

Lecture 2. Intensity Transformation and Spatial Filtering



Geometric Spatial Transformations

- ▶ Geometric transformation (rubber-sheet transformation)
 - A **spatial transformation** of coordinates

$$(x, y) = T\{(v, w)\}$$

- **intensity interpolation** that assigns intensity values to the spatially transformed pixels.

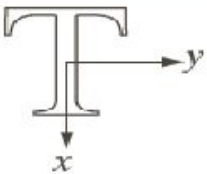
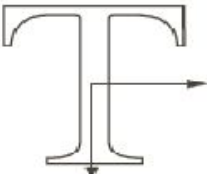
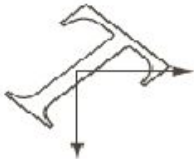
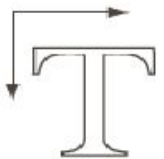
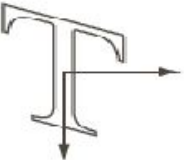
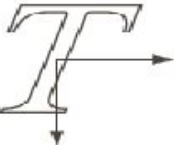
- ▶ Affine transform

$$\begin{bmatrix} x & y & 1 \end{bmatrix} = \begin{bmatrix} v & w & 1 \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

Application:

1. Rescale the graphics content when displaying it on a desktop vs mobile.
2. Artificially generate more data is to randomly apply an affine transformation to the input data. A technique also known as **augmentation**

TABLE 2.2
Affine transformations based on Eq. (2.6.–23).

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= v \\ y &= w \end{aligned}$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= c_x v \\ y &= c_y w \end{aligned}$	
Rotation	$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= v \cos \theta - w \sin \theta \\ y &= v \sin \theta + w \cos \theta \end{aligned}$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$\begin{aligned} x &= v + t_x \\ y &= w + t_y \end{aligned}$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= v + s_v w \\ y &= w \end{aligned}$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{aligned} x &= v \\ y &= s_h v + w \end{aligned}$	

Intensity Assignment

▶ Forward Mapping

$$(x, y) = T\{(v, w)\}$$

It's possible that two or more pixels can be transformed to the same location in the output image.

▶ Inverse Mapping

$$(v, w) = T^{-1}\{(x, y)\}$$

$$(x, y) = T[(w, z)] = (w/2, z/2)$$

$$(w, z) = T^{-1}[(x, y)] = (2x, 2y)$$

The nearest input pixels to determine the intensity of the output pixel value.

Inverse mappings are more efficient to implement than forward mappings.

Intensity Transform: Spatial Domain vs. Transform Domain

► Spatial domain

image plane itself, directly process the intensity values of the image plane

► Transform domain

process the transform coefficients, not directly process the intensity values of the image plane

Spatial Domain Process

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

$f(x, y)$: input image

$g(x, y)$: output image

T : an operator on f defined over
a neighborhood of point (x, y)

Spatial Domain Process

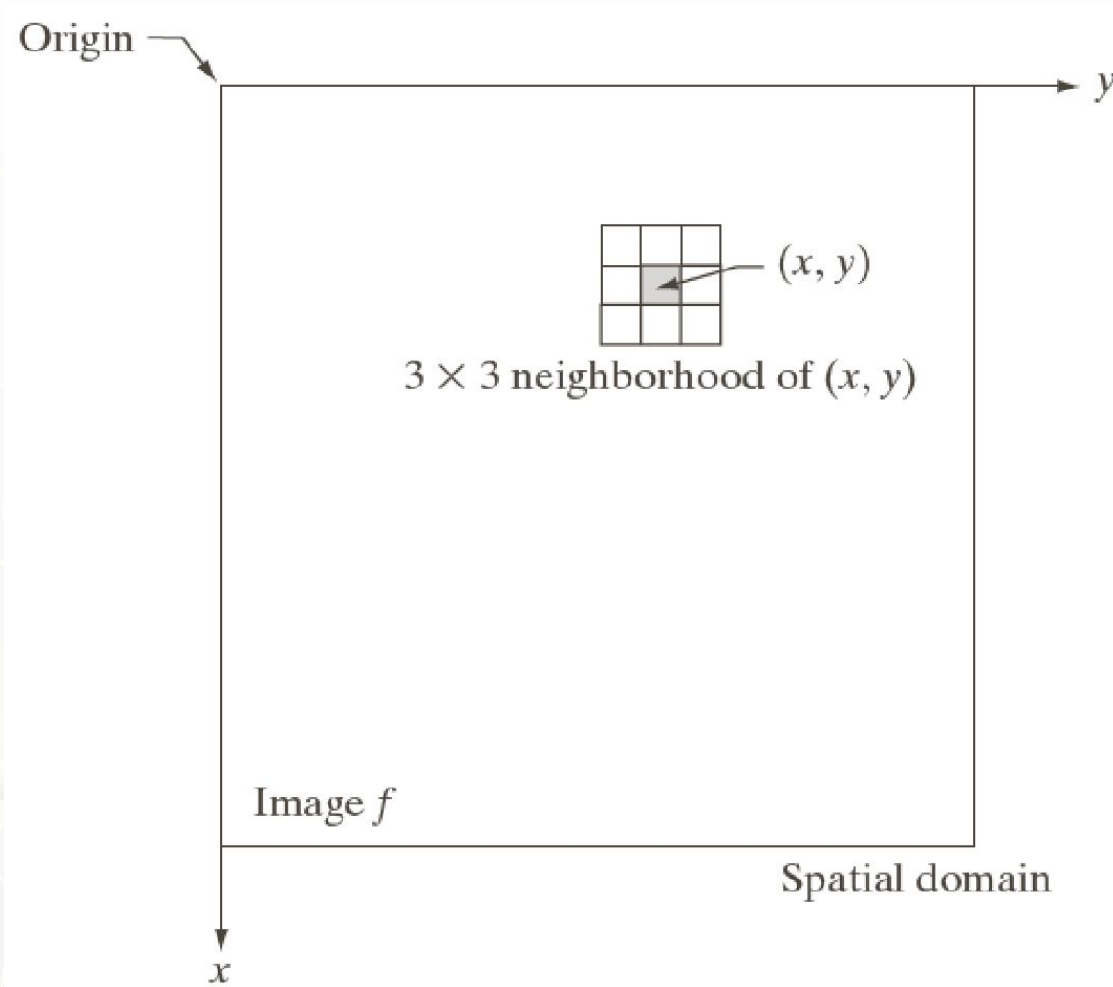


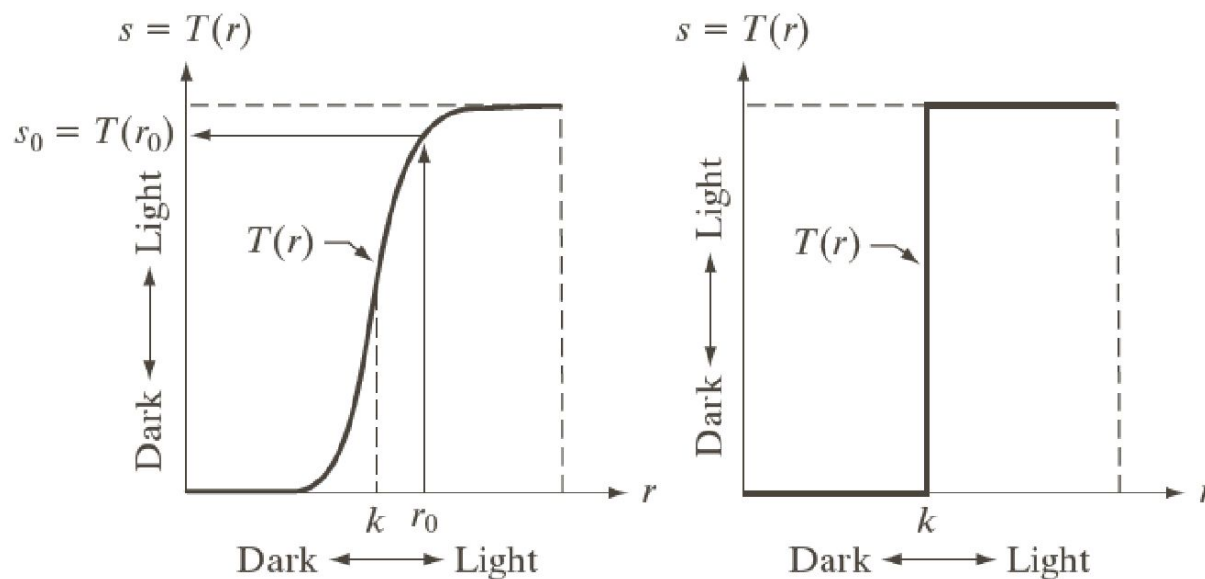
FIGURE 3.1

A 3×3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

Spatial Domain Process

Intensity transformation function

$$s = T(r)$$



a b

FIGURE 3.2

Intensity transformation functions.

(a) Contrast-stretching function.

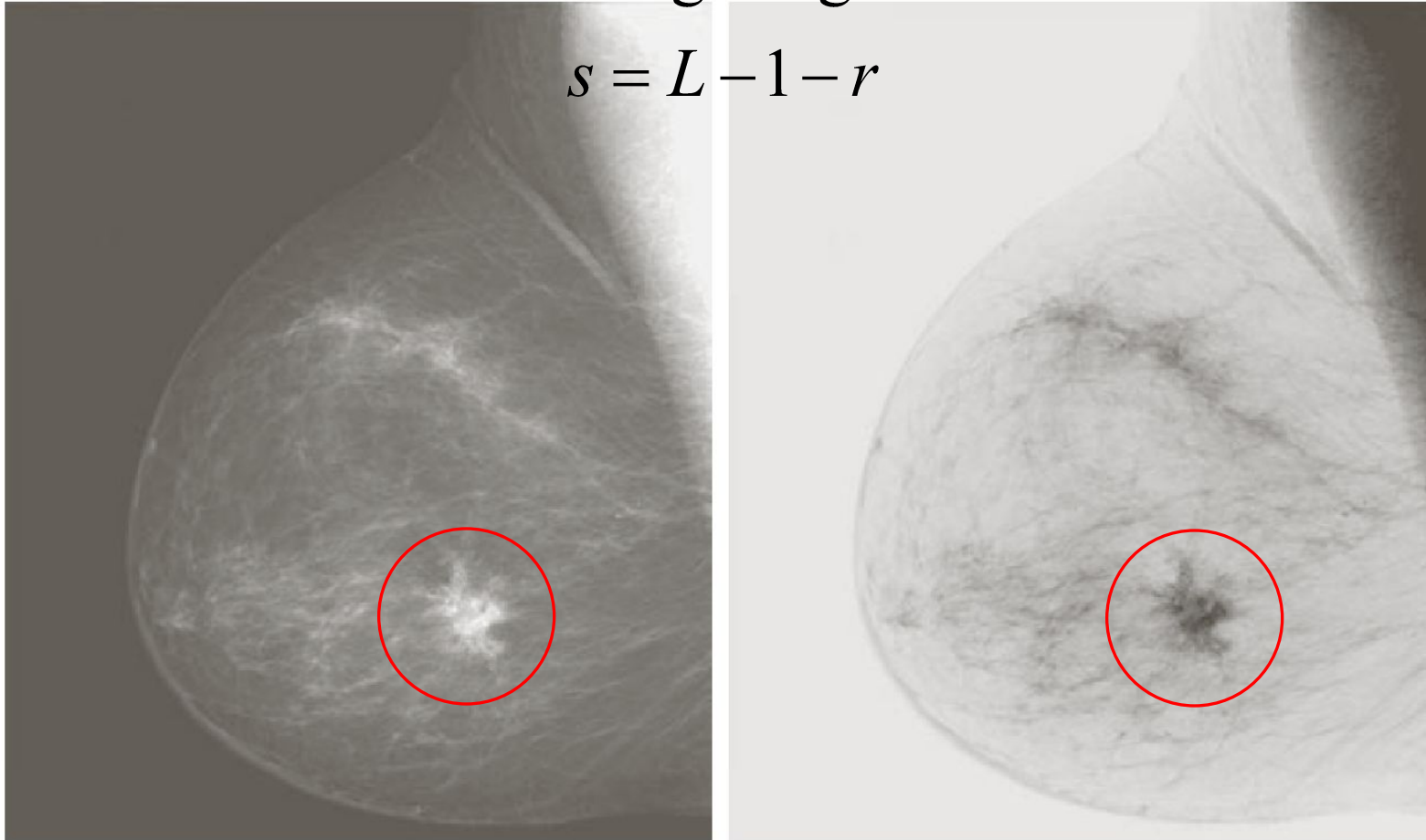
(b) Thresholding function.

In contrast stretching, if $r_1=s_1$ and $r_2=s_2$, then transformation is a linear function that produces no changes in gray levels

Example: Image Negatives

Image negatives

$$s = L - 1 - r$$



a b

FIGURE 3.4

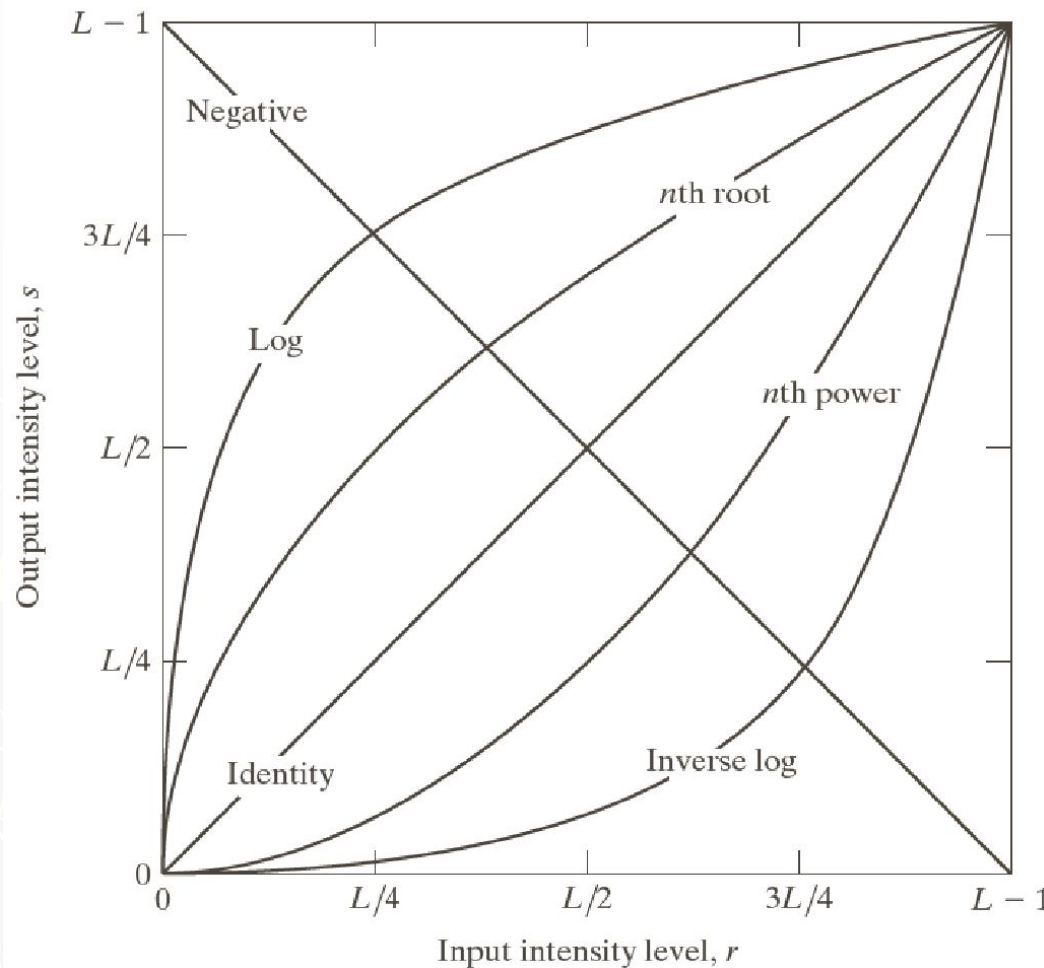
(a) Original digital mammogram.

(b) Negative image obtained using the negative transformation in Eq. (3.2-1).

(Courtesy of G.E. Medical Systems.)

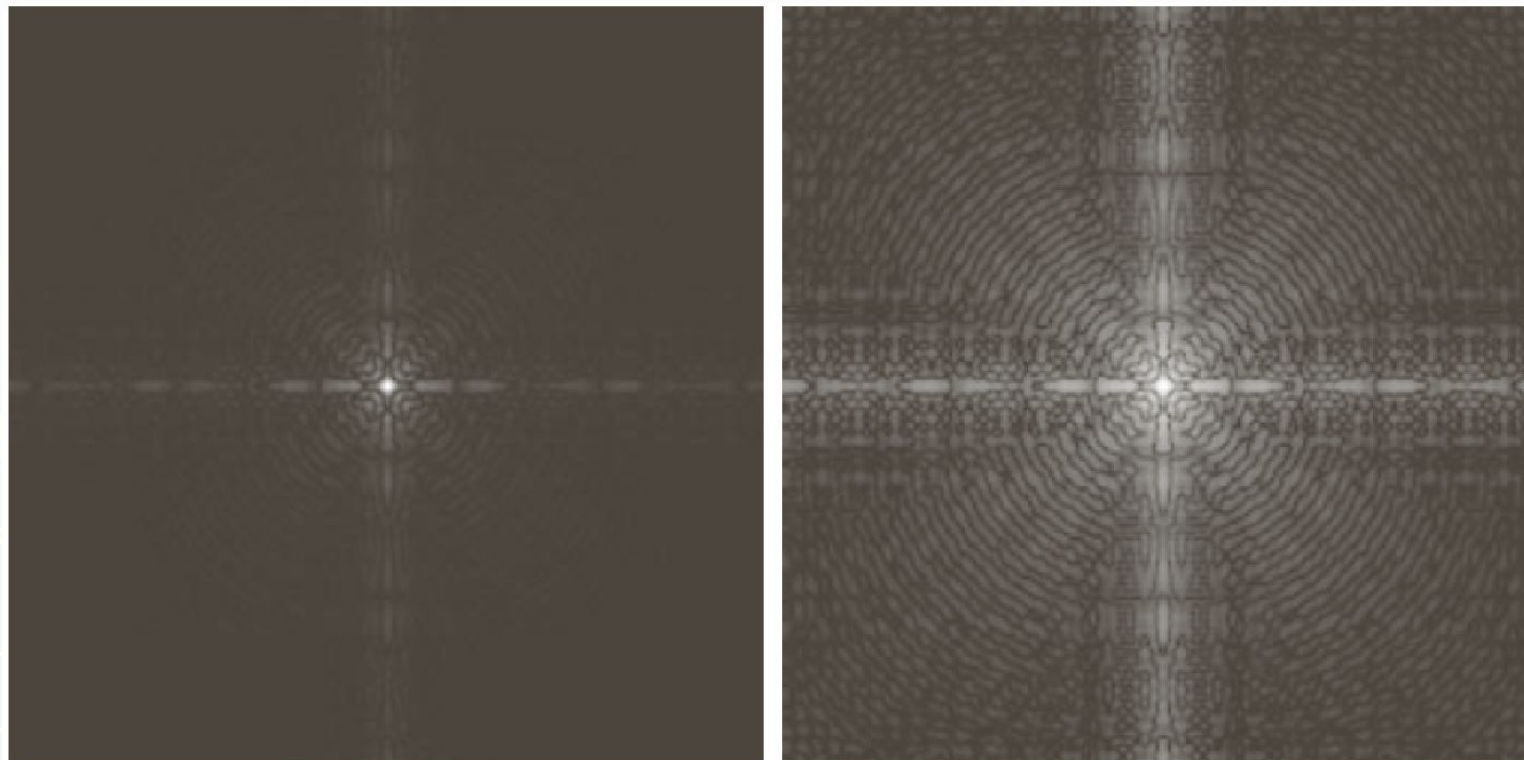
Small
lesion

Log Transformations



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

Example: Log Transformations



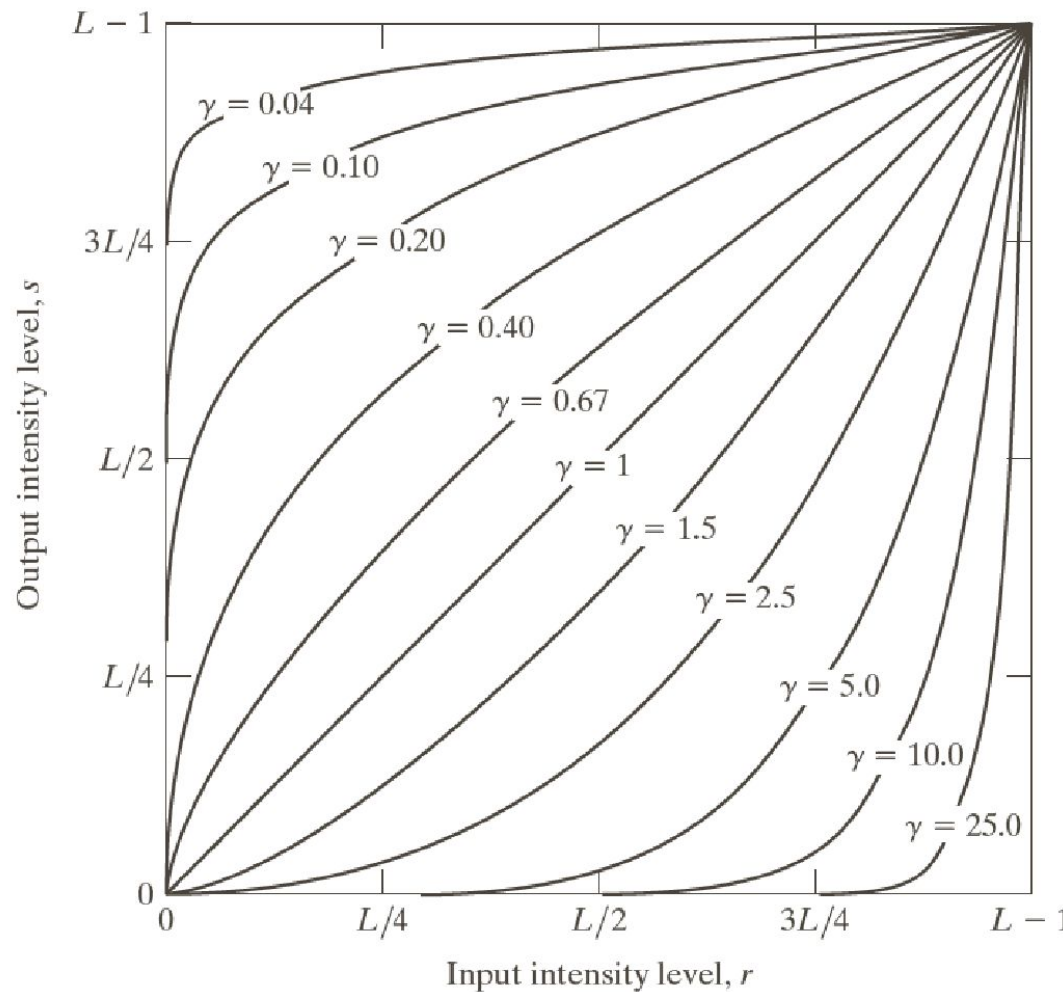
a b

FIGURE 3.5

(a) Fourier spectrum.

(b) Result of applying the log transformation in Eq. (3.2-2) with $c = 1$.

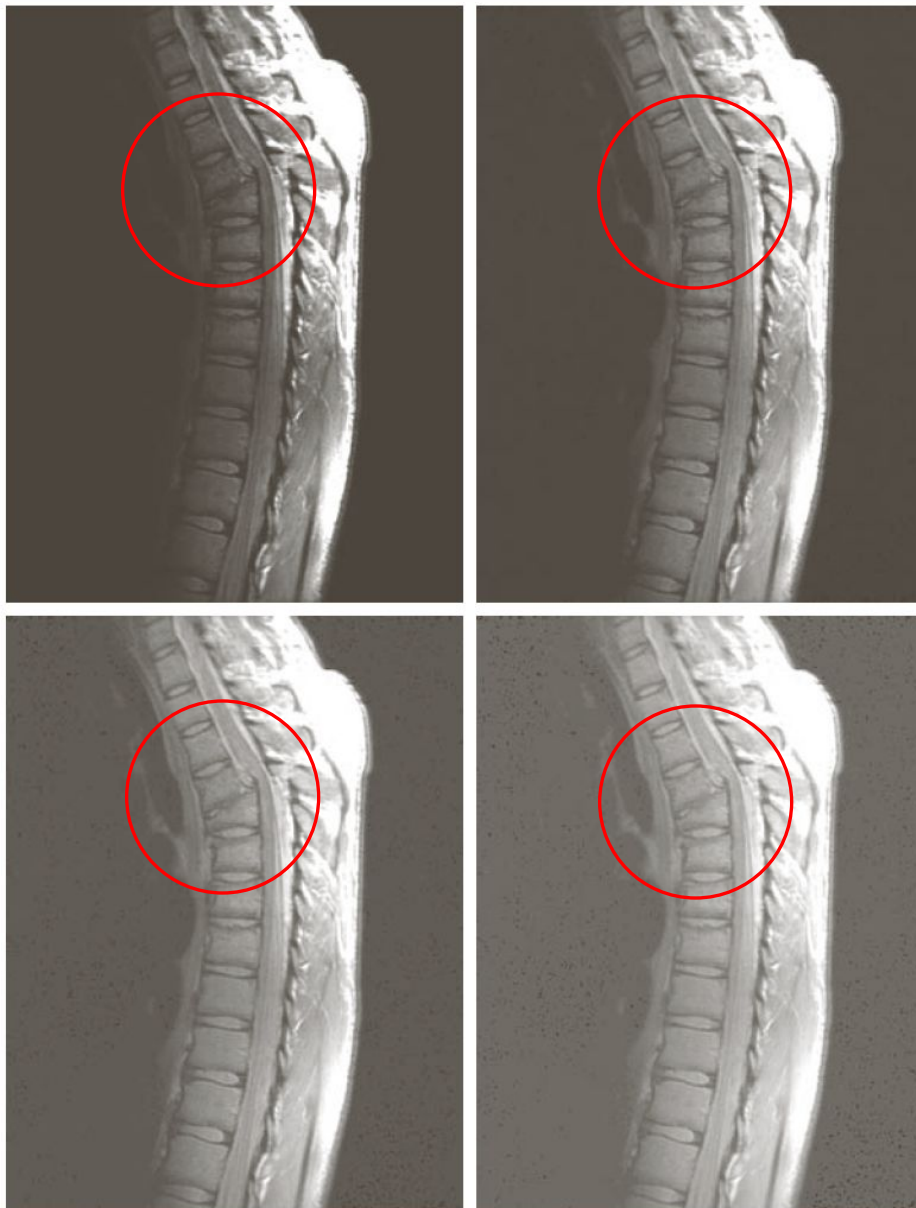
Power-Law (Gamma) Transformations



$$s = cr^\gamma$$

FIGURE 3.6 Plots of the equation $s = cr^\gamma$ for various values of γ ($c = 1$ in all cases). All curves were scaled to fit in the range shown.

Example: Gamma Transformations



a	b
c	d

FIGURE 3.8

(a) Magnetic resonance image (MRI) 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 courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

Histogram Processing

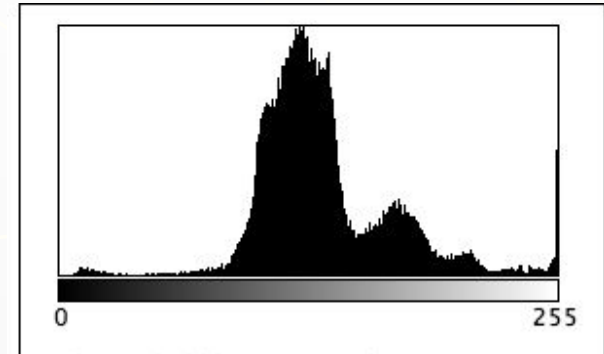
► Histogram Equalization



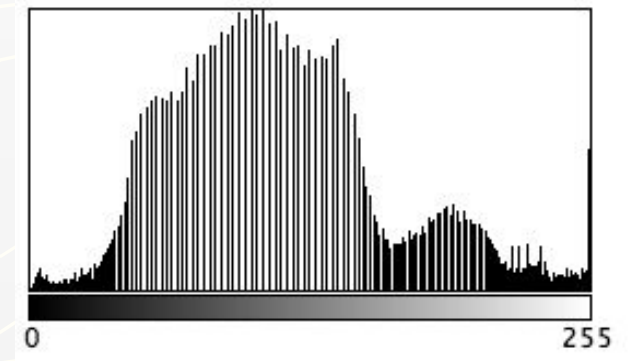
Example: Histogram Equalization

Histogram equalization yields an image whose pixels are (in theory) uniformly distributed among all gray levels.

► Original



► Modified



Example: Histogram Equalization

Suppose that a 3-bit image ($L=8$) of size 3×4 pixels ($MN = 12$) has the intensity distribution shown in following table.

Get the histogram equalization transformation function and give the $p_s(s_k)$ for each s_k .

A=

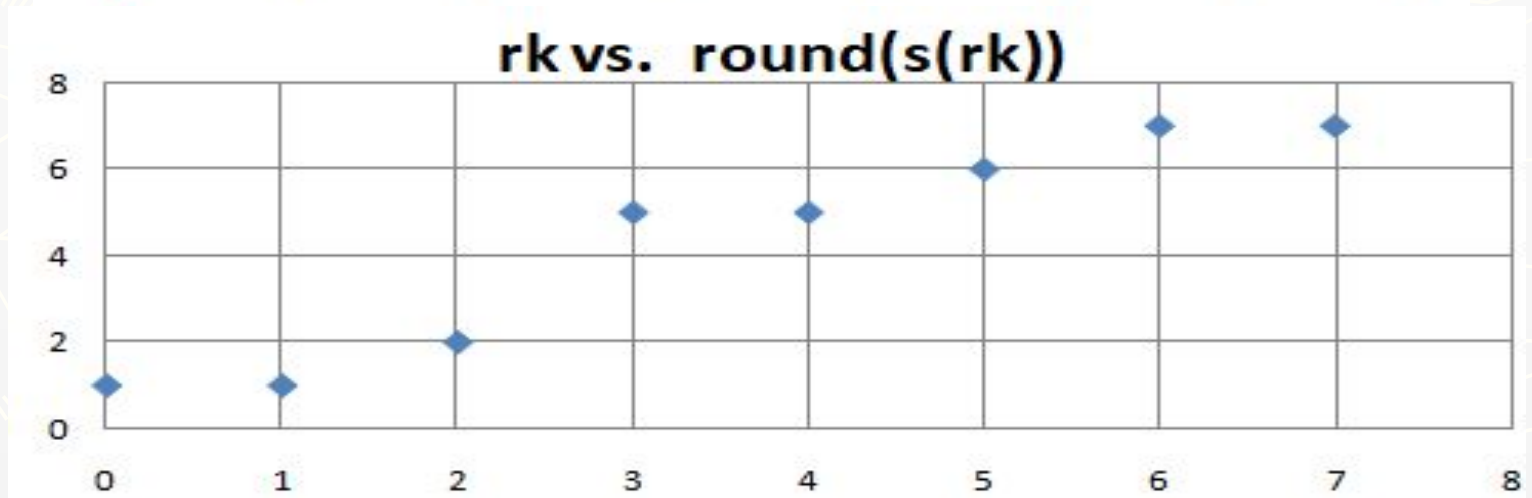
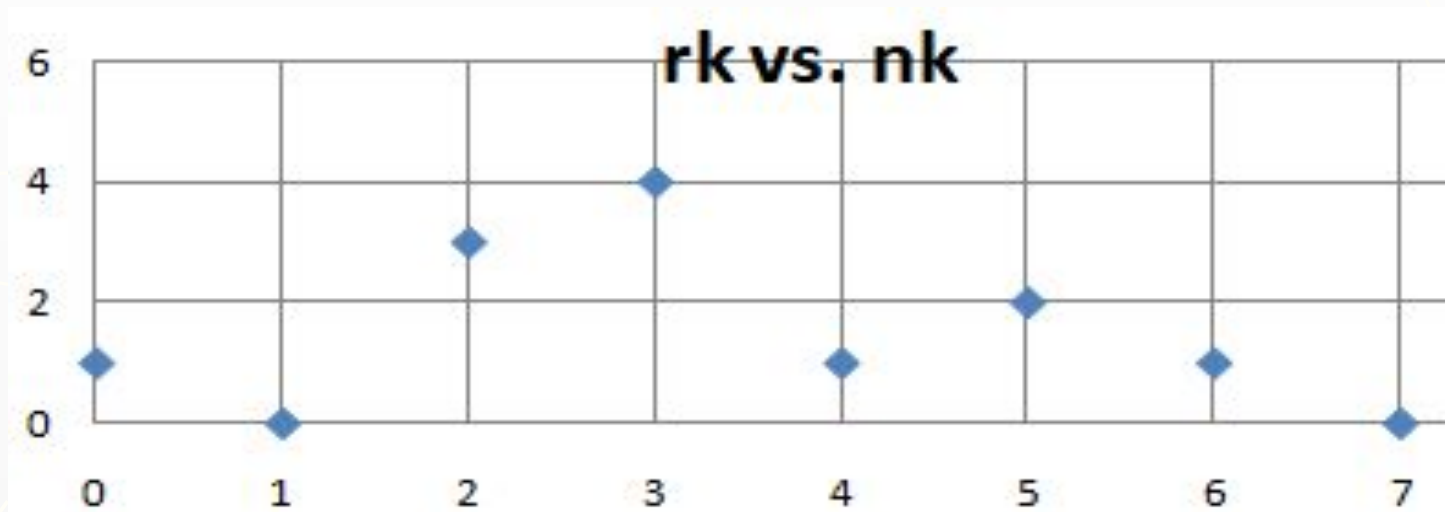
2	3	5
5	6	2
2	3	4
3	3	0

Equalized image look
up table
 $\text{img} = \text{cdf}[A]$

n= 12 rk=[0-7] gray level=L=8

rk	nk	$p(\text{rk})=\text{nk}/n$	$T(\text{rk})=7*P(\text{rk})$	$S(\text{rk})=\text{cdf}(T(\text{rk}))$	$\text{floor}(s(\text{rk}))$
0	1	0.08	0.58	0.58	0
1	0	0.00	0.00	0.58	0
2	3	0.25	1.75	2.33	2
3	4	0.33	2.33	4.67	4
4	1	0.08	0.58	5.25	5
5	2	0.17	1.17	6.42	6
6	1	0.08	0.58	7.00	7
7	0	0.00	0.00	7.00	7

Example: Histogram Equalization



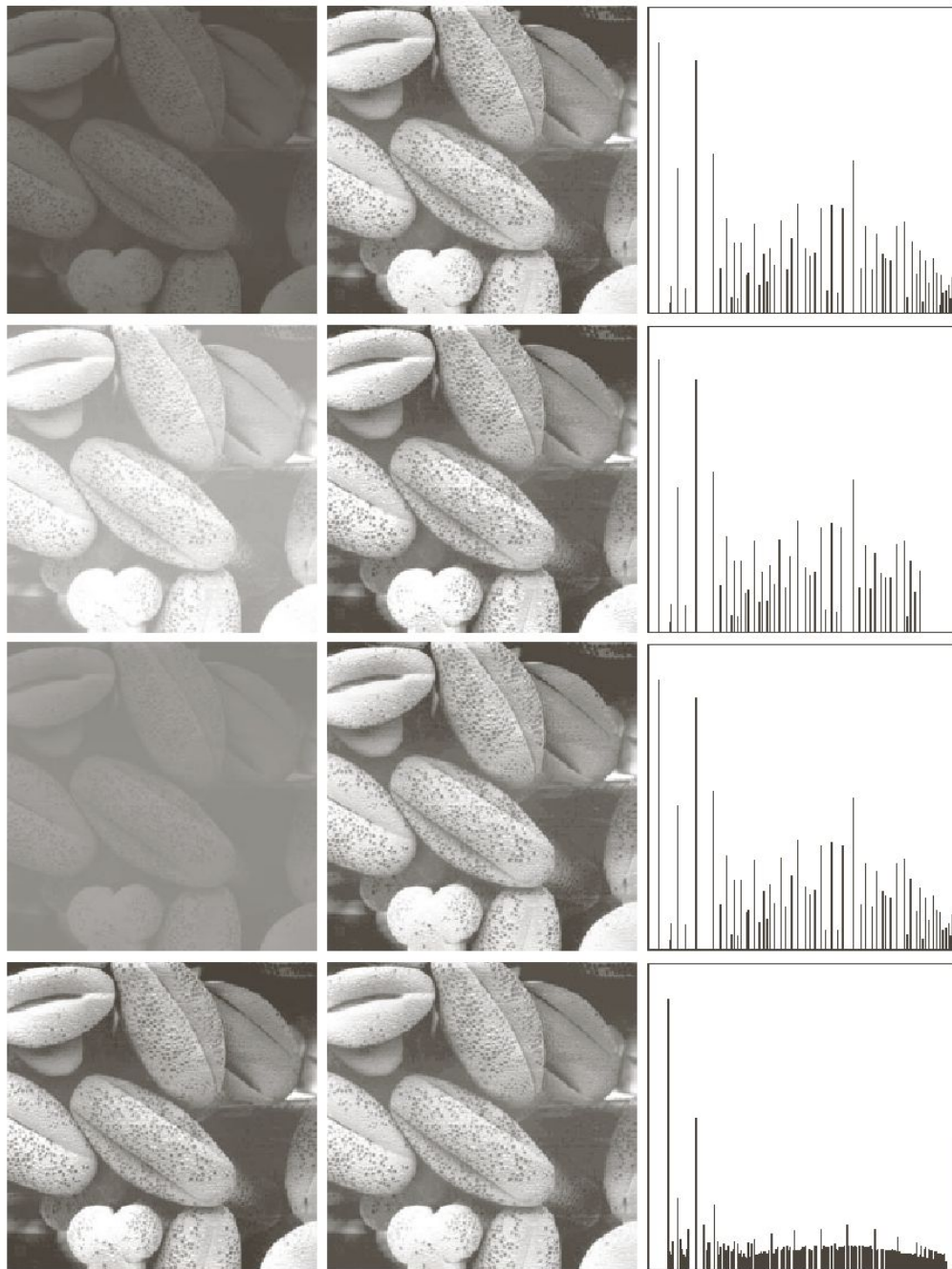


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

Question

1. Write python code to perform histogram equalization.



Spatial Filtering

A spatial filter consists of (a) **a neighborhood**, and (b) **a predefined operation**

Linear spatial filtering of an image of size MxN with a filter of size mnxn is given by the expression.

$w(x,y)$ is a filter, and $f(x,y)$ is the image

$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

Spatial Correlation

The correlation of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted as $w(x, y) \star f(x, y)$

$$w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

Spatial Convolution

The convolution of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted as $w(x, y) \star f(x, y)$

$$w(x, y) \star f(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x-s, y-t)$$

- Linear filtering of an image is accomplished through an operation called *convolution*.
- Convolution is a neighborhood operation in which each output pixel is the weighted sum of neighboring input pixels.
- The matrix of weights is called the *convolution kernel*, also known as the *filter*.
- A convolution kernel is a correlation kernel that has been rotated 180 degrees.

↖ Origin	$f(x, y)$					
	0	0	0	0	0	
	0	0	0	0	0	$w(x, y)$
	0	0	1	0	0	1 2 3
	0	0	0	0	0	4 5 6
	0	0	0	0	0	7 8 9

(a)

FIGURE 3.30
Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

Smoothing Spatial Filters

Smoothing filters are used for blurring and for noise reduction

Blurring is used in removal of small details and bridging of small gaps in lines or curves

Smoothing spatial filters include linear filters and nonlinear filters.

Spatial Smoothing Linear Filters

The general implementation for filtering an $M \times N$ image with a weighted averaging filter of size $m \times n$ is given

$$g(x, y) = \frac{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)}{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t)}$$

where $m = 2a + 1$, $n = 2b + 1$.

Two Smoothing Averaging Filter Masks

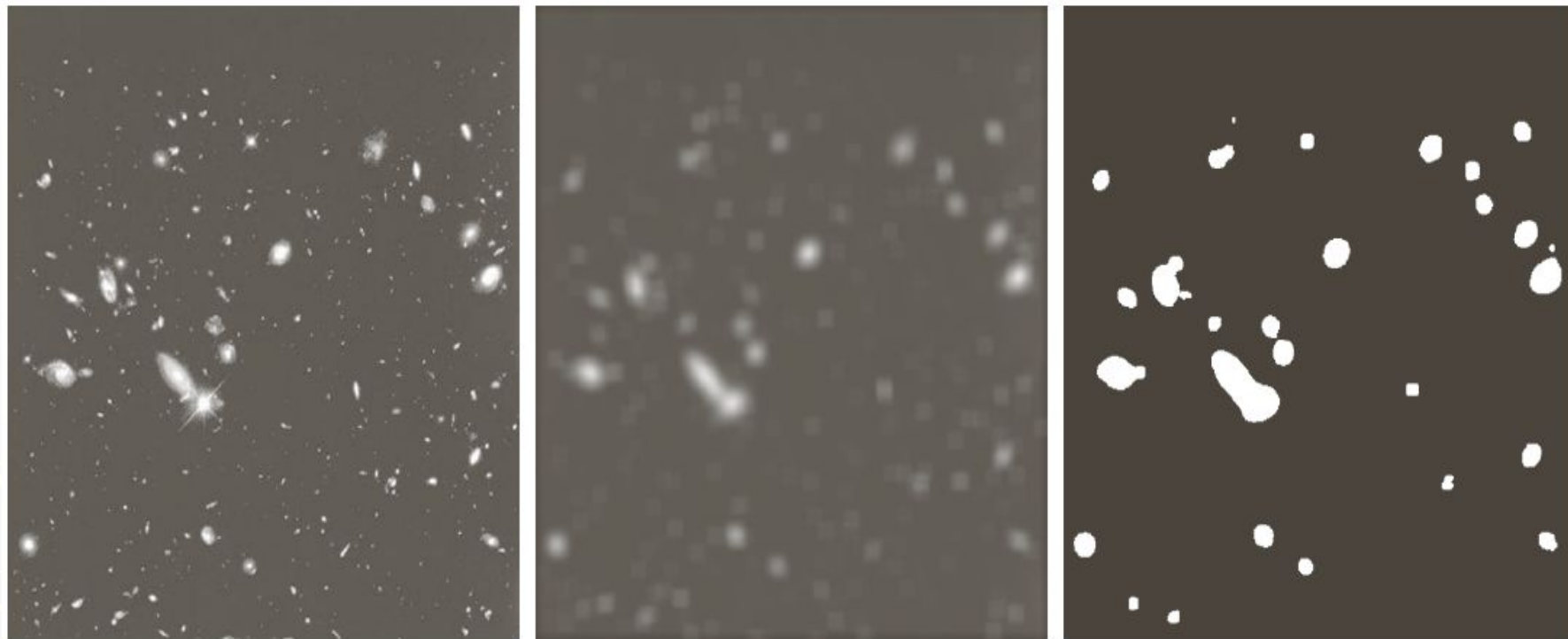
$\frac{1}{9} \times$	1	1	1
	1	1	1
	1	1	1

$\frac{1}{16} \times$	1	2	1
	2	4	2
	1	2	1

a b

FIGURE 3.32 Two 3×3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to 1 divided by the sum of the values of its coefficients, as is required to compute an average.

Example: Gross Representation of Objects



a b c

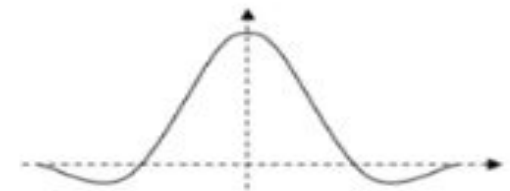
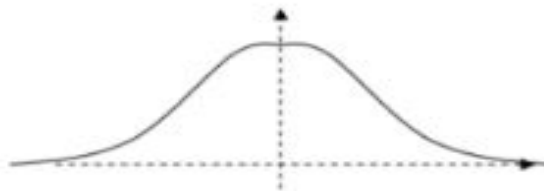
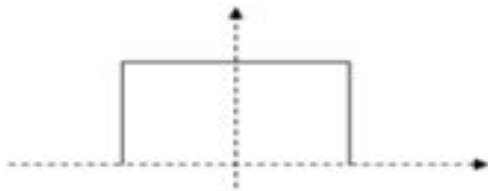
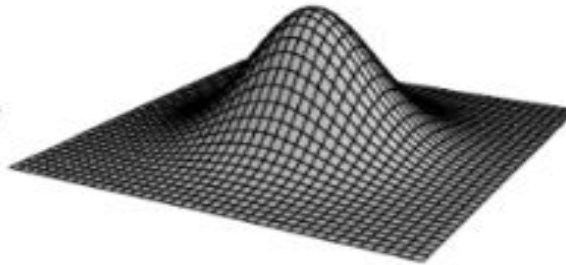
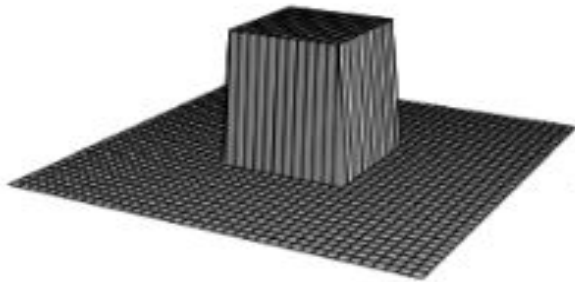
FIGURE 3.34 (a) Image of size 528×485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15×15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Order-statistic (Nonlinear) Filters

- Based on ordering (ranking) the pixels contained in the filter mask
- Replacing the value of the center pixel with the value determined by the ranking result

E.g., median filter, max filter, min filter

Example: filters



0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

(a)

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

(b)

0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

(c)

3D structure, 2D structure and example of filter (a) Box filter (b) Gaussian filter and (c) Laplace filter

Box filter (max,min,median) . All members of this filter are the same

Sharpening Spatial Filters

- ▶ Foundation
- ▶ Laplacian Operator
- ▶ Highboost Filtering
- ▶ Using First-Order Derivatives for Nonlinear Image Sharpening — The Gradient

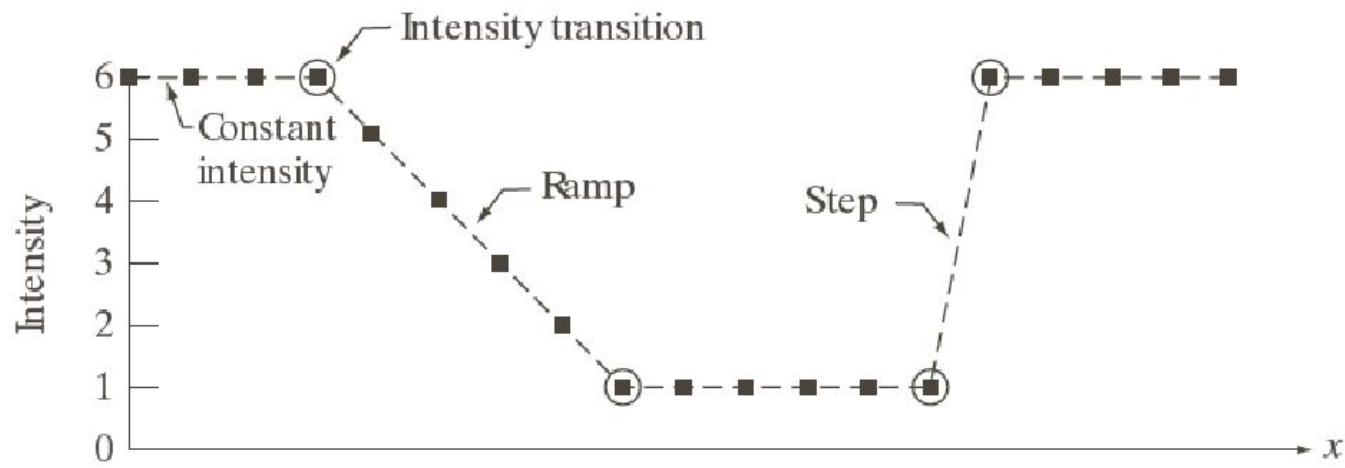
Sharpening Spatial Filters: Foundation

- ▶ The first-order derivative of a one-dimensional function $f(x)$ is the difference

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- ▶ The second-order derivative of $f(x)$ as the difference

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$



a
b
c

FIGURE 3.36 Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

Sharpening Spatial Filters: Laplace Operator

The second-order isotropic derivative operator is the Laplacian for a function (image) $f(x,y)$

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

Sharpening Spatial Filters: Laplace Operator

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1

0	-1	0	-1	-1	-1
-1	4	-1	-1	8	-1
0	-1	0	-1	-1	-1

a b
c d

FIGURE 3.37

(a) Filter mask used to implement Eq. (3.6-6).

(b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.

Sharpening Spatial Filters: Laplace Operator

Image sharpening in the way of using the Laplacian:

$$g(x, y) = f(x, y) + c \left[\nabla^2 f(x, y) \right]$$

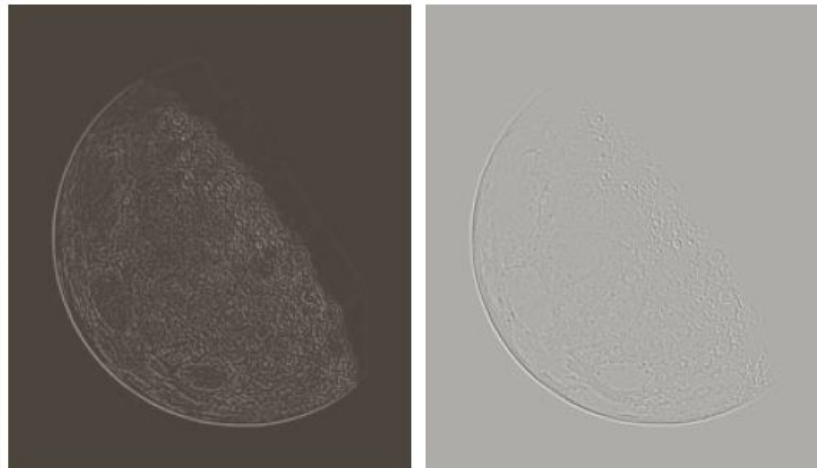
where,

$f(x, y)$ is input image,

$g(x, y)$ is sharpened images,

$c = -1$ if $\nabla^2 f(x, y)$ corresponding to Fig. 3.37(a) or (b)

and $c = 1$ if either of the other two filters is used.



a	
b	c
d	e

FIGURE 3.38

(a) Blurred image of the North Pole of the moon.

(b) Laplacian without scaling.

(c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a).

(e) Result of using the mask in Fig. 3.37(b).

(Original image courtesy of NASA.)

Unsharp Masking and Highboost Filtering

► Unsharp masking

Sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image
e.g., printing and publishing industry

► Steps

1. Blur the original image
2. Subtract the blurred image from the original
3. Add the mask to the original

Unsharp Masking and Highboost Filtering

Let $\bar{f}(x, y)$ denote the blurred image, unsharp masking is

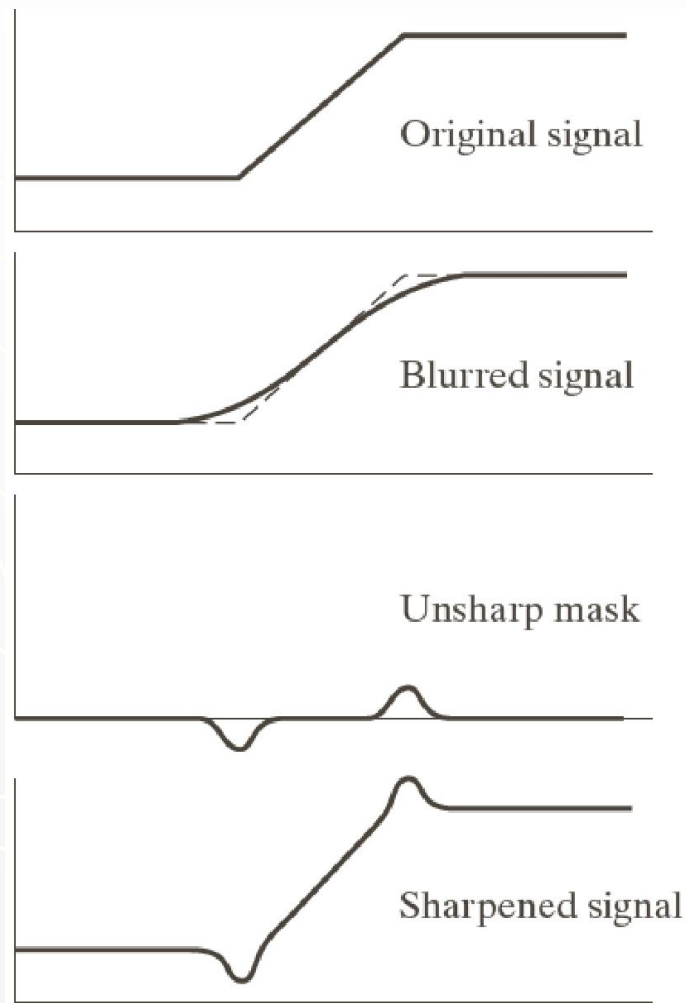
$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y)$$

Then add a weighted portion of the mask back to the original

$$g(x, y) = f(x, y) + k * g_{mask}(x, y) \quad k \geq 0$$

when $k > 1$, the process is referred to as highboost filtering.

Unsharp Masking: Demo



a
b
c
d

FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

Unsharp Masking and Highboost Filtering: Example



a
b
c
d
e

FIGURE 3.40

(a) Original image.
(b) Result of blurring with a Gaussian filter.
(c) Unsharp mask. (d) Result of using unsharp masking.
(e) Result of using highboost filtering.

Image Sharpening based on First-Order Derivatives

For function $f(x, y)$, the gradient of f at coordinates (x, y) is defined as

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

The *magnitude* of vector ∇f , denoted as $M(x, y)$

Gradient Image $M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$

Image Sharpening based on First-Order Derivatives

The *magnitude* of vector ∇f , denoted as $M(x, y)$

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

$$M(x, y) \approx |g_x| + |g_y|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$M(5)=$

$$M(x, y) = |z_8 - z_5| + |z_6 - z_5|$$

Image Sharpening based on First-Order Derivatives

Roberts Cross-gradient Operators

$$M(x, y) \approx |z_9 - z_5| + |z_8 - z_6|$$

Sobel Operators

$$M(x, y) \approx |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| \\ + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Image Sharpening based on First-Order Derivatives

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

-1	0	0	-1
0	1	1	0

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

a
b c
d e

FIGURE 3.41

A 3×3 region of an image (the z s are intensity values).

(b)–(c) Roberts cross gradient operators.

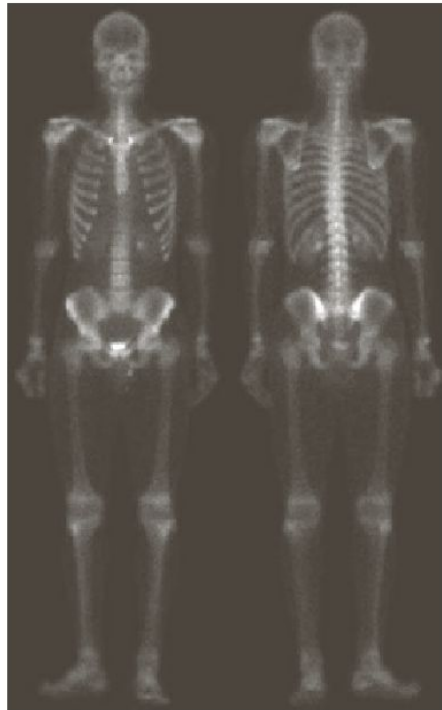
(d)–(e) Sobel operators. All the mask coefficients sum to zero, as expected of a derivative operator.

Example:

Combining Spatial Enhancement Methods

Goal:

Enhance the
image by
sharpening it
and by bringing
out more of the
skeletal detail



a	b
c	d

FIGURE 3.43

(a) Image of whole body bone scan.

(b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b). (d) Sobel gradient of (a).

What is mirroring and padding?

