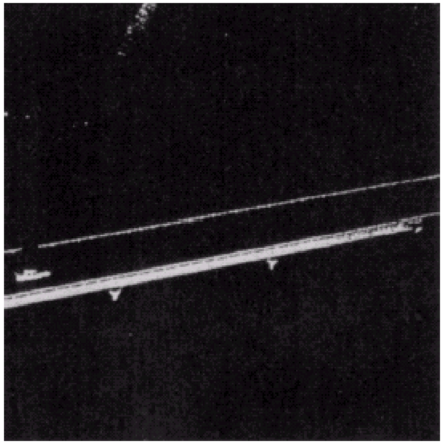
Highlights a specific range of grey levels

- Similar to thresholding
- Other levels can be suppressed or maintained
- Useful for highlighting features in an image







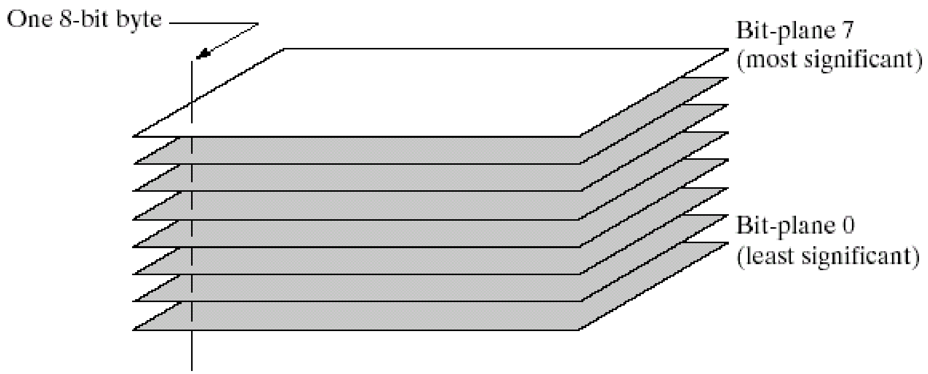


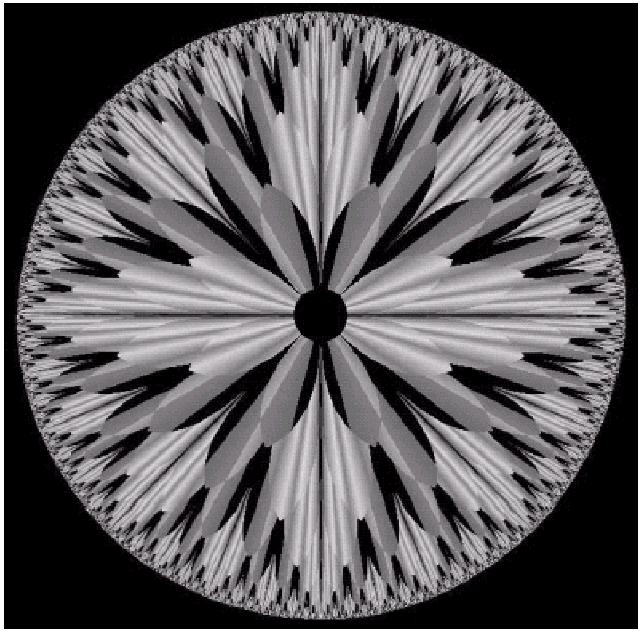
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Bit Plane Slicing

Often by isolating particular bits of the pixel values in an image we can highlight interesting aspects of that image

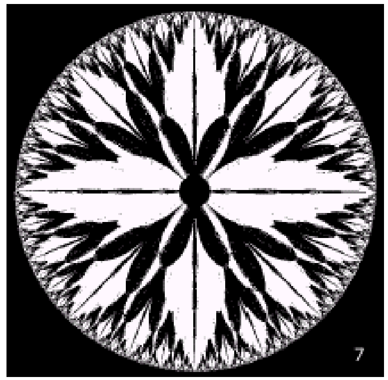
- Higher-order bits usually contain most of the significant visual information
- Lower-order bits contain subtle details

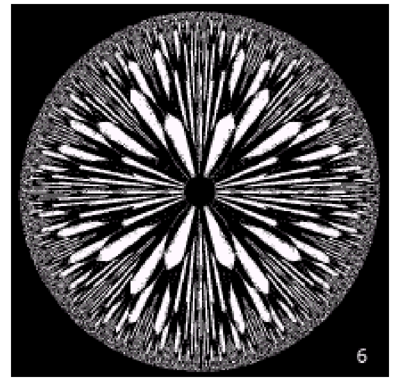




Bit Plane Slicing (cont…)

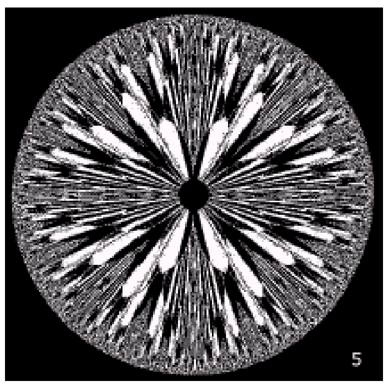
[10000000]

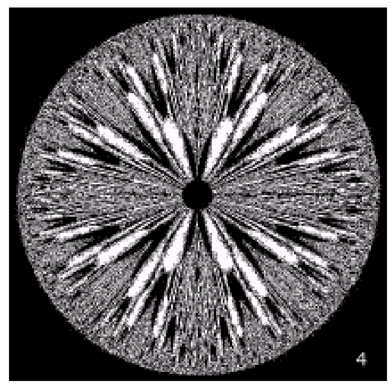


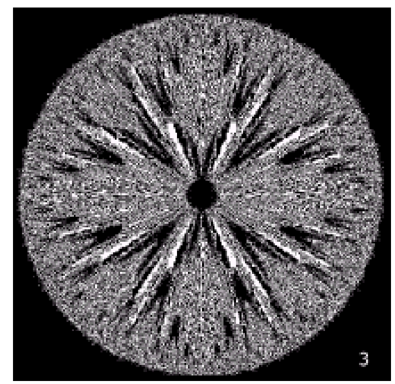


[01000000]

[00100000]

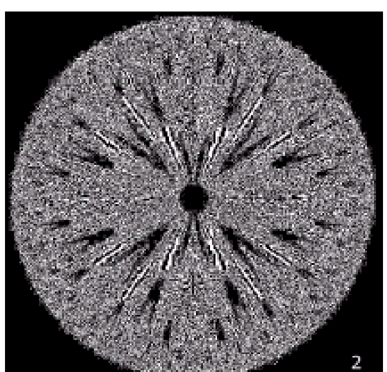


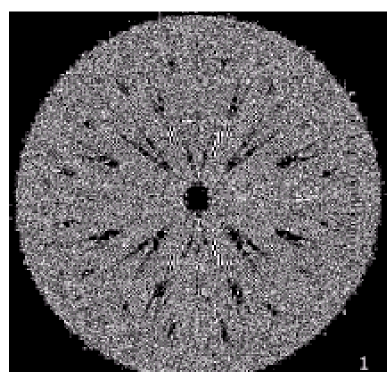


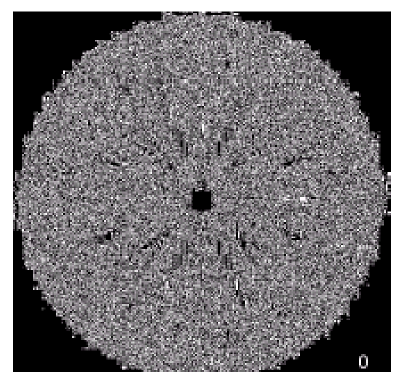


[00001000]

[00000100]



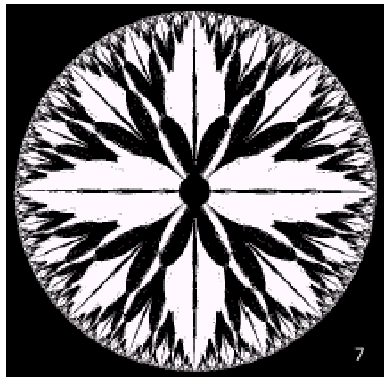


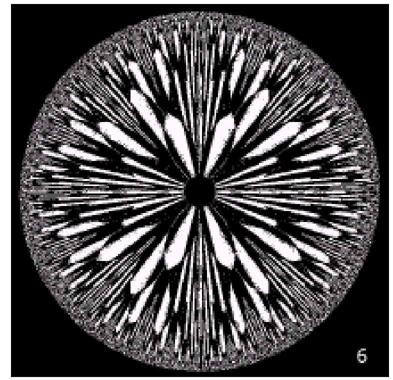


[00000001]

Bit Plane Slicing (cont…)

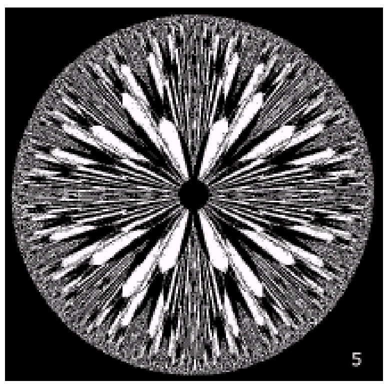
[10000000]

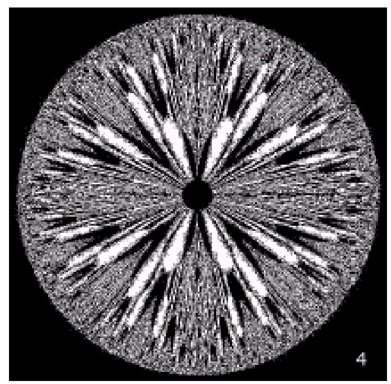


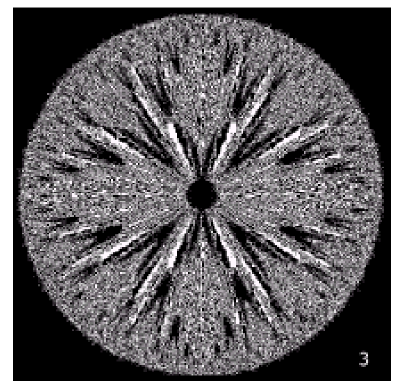


[01000000]

[00100000]

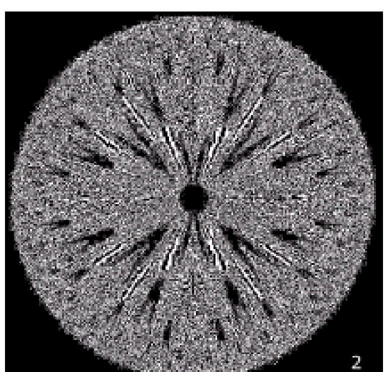


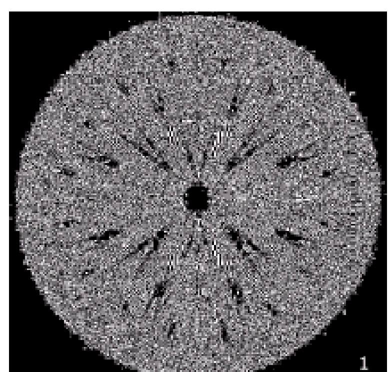


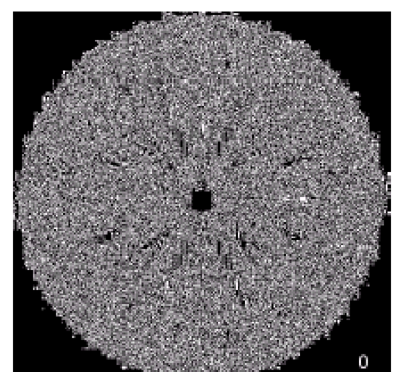


[00001000]

[00000100]







[00000001]

Bit Plane Slicing (cont…)

a	b	c
d	e	f
g	h	i

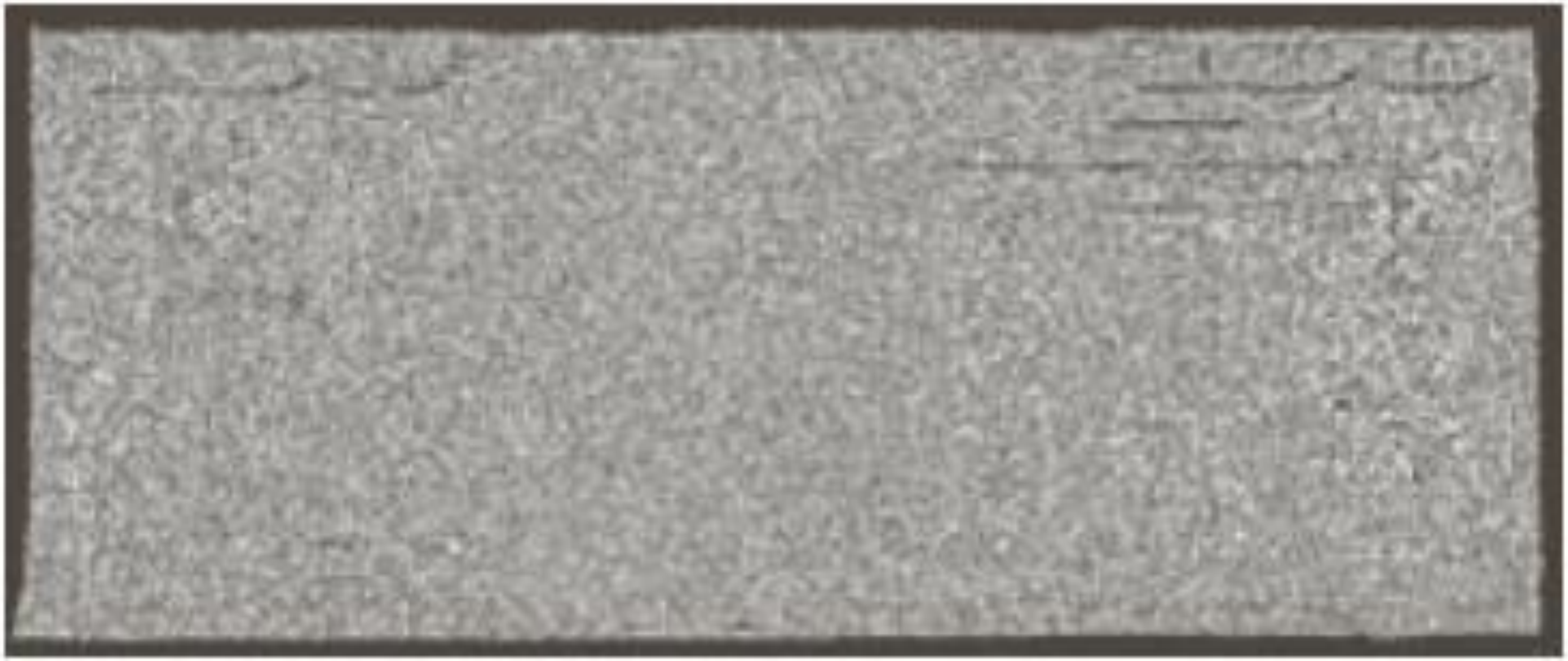
FIGURE 3.14 (a) An 8-bit gray-scale image of size 500 × 1192 pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

Bit Plane Slicing (cont…)

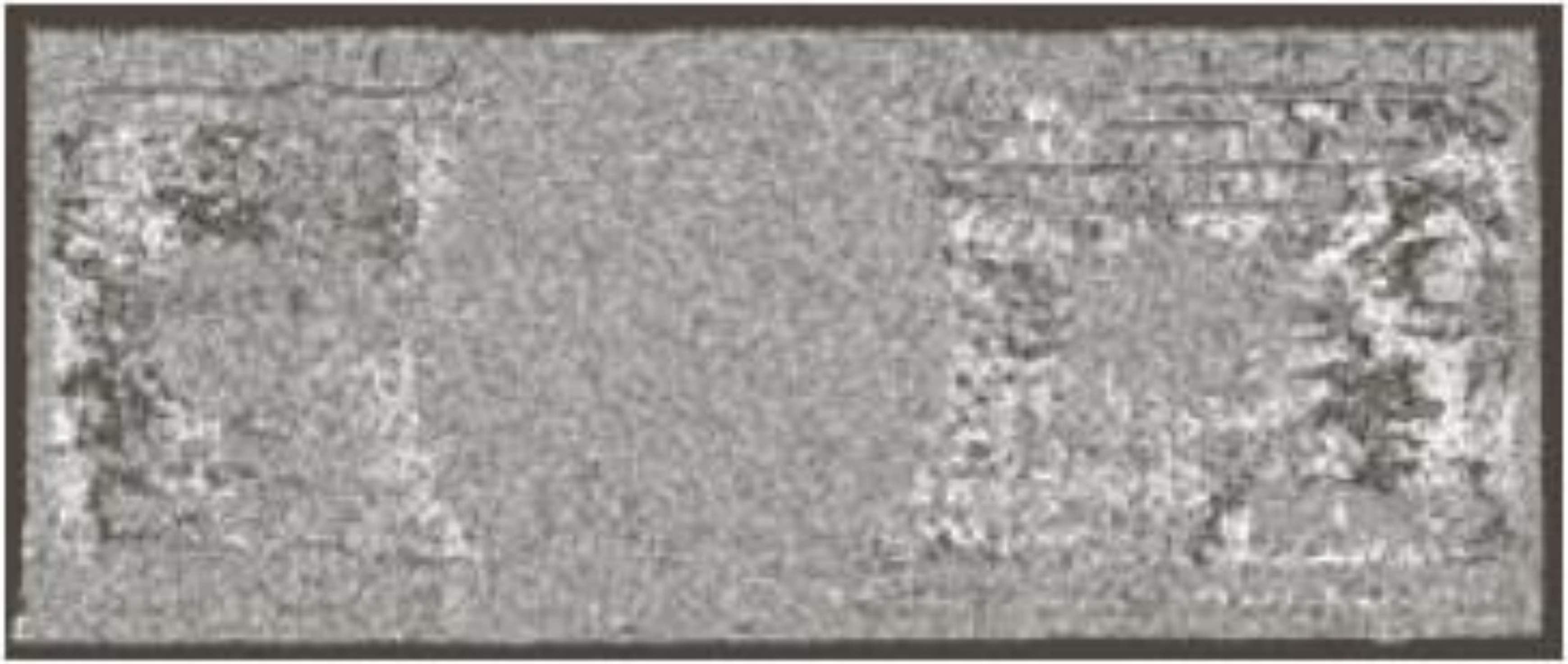
Bit Plane Slicing (cont…)

Bit Plane Slicing (cont…)

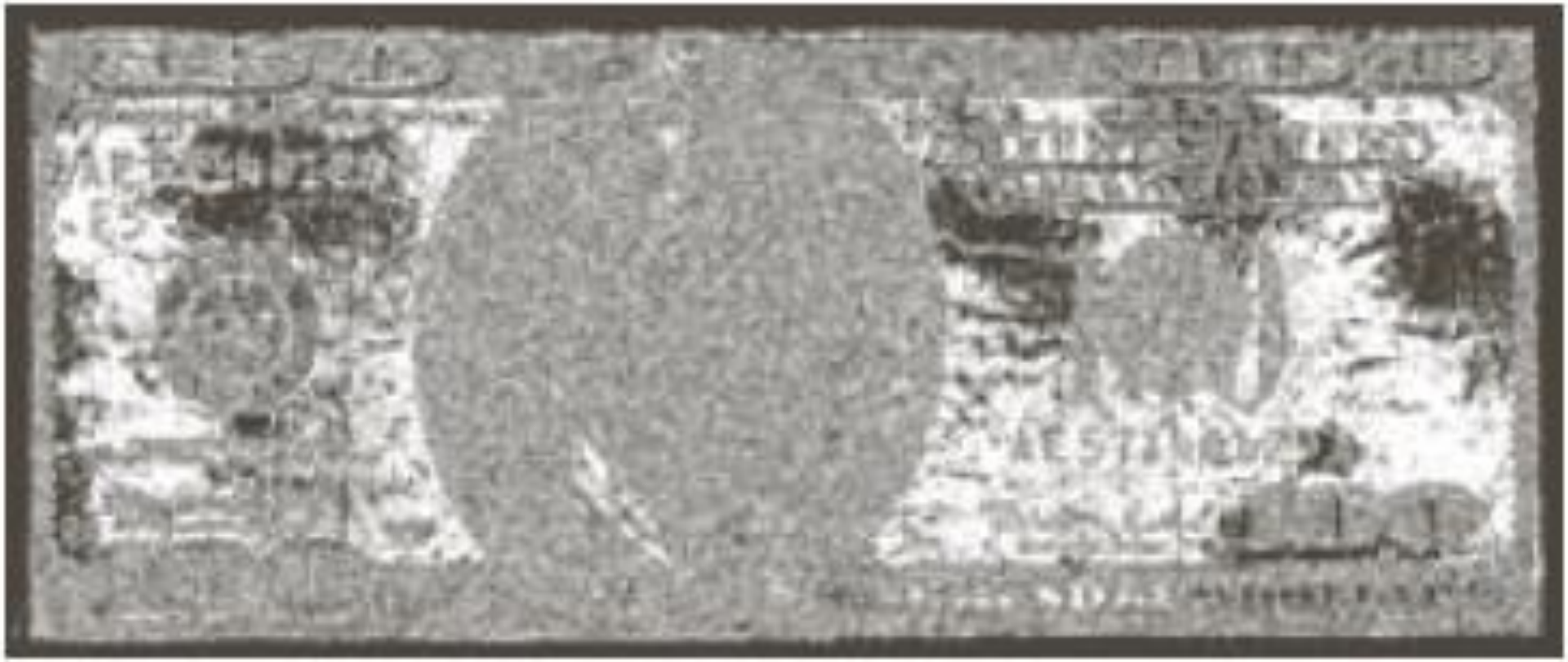
Bit Plane Slicing (cont…)



Bit Plane Slicing (cont…)



Bit Plane Slicing (cont…)



Bit Plane Slicing (cont…)

Bit Plane Slicing (cont…)

Bit Plane Slicing (cont…)

Bit Plane Slicing (cont…)

Reconstructed image using only bit planes 8 and 7

Reconstructed image using only bit planes 8, 7 and 6

Reconstructed image using only bit planes 7, 6 and 5

Summary

We have looked at different kinds of point processing image enhancement

Next time we will start to look at neighbourhood operations – in particular *filtering* and *convolution*