Image Enhancement in the Spatial Domain

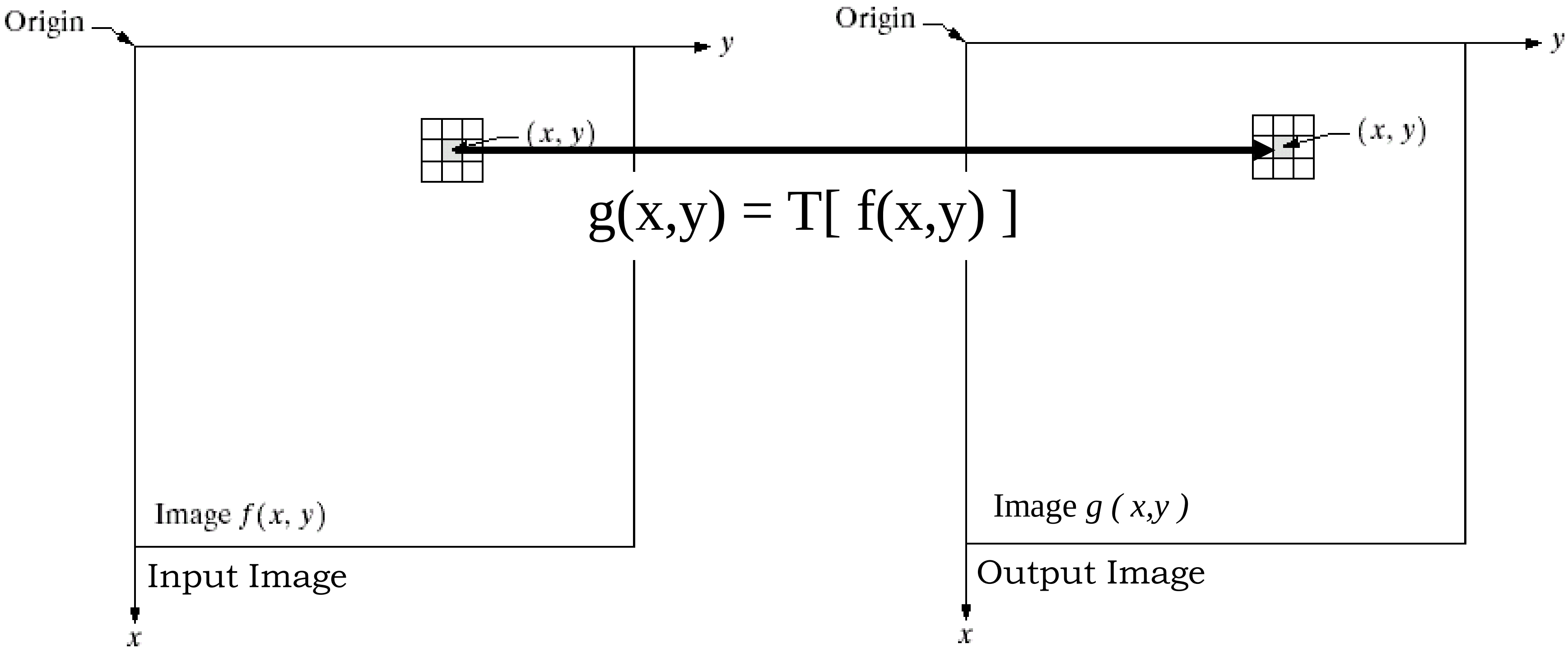
Enhancement

- Objectives
  - It makes an image more suitable than original for specific and problem-oriented applications.
  - It is often for human perception, the viewer is the ultimate judge of how well a particular works.
- Two broad categories
  - Spatial domain
    - Works with pixels
  - Frequency domain
    - Works with frequencies

Spatial Domain

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

- f(x,y) input image
- g(x,y) output, processed or transformed image
- T is an operator on f (T = intensity transformation function)
  - T is defined over some neighborhood of (x,y), e.g., square or rectangular sub-image area, or a point of 1x1 size.
  - T can be operated on a single image or a set of images.



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

T = intensity transformation function

FIGURE 3.1 A 3×3 neighborhood about a point (x, y) in an image.

$$g(x,y) = T[\, f_1(x,y),\, f_2(x,y),\, \ldots,\, f_n(x,y)]$$

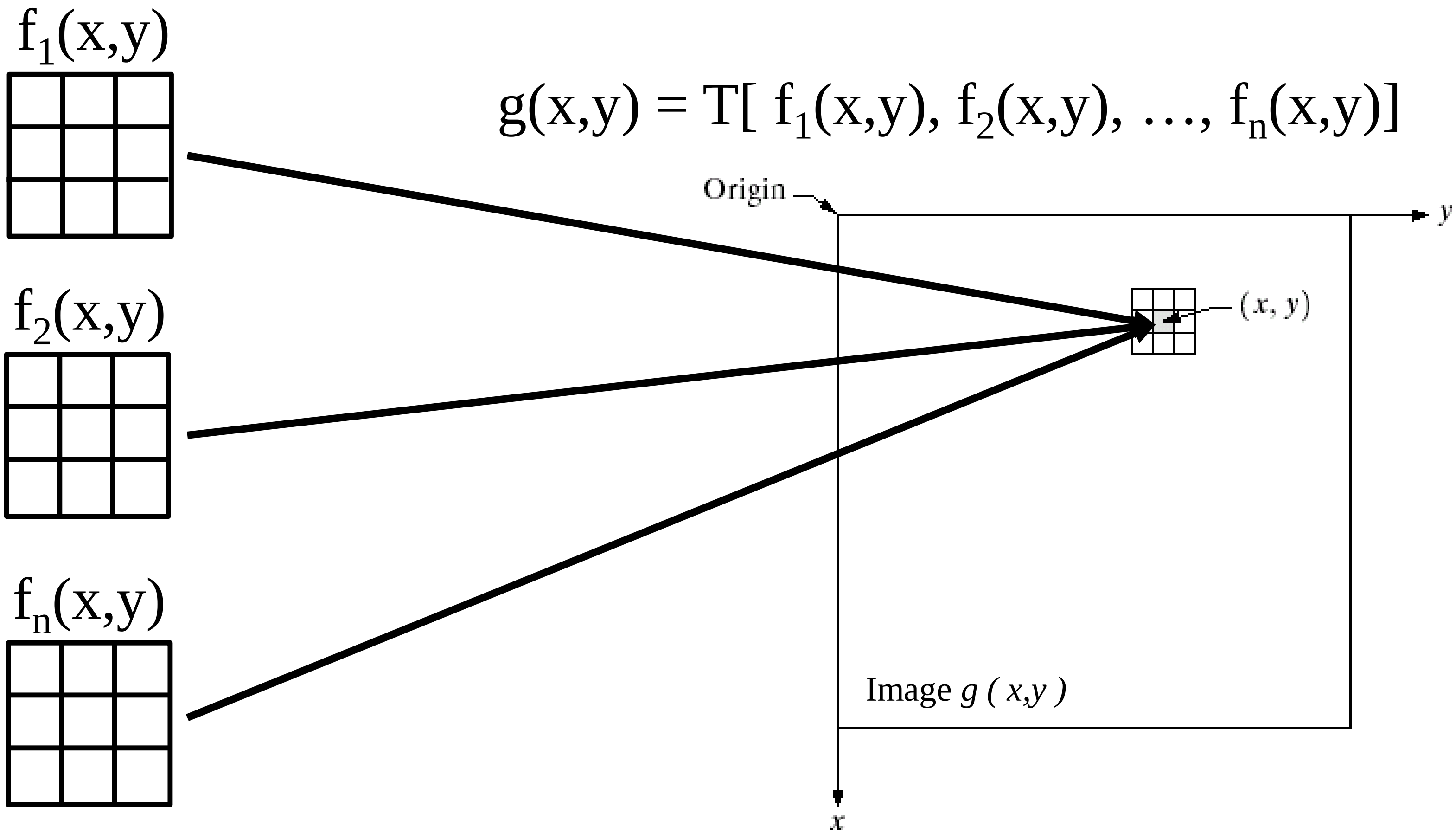


FIGURE 3.1 A 3×3 neighborhood about a point (x, y) in an image.

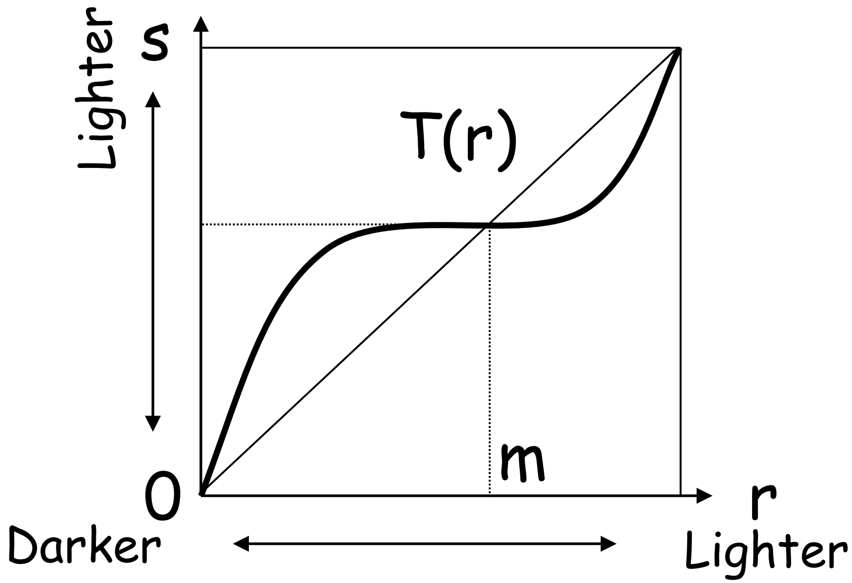
T = intensity transformation function

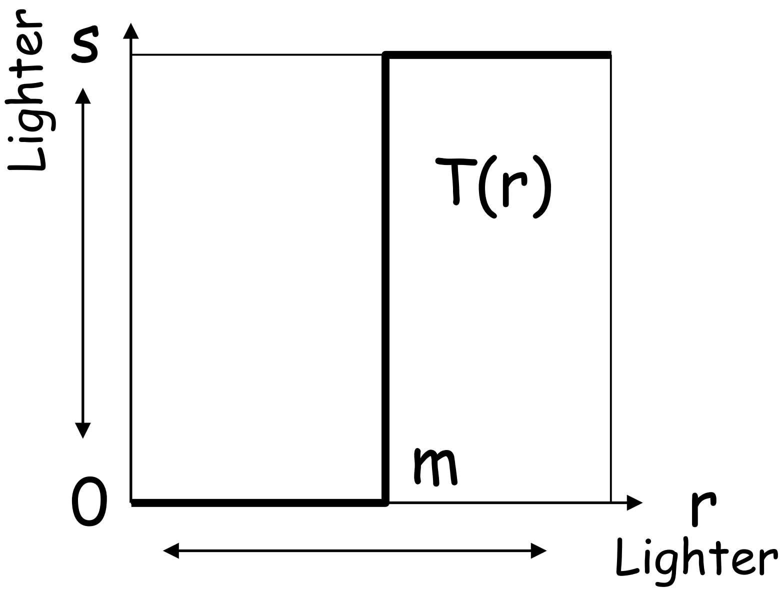
Gray-level transformation

- Simplest form of T
  - neighborhood is 1 x 1 (one pixel)

- Notation: $s = T(r)$
  - r, s denote gray levels of $f(x,y)$ and $g(x,y)$ for any point (x,y) respectively

Examples of T(r)



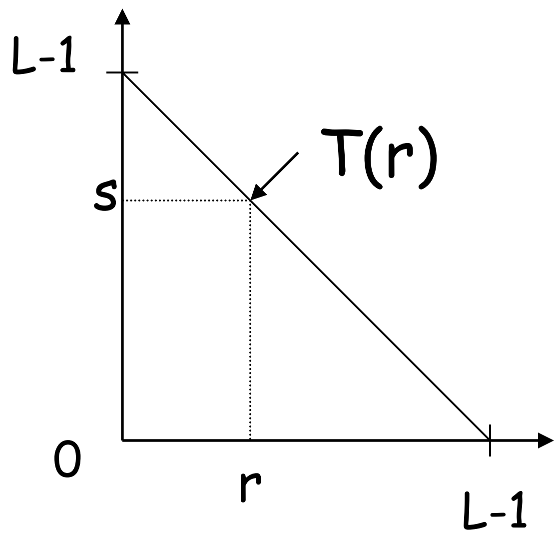


It produces an image of lower contrast than the original by (1) darkening the intensity levels above m and (2) brightening the leaves below m in the original image.

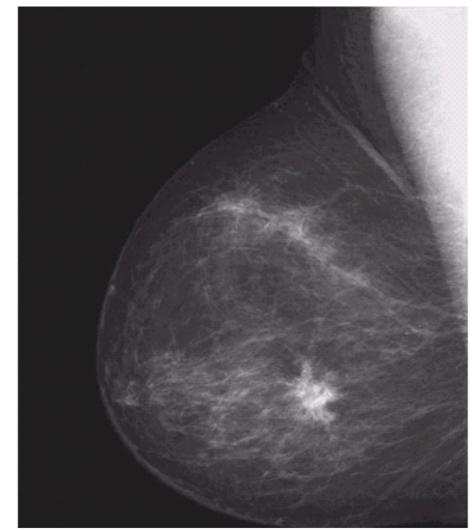
It produces a binary image with an intensity threshold at m.

Low intensity levels below m map to zero and high intensity levels above m map to s.

Basic Gray Level Transforms: Inverse or Image Negatives



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



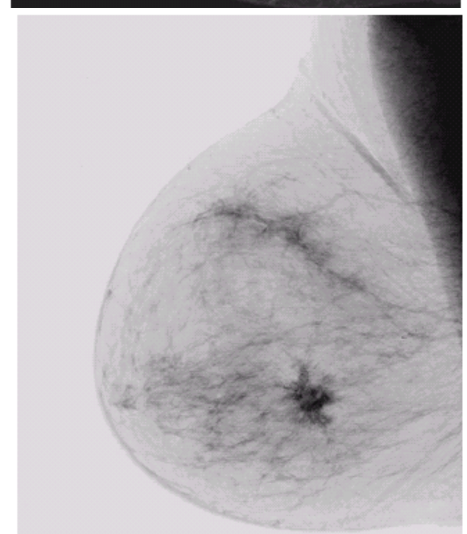
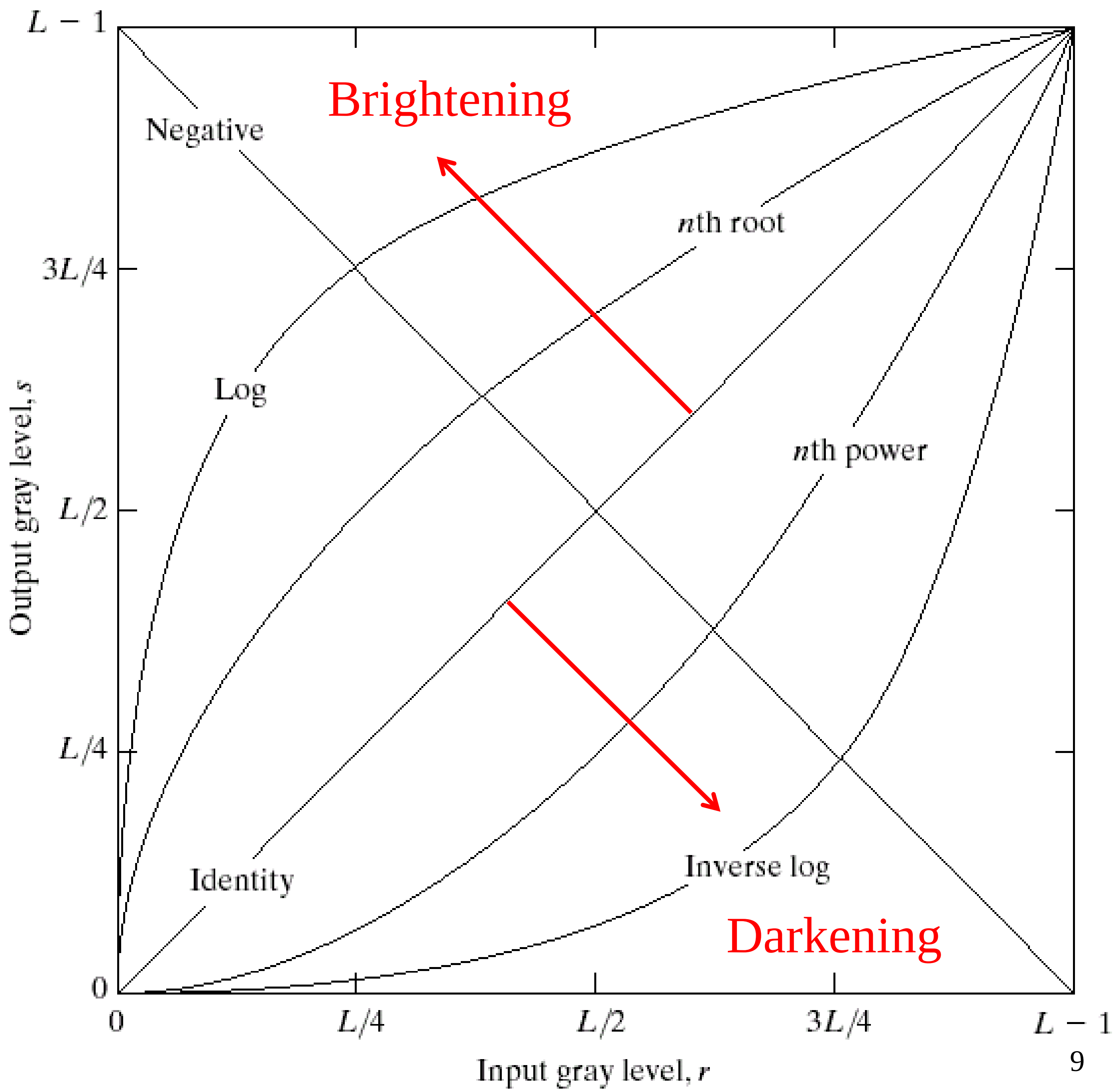
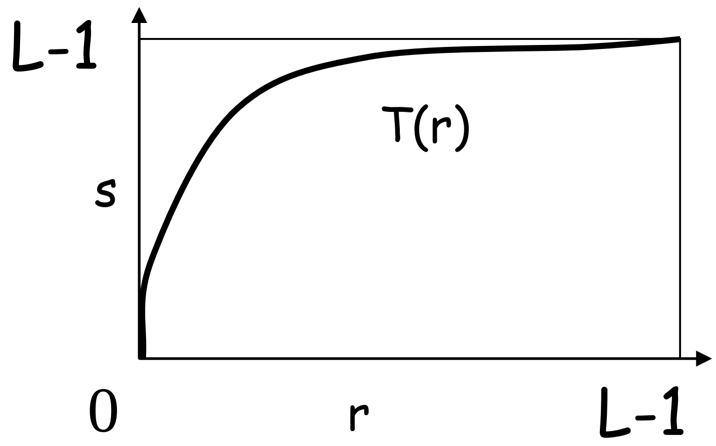


FIGURE 3.3 Some basic gray-level transformation functions used for image enhancement.



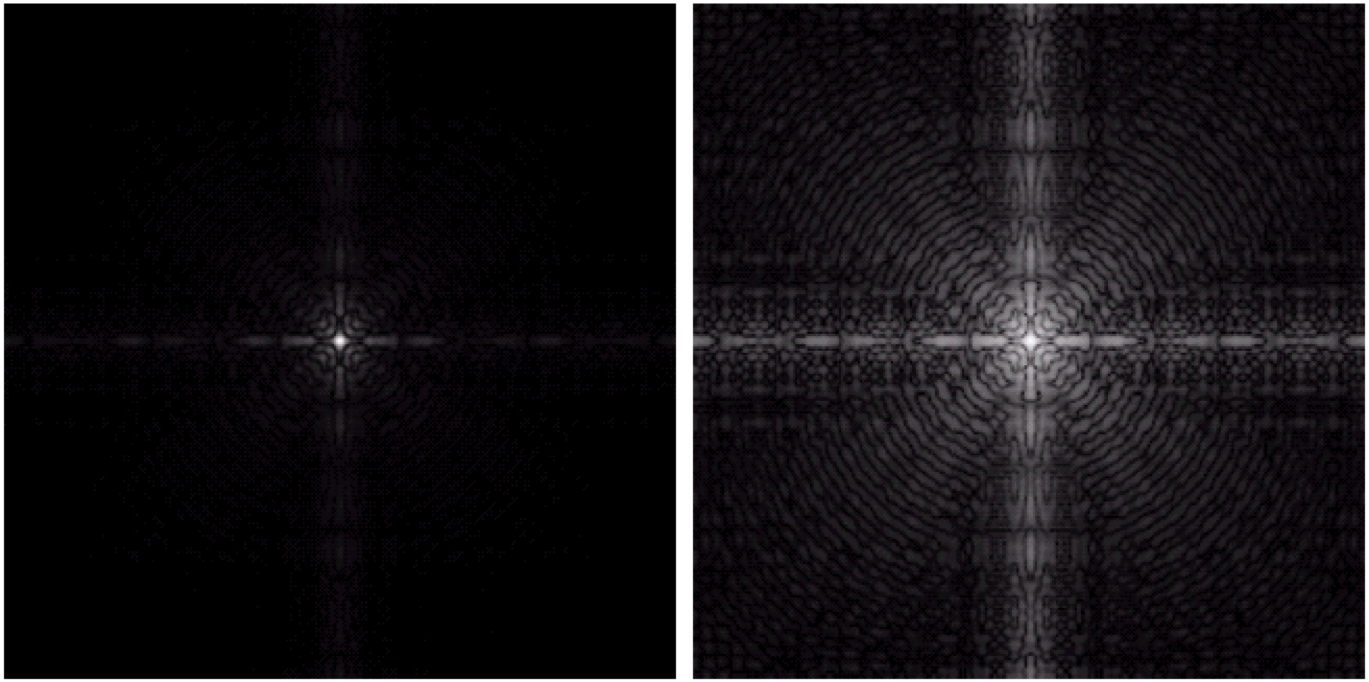
Log Transformations



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

(c is some scale-factor)
(intensity rescaling maybe needed for s to fit s into the range [0, L-1])

The image becomes brighter and details are more visible.



a b

FIGURE 3.5 (a) Fourier spectrum. (b) Result of applying the log transformation given in Eq. (3.2-2) with $c = 1$.

Power-Law Transformations

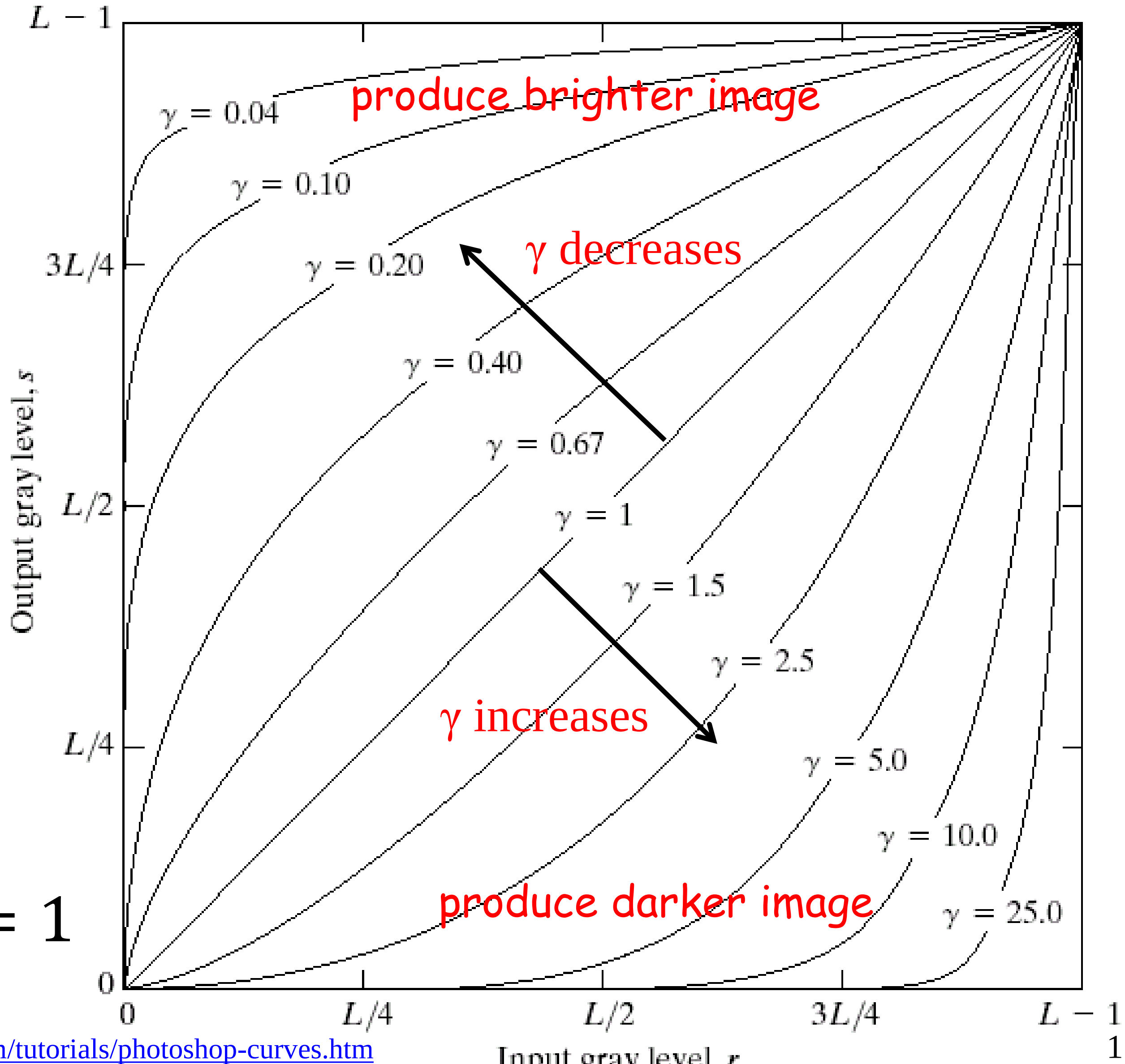
$$s = cr^{\gamma}$$

where C and γ are positive constants.

Intensity rescaling may be needed for *s* to fit *s* into the range [0, L-1].

Also, known as *gamma* (γ) *correction.*

$$C = 1$$



Examples: http://www.cambridgeincolour.com/tutorials/photoshop-curves.htm

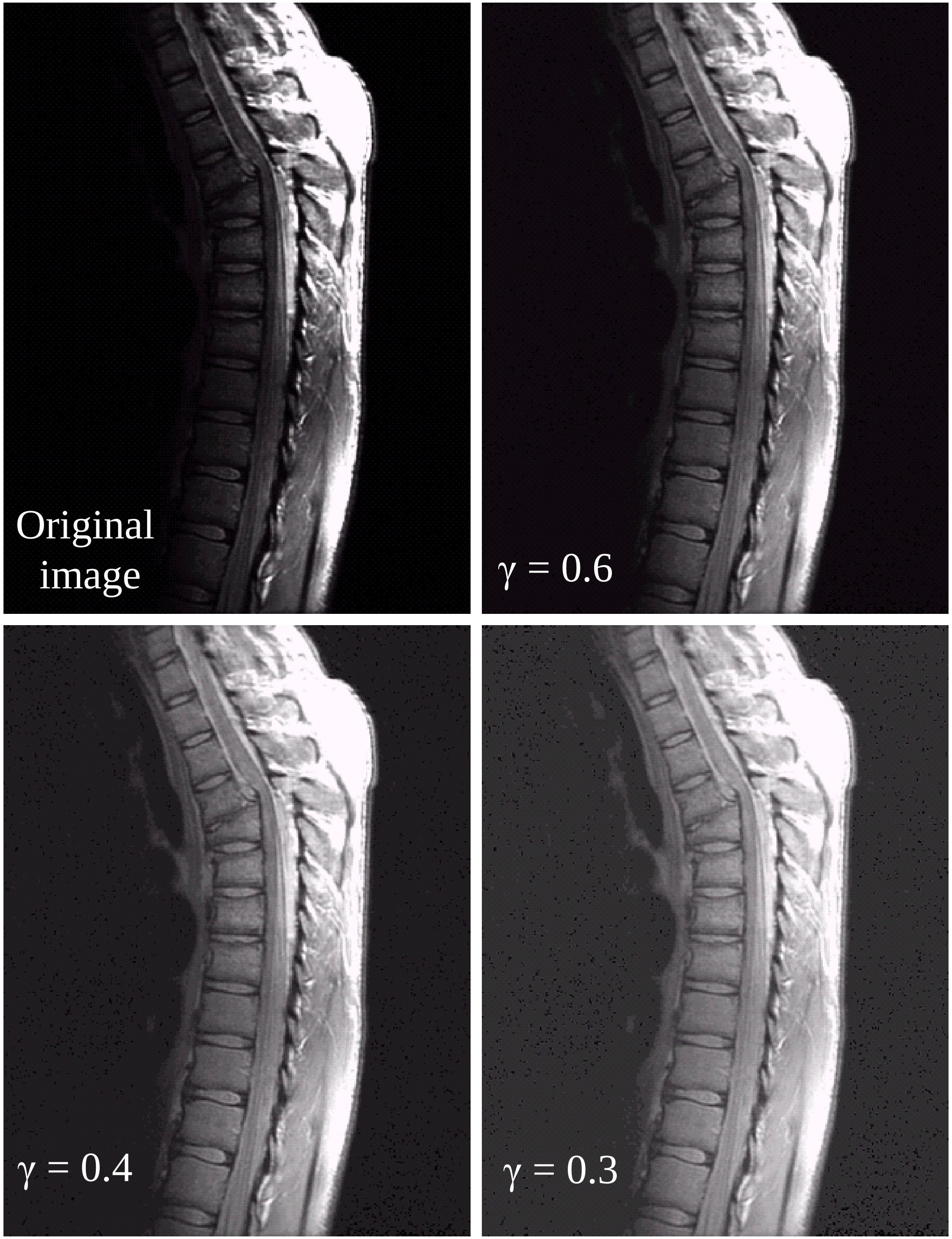
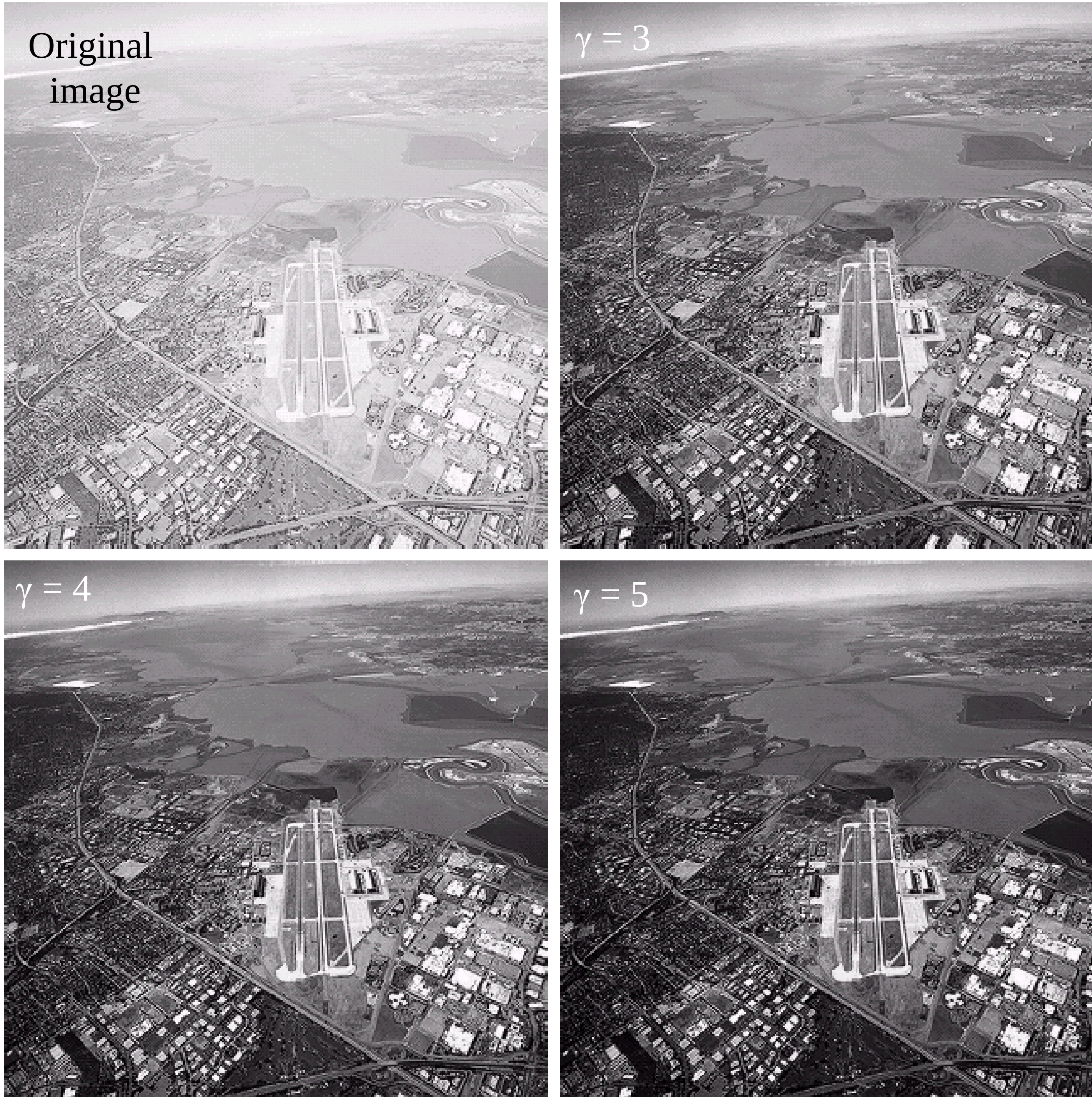


FIGURE 3.8 (a) Magnetic resonance (MR) image of a fractured human spine. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 0.6, 0.4,$ and 0.3, respectively. (Original image for this example courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

With the help of power-law transformation, fractures near the vertical centre of the spine are now made more visible in the magnetic resonance image.

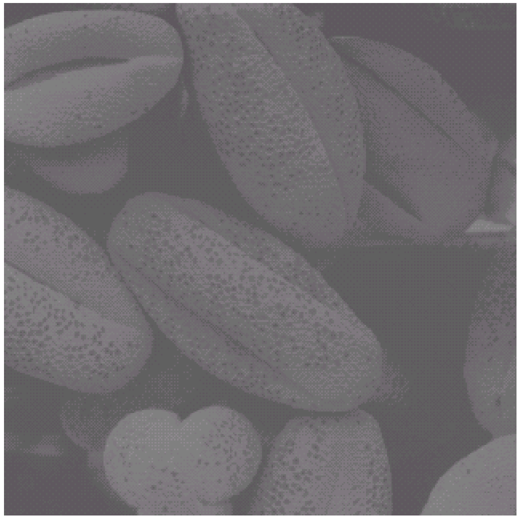
a b
c d

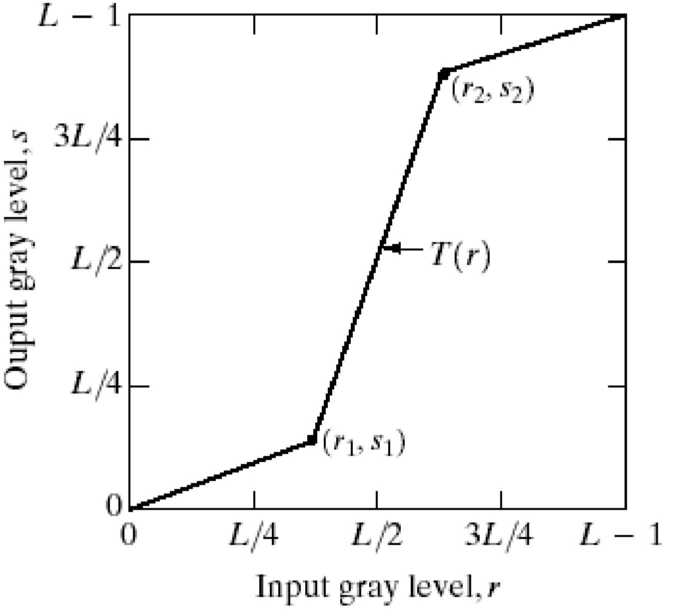
FIGURE 3.9 (a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0, 4.0,$ and 5.0, respectively. (Original image for this example courtesy of NASA.)



Contrast Stretching

A low-contrast image





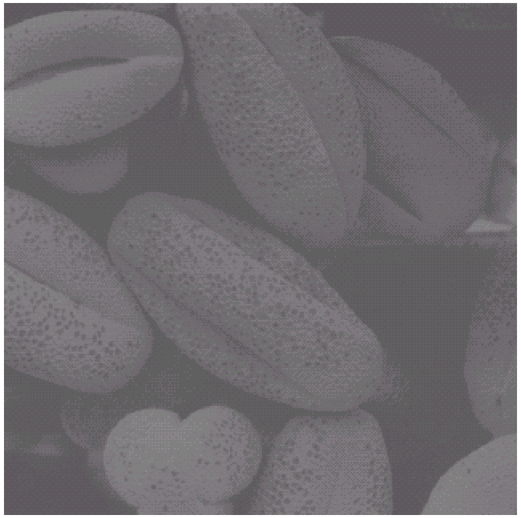
Output image

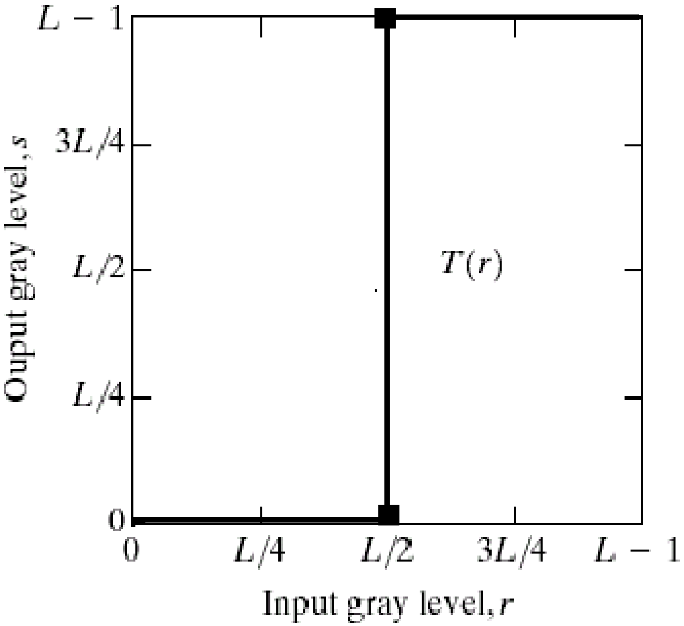
If $r_1 = r_{\min} \quad s_1 = 0$
$r_2 = r_{\max} \quad s_2 = L-1$

then intensity levels are stretched to the full intensity range [0, L – 1].

Contrast Stretching

A low-contrast image



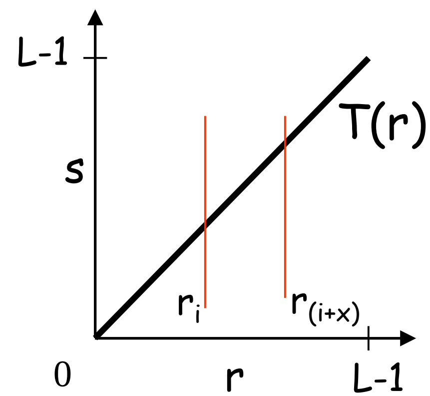


Output binary image

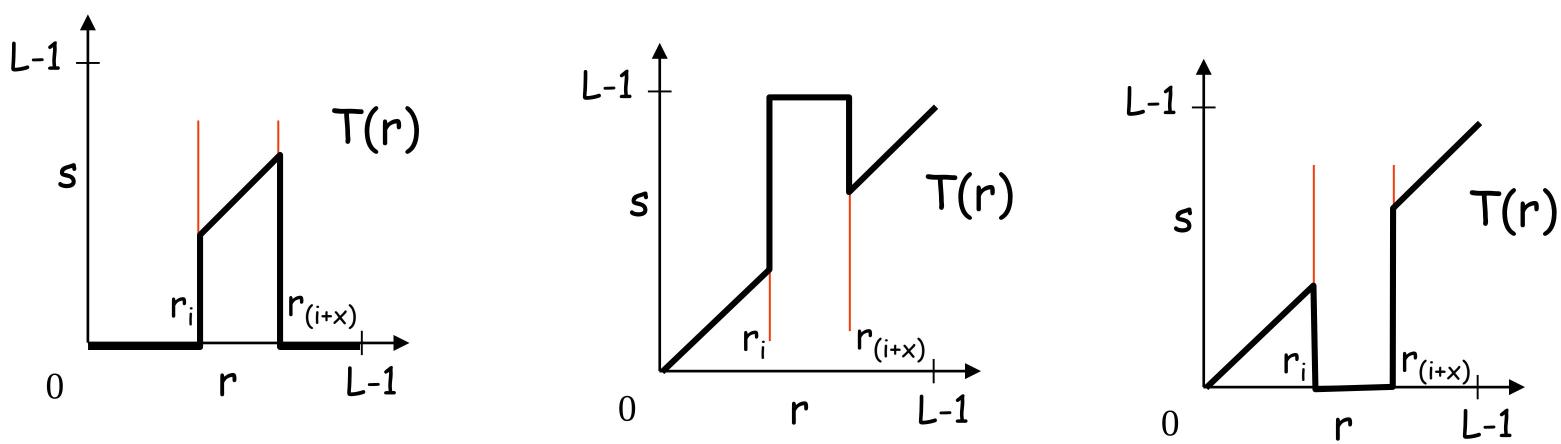
If $r_1 = r_2$ $s_1 = 0$ $s_2 = L-1$

then the image is thresholded and a binary image is produced.

Windowing or Gray Level Slicing

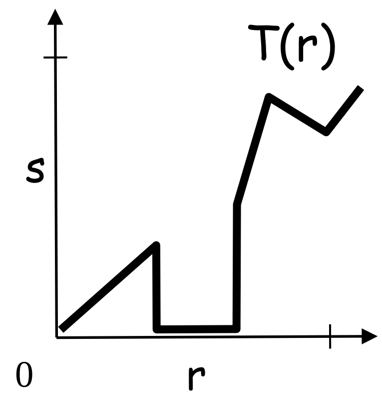


Operating on a "window" of gray levels r_i to $r_{(i+x)}$



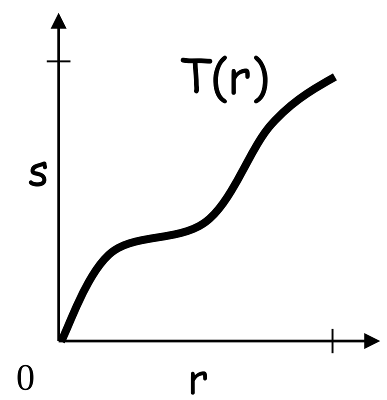
Properties of T(r)

- Non-increasing
  or non-decreasing



•No-inverse due to ambiguity
•Does not preserve gray level ordering
•For non-inverse transformation, it is not an one-to-one mapping and could be an one-to-many/ many-to-one mapping.

- Increasing
  or decreasing



•Inverse
•Preserves gray level ordering (increasing)

Image Histogram

- Image Histogram
  - digital image with gray levels [0, L-1]
  - $p(r_k) = n_k/N$, probability of occurrence of gray level r_k
  - r_k is the k^{th} gray level
  - n_k = number of pixels with k^{th} gray level
  - N = total number of pixels
  - $k = 0, 1, 2, 3, 4, 5 \ldots, L-1$

Image Histogram

- $p(r_k)$ is the probability of the occurrence of gray-level r_k

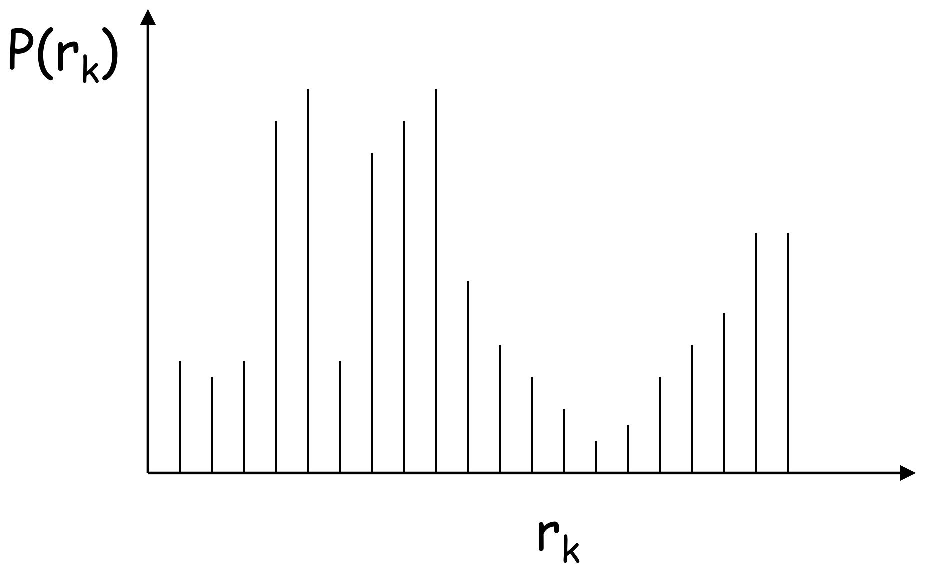


Image Histogram

- Histogram can tell you a lot about an image
- Gray level distribution
- It can be modelled by statistical distribution (red line)

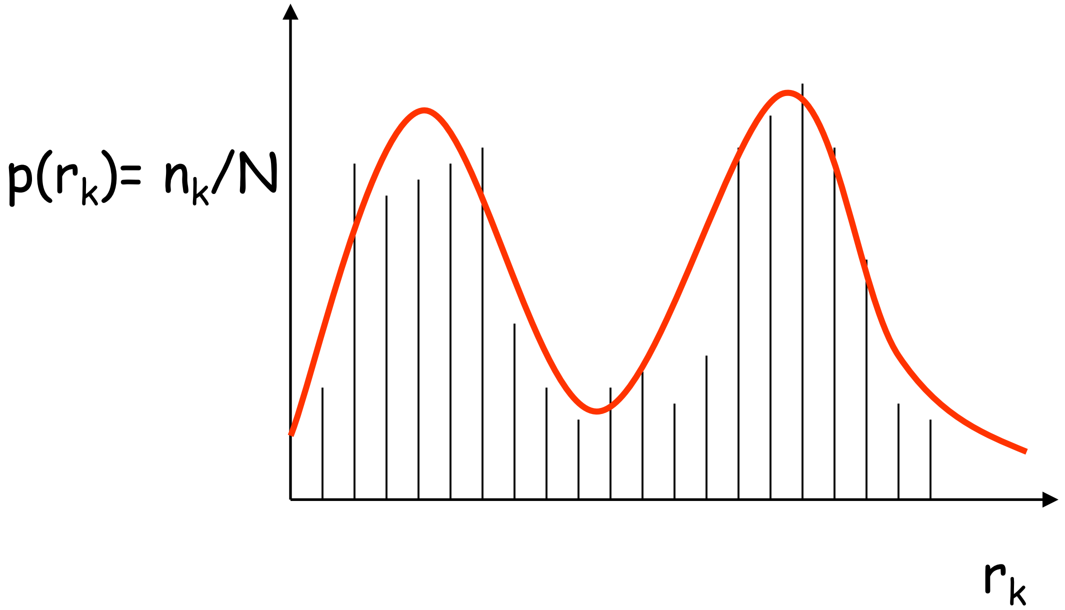


Image Histogram

- Gray level distribution helps define intensity threshold

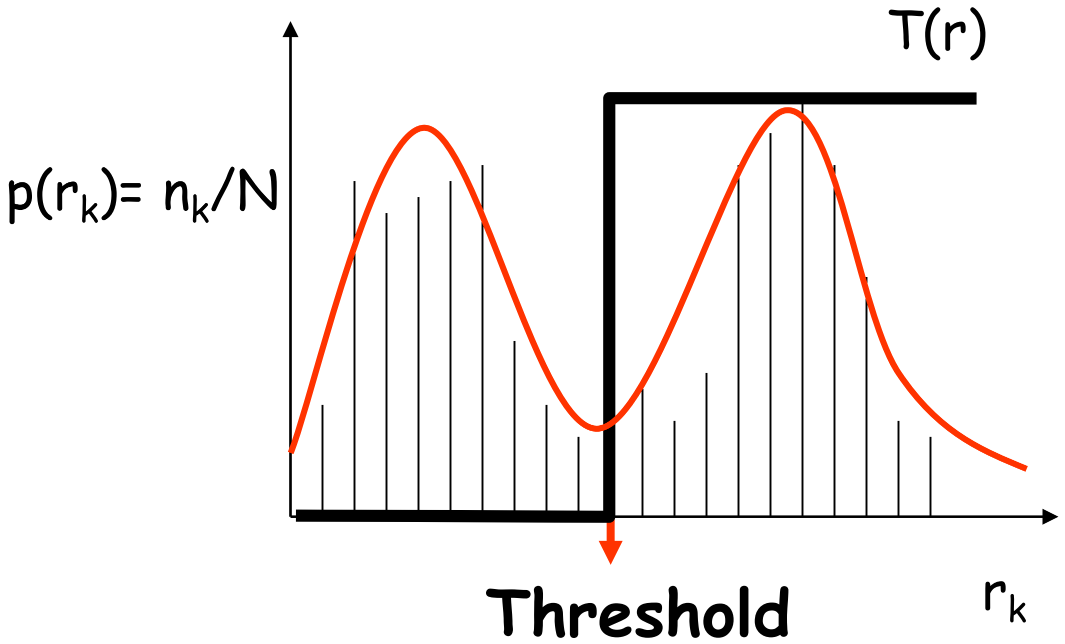


Image Histogram

- Histograms are not unique. Two images below give the same image histogram.
- No spatial information is captured

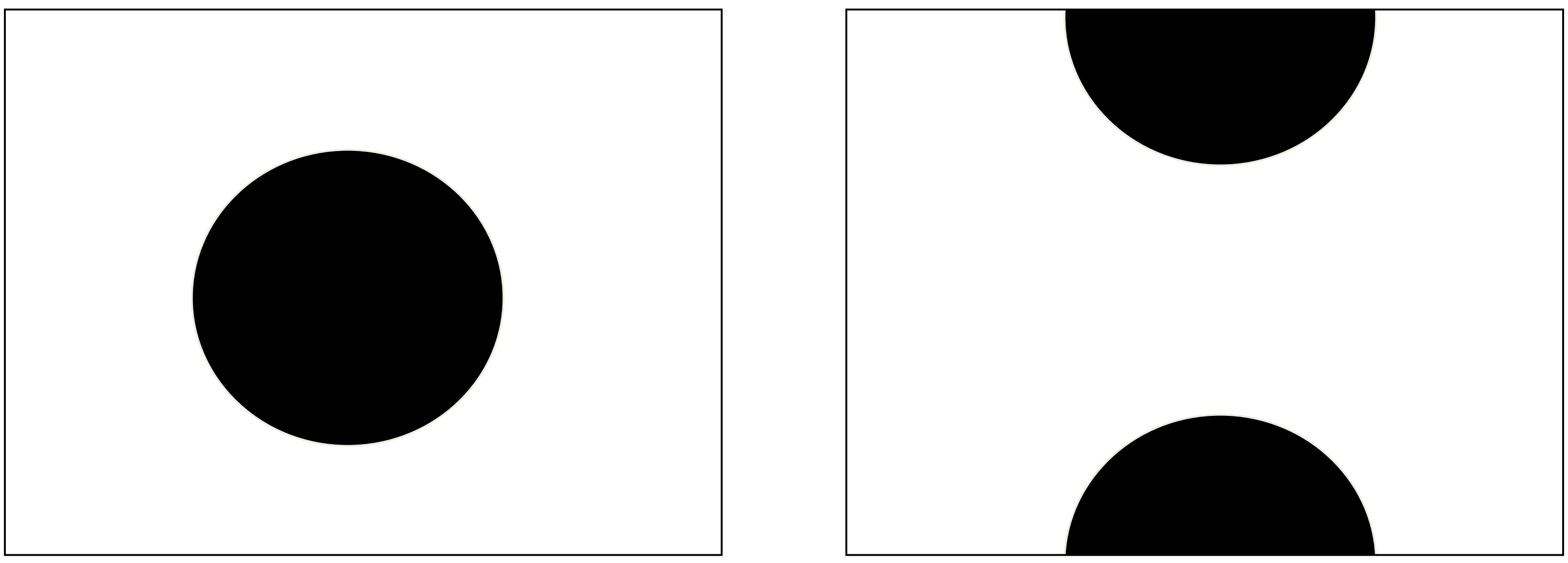
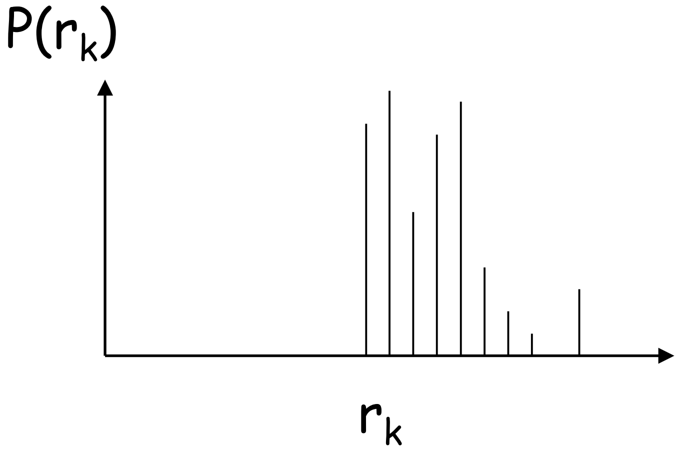


Image Histogram

- Types of Image

Bright Image



Dark Image

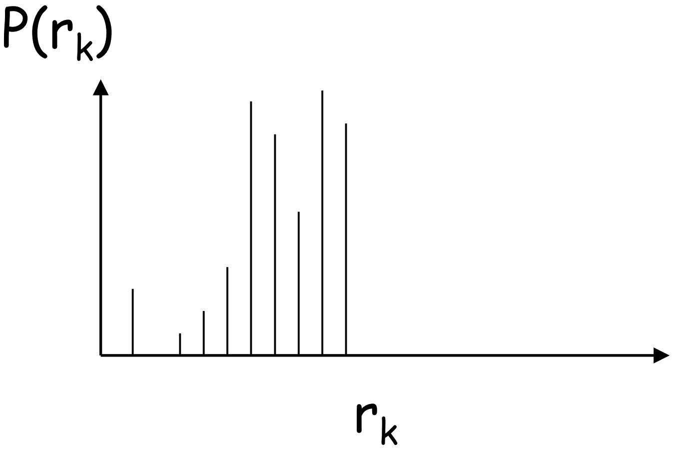
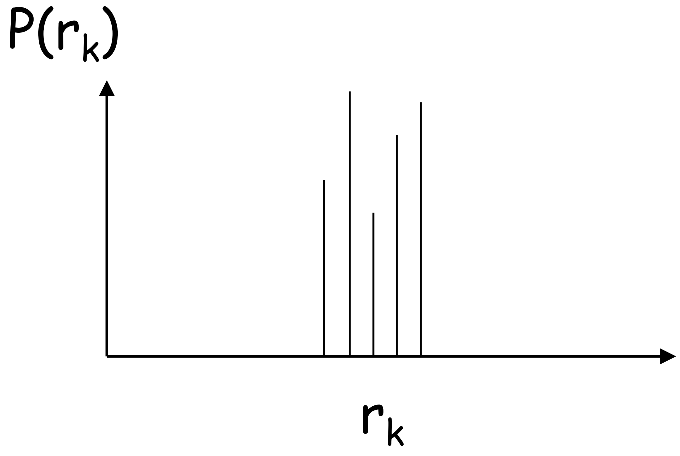


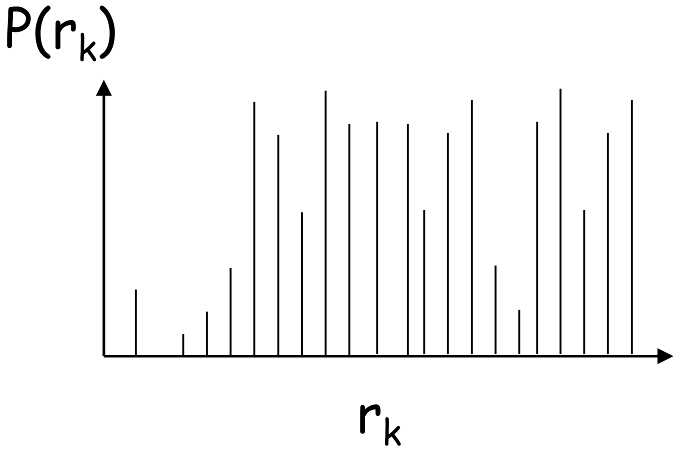
Image Histogram

- Types of Image

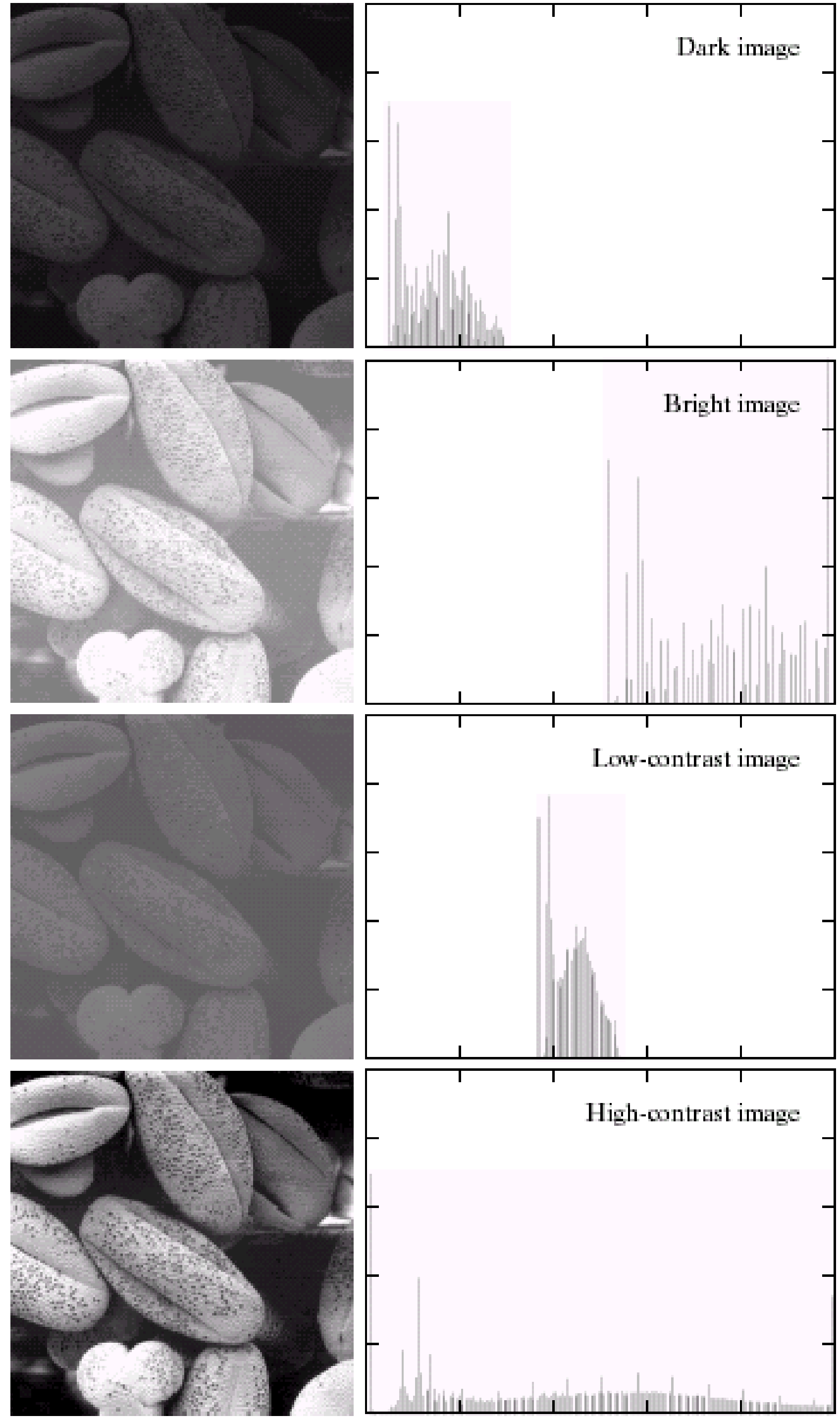
Low Contrast



High Contrast

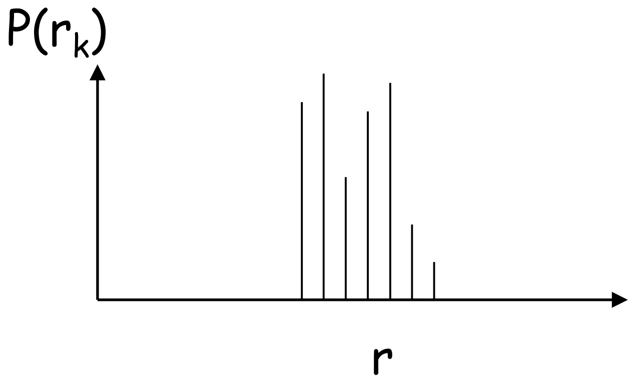


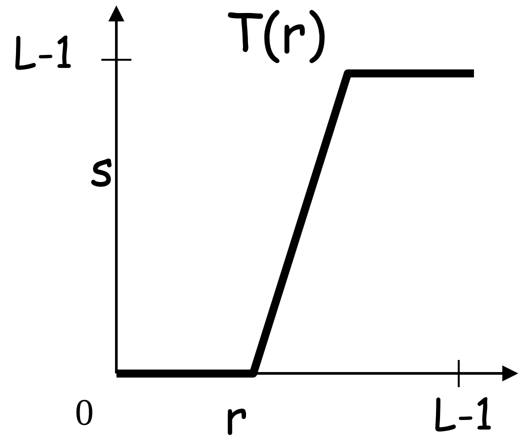
Some Examples

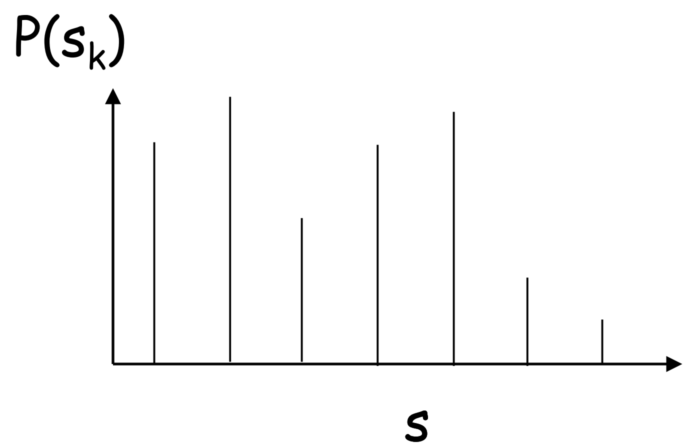


Contrast Stretching

Low Contrast

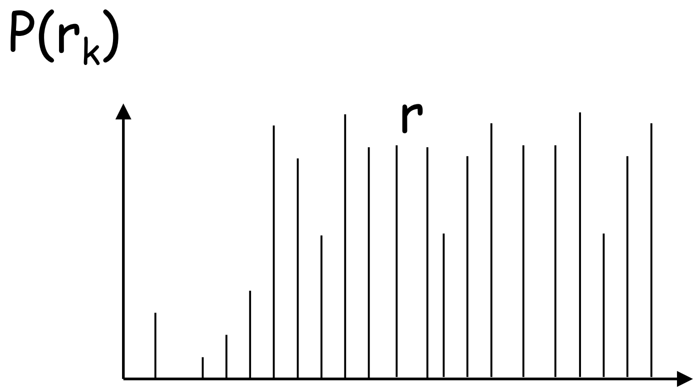


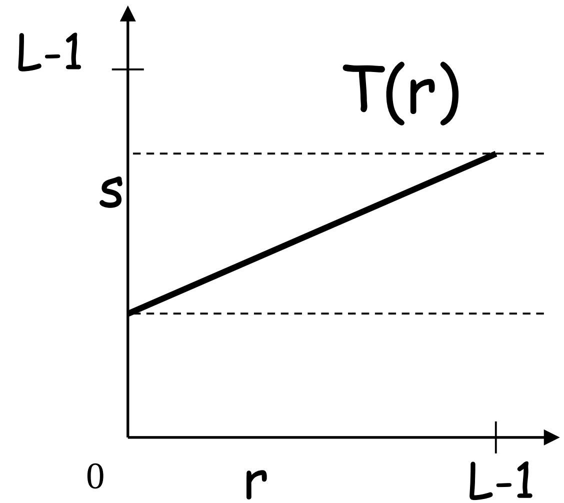


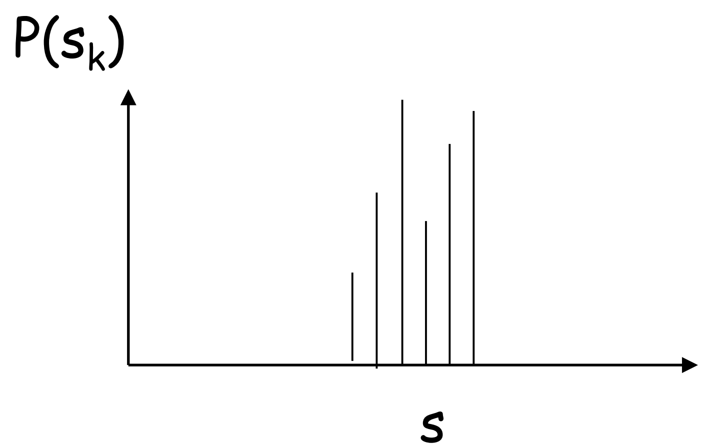


Contrast Compressing

High Contrast







Histogram Equalization

- We want an image with equally many pixels at every gray level, or the output intensity approx. follows *uniform distribution*.

- That is, a flat histogram, where each gray level, r_k, appears (N/r_m) times

  - where "r_m" is the maximum gray level
  - N is number of pixels in the image

- Example
  - http://www.mathworks.com/help/toolbox/images/ref/histeq.html

Histogram Equalization Image Transformation

- Cumulative distribution function (CDF) of r is represented by

$$s = T(r) = \int_0^r p_r(w)dw$$

T(r) to equalize the histogram

- Assume the input variable “r” has been normalized between [0,1].

- s = T(r), there are two properties
  - (a) T(r) is single-valued in the interval 0 <= r <= 1
  - (b) 0 <= T(r) <= 1 for 0 <= r <= 1

Conditions (a) and (b)

- (a) $T(r)$ is single-valued and non-decreasing in the interval $0 <= r <= 1$
  - Can preserve the order from black to white in the gray scale.
  - Can preserve the basic appearance of an image.

- (b) for $0 <= r <= 1$, $0 <= T(r) <= 1$
  - Can guarantee a mapping that is consistent with the allowed range of pixel values. No intensity rescaling is needed.

CDF is for continuous functions

- Let's look at the discrete approximation

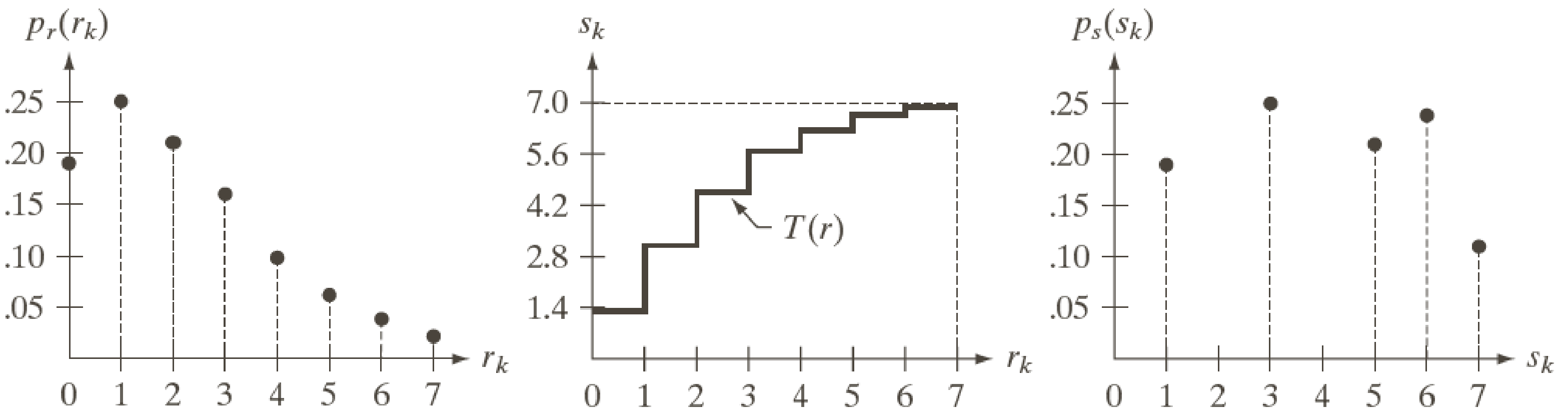
$$s_k = T(r_k) = \sum_{j=0}^{k} n_j / N = \sum_{j=0}^{k} p_r(r_j)$$

s_k is output intensity
r_k is input intensity
n_j is number of pixels with jth gray level
k = 0, 1, 2, 3, . . L-1 (gray levels)

Example

r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

TABLE 3.1 Intensity distribution and histogram values for a 3-bit, 64×64 digital image.



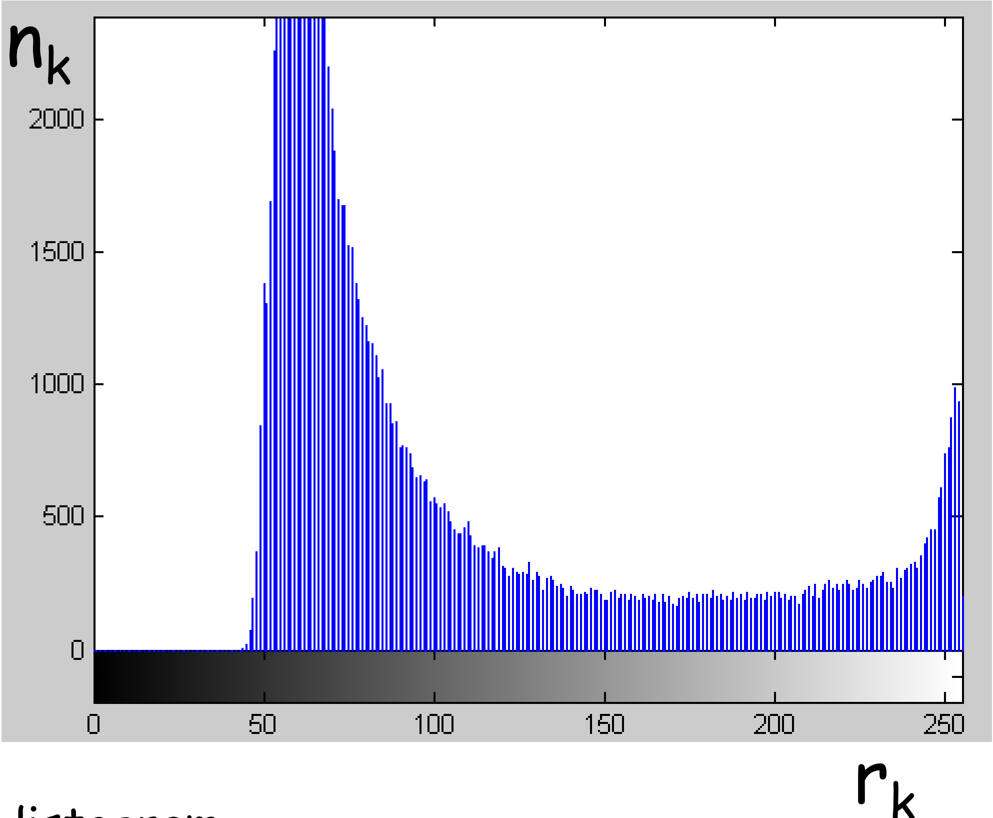
a b c

FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

Example

http://www.mathworks.com/help/toolbox/images/ref/histeq.html

Image

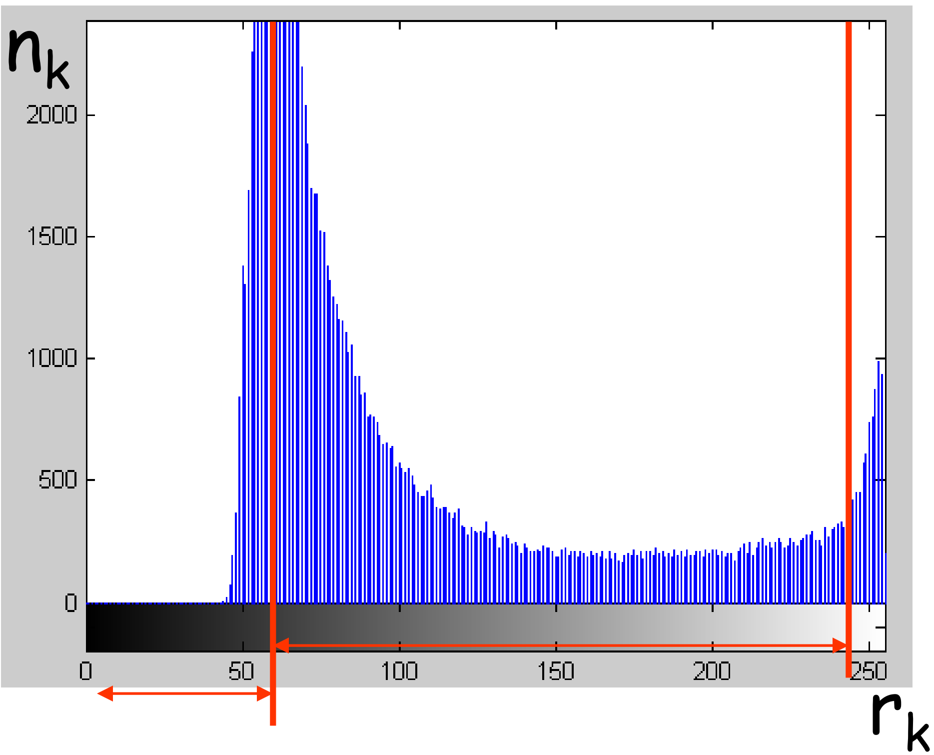


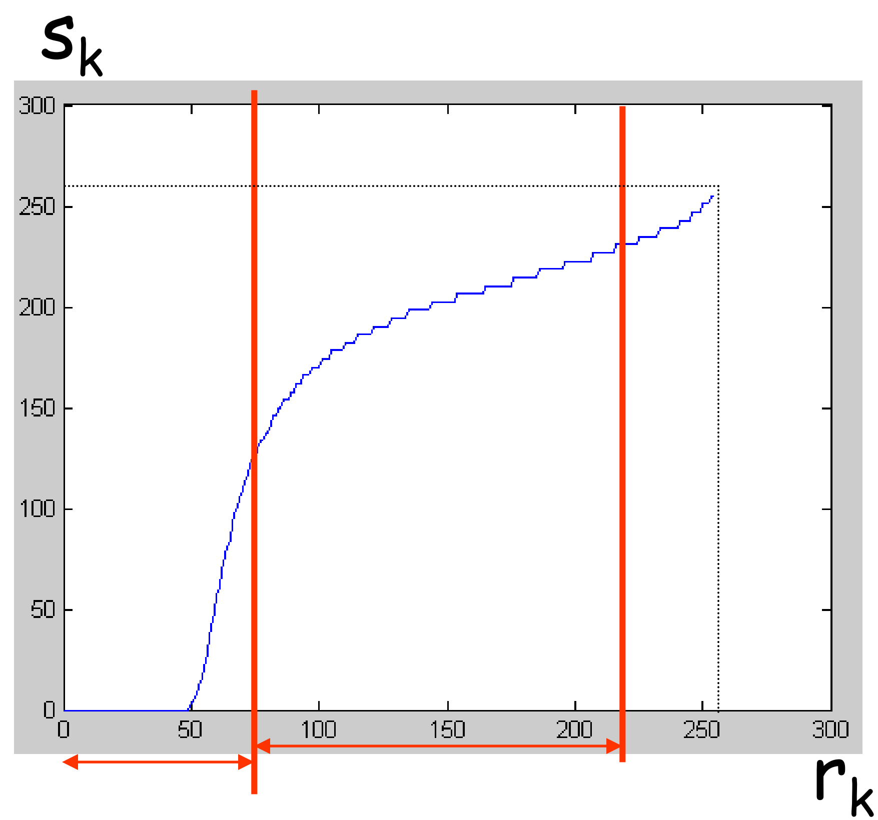
Histogram

Notice, this is not normalized, y axis is n_k.

To normalize, let y axis = $p(r_k) = n_k / N$

Example





stretching compressing

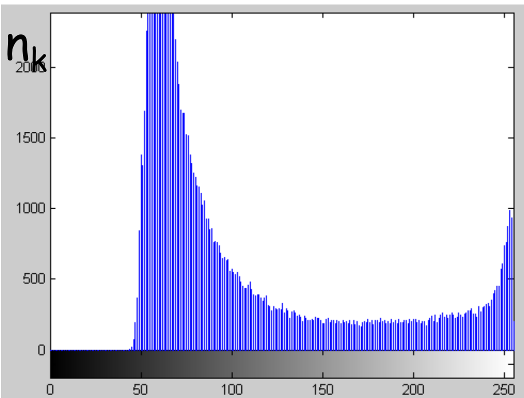
Histogram

Notice, this is not normalized, y axis is n_k.

To normalize, let y axis = $p(r_k) = n_k / N$

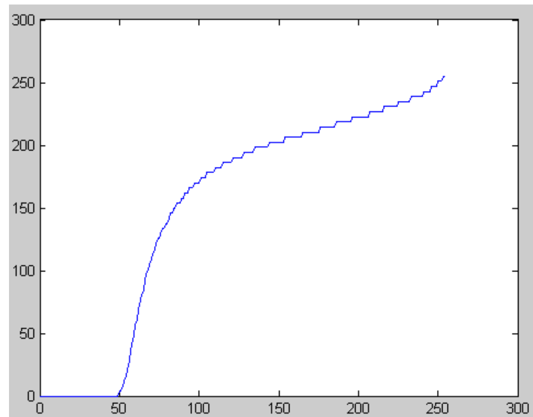
Example

n_k



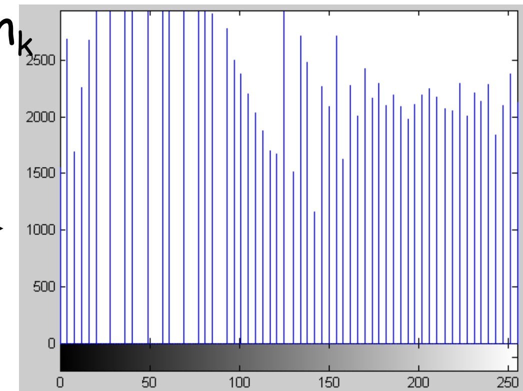
r_k

s_k



$T(r)$

n_k



s_k

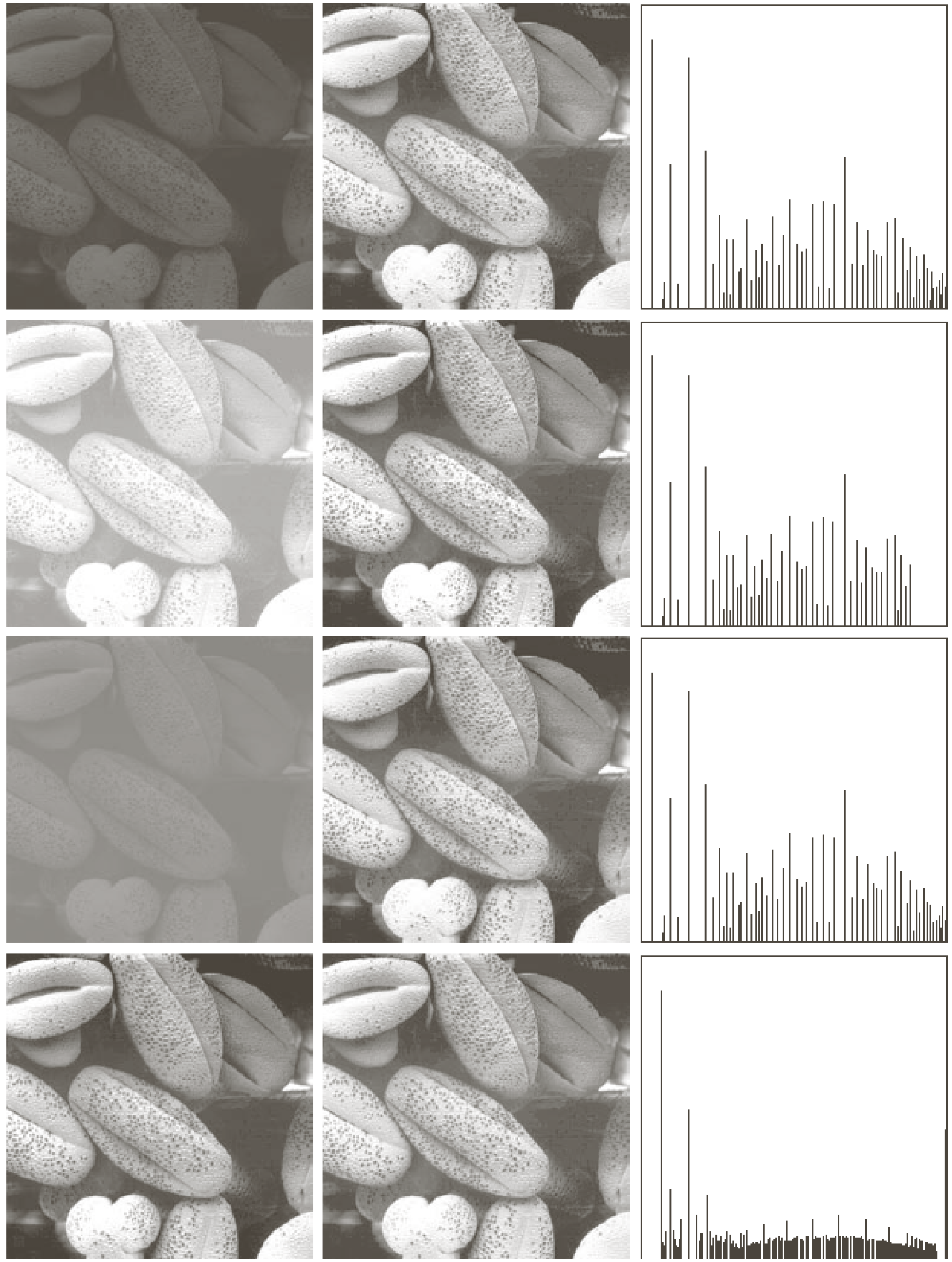


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

Histogram Equalization

- Can significantly improve image appearance
  - Automatic
  - users doesn’t have to perform manual windowing or manual gray level slicing

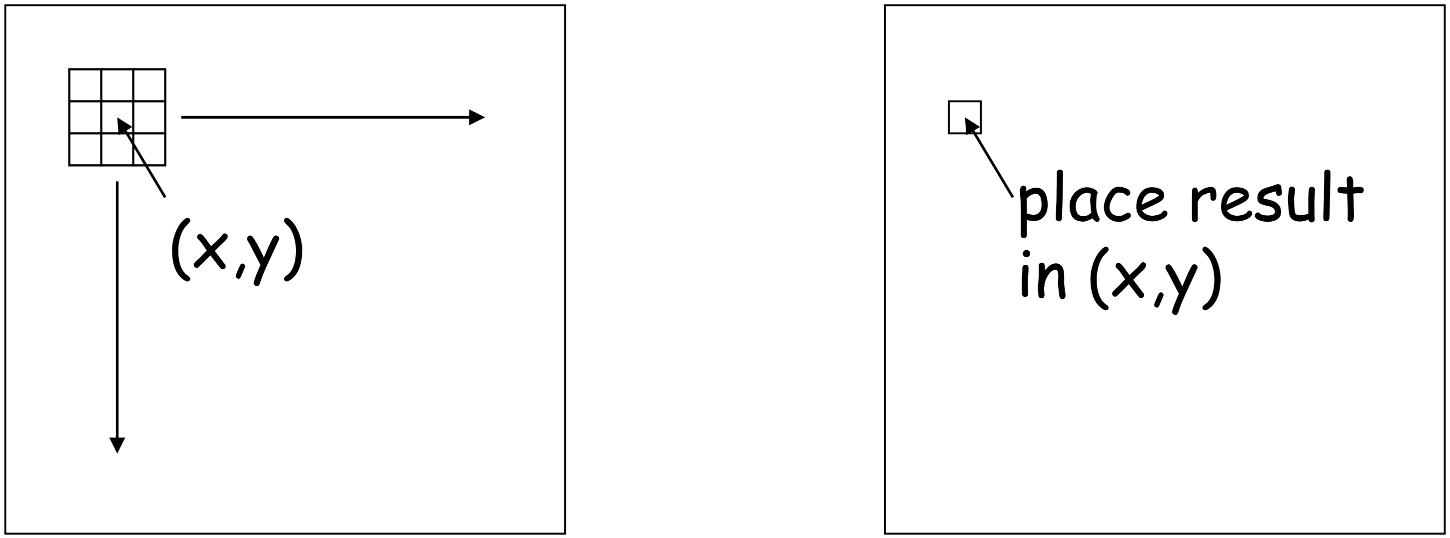
- Nice pre-processing step before image comparison
  - Account for different lighting conditions
  - Account for different camera/device properties

Local Enhancement

- Histogram equalization is a global operation
  - Each pixel is processed based on information of the entire image
  - Often enhances global details

- We would like to enhance details over small areas
  - Each pixel is processed based on information of a small area/sub-image

Local Histogram Equalization



Result

calculate histogram
using neighborhood of mxm
about (x,y)

Local Histogram

- Apply histogram equalization about a neighborhood around (x,y)

- Transform the gray level for pixel (x,y)

- Move the neighborhood over the rest of the image

Local Histogram

Reveals detail in local areas

Original Global Histogram Local Histogram

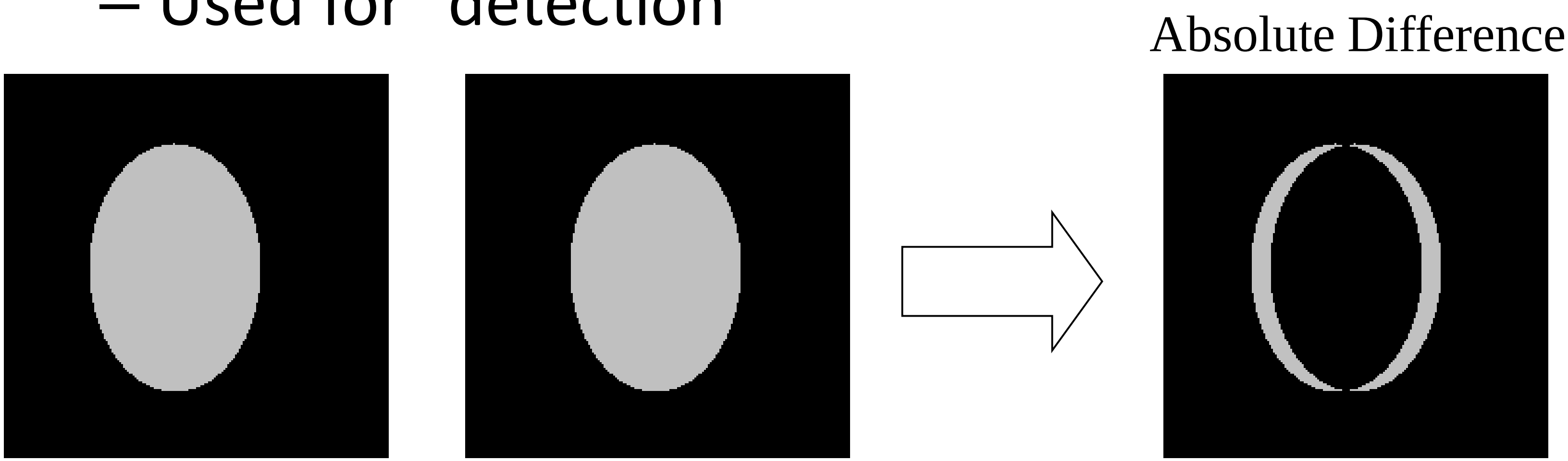
Image Operations

- Arithmetic Operations (p and q are images)
  - p+q
  - p-q
  - p*q   (also stated as pq, or p x q)
  - p%q

- Logic Operations (p and q are binary images)
  - p AND q
  - p OR q
  - Not q
  - p XOR q

Image Operations

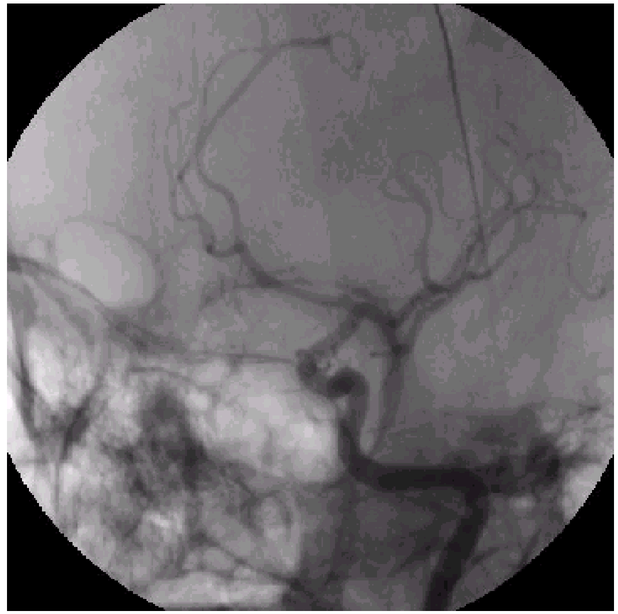
- Subtraction or Absolute Difference
  - Very useful for determining the difference between two images
  - Used for “detection”



Grey = non-zero, Background = 0.

Example

X-ray brain image
without contrast agent
injected



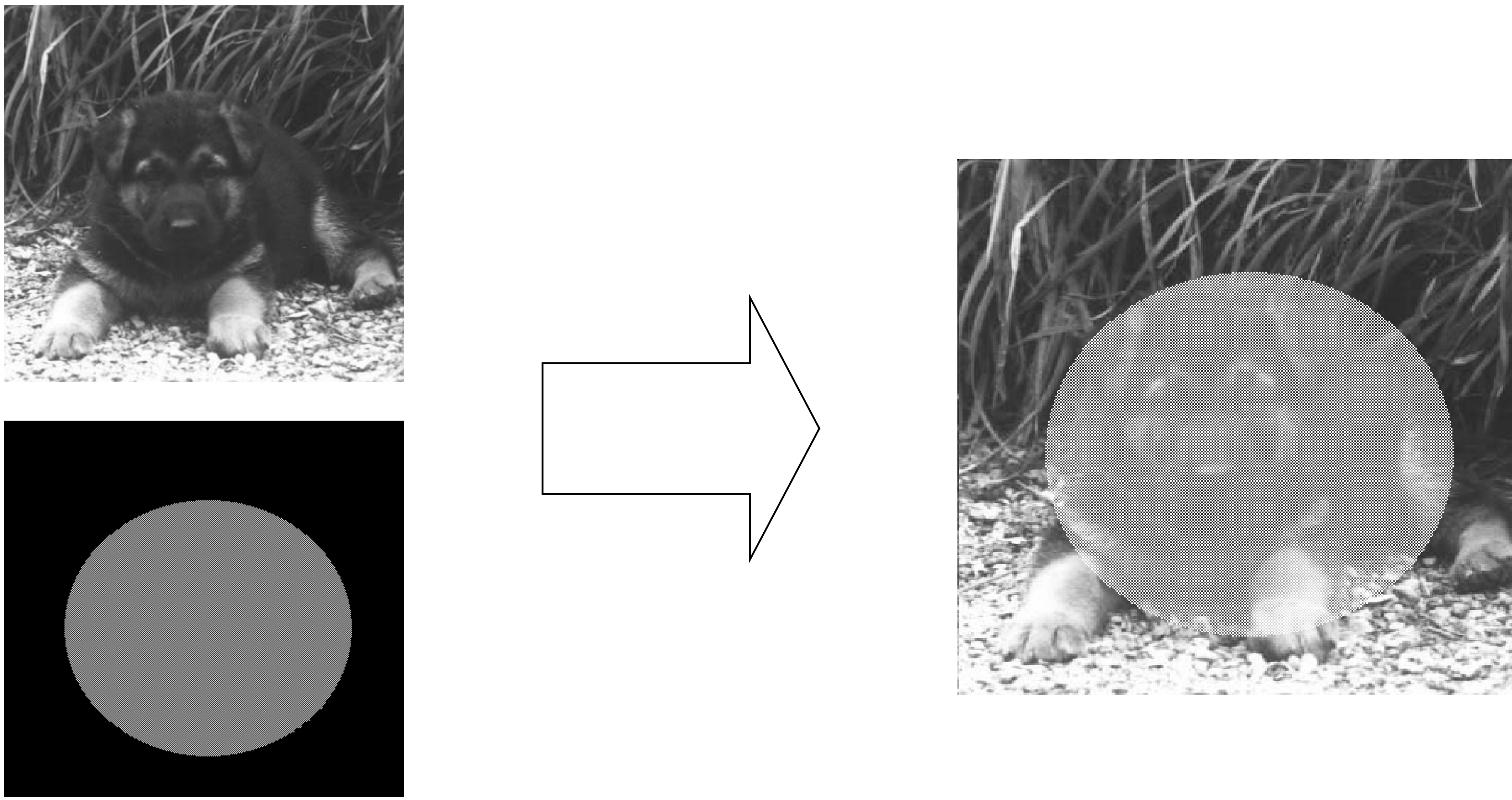
X-ray brain image
with contrast agent injected

Digital subtraction angiogram
(DSA)

http://en.wikipedia.org/wiki/Digital_subtraction_angiography

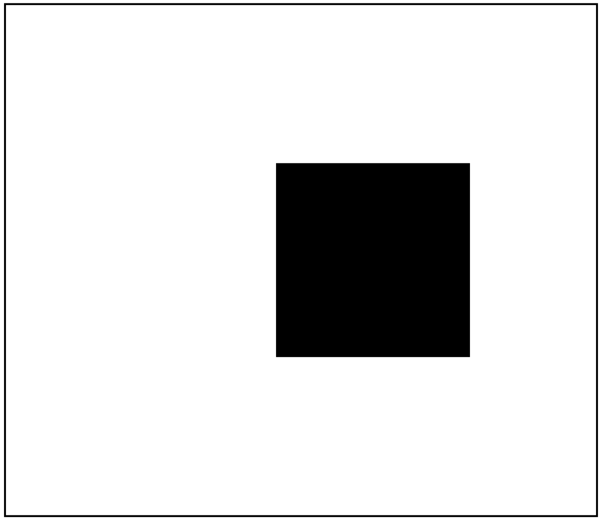
Image Operation

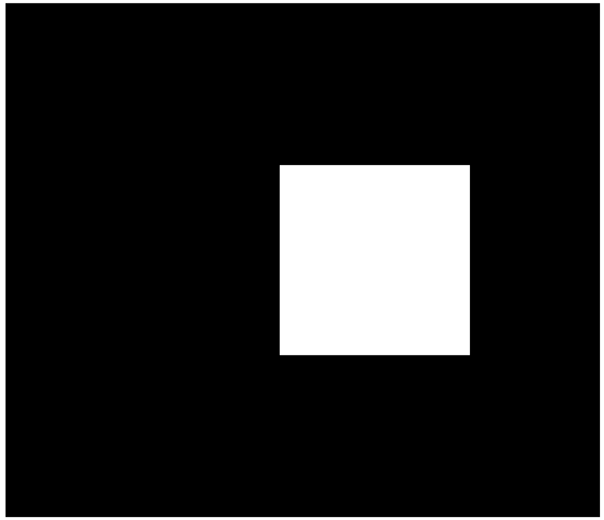
- Addition can blend two images



Logic Operators

- NOT



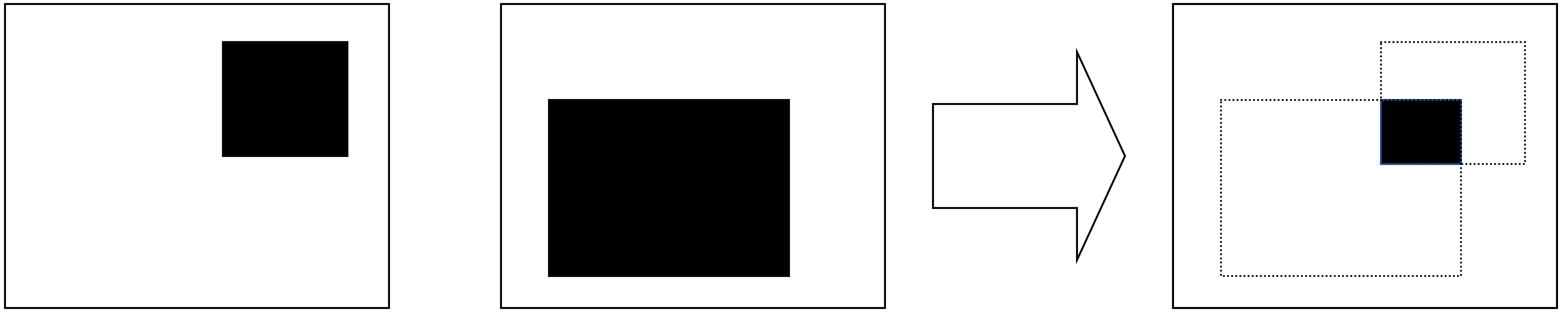


black = 1, white = 0.

Logic Operators

- AND

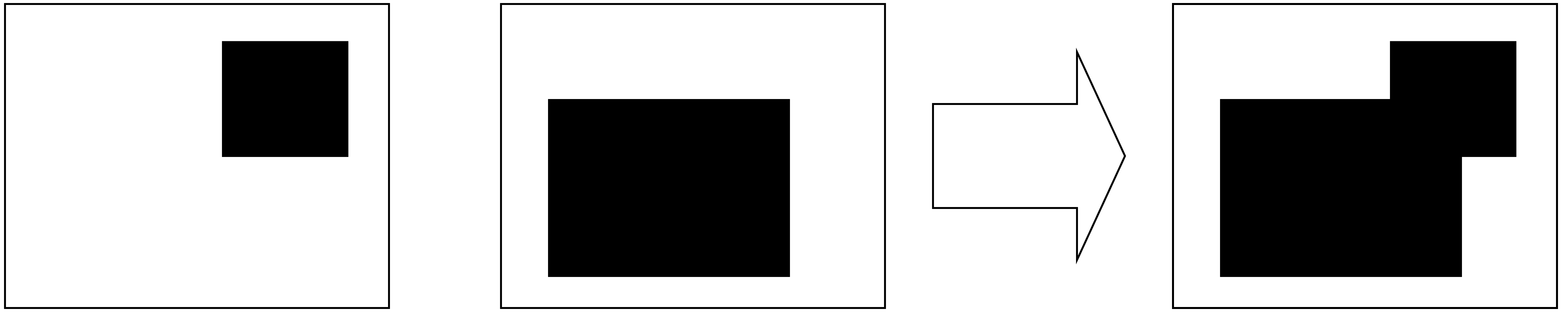
black = 1, white = 0.



Logic Operators

- OR

black = 1, white = 0.



- XOR?

Image Averaging

- If you can capture a scene or acquire an image repeatedly, then averaging all the observed images will give you a clear image with less noise.

- $g_i(x,y)$ = i^{th} observed noisy image, i = 1, ..., K, and K = total number of images.
- f(x,y) = 'true', original image
- $\eta(x,y)$ = (*eta*) is zero mean, random Gaussian noise
- the final average image is given by

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

Image Averaging

- It follows that

Expected value of the average image

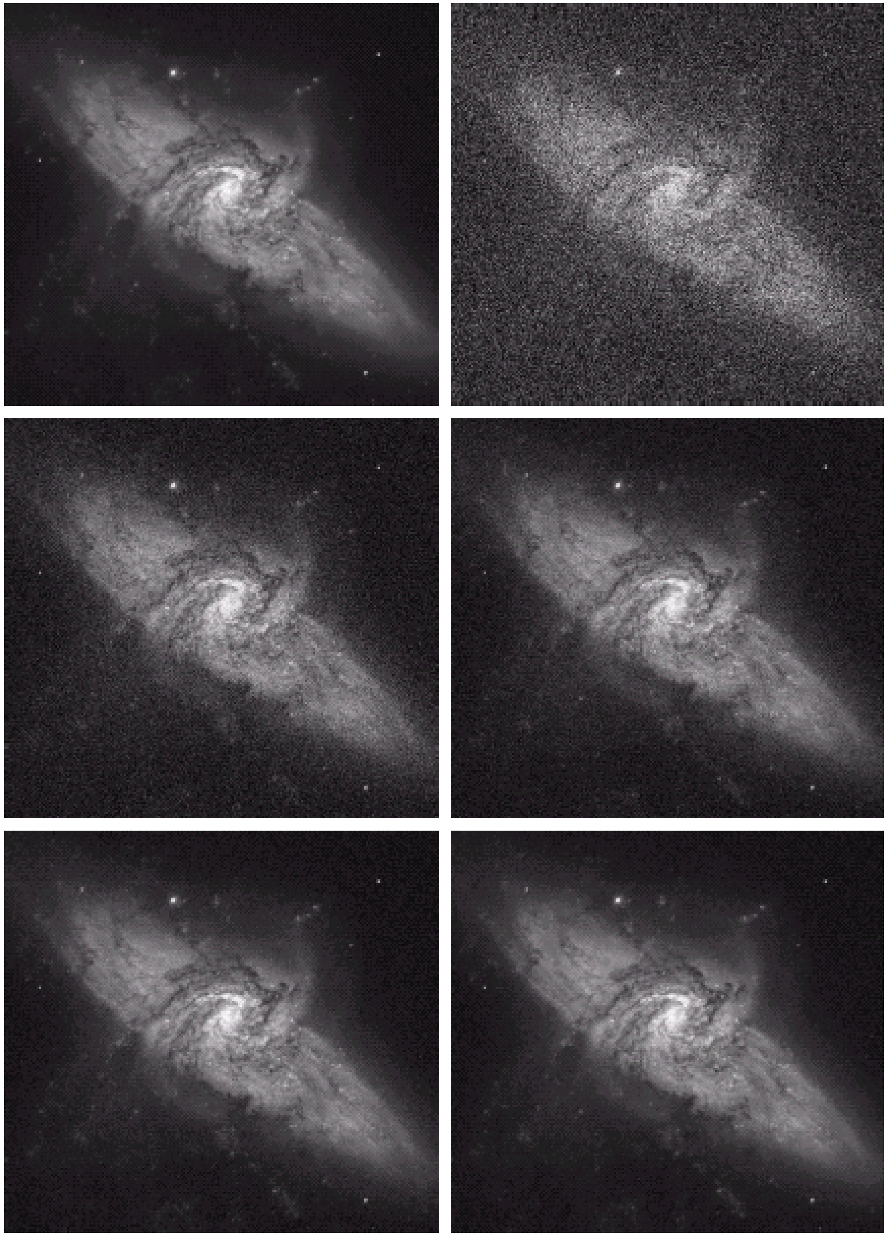
$$E\left\{\overline{g}(x,y)\right\} = f(x,y)$$

Variance of the average image

$$\sigma^2_{\overline{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)}$$

- If K is sufficiently large, then the average image will be very close to the true, original image, and the variance (noise) will be small.

Example



a	b
c	d
e	f

FIGURE 3.30 (a) Image of Galaxy Pair NGC 3314. (b) Image corrupted by additive Gaussian noise with zero mean and a standard deviation of 64 gray levels. (c)–(f) Results of averaging $K = 8, 16, 64$, and 128 noisy images. (Original image courtesy of NASA.)

More examples

1. Reducing noise by image averaging
http://www.school-of-digital-photography.com/2014/03/reducing-noise-by-image-averaging.html

2. The best noise reduction technique
http://ezbackgrounds.com/blog/noise-reduction-averaging-photoshop.php

3. Noise reduction by image averaging
http://www.cambridgeincolour.com/tutorials/image-averaging-noise.htm

Spatial Filtering using a Mask

- Neighborhood operators

mask/filter

z1	z2	z3
z4	z5	z6
z7	z8	z9

w1	w2	w3
w4	w5	w6
w7	w8	w9

Response of a linear mask/filter, R,

$$R = (w1z1 + w2z2 + w3z3 \ldots + w9z9) = \sum_{i=1}^{9} wizi$$

f(x,y) is centered around "z5"

so g(x,y) = R =filter*f(x,y), *=convolution symbol

Response of a *linear mask*

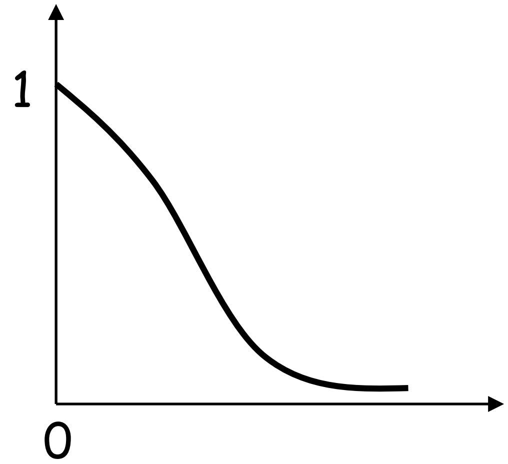
$$R = w1z1 + w2z2 + w3z3 \ldots + w9z9$$

where w_i are mask coefficients (weights) and z_i are pixel intensities.

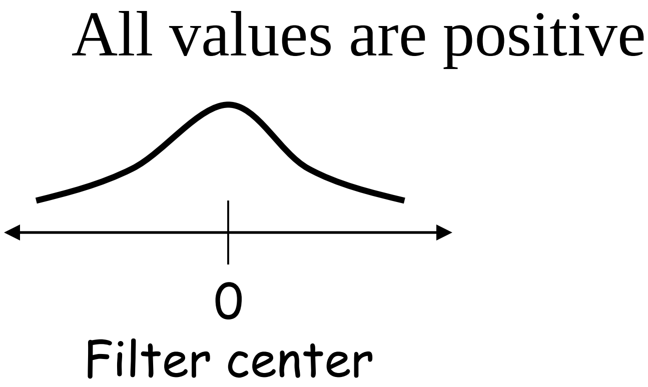
Smoothing Filters

- Smoothing Filters
  - blurring
    - "pre-processing"
      - removal of small details before object extraction
  - noise reduction
    - removal of noise in an image
- Often called "Low-Pass" Filters
  - Filter lets low-frequencies pass
  - Stops high-frequencies

Low-pass spatial filtering



Frequency Domain



Spatial Domain

Low-pass spatial filtering

w1	w2	w3
w4	w5	w6
w7	w8	w9

- Only requirement for a low-pass filter is that w_i be positive

- Note that the result can be larger than the valid output range(L-1)

- Can pre-scale the filter

$$\text{scale\_factor} = \left(\sum w_i\right)^{-1}$$

Low-pass spatial filter

$$\frac{1}{9} * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Average Filter

Extends to larger filters

$$1/9 * \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$1/81 * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Examples

original

n=5 (nxn mask)

n=15 (nxn mask)

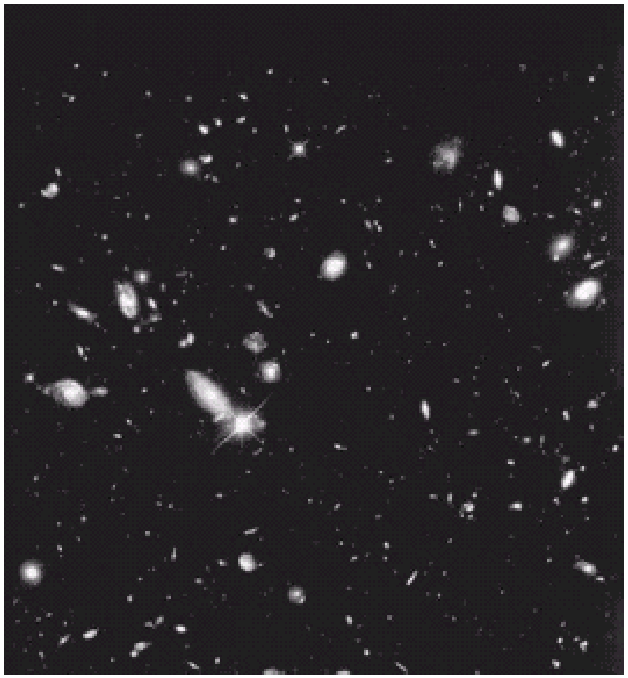
n=25 (nxn mask)

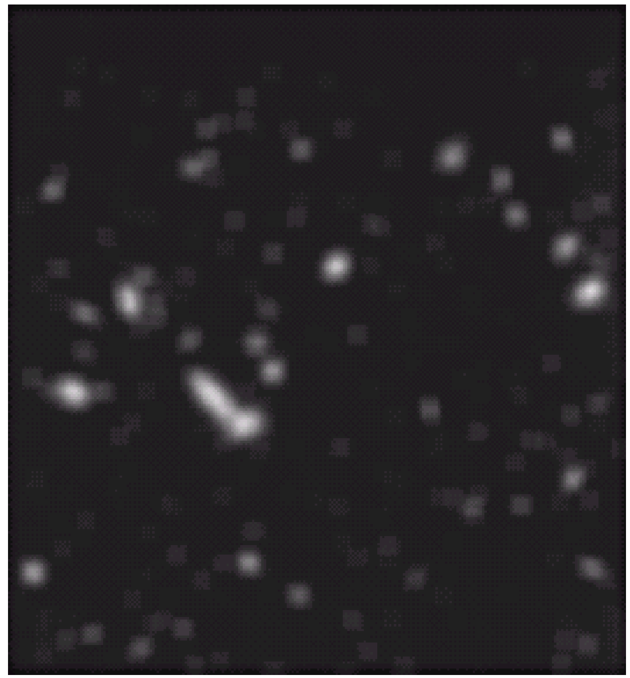
Example

Original image

Smooth image
15x15 averaging filter

thresholded image

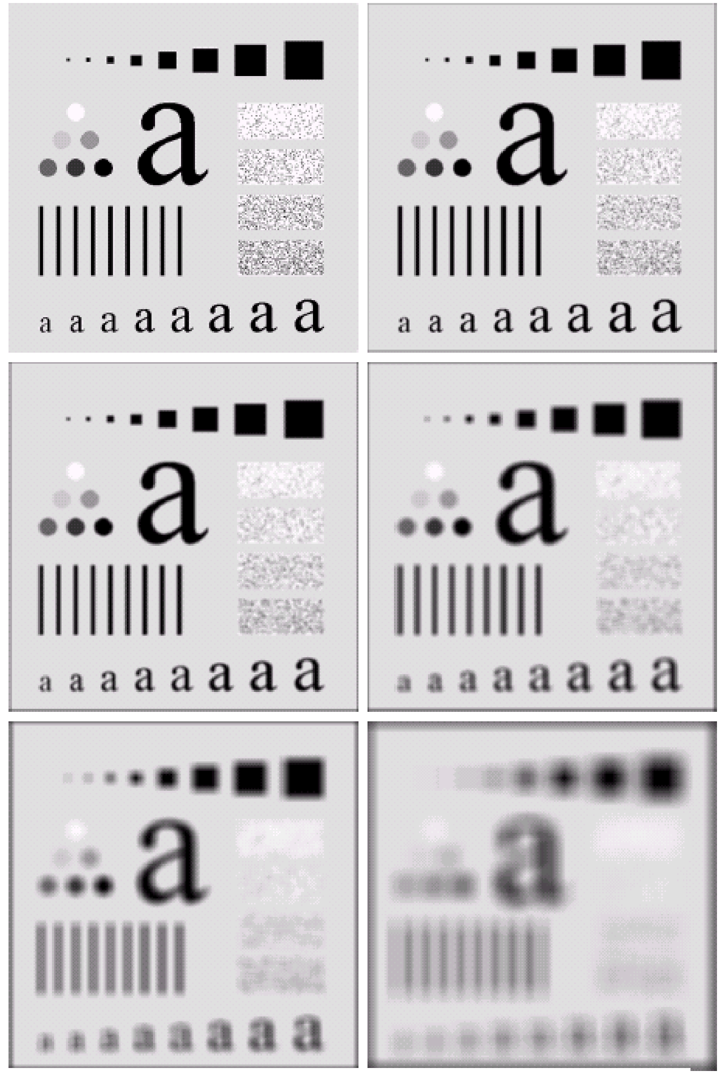




a b c

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Examples



a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Other arrangements

1	1	1
1	2	1
1	1	1

1	1	1	1	1
1	2	3	2	1
1	3	4	3	1
1	2	3	2	1
1	1	1	1	1

Remember that the resulting gray-levels may be out of the range of the original image.

Linear vs. Non-linear

- Linear Filters
  - Linear operation
  - Have corresponding frequency domain filter

- Non-linear Filters
  - Examine neighbors using various orderings
    - Often use Rank or Order Statistics

Median Filter

- Very popular non-linear filter
- Find the median of the window
- Preserves edges
- Removes impulse noise, avoids excessive smoothing

2	3	8
3	4	10
4	2	9

pixel values about (x,y)
window 3x3

neighbor sort = {2,2,3,3,$\boxed{4}$,4,8,9,10}

f(x,y) = median

Examples

Original

(a)

Noisy Image

(b)

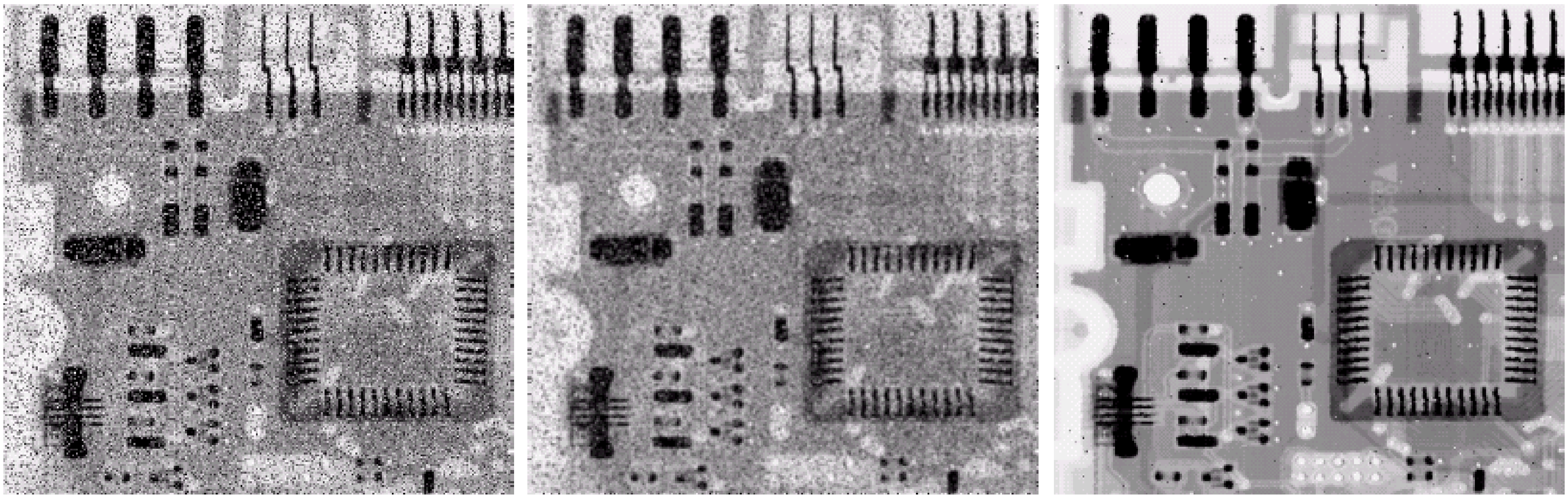
Low-Pass Filter

(c)

Median Filter

(d)

Examples



a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Median Filter

- Often referred to as “de-speckle operation”

- It converges
  - That is, if you perform it over and over for many times, eventually the image will not change.

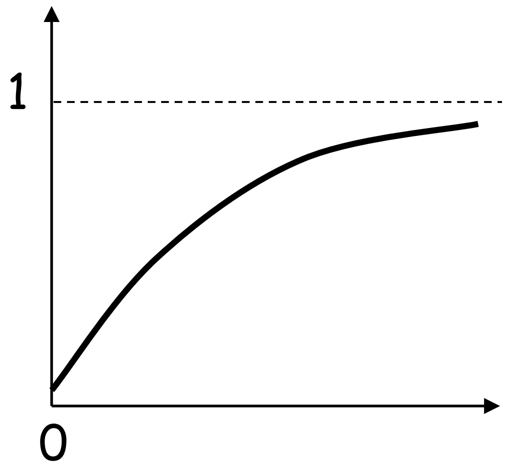
Iterative Smoothing

- Local Averaging
  - Converges to an image with constant intensity
- Median Filter
  - Converges to an image invariant to the filter

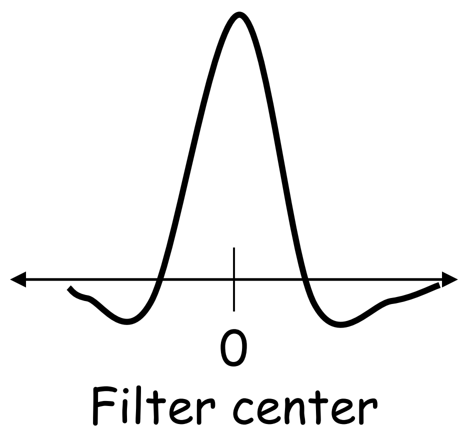
Sharpening

- Objectives of sharpening
  - Highlight fine details in an image
  - Enhance details that have been blurred
  - We can think of this as high-pass filtering
    - Letting high frequencies pass
    - Removing low frequencies

Sharpening



Values can be either positive or negative



Frequency Domain

Spatial Domain

Sharpening spatial filter

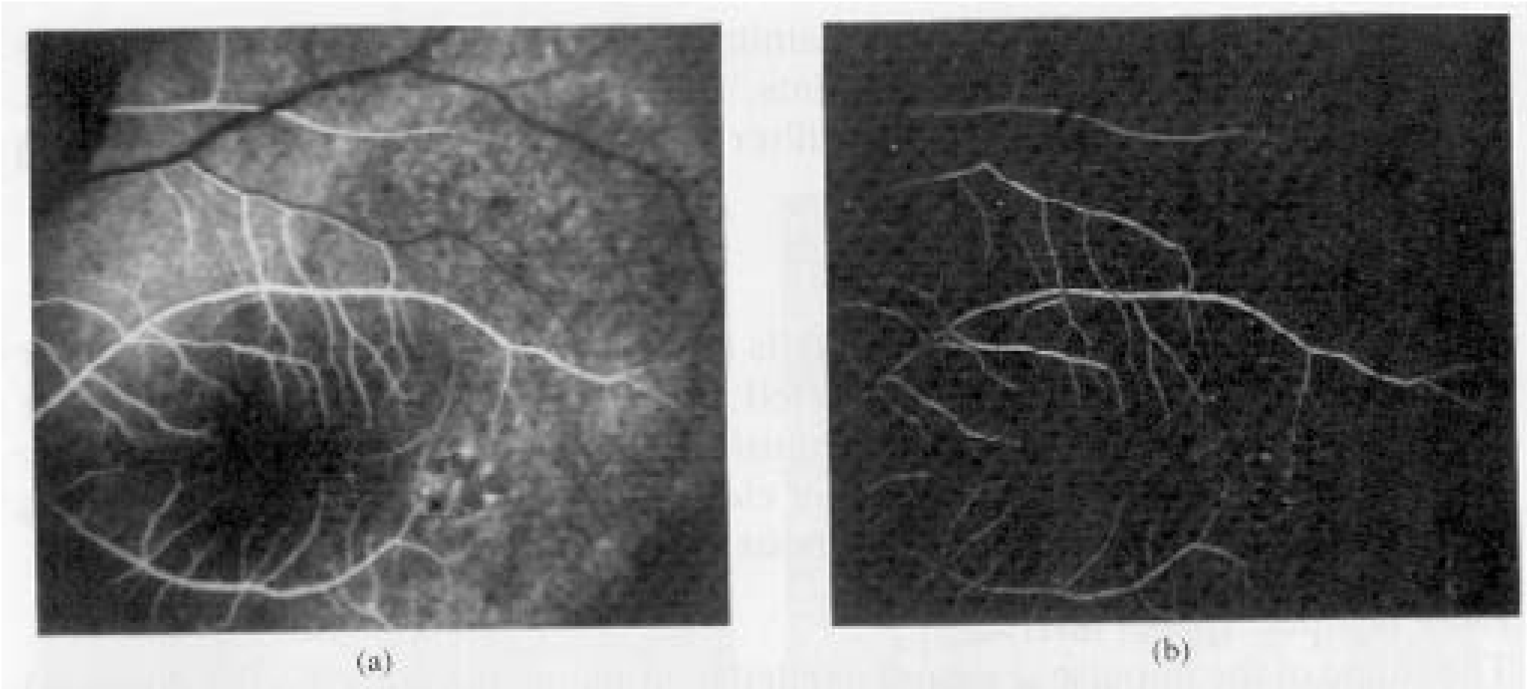
$$\frac{1}{9} * \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

- Positive coefficients near its center
- Negative coefficients near the outer periphery
- Note, the sum of the coefficients is zero
- Thus, when the mask is applied over an area of constant intensity, the result is zero
- Output = 0 if the intensity values are constant.

Sharpening spatial filter

- The results may be negative.
- You'll need to scale and/or clip so that the gray levels of the result span [0, L-1].

Example



Original **High-Pass**

First Order Derivatives

- Gradient
- Function of 2 variables x, y

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

First Order Derivatives

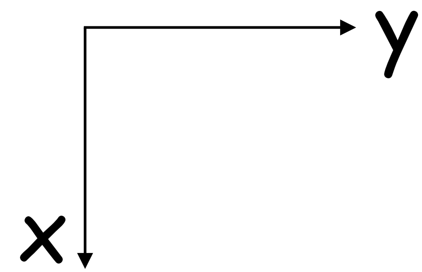
- For each (x,y) you are storing two values:

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- Often have two images to represent this
- X-Gradient and Y-Gradient
  – Can be computed independently

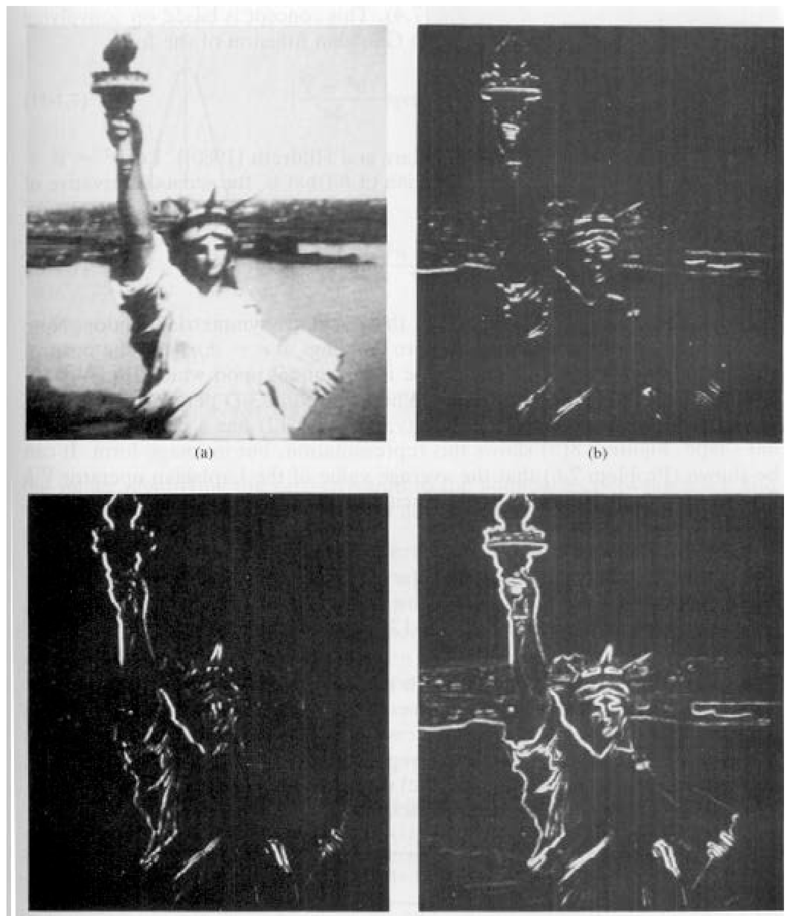
First Order Derivatives

- X-Gradient and Y-Gradient



Original

$|\partial f/\partial x|$



$|\partial f/\partial y|$

mag(∇f)

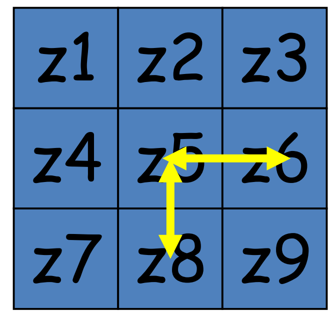
Gradient

- Gradient Magnitude

$$\mathrm{mag}(\nabla f) = \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2}$$

Basic Derivative

- Consider the pixels

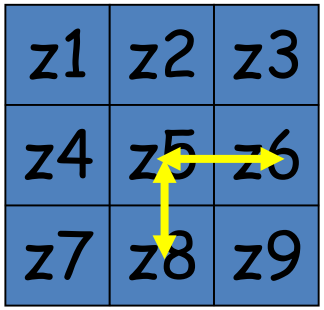


- $\mathrm{mag}(\nabla \mathrm{f})$ at z_5 can be computed

$$\mathrm{mag}(\nabla f) \approx [(z_5 - z_6)^2 + (z_5 - z_8)^2]^{1/2}$$

Basic Derivative

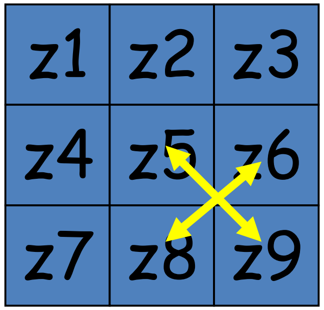
- Consider the pixels



- $\mathrm{mag}(\nabla \mathrm{f})$ at z_5 can be computed quicker

$$\mathrm{mag}(\nabla f) \approx | z_5 - z_6 | \; + | z_5 - z_8 |$$

Basic Derivative



- $\text{mag}(\nabla \text{f})$ sometimes is computed using the "cross" difference

$$\text{mag}(\nabla f) \approx |z_6 - z_8| + |z_5 - z_9|$$

Even Sized Masks

$$\text{mag}(\nabla f) \approx |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| +$$
$$|(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

- Difference between first and third rows (∂f/∂x)
- Difference between first and third columns (∂f/∂y)

Even sized mask

- Prewitt operator

z1	z2	z3
z4	z5	z6
z7	z8	z9

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

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

$$\text{mag}(\nabla f) \approx |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| +$$
$$|(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)|$$

Even sized mask

- Sobel operator

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

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

- Weights closer neighbor a little more