DIGITAL IMAGE PROCESSING (WLE-306)

UNIT-I: Digital Image Fundamentals

Presented by:

Shahnawaz Uddin

Digital Image Processing (DIP)

"One picture is worth more than ten thousand words"

•Anonymous

What is a Digital Image?

- An image is defined as "*a two-dimensional function, f(x,y), where x and y are spatial (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity (gray level of the image) at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image.*"
- *The field of DIP refers to processing digital images by means of a digital computer*
- *A digital image composed of a finite number of elements, each of which has a particular location and value, which are called picture elements/pixel/pels.*

Digital Image Processing (DIP)

<u>We can define three types of computerized processes:</u>

Low-, mid-, and high-level.
Low: image preprocessing, noise reduction, enhance contrast etc.
Mid: segmentation, sorting and classification.
High: assembly of all components into a meaningful coherent form

Low Level Process	Mid Level Process	High Level Process
Input: Image **Output:** Image	**Input:** Image **Output:** Attributes	**Input:** Attributes **Output:** Understanding
Examples: Noise removal, image sharpening	**Examples:** Object recognition, segmentation	**Examples:** Scene understanding, autonomous navigation

Fundamental Steps in DIP

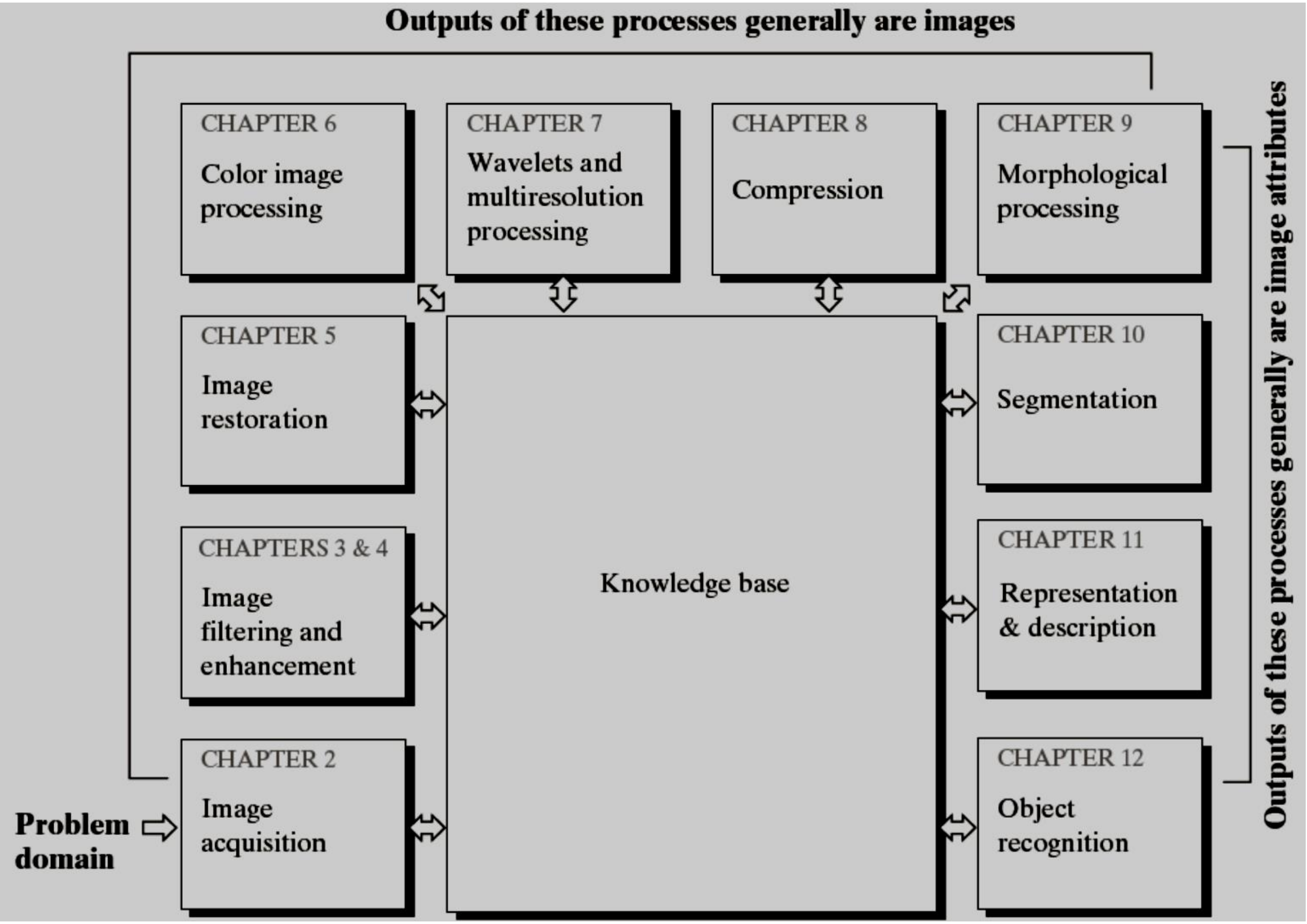
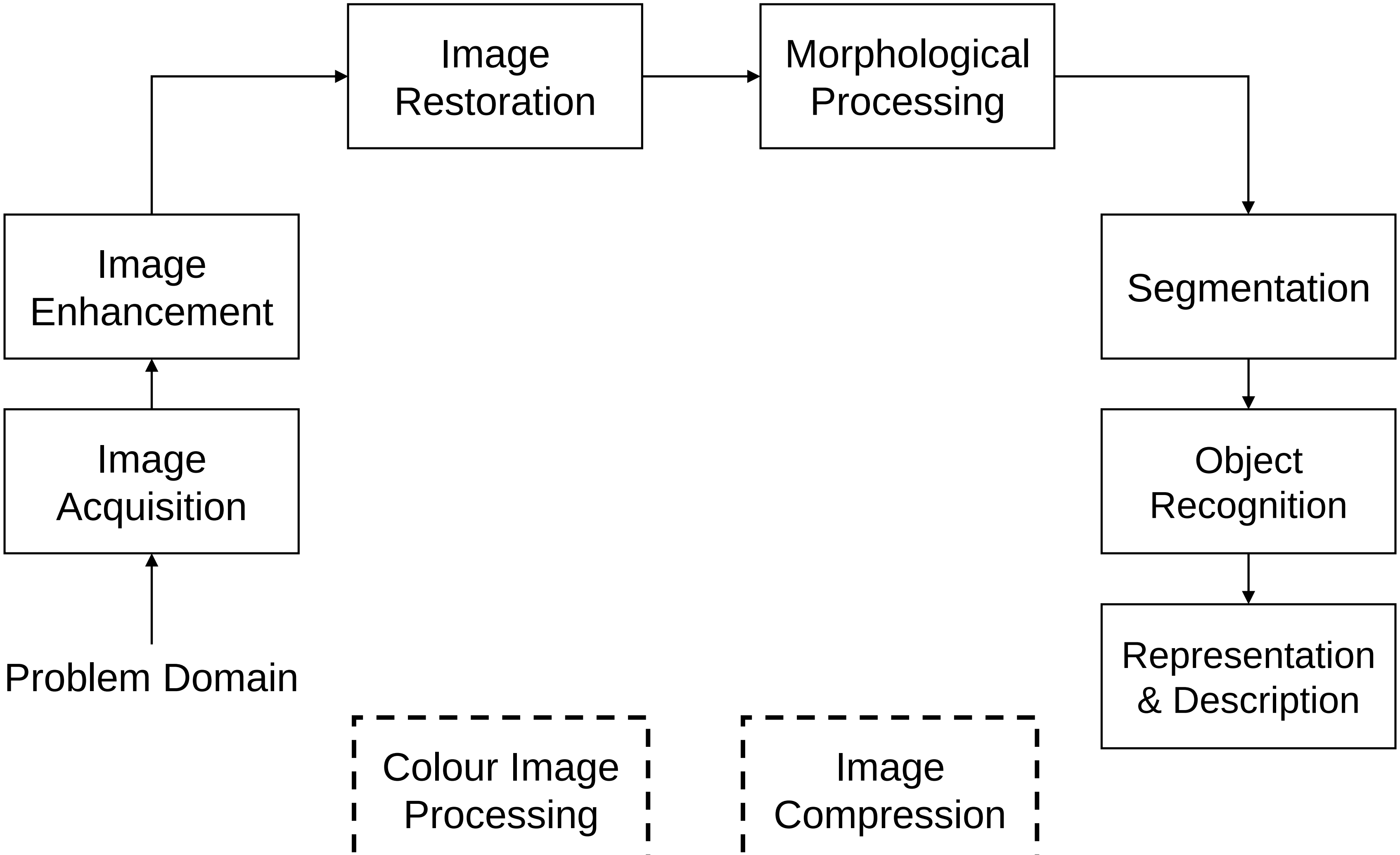


FIGURE 1.23 Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.

Key Stages in Digital Image Processing



Key Stages in Digital Image Processing: Image Aquisition

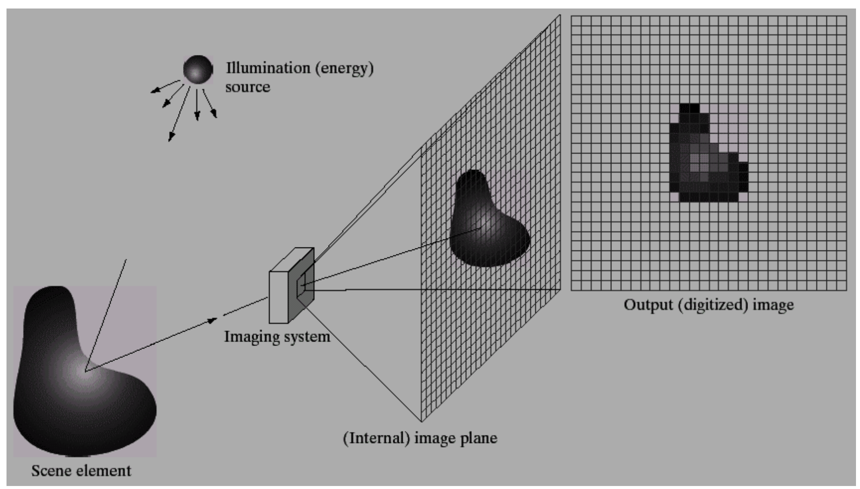
Image Restoration

Morphological Processing

Image Enhancement

Image Acquisition

Problem Domain



Segmentation

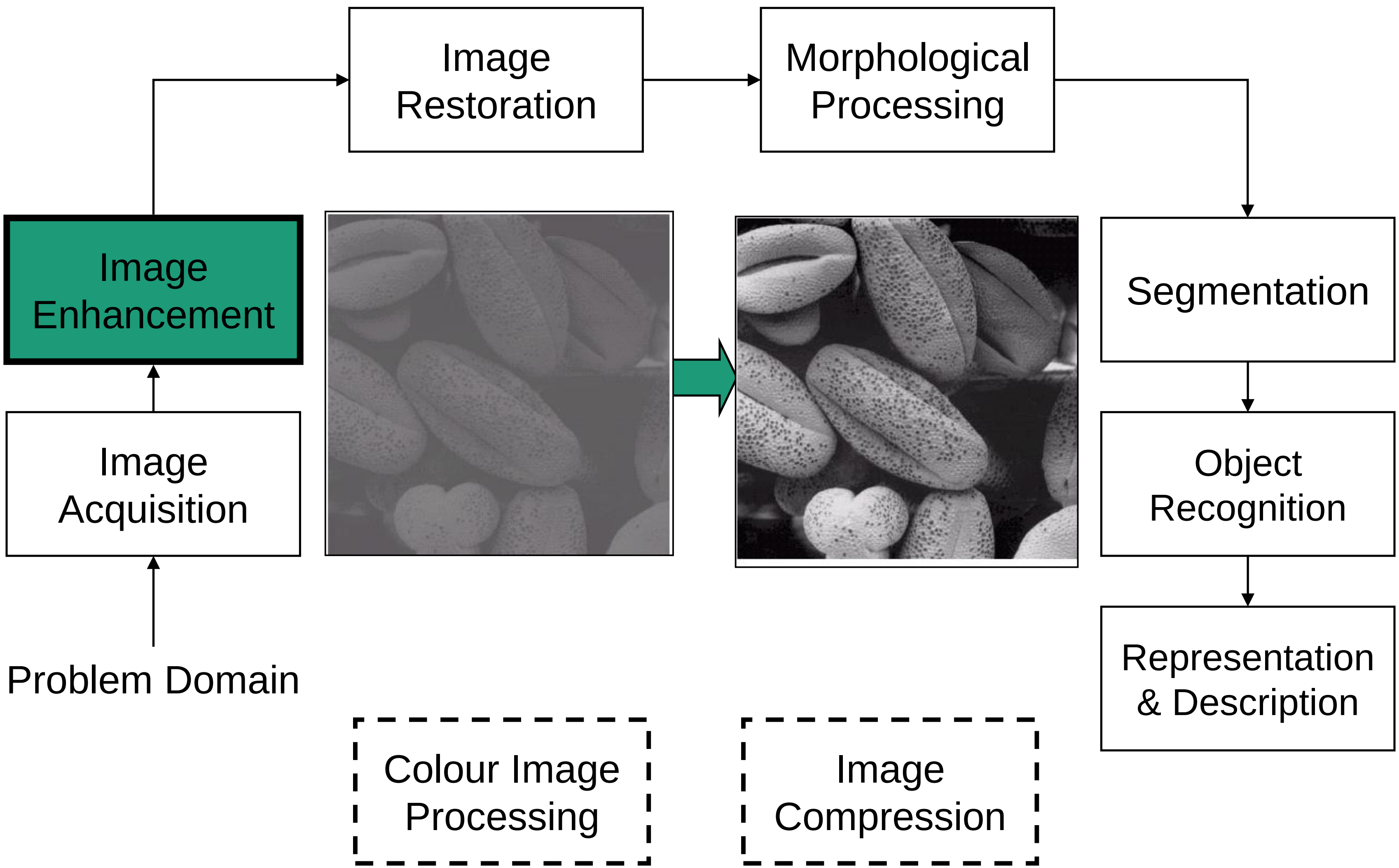
Object Recognition

Representation & Description

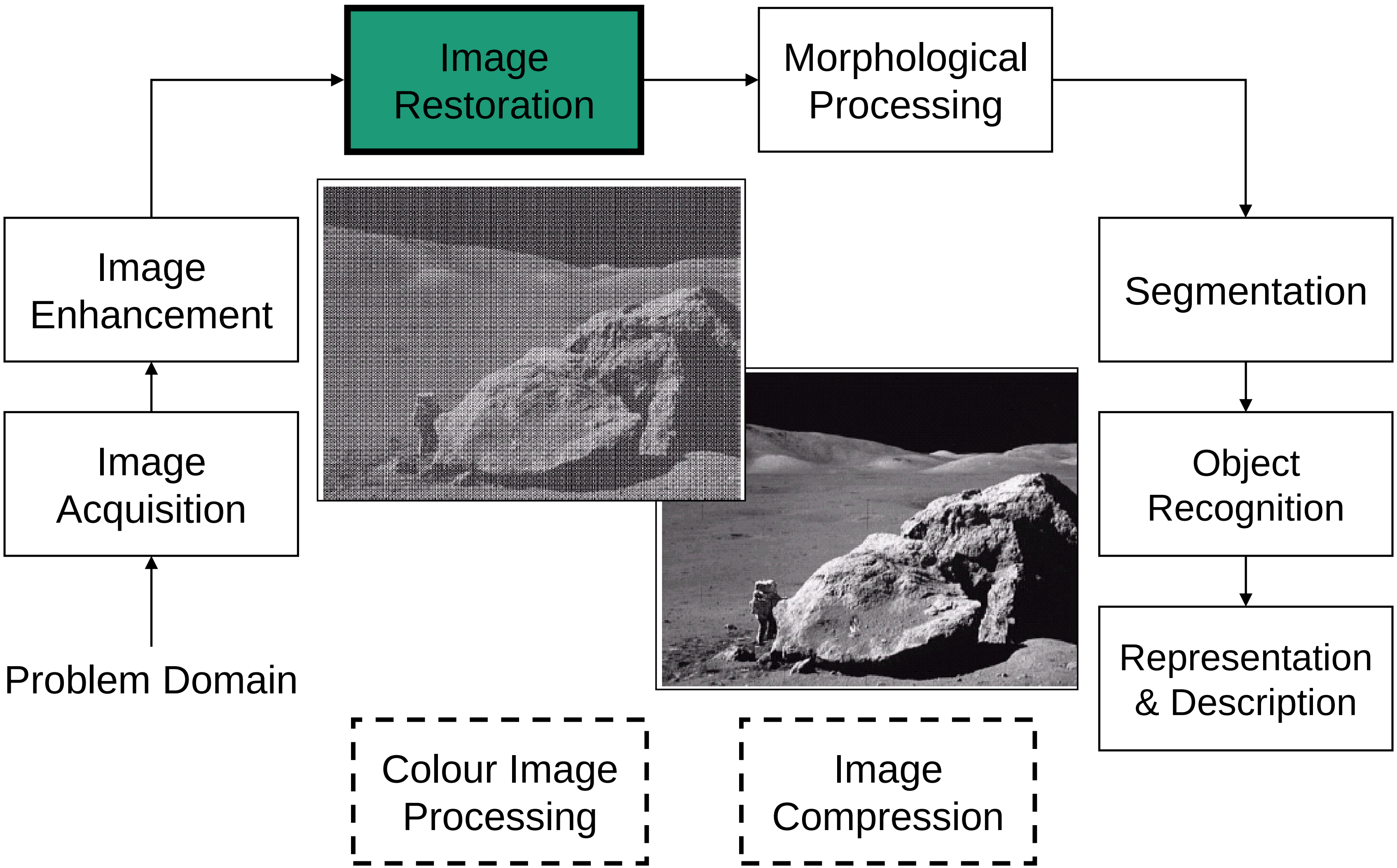
Colour Image Processing

Image Compression

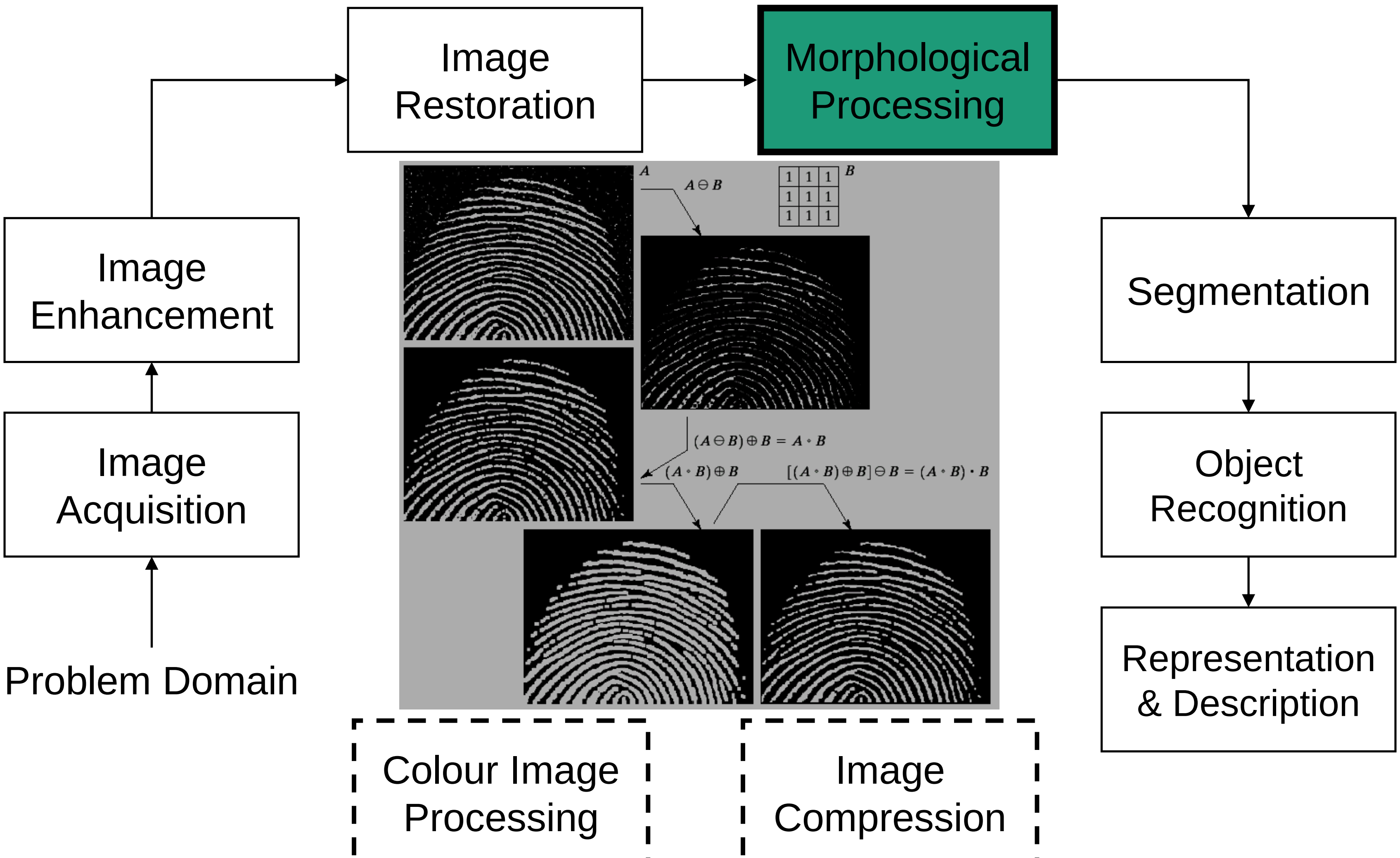
Key Stages in Digital Image Processing: Image Enhancement



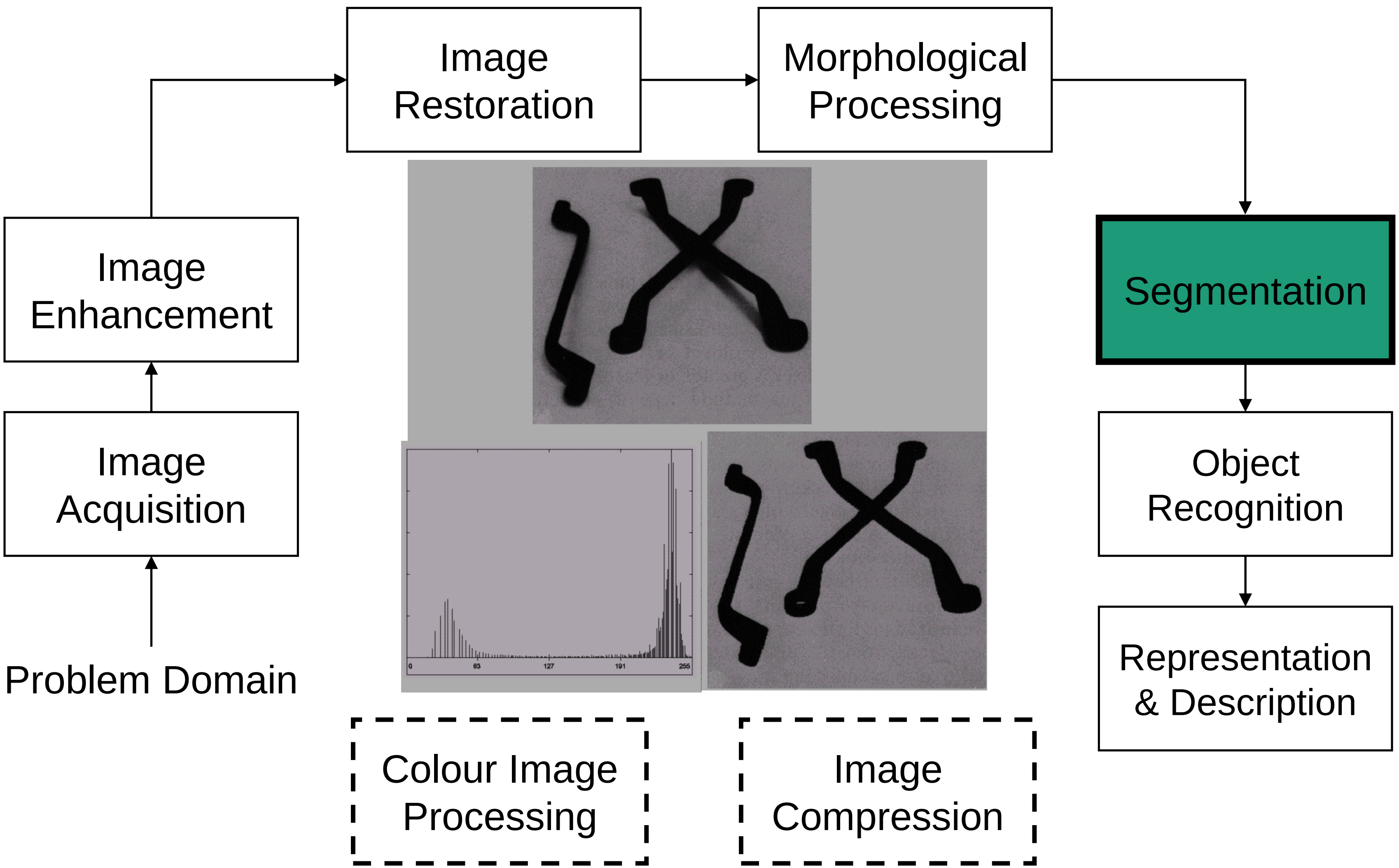
Key Stages in Digital Image Processing: Image Restoration



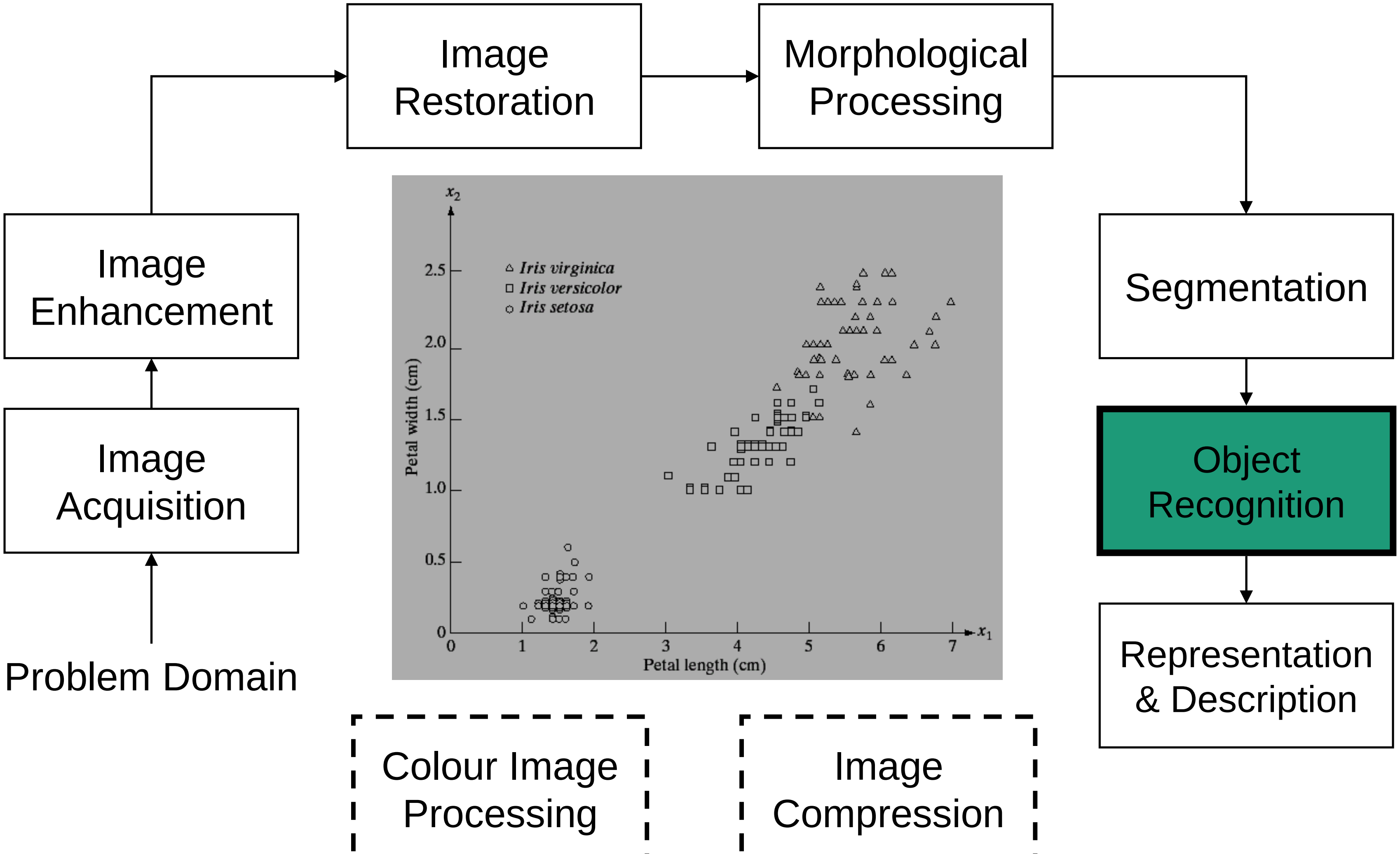
Key Stages in Digital Image Processing: Morphological Processing



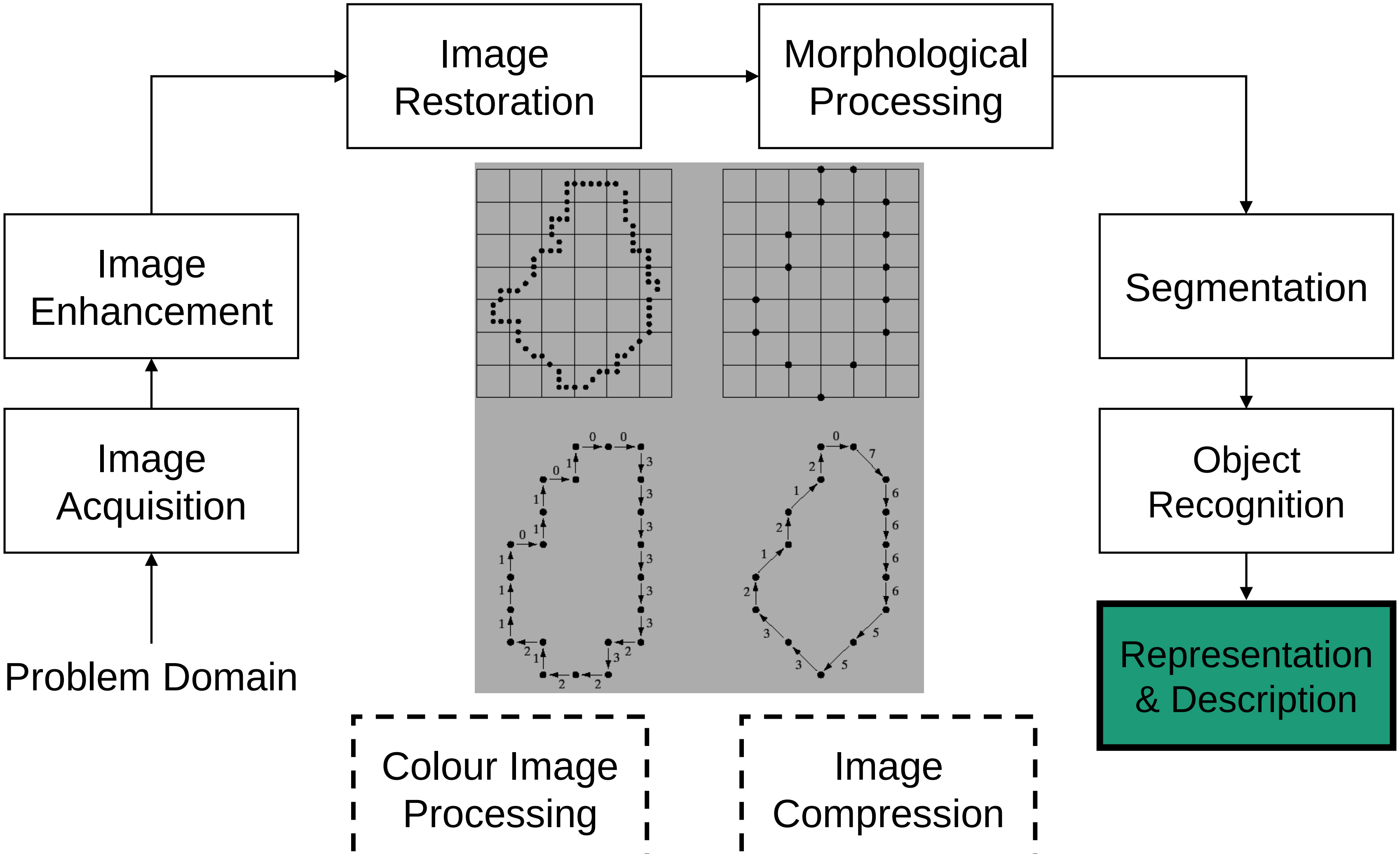
Key Stages in Digital Image Processing: Segmentation



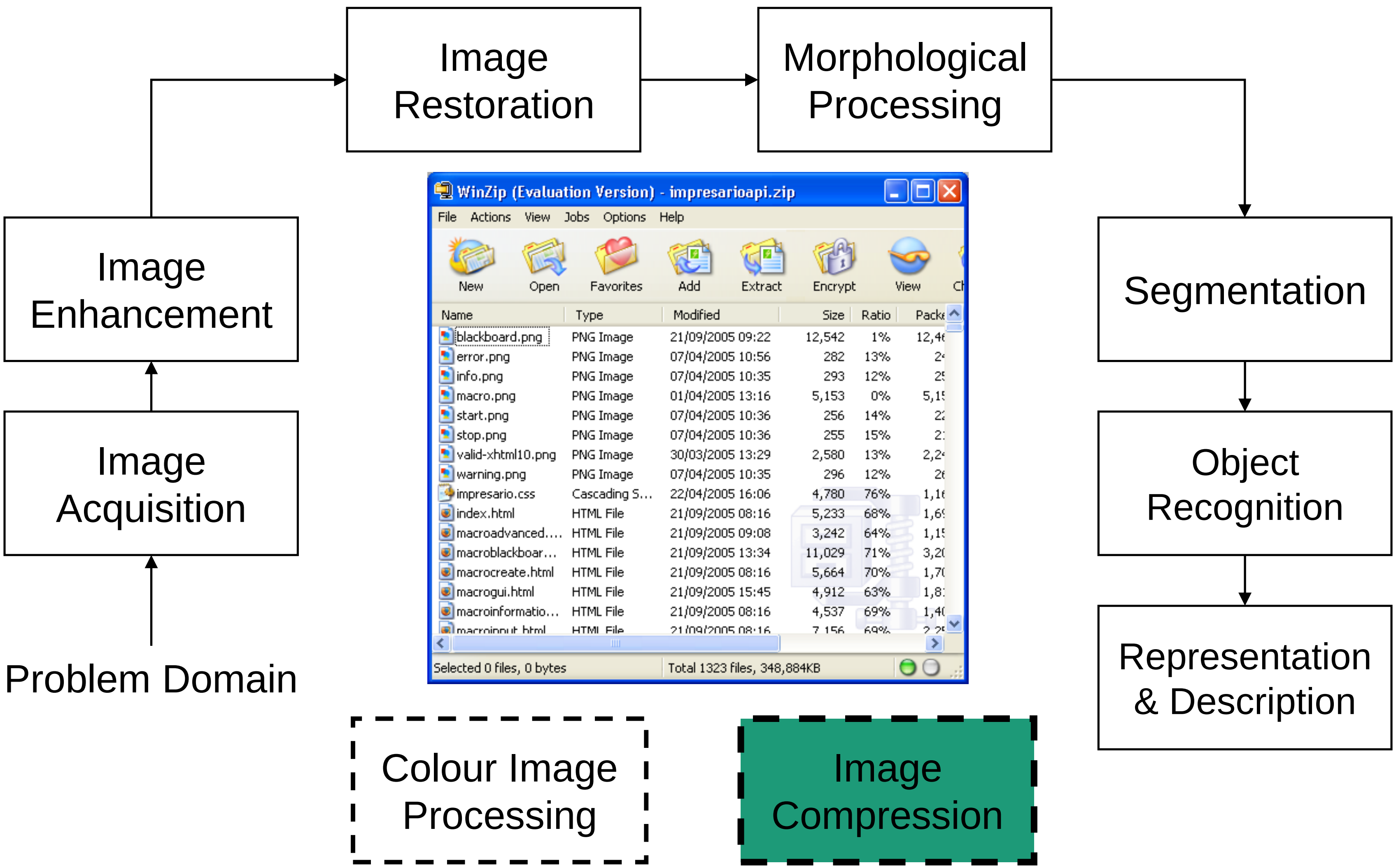
Key Stages in Digital Image Processing: Object Recognition



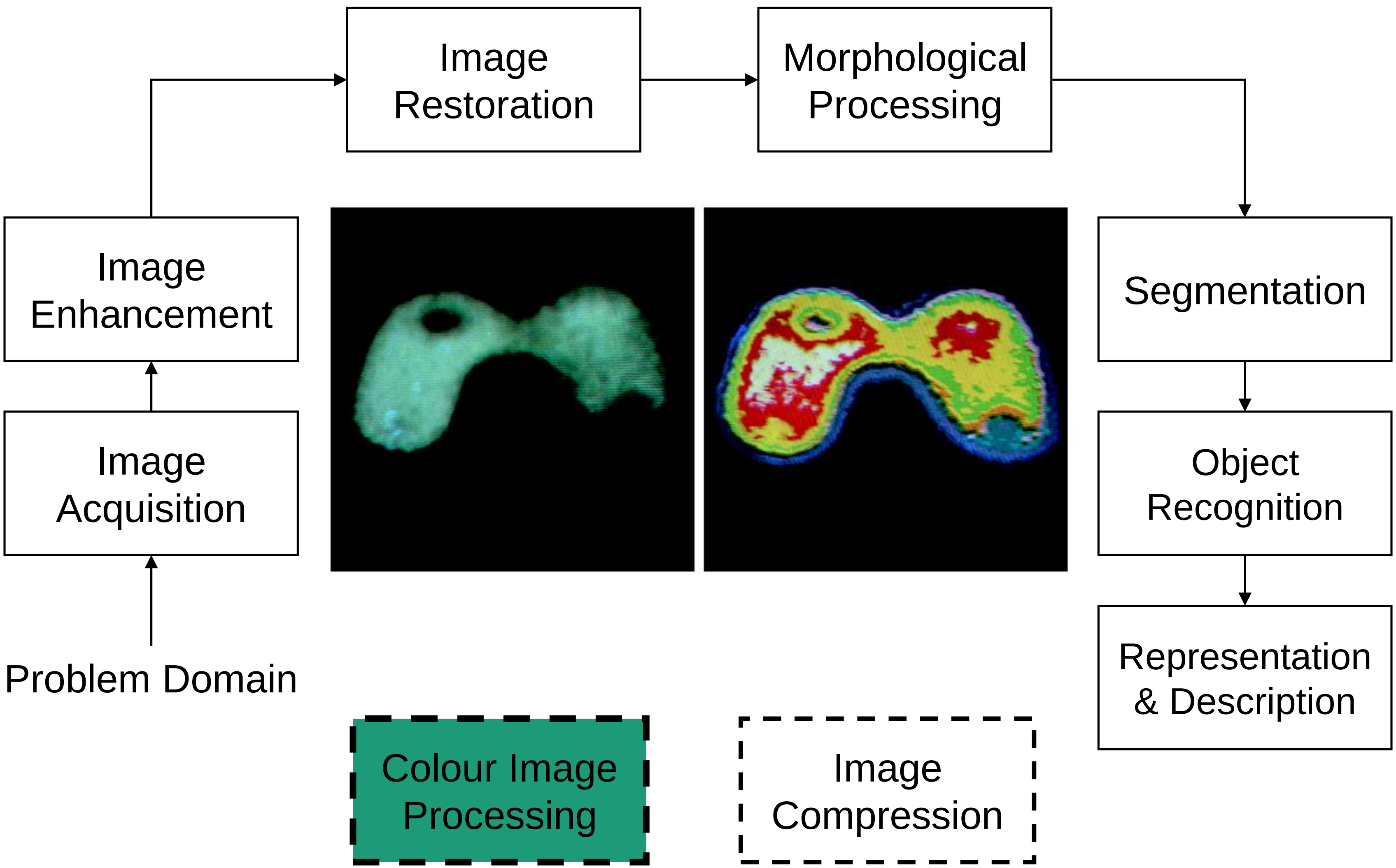
Key Stages in Digital Image Processing: Representation & Description



Key Stages in Digital Image Processing: Image Compression



Key Stages in Digital Image Processing: Colour Image Processing



Components of an Image Processing System

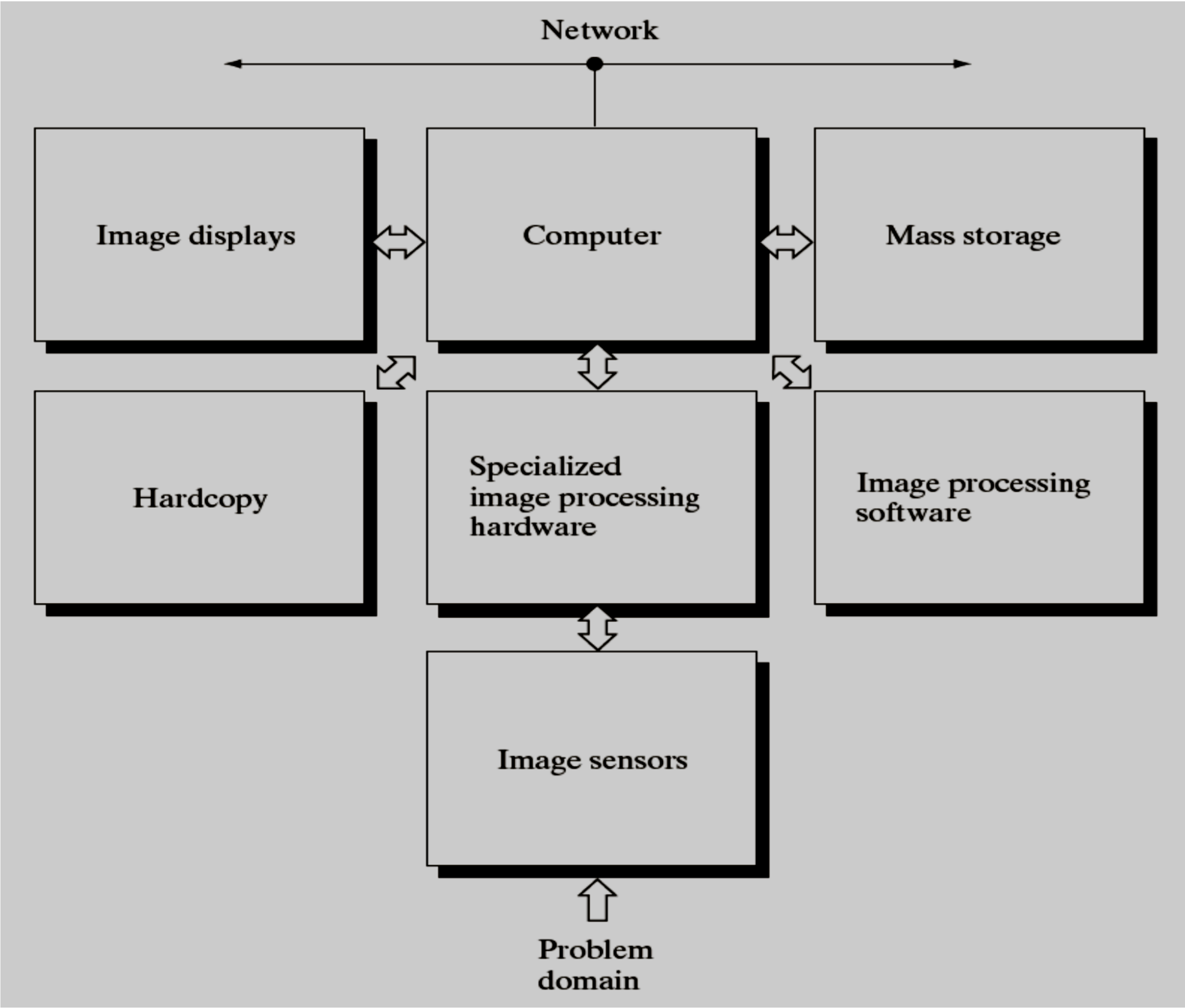


FIGURE 1.24 Components of a general-purpose image processing system.

Light & the Electromagnetic Spectrum

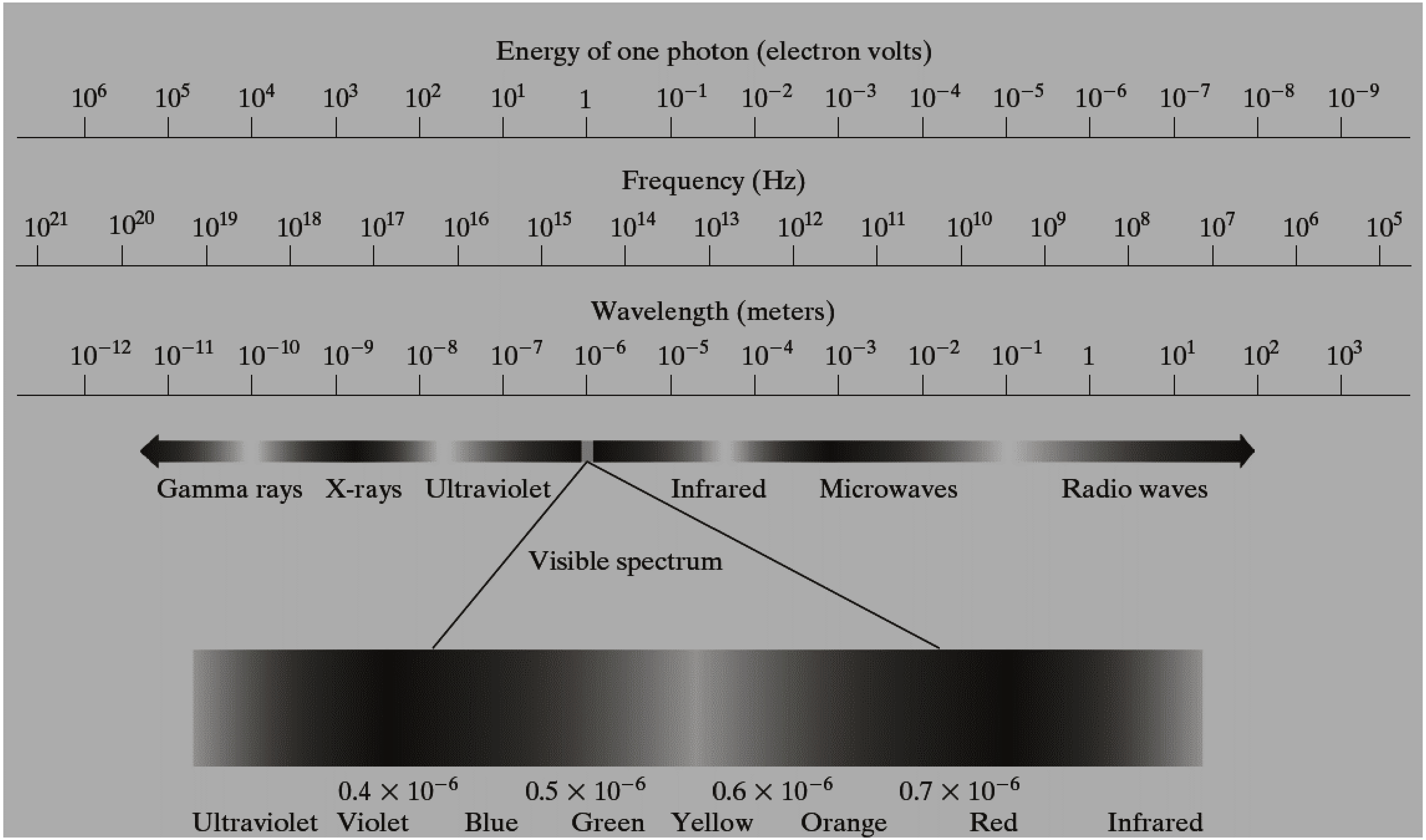


FIGURE 2.10 The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanation, but note that the visible spectrum is a rather narrow portion of the EM spectrum.

Light & the Electromagnetic Spectrum

- Chromatic (color) light spans the em energy spectrum from approx. 0.43 to 0.79 μm. In addition to frequency, three basic quantities are used to describe the quality of a chromatic light source: **radiance, luminance, & brightness**
- **Radiance-** the total amount of energy radiated by the light source and usually measured in watts (W)
- **Luminance-** it is the measure of energy an observer perceives from a light source and is measured in lumens (lm)
- **Brightness-** it is the subjective descriptor of light perception that is practically impossible to measure

Light & the Electromagnetic Spectrum

- Radiance (watt):
  - Total amount of energy flow from the light source.
- Luminance (lumens, lm):
  - measure of amount of energy an observer perceives from a light source. It varies based on distance from the source, wavelength, etc.
- Brightness:
  - a subjective descriptor, describing color sensation.
- Primary colors of pigment (subtractive):
  - Magenta,
  - Cyan, and
  - Yellow

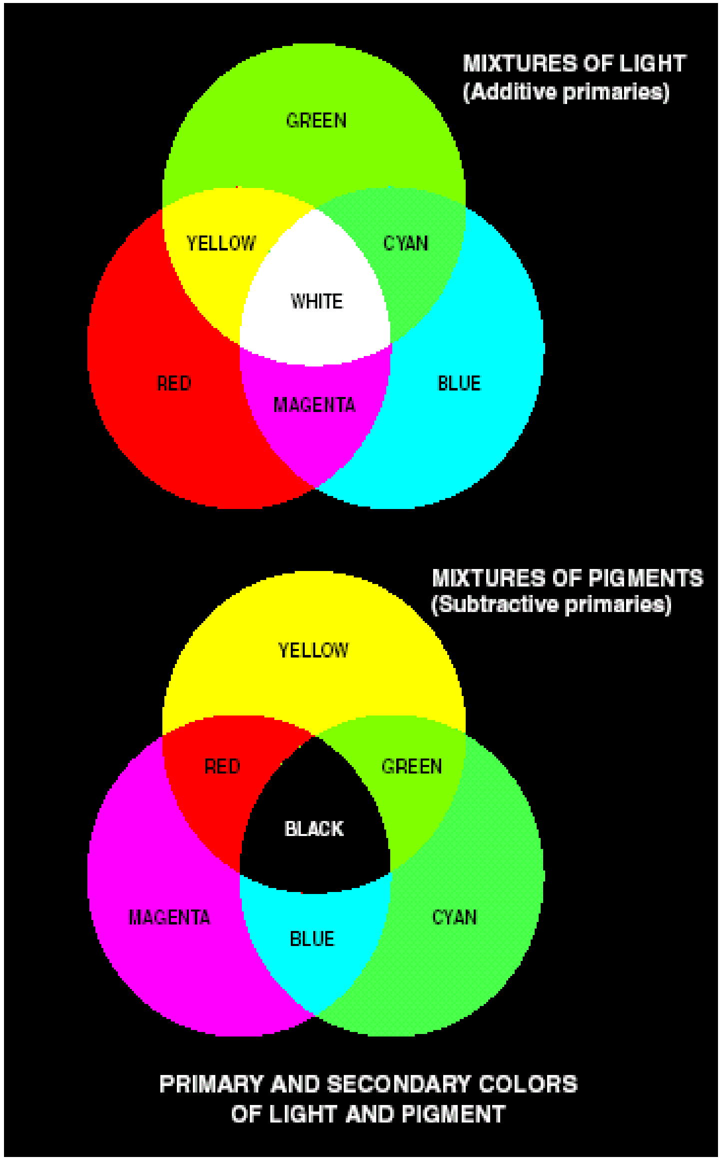
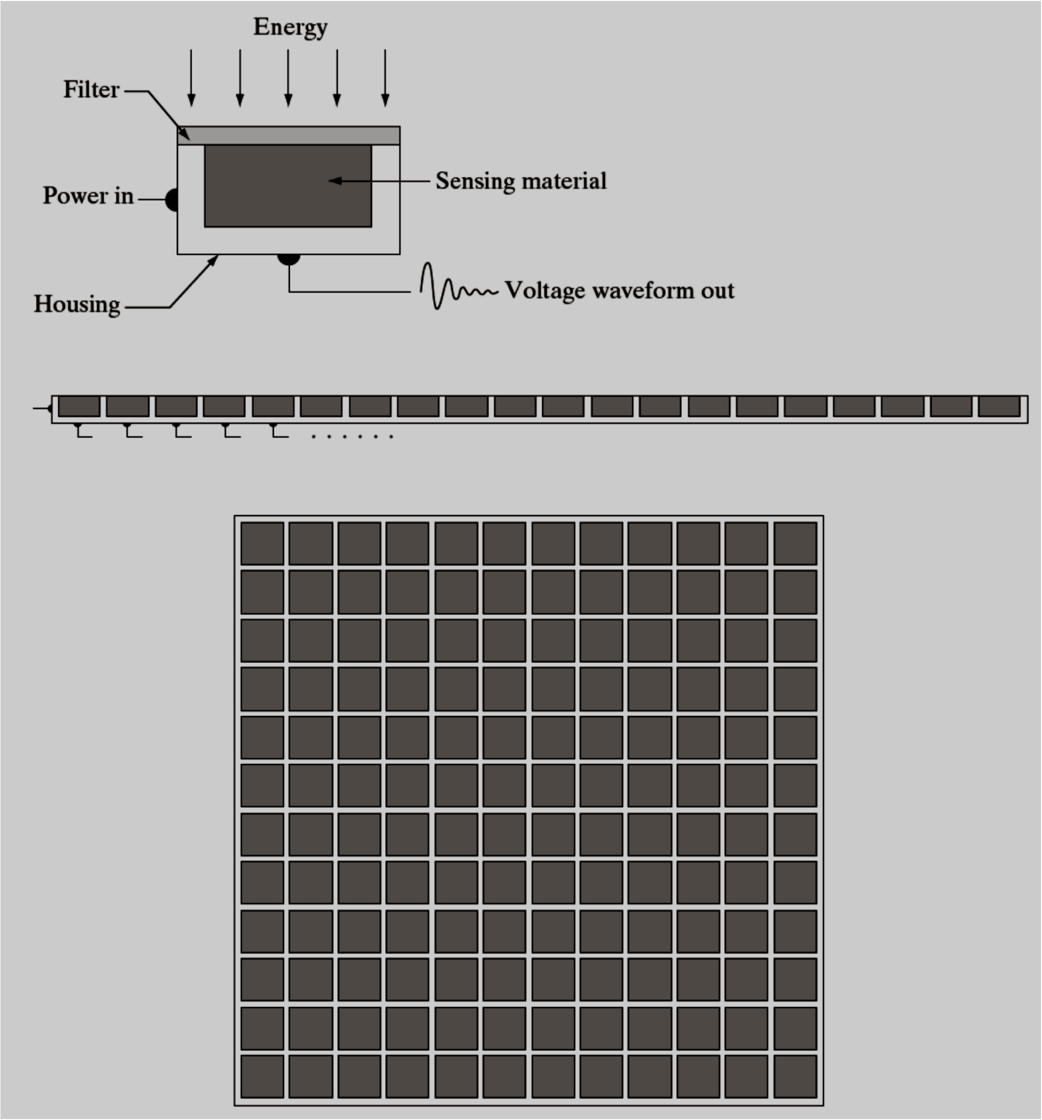


Image Sensing & Acquisition



a
b
c

FIGURE 2.12
(a) Single imaging sensor.
(b) Line sensor.
(c) Array sensor.

Image Acquisition using Single Sensor

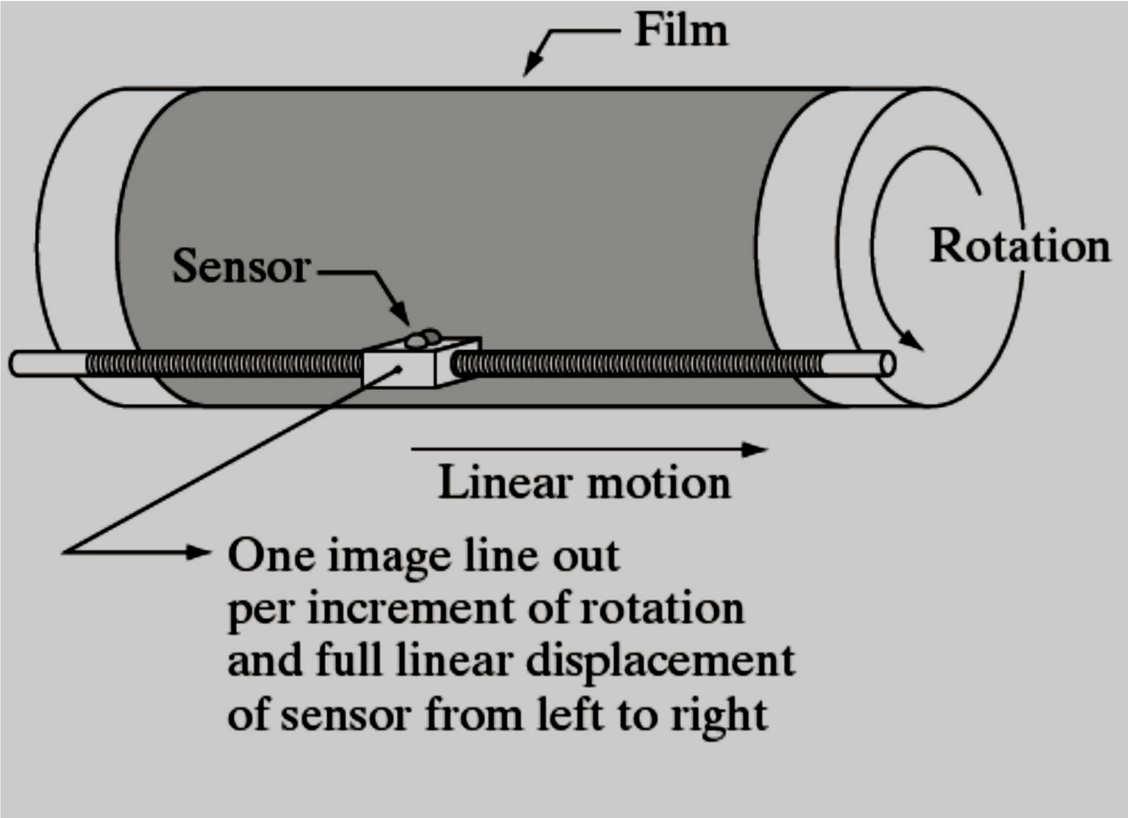
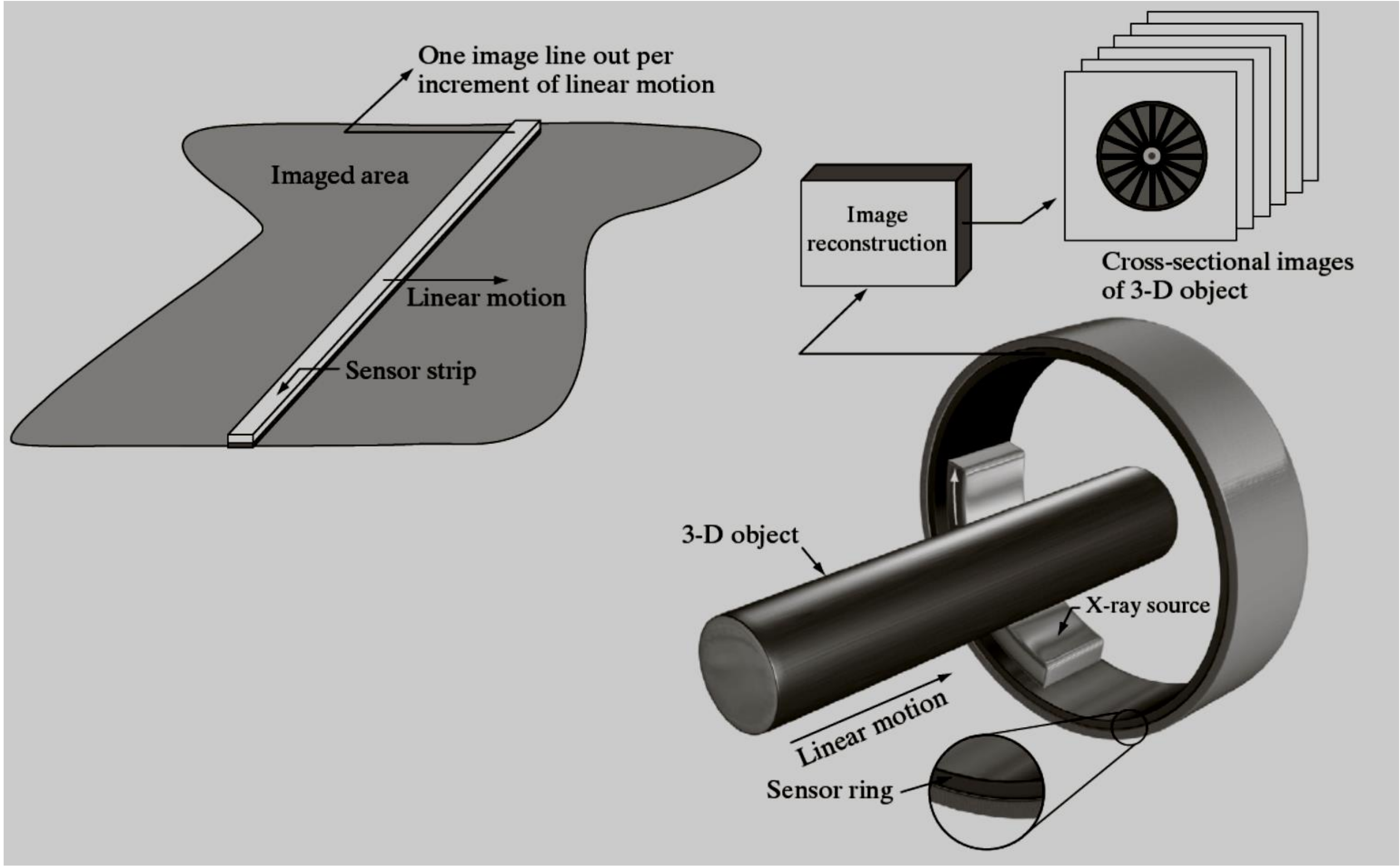


FIGURE 2.13 Combining a single sensor with motion to generate a 2-D image.

Image Acquisition using Sensor Strip



a b

FIGURE 2.14 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

Image Acquisition using Sensor Arrays

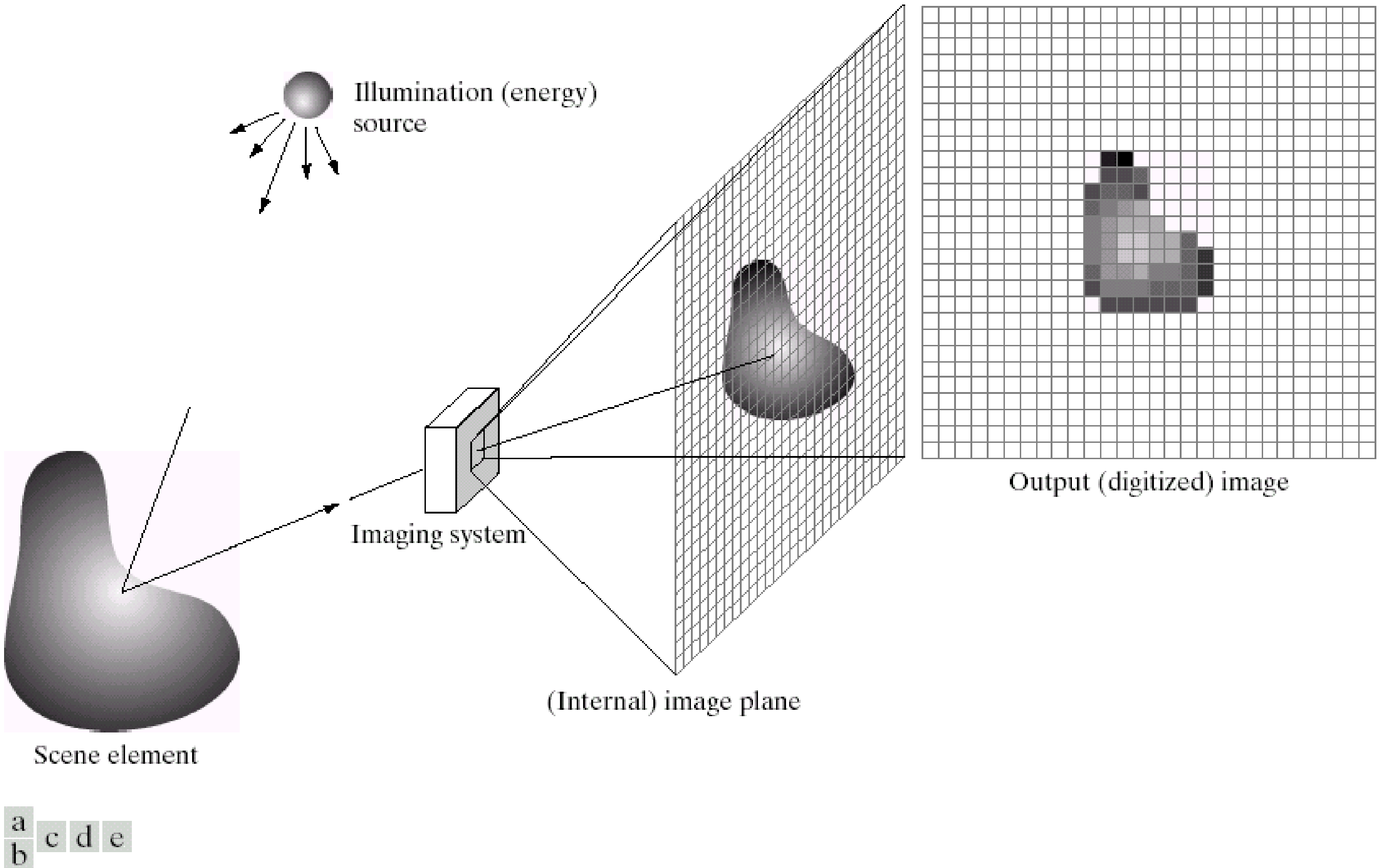


FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

Simple Image Formation Model

$$0 < f(x, y) < \infty$$

$$\ell = f(x_0, y_0)$$

$$f(x, y) = i(x, y)r(x, y)$$

$$L_{\min} \le \ell \le L_{\max}$$

$$0 < i(x, y) < \infty$$

$$0 < r(x, y) < 1$$

EXAMPLE 2.1: Some typical values of illumination and reflectance.

■ The values given in Eqs. (2.3-3) and (2.3-4) are theoretical bounds. The following *average* numerical figures illustrate some typical ranges of $i(x, y)$ for visible light. On a clear day, the sun may produce in excess of 90,000 lm/m^2 of illumination on the surface of the Earth. This figure decreases to less than 10,000 lm/m^2 on a cloudy day. On a clear evening, a full moon yields about 0.1 lm/m^2 of illumination. The typical illumination level in a commercial office is about 1000 lm/m^2. Similarly, the following are typical values of $r(x, y)$: 0.01 for black velvet, 0.65 for stainless steel, 0.80 for flat-white wall paint, 0.90 for silver-plated metal, and 0.93 for snow. ■

Image Sampling & Quantization

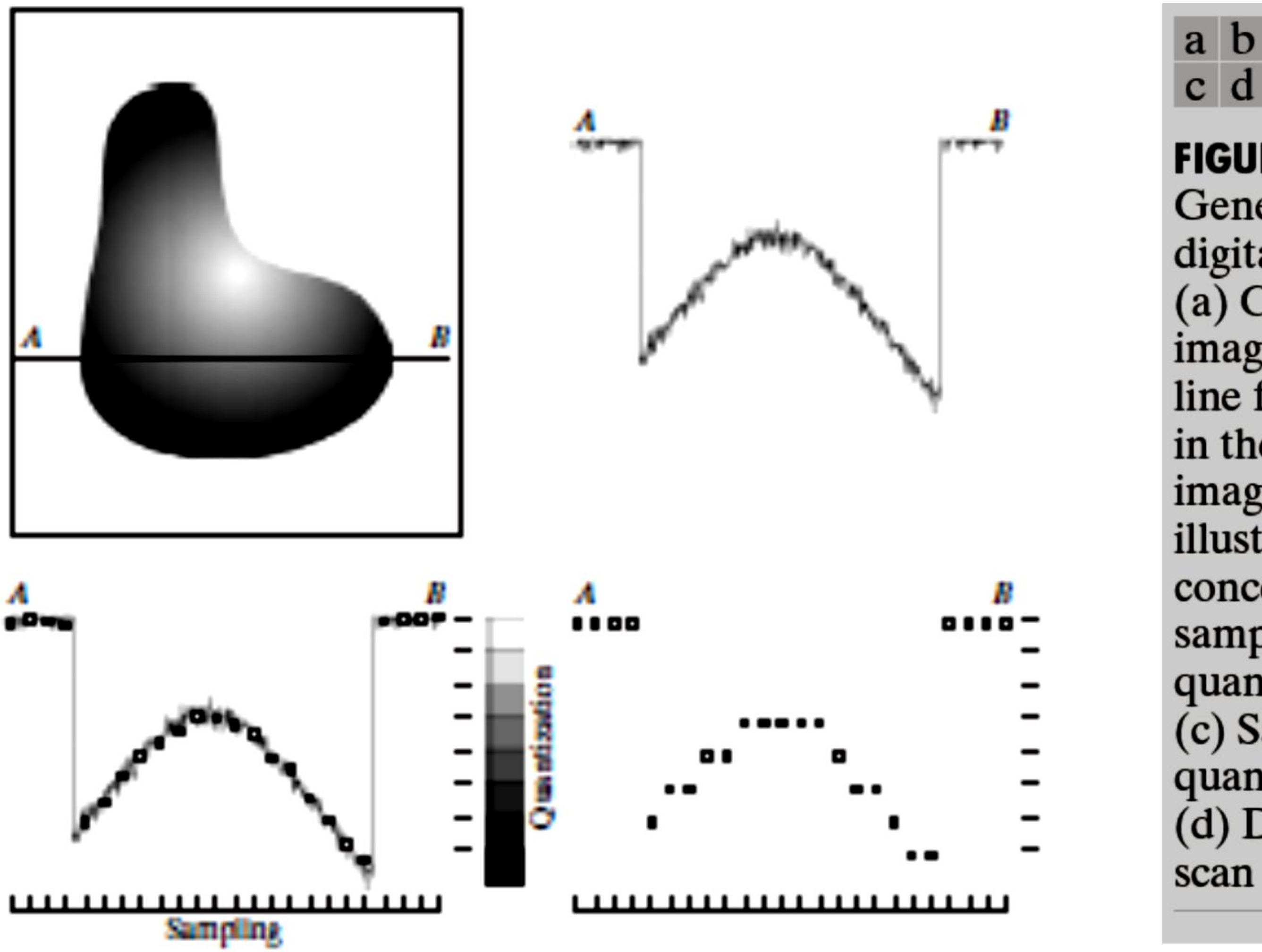


FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

Image Sampling & Quantization

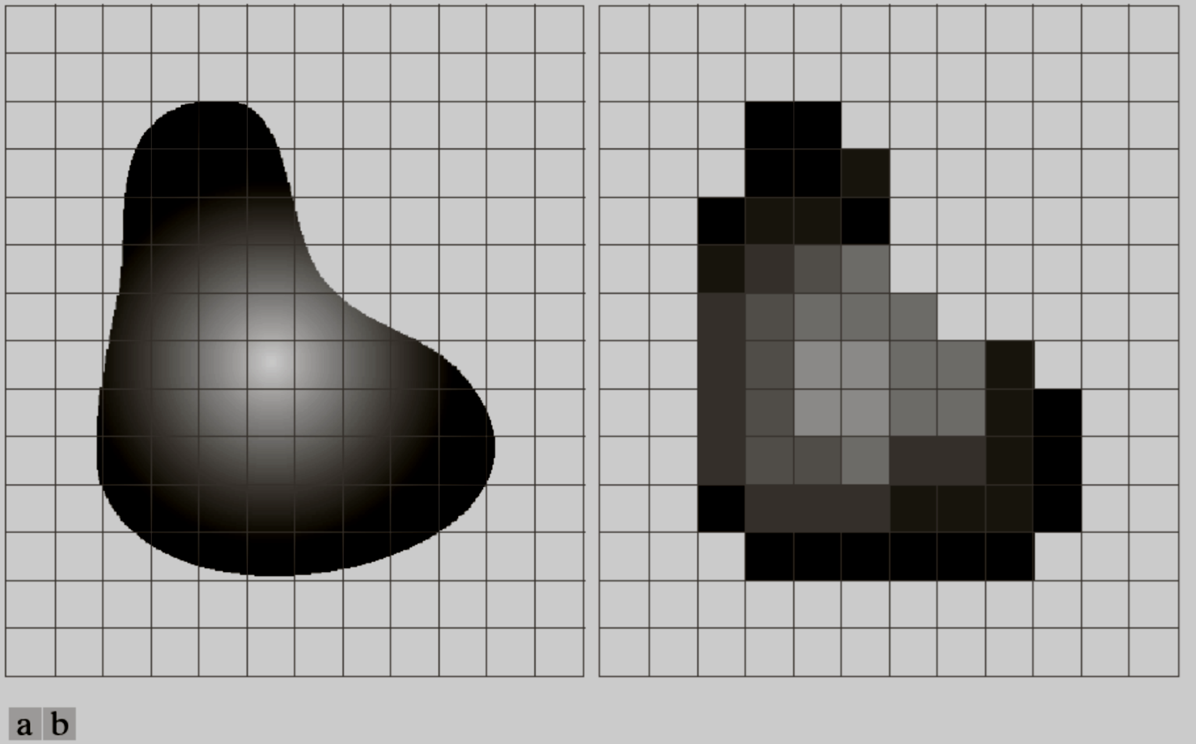
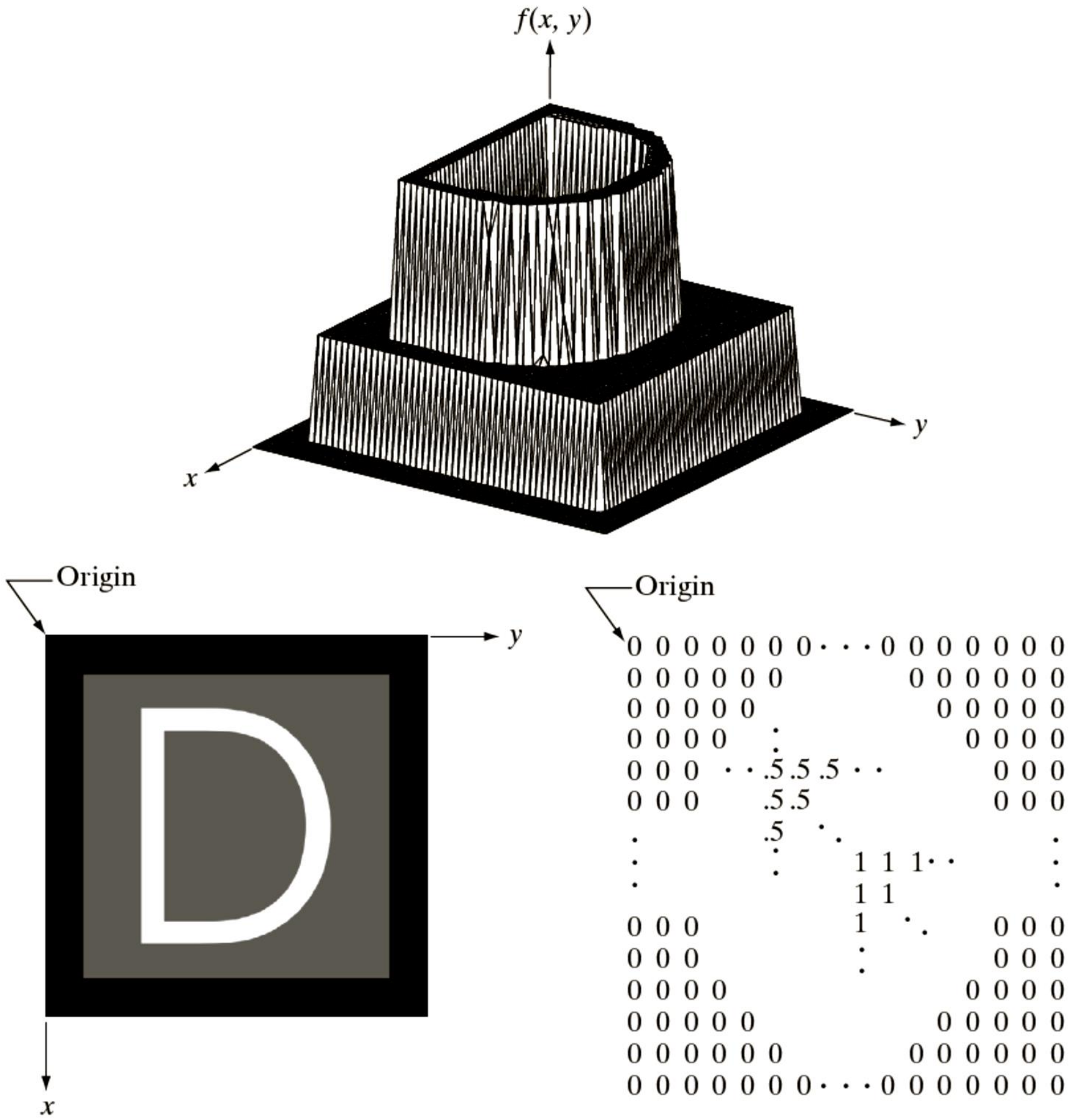


FIGURE 2.17 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

Representing Digital Images



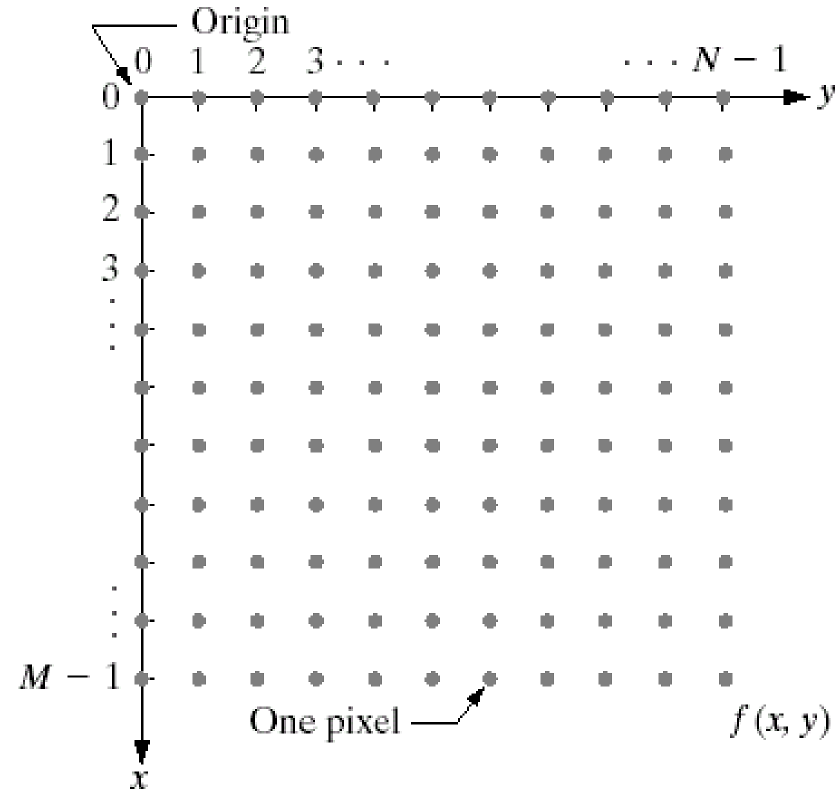
a

b c

FIGURE 2.18
(a) Image plotted as a surface.
(b) Image displayed as a visual intensity array.
(c) Image shown as a 2-D numerical array (0, .5, and 1 represent black, gray, and white, respectively).

Digital Image Representation

- An image is a function defined on a 2D coordinate $f(x,y)$
- The value of $f(x,y)$ is the intensity
- 3 such functions can be defined for a color image, each represents one color component
- A digital image can be represented as a matrix



$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0, N-1) \\ f(1,0) & f(1,1) & \cdots & f(1, N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1, N-1) \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix}$$

$$a_{ij} = f(x = i, y = j) = f(i, j)$$

$$L = 2^k \qquad b = M \times N \times k \qquad b = N^2 k$$

- Sometimes, the range of values spanned by the gray scale is referred to informally as the **dynamic range-** the ratio of the maximum measurable intensity to the minimum detectable intensity level in the system
- **Contrast-** the difference between the highest and the lowest intensity levels in an image

An Image Exhibiting Saturation & Noise

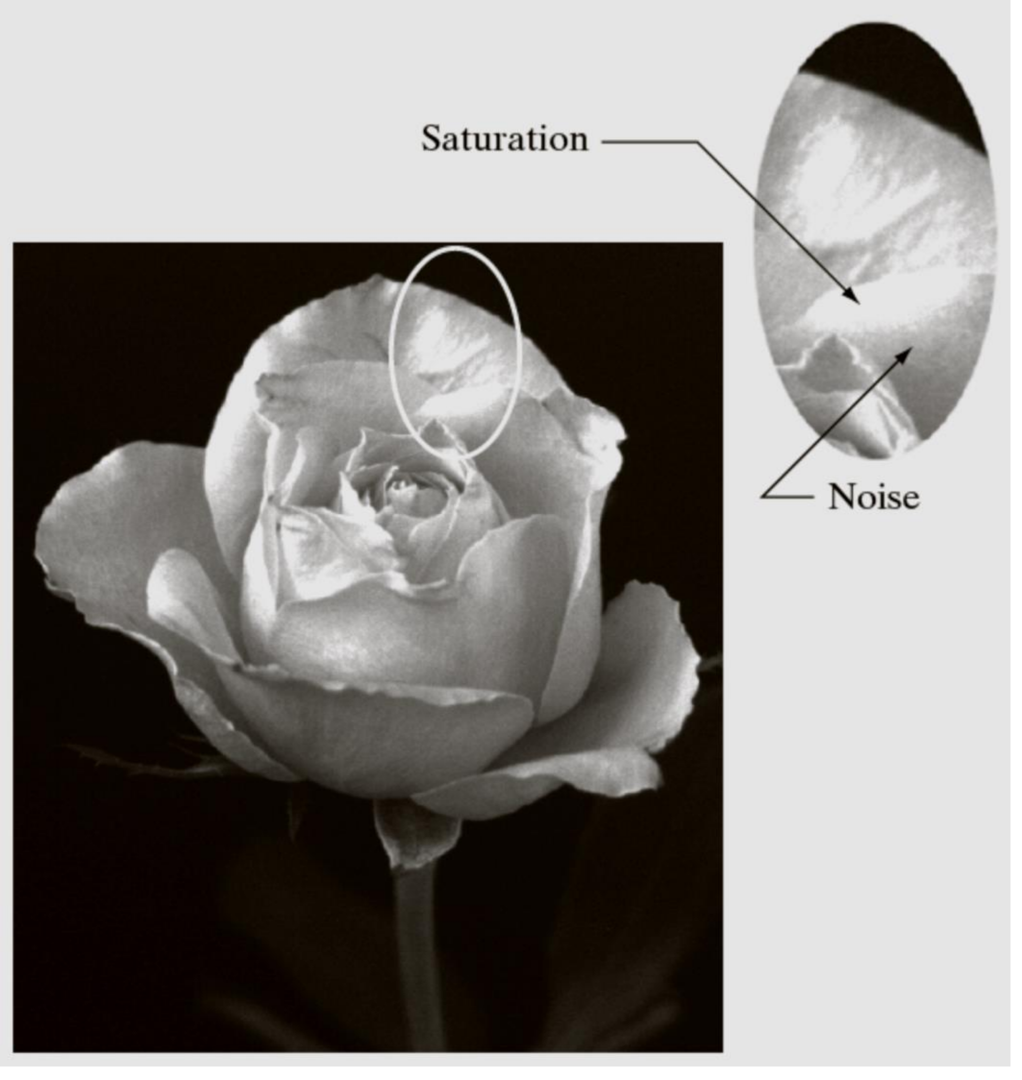


FIGURE 2.19 An image exhibiting saturation and noise. Saturation is the highest value beyond which all intensity levels are clipped (note how the entire saturated area has a high, *constant* intensity level). Noise in this case appears as a grainy texture pattern. Noise, especially in the darker regions of an image (e.g., the stem of the rose) masks the lowest detectable true intensity level.

Number of Storage Bits for various values of N & k

TABLE 2.1
Number of storage bits for various values of N and k.

N/k	1 (L = 2)	2 (L = 4)	3 (L = 8)	4 (L = 16)	5 (L = 32)	6 (L = 64)	7 (L = 128)	8 (L = 256)
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

Spatial and Gray Level Resolution

- Spatial resolution is rather intuitive, and is determined by the quality and “density” of the sampling

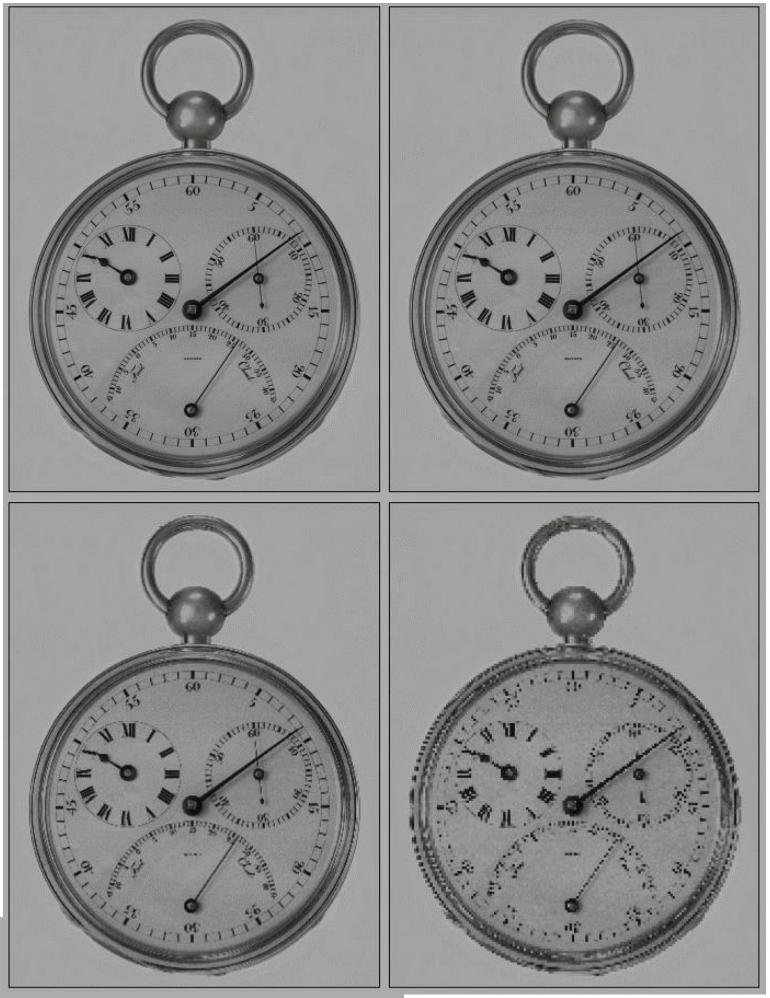
- Sampling theories (e.g., Nyquist-Shannon) state that sampling should be performed at a rate that **is at least** twice the size of the smallest object/highest frequency
- Based on this, over-sampling and under-sampling (=spatial aliasing) can occur

- Gray level resolution is a term used to describe the binning of the signal rather than the actual difference we managed to obtain when we quantized the signal. 8-bit and 16-bit images are the most common ones, but 10- and 12-bit images can also be found

Spatial and Gray Level Resolution

- Spatial resolution:
  - Number of samples per unit length or area
  - DPI: dots per inch specifies the size of an individual pixel
  - If pixel size is kept constant, the size of an image will affect spatial resolution
- Gray level resolution:
  - Number of bits per pixel
  - Usually 8 bits
  - Color image has 3 image planes to yield 8 x 3 = 24 bits/pixel
  - Too few levels may cause *false contour*

Spatial Resolution



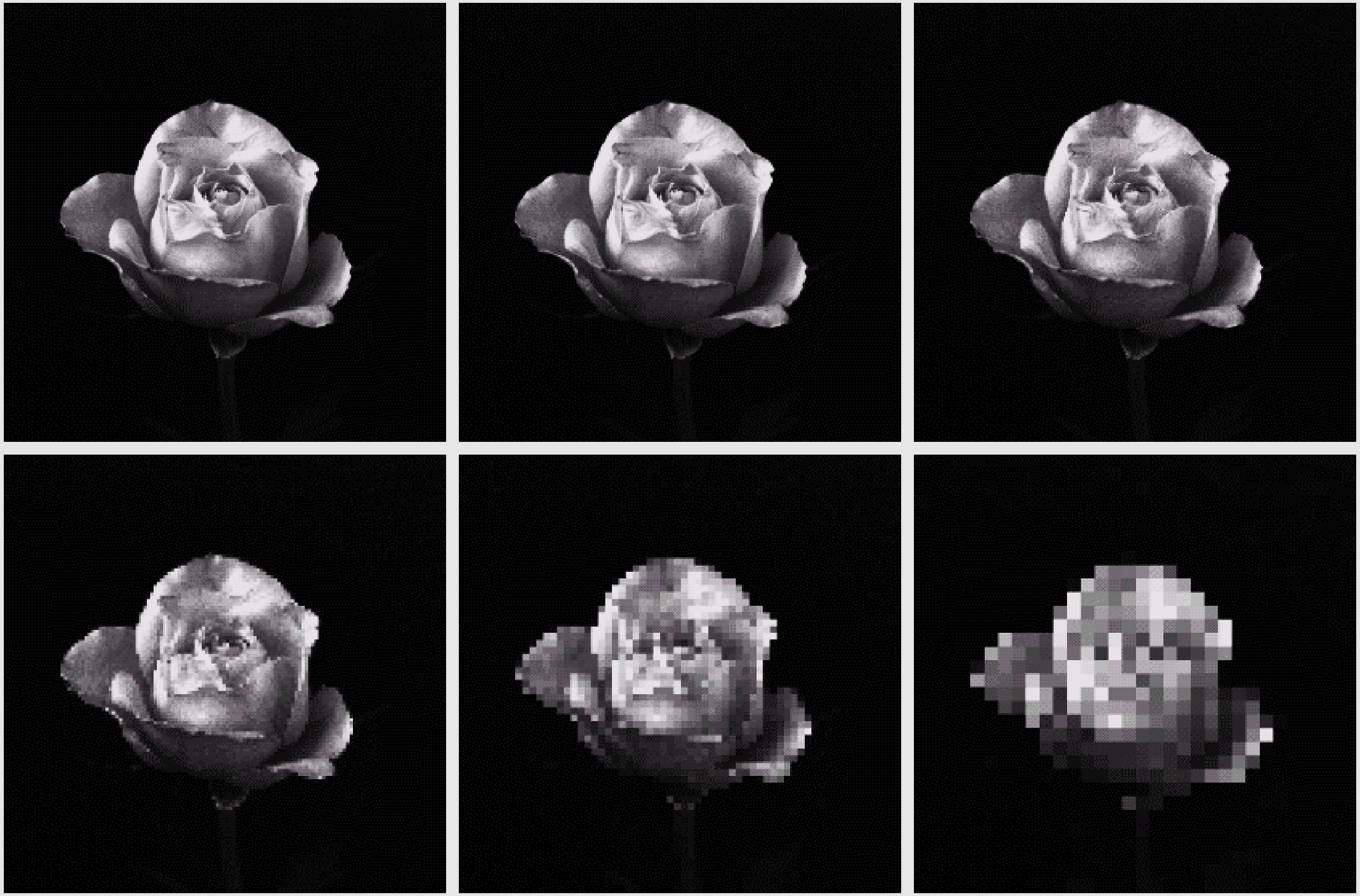
a b
c d

FIGURE 2.20 Typical effects of reducing spatial resolution. Images shown at: (a) 1250 dpi, (b) 300 dpi, (c) 150 dpi, and (d) 72 dpi. The thin black borders were added for clarity. They are not part of the data.

Same Pixel Size, different Sizes

FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

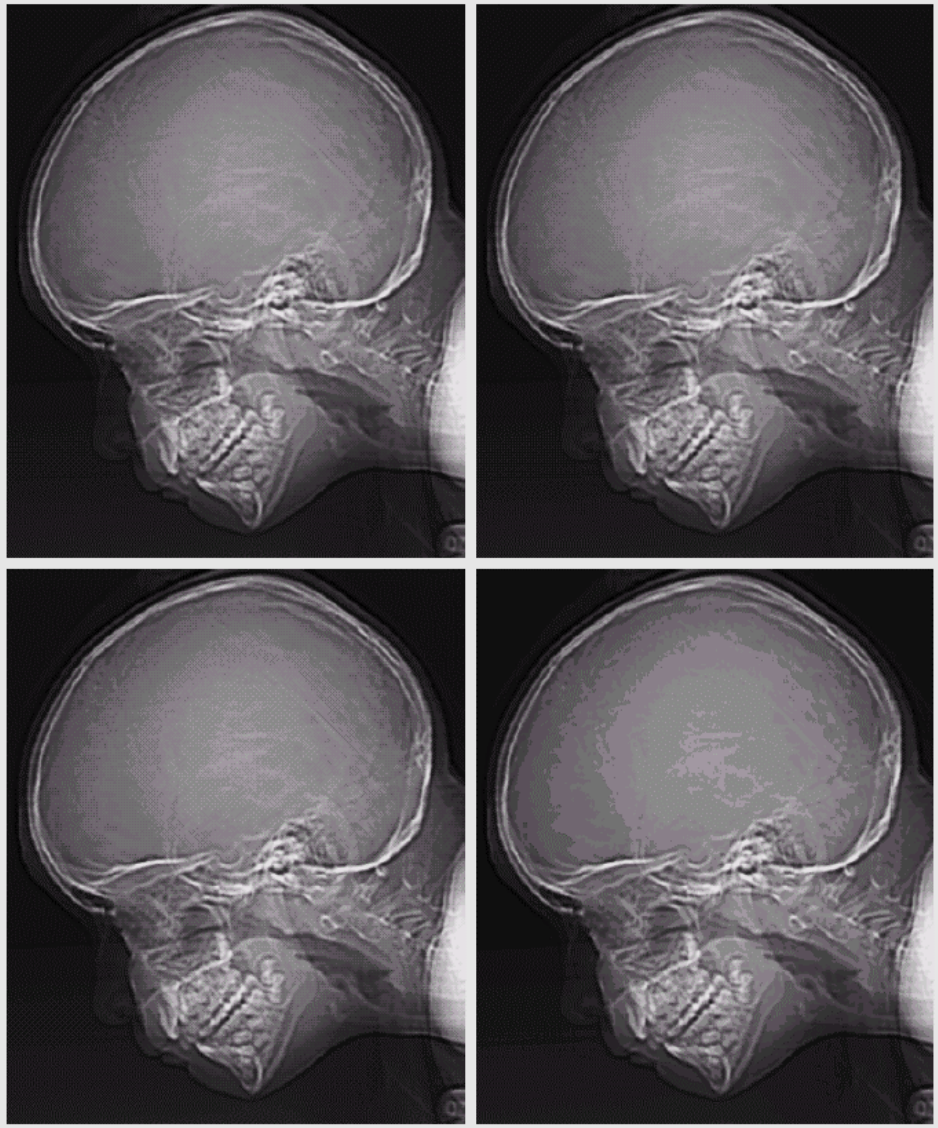
Same Size, Different Pixel Sizes



a	b	c
d	e	f

FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

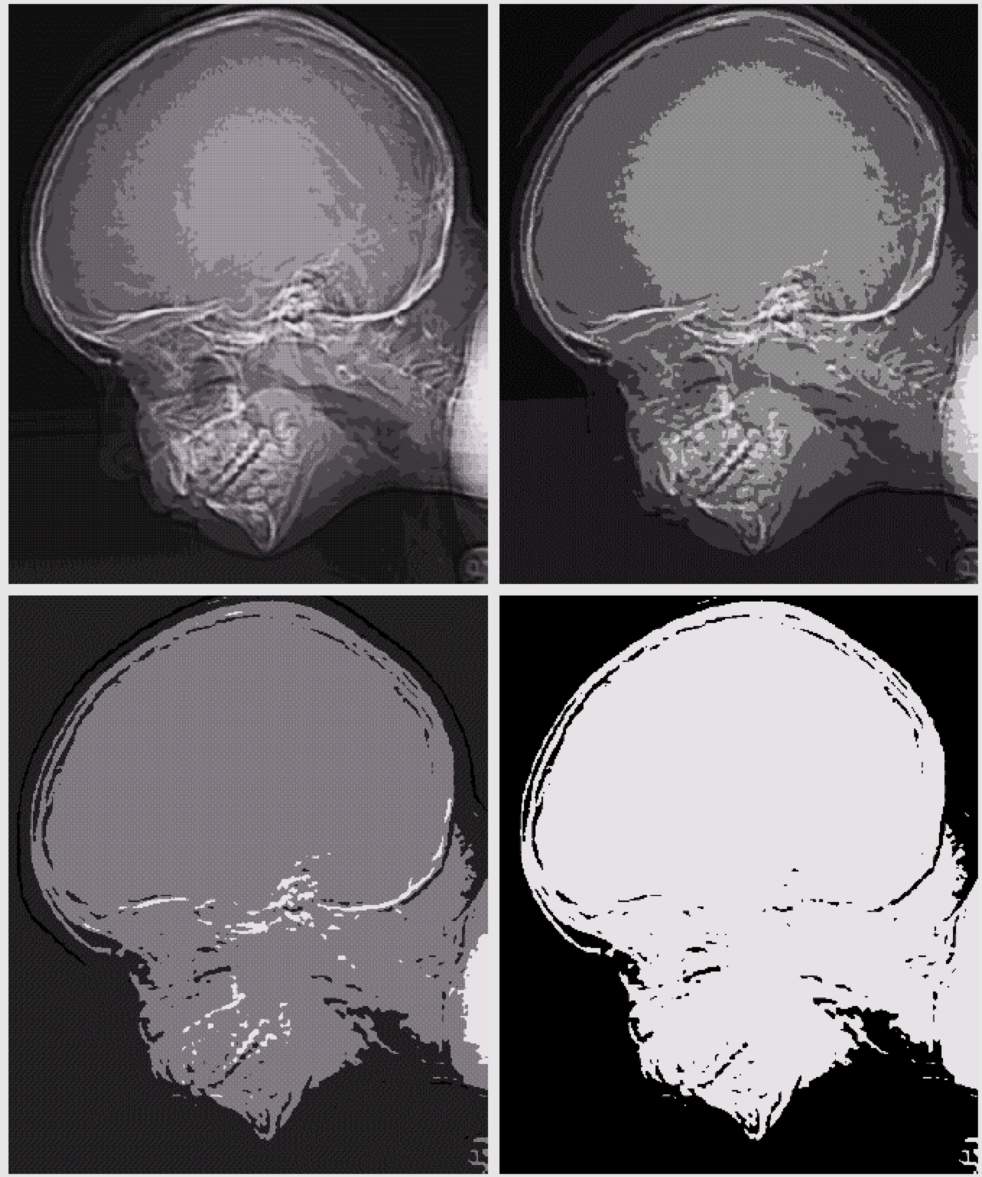
Effects of Varying the Gray Level Resolution in a Digital Image



a b
c d

FIGURE 2.21 (a) 452 × 374, 256-level image. (b)–(d) Image displayed in 128, 64, and 32 gray levels, while keeping the spatial resolution constant.

Effects of varying the number of Intensity levels in a digital Image



e f
g h

FIGURE 2.21 *(Continued)* (e)–(h) Image displayed in 16, 8, 4, and 2 gray levels. (Original courtesy of Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

Images with varying Degree of Details

a b c

FIGURE 2.22 (a) Image with a low level of detail. (b) Image with a medium level of detail. (c) Image with a relatively large amount of detail. (Image (b) courtesy of the Massachusetts Institute of Technology.)

Isopreference Curves

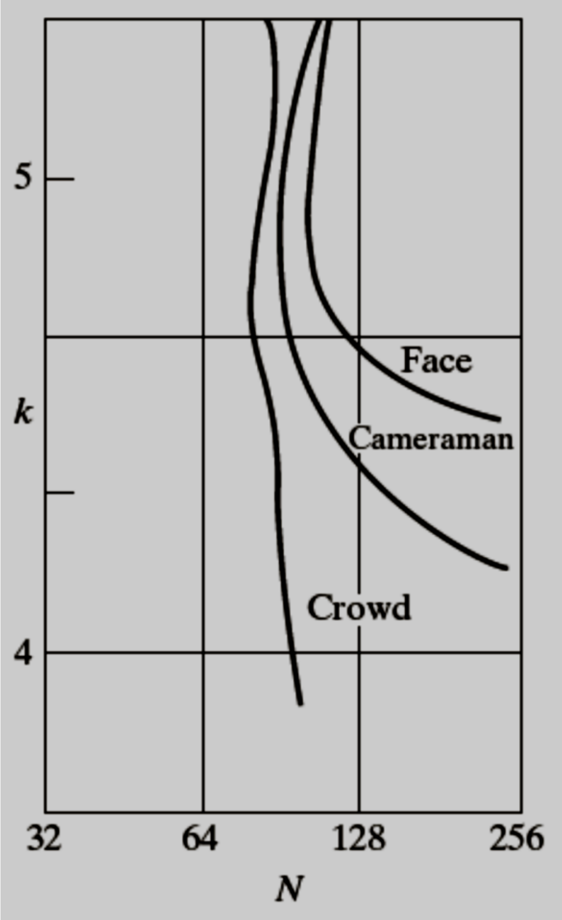


Fig (2.23) Typical Isopreference curves for 3 types of Images in Fig. (2.22)

Image Interpolation

- Image interpolation refers to the "guess" of intensity values at missing locations, i.e., x and y can be arbitrary
- Note that it is just a guess (Note that all sensors have finite sampling distance)

(1) Nearest Neighbor Interpolation
(2) Bilinear Interpolation
(3) Bicubic Interpolation
(4) Other complex techniques such as using splines and wavelets

a b c
d e f

FIGURE 2.24 (a) Image reduced to 72 dpi and zoomed back to its original size (3692 × 2812 pixels) using nearest neighbor interpolation. This figure is the same as Fig. 2.20(d). (b) Image shrunk and zoomed using bilinear interpolation. (c) Same as (b) but using bicubic interpolation. (d)–(f) Same sequence, but shrinking down to 150 dpi instead of 72 dpi [Fig. 2.24(d) is the same as Fig. 2.20(c)]. Compare Figs. 2.24(e) and (f), especially the latter, with the original image in Fig. 2.20(a).

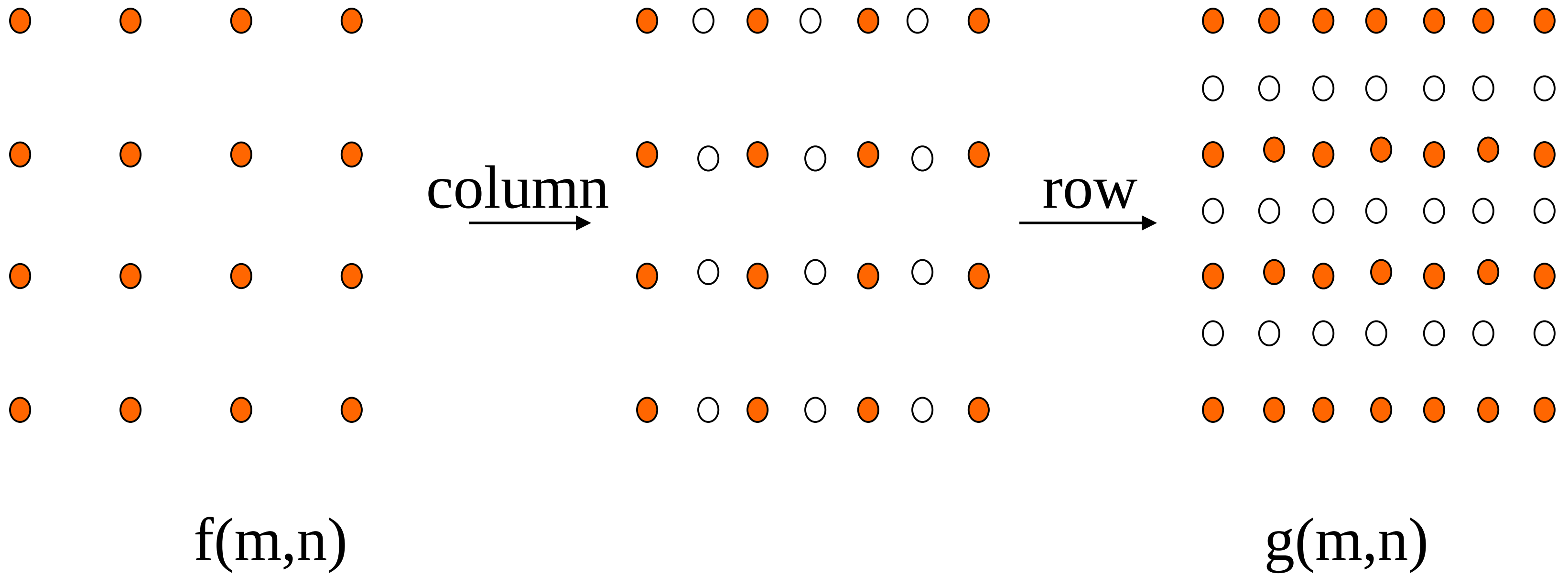
Image Interpolation

Bilinear Interpolation $v(x, y) = ax + by + cxy + d$

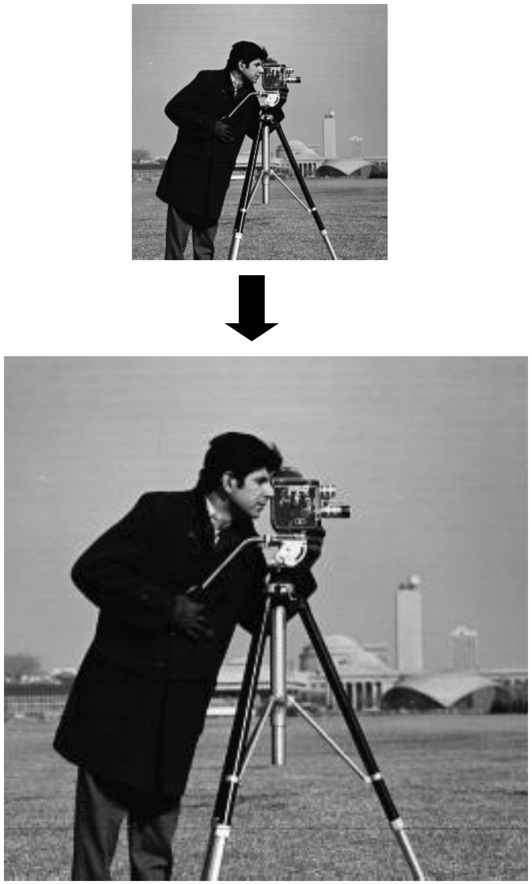
Bicubic Interpolation $v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$

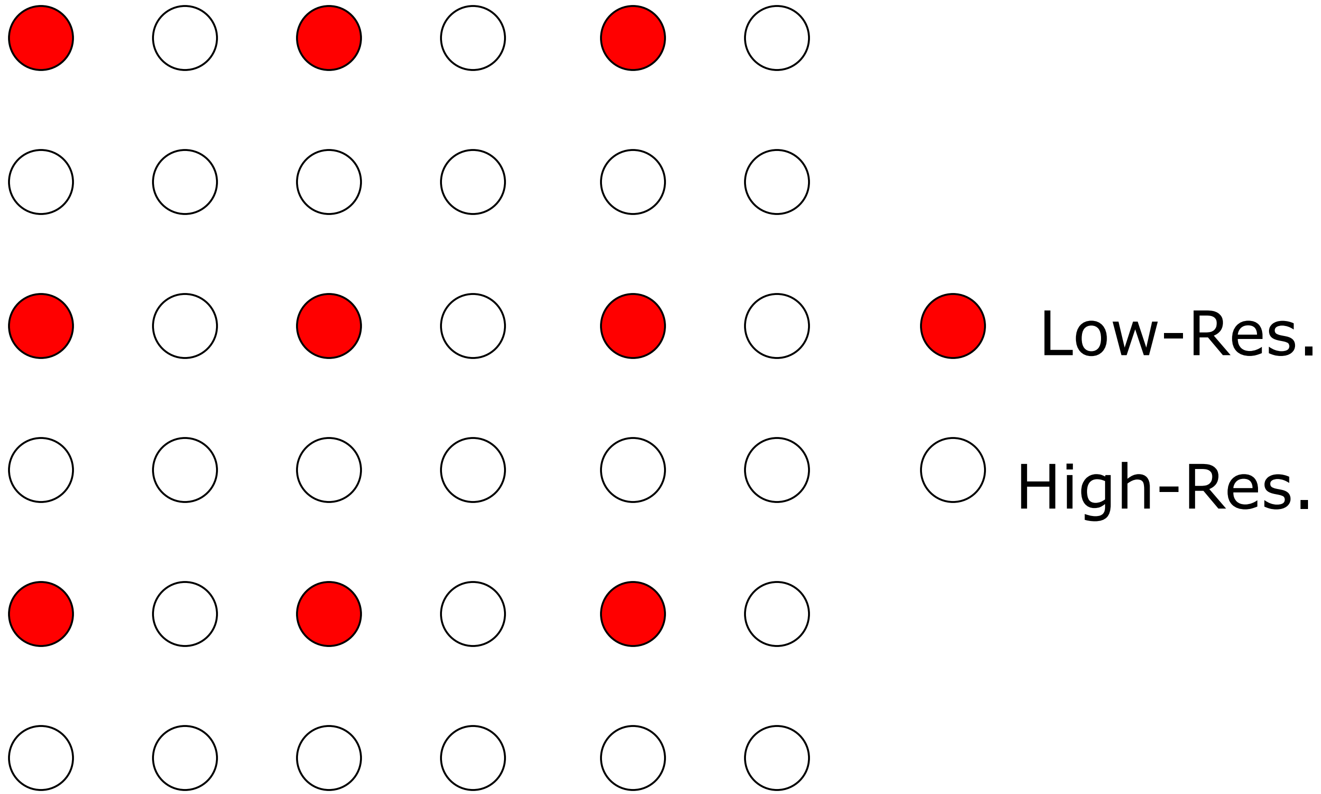
- Why do we need image interpolation?
  - We want BIG images
    - When we see a video clip on a PC, we like to see it in the full screen mode
  - We want GOOD images
    - If some block of an image gets damaged during the transmission, we want to repair it
  - We want COOL images
    - Manipulate images digitally can render fancy artistic effects as we often see in movies

Graphical Interpretation of Interpolation at Half-pel

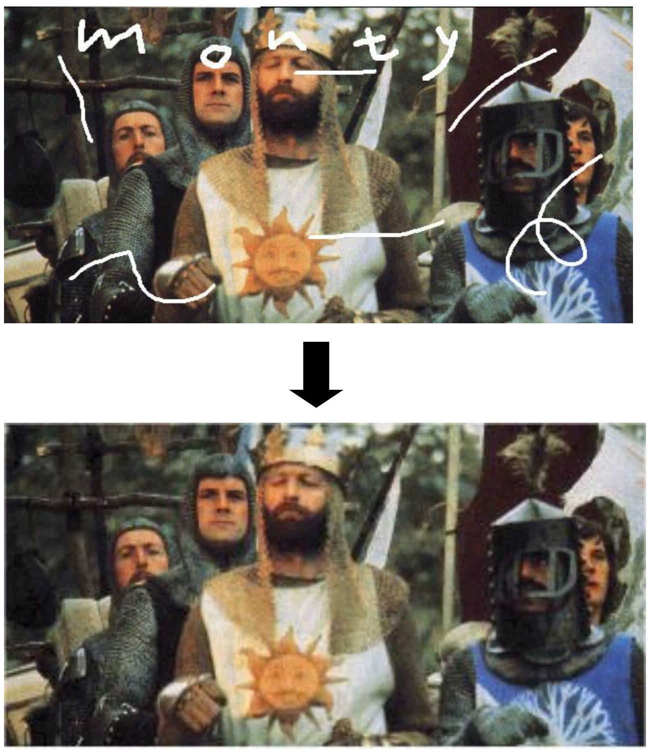


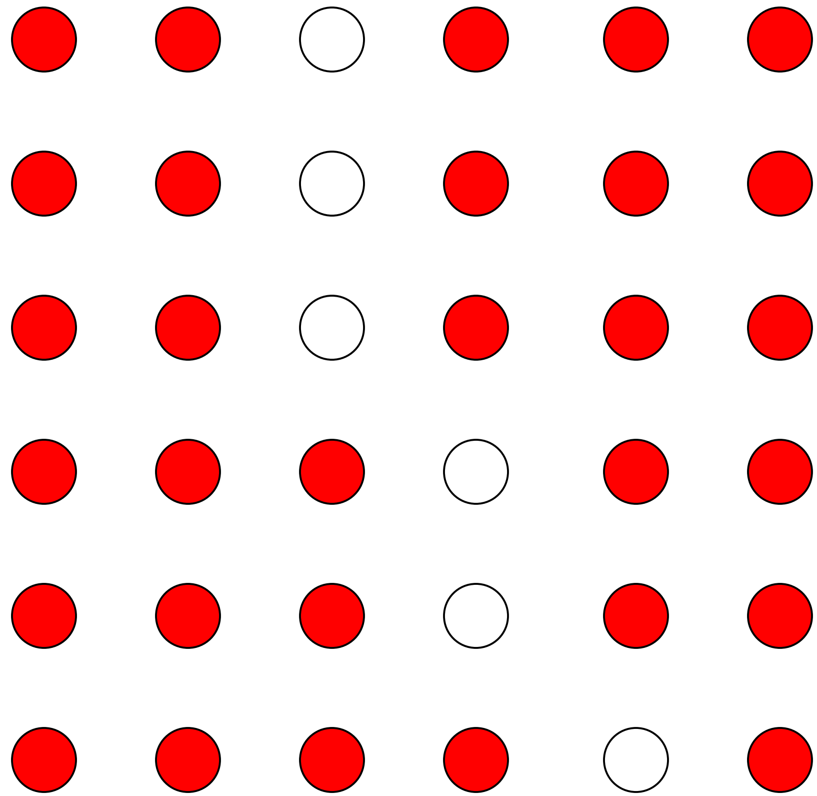
Scenario I: Resolution Enhancement





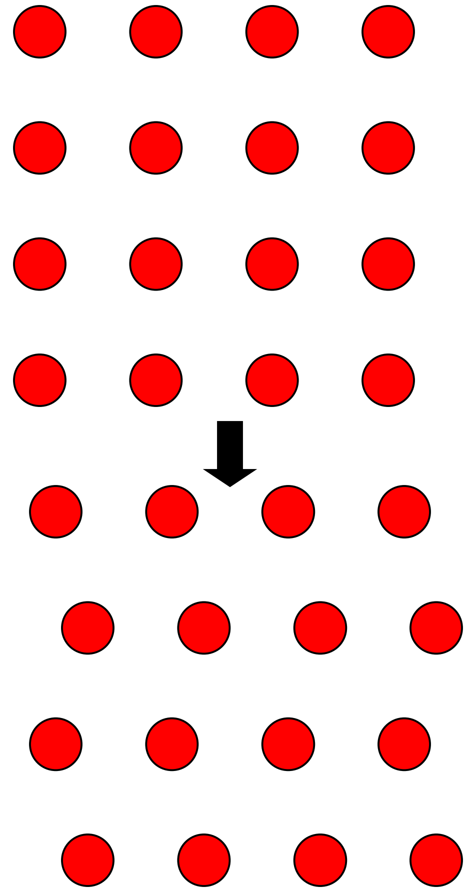
Scenario II: Image Inpainting



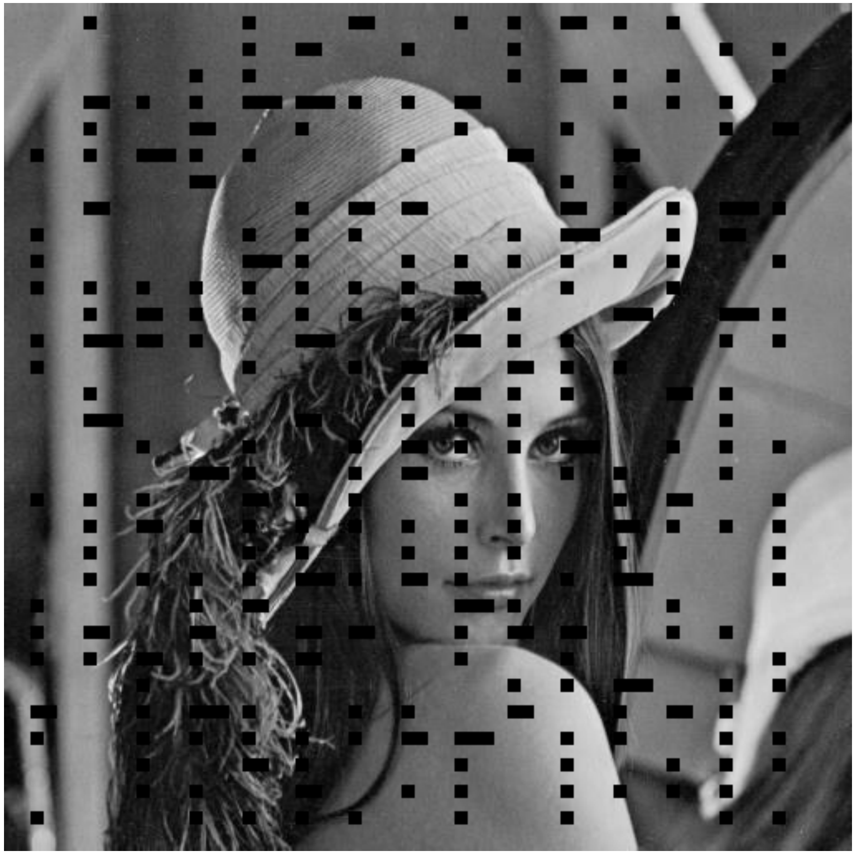


Non-damaged Damaged

Scenario III: Image Warping



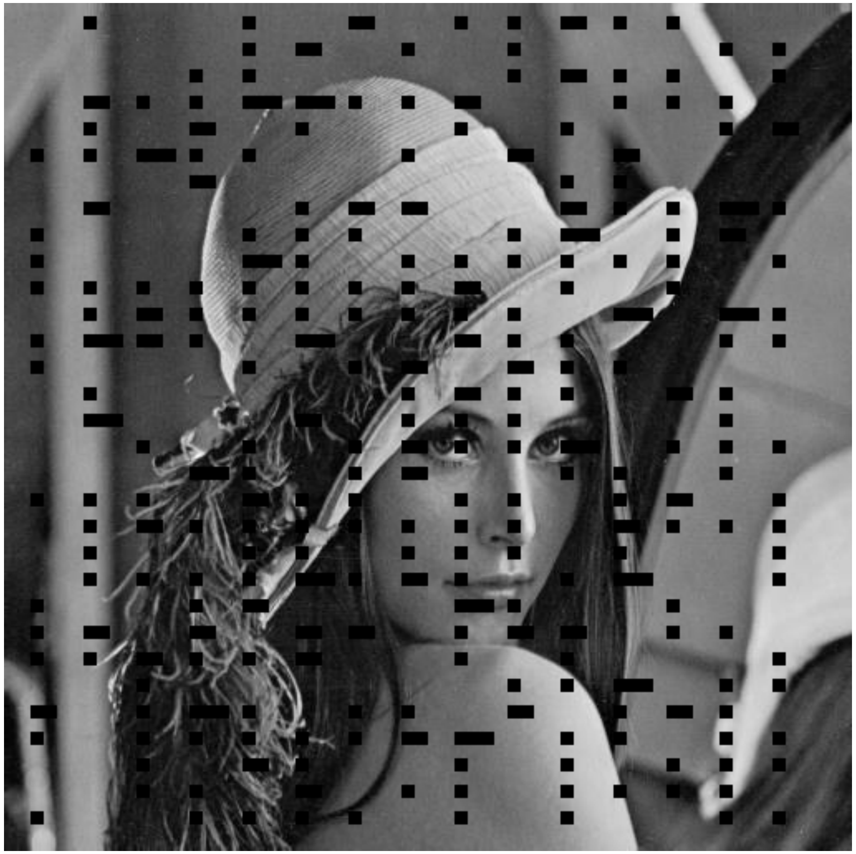
Error Concealment



damaged

interpolated

Error Concealment



damaged

interpolated

Basic Relationship between Pixels

Neighbors of a Pixel:

- **4-*neighbors***
- **Diagonal *neighbors***
- **8-*neighbors***

Adjacency, Connectivity, Regions, & Boundaries:

- **4-adjacency**
- **8-adjacency**
- **m-adjacency**

Neighbors of a Pixel

- A pixel *p* at coordinates (*x*,*y*) has four *horizontal* and *vertical* neighbors whose coordinates are given by:

  (x,y-1), (x, y+1), (x-1, y), (x+1,y)

	(x-1, y)	
(x, y-1)	*P (x, y)*	(x, y+1)
	(x+1, y)	

- This set of pixels, called the ***4-neighbors* of *p***, is denoted by $\boldsymbol{N_4(p)}$. Each pixel is one unit distance from (x, y) and some of the neighbors of p lie outside the digital image if (x, y) is on the border of the image.

Neighbors of a Pixel

- The four ***diagonal* neighbors of** *p* have coordinates:

  (x-1, y-1), (x+1, y-1), (x-1, y+1), (x+1, y+1), denoted by $\boldsymbol{N_D}$ **(*p*)**.

(x-1, y-1)		(x-1, y+1)
	P (x, y)	
(x+1, y-1)		(x+1, y+1)

- These points, together with the 4-neighbors, are called the **8-neighbors of p**, denoted by $\boldsymbol{N_8}$ **(*p*)**.

(x-1, y-1)	(x-1, y)	(x-1, y+1)
(x, y-1)	*P (x, y)*	(x, y+1)
(x+1, y-1)	(x+1, y)	(x+1, y+1)

- As before, some of the points in $N_D(p)$ and $N_8(p)$ fall outside the image if (x, y) is on the border of the image.

Adjacency and Connectivity

- Let V: a set of intensity values used to define adjacency and connectivity.
- In a binary image, $V = \{1\}$, if we are referring to adjacency of pixels with value 1.
- In a gray-scale image, the idea is the same, but V typically contains more elements, for example, $V = \{180, 181, 182, \ldots, 200\}$
- If the possible intensity values 0 – 255, V set can be any subset of these 256 values.

Types of Adjacency

1. **4-adjacency:** Two pixels p and q with values from V are 4-adjacent if q is in the set $N_4(p)$
2. **8-adjacency:** Two pixels p and q with values from V are 8-adjacent if q is in the set $N_8(p)$
3. **m-adjacency =mixed-adjacency**

Types of Adjacency

- **m-adjacency:**

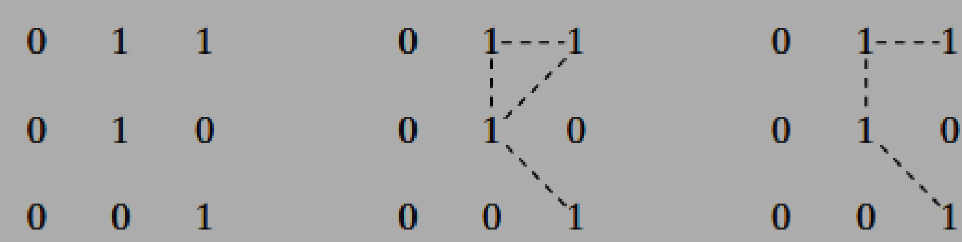
  Two pixels *p* and *q* with values from *V* are m-adjacent if :

  - q is in $N_4(p)$ or
  - q is in $N_D(p)$ and the set $N_4(p) \cap N_4(q)$ has no pixel whose values are from V (no intersection)

- **Important Note:** the type of adjacency used must be specified

Types of Adjacency

- Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used.
- For example:



a b c

FIGURE 2.26 (a) Arrangement of pixels; (b) pixels that are 8-adjacent (shown dashed) to the center pixel; (c) *m*-adjacency.

Types of Adjacency

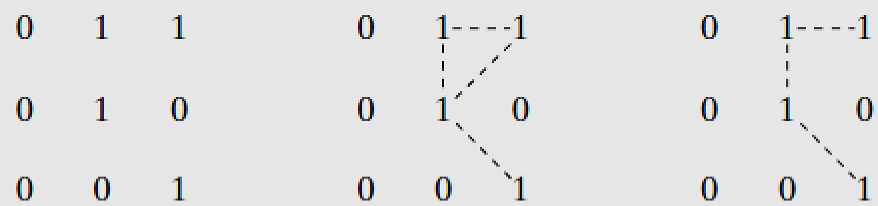
- In this example, we can note that to connect between two pixels (finding a path between two pixels):
  - In 8-adjacency way, you can find multiple paths between two pixels
  - While, in m-adjacency, you can find only one path between two pixels
- So, m-adjacency has eliminated the multiple path connection that has been generated by the 8-adjacency.
- Two subsets *S1* and *S2* are adjacent, if some pixel in *S1* is adjacent to some pixel in *S2*. Adjacent means, either 4-, 8- or m-adjacency.

A Digital Path

- A digital path (or curve) from pixel p with coordinate (x,y) to pixel q with coordinate (s,t) is a sequence of distinct pixels with coordinates (x_0,y_0), (x_1,y_1), …, (x_n, y_n) where $(x_0,y_0) = (x,y)$ and $(x_n, y_n) = (s,t)$ and pixels (x_i, y_i) and (x_{i-1}, y_{i-1}) are adjacent for $1 \le i \le n$
- n is the length of the path
- If $(x_0,y_0) = (x_n, y_n)$, the path is closed.
- We can specify 4-, 8- or m-paths depending on the type of adjacency specified.

A Digital Path

- Return to the previous example:



a b c

FIGURE 2.26 (a) Arrangement of pixels; (b) pixels that are 8-adjacent (shown dashed) to the center pixel; (c) *m*-adjacency.

In figure (b) the paths between the top right and bottom right pixels are 8-paths. And the path between the same 2 pixels in figure (c) is m-path

Connectivity

- Let *S* represent a subset of pixels in an image, two pixels *p* and *q* are said to be connected in *S* if there exists a path between them consisting entirely of pixels in *S*
- For any pixel *p* in *S*, the **set of pixels** that are **connected to p in *S*** is called **a *connected component* of *S***
- If the set S has only **one connected component**, then set *S* is called a ***connected set***

Region:

- Let *R* be a subset of pixels in an image, we call *R* a region of the image if ***R* is a connected set**
- Two regions R_i & R_j are said to be **adjacent regions** if their union makes a **connected set**
- The regions which are not adjacent are said to be **disjoint regions**
- To make sense, the **types of adjacency** must be specified
- When referring to regions, we consider **4- & 8-adjacency**

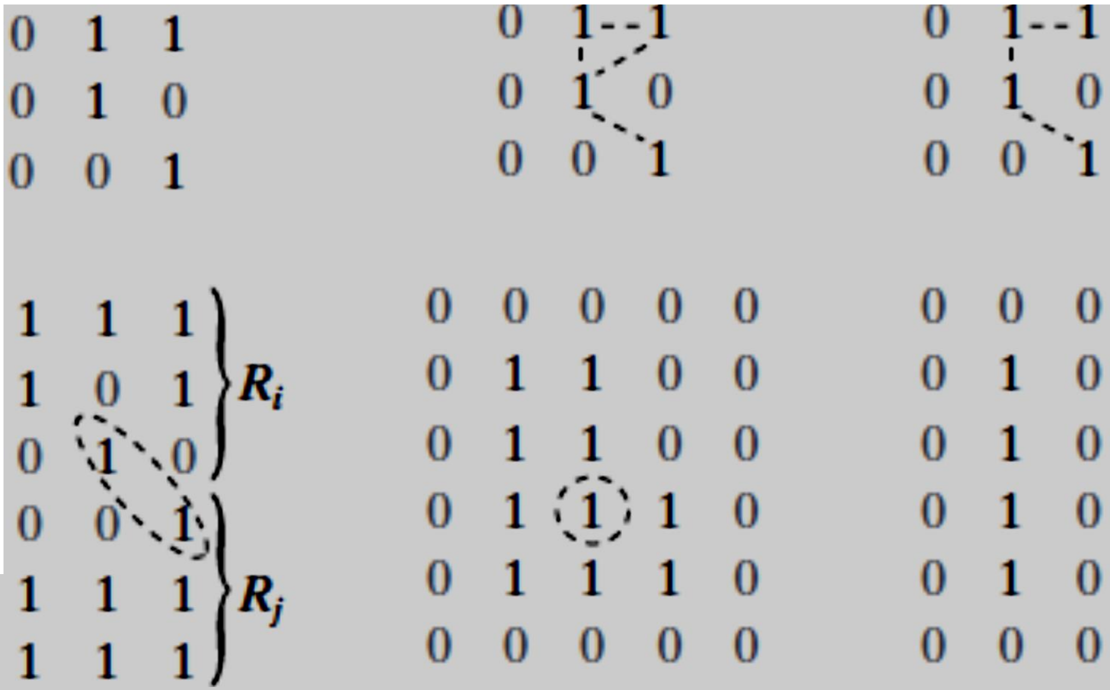
Region and Boundary (-contd.)

- If an image has **K-disjoint regions** and none of them touches the image border. Let R_u denotes the **union of all the K-disjoint regions** and $(R_u)^c$ denotes its **complement**
- Then we call the pixels in R_u the **foreground** and all pixels in $(R_u)^c$ the **background** of the image

Boundary:

- The ***boundary*** (also called *border* or *contour*) of a region *R* is the **set of pixels in the region that have one or more neighbors that are not in *R***
- If *R* happens to be an entire image, then its boundary is defined as the set of pixels in the first and last rows and columns in the image
- This extra definition is required because an image has no neighbors beyond its borders
- Normally, when we refer to a region, we are referring to subset of an image, and any pixels in the boundary of the region that happen to coincide with the border of the image are included implicitly as part of the region boundary

Region and Boundary



a b c
d e f

FIGURE 2.25 (a) An arrangement of pixels. (b) Pixels that are 8-adjacent (adjacency is shown by dashed lines; note the ambiguity). (c) *m*-adjacency. (d) Two regions (of 1s) that are adjacent if 8-adjecency is used. (e) The circled point is part of the boundary of the 1-valued pixels only if 8-adjacency between the region and background is used. (f) The inner boundary of the 1-valued region does not form a closed path, but its outer boundary does.

Distance Measures

- For pixels p, q and z, with coordinates (x,y), (s,t) and (v,w), respectively, D is a distance function if:

  (a) $D(p,q) \geq 0$ ($D(p,q) = 0$ iff $p = q$),

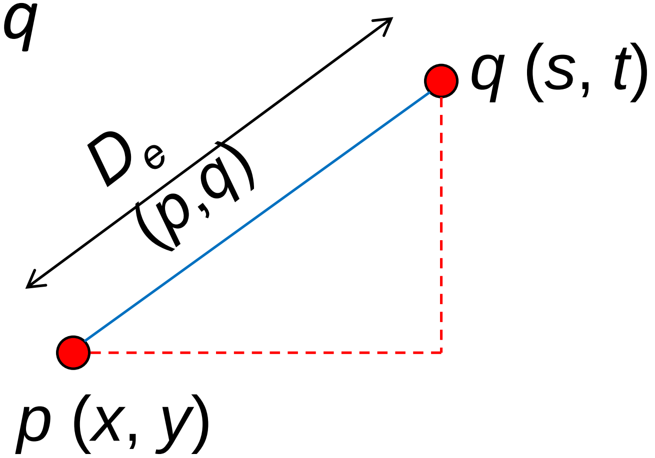
  (b) $D(p,q) = D(q, p)$, and

  (c) $D(p,z) \leq D(p,q) + D(q,z)$.

Distance Measures

- The *Euclidean Distance* between *p* and *q* is defined as:

  $D_e\,(p,q) = [(x - s)^2 + (y - t)^2]^{1/2}$



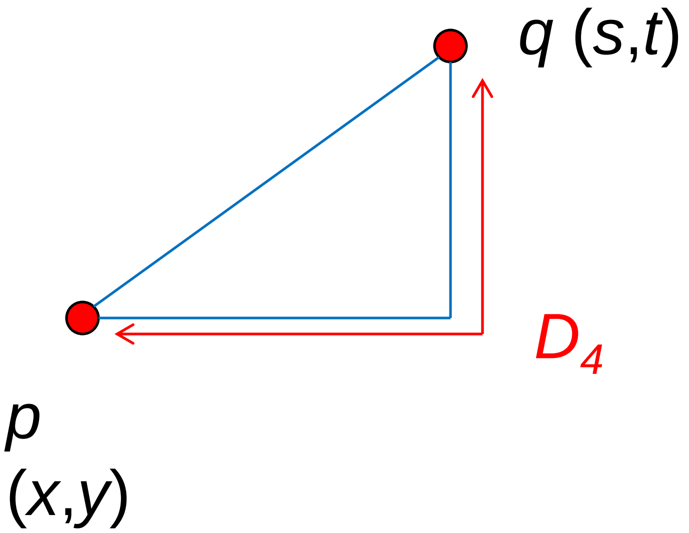
- Pixels having a distance less than or equal to some value r from (x, y) are the points contained in a disk of radius r centered at (x, y)

Distance Measures

- The D_4 *distance* (also called *city-block distance*) between p and q is defined as:

  $$D_4(p,q) = | x - s | + | y - t |$$

Pixels having a D_4 distance from (x,y), less than or equal to some value r form a Diamond centered at (x,y)



Distance Measures

Example:

The pixels with distance $D_4 \leq 2$ from (x,y) form the following contours of constant distance.

The pixels with $D_4 = 1$ are the 4-neighbors of (x,y)

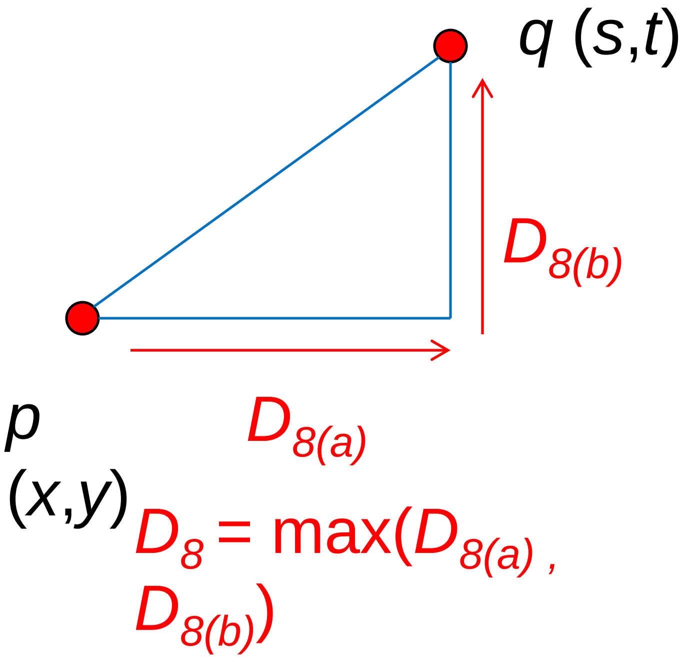
		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

Distance Measures

- The D_8 *distance* (also called *chessboard distance*) between p and q is defined as:

$$D_8(p,q) = \max(|x - s|, |y - t|)$$

Pixels having a D_8 distance from (x,y), less than or equal to some value r form a square Centered at (x,y)



Distance Measures

Example:

D_8 distance ≤ 2 from (x,y) form the following contours of constant distance.

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Distance Measures

- **Dm distance:**

  is defined as the shortest m-path between the points.

  In this case, the distance between two pixels will depend on the values of the pixels along the path, as well as the values of their neighbors.

Distance Measures

- Example:

  Consider the following arrangement of pixels and assume that p, p_2, and p_4 have value 1 and that p_1 and p_3 can have can have a value of 0 or 1

  Suppose that we consider

  the adjacency of pixels

  values 1 (i.e. $V = \{1\}$)

$$\begin{array}{ccc} & p_3 & p_4 \\ p_1 & p_2 & \\ p & & \end{array}$$

Distance Measures

- Cont. Example:

  Now, to compute the D_m between points p and p_4

  Here we have 4 cases:

  Case1: If p_1 =0 and p_3 = 0

  The length of the shortest m-path

  (the D_m *distance) is 2 (*p, p_2, p_4*)*

	0	1
0	1	
1		

Distance Measures

- Cont. Example:

Case2: If p_1 =1 and p_3 = 0

now, p_1 *and* p *will no longer be adjacent (see m-adjacency definition)*

then, the length of the shortest

path will be 3 (p, p_1, p_2, p_4)

	0	1
1	1	
1		

Distance Measures

- Cont. Example:

  Case3: If p_1 =0 and p_3 = 1

  The same applies here, and the shortest –m-path will be 3 (p, p_2, p_3, p_4)

	1	1
0	1	
1		

Distance Measures

- Cont. Example:

  Case4: If p_1 =1 and p_3 = 1

  The length of the shortest m-path will be 4 (p, p_1 , p_2, p_3, p_4)

	1	1
1	1	
1		

Mathematical Tools Used in Digital Image Processing

Array Versus Matrix Operations

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$$

Linear versus Nonlinear Operations

$$H\left[f(x, y)\right] = g(x, y)$$

$$\begin{aligned} H\left[a_i f_i(x, y) + a_j f_j(x, y)\right] &= a_i H\left[f_i(x, y)\right] + a_j H\left[f_j(x, y)\right] \\ &= a_i g_i(x, y) + a_j g_j(x, y) \end{aligned}$$

$$\begin{aligned} \sum\left[a_i f_i(x, y) + a_j f_j(x, y)\right] &= \sum a_i f_i(x, y) + \sum a_j f_j(x, y) \\ &= a_i \sum f_i(x, y) + a_j \sum f_j(x, y) \\ &= a_i g_i(x, y) + a_j g_j(x, y) \end{aligned}$$

Linear versus Nonlinear Operations

$$f_1 = \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} \quad \text{and} \quad f_2 = \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}$$

$$a_1 = 1 \text{ and } a_2 = -1$$

$$\max\left\{(1)\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} + (-1)\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = \max\left\{\begin{bmatrix} -6 & -3 \\ -2 & -4 \end{bmatrix}\right\}$$
$$= -2$$

$$(1)\max\left\{\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}\right\} + (-1)\max\left\{\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = 3 + (-1)7$$
$$= -4$$

Arithmetic Operations

$$s(x, y) = f(x, y) + g(x, y)$$
$$d(x, y) = f(x, y) - g(x, y)$$
$$p(x, y) = f(x, y) \times g(x, y)$$
$$v(x, y) = f(x, y) \div g(x, y)$$

EXAMPLE 2.5: Addition (averaging) of noisy images for noise reduction.

■ Let $g(x, y)$ denote a corrupted image formed by the addition of noise, $\eta(x, y)$, to a noiseless image $f(x, y)$; that is,

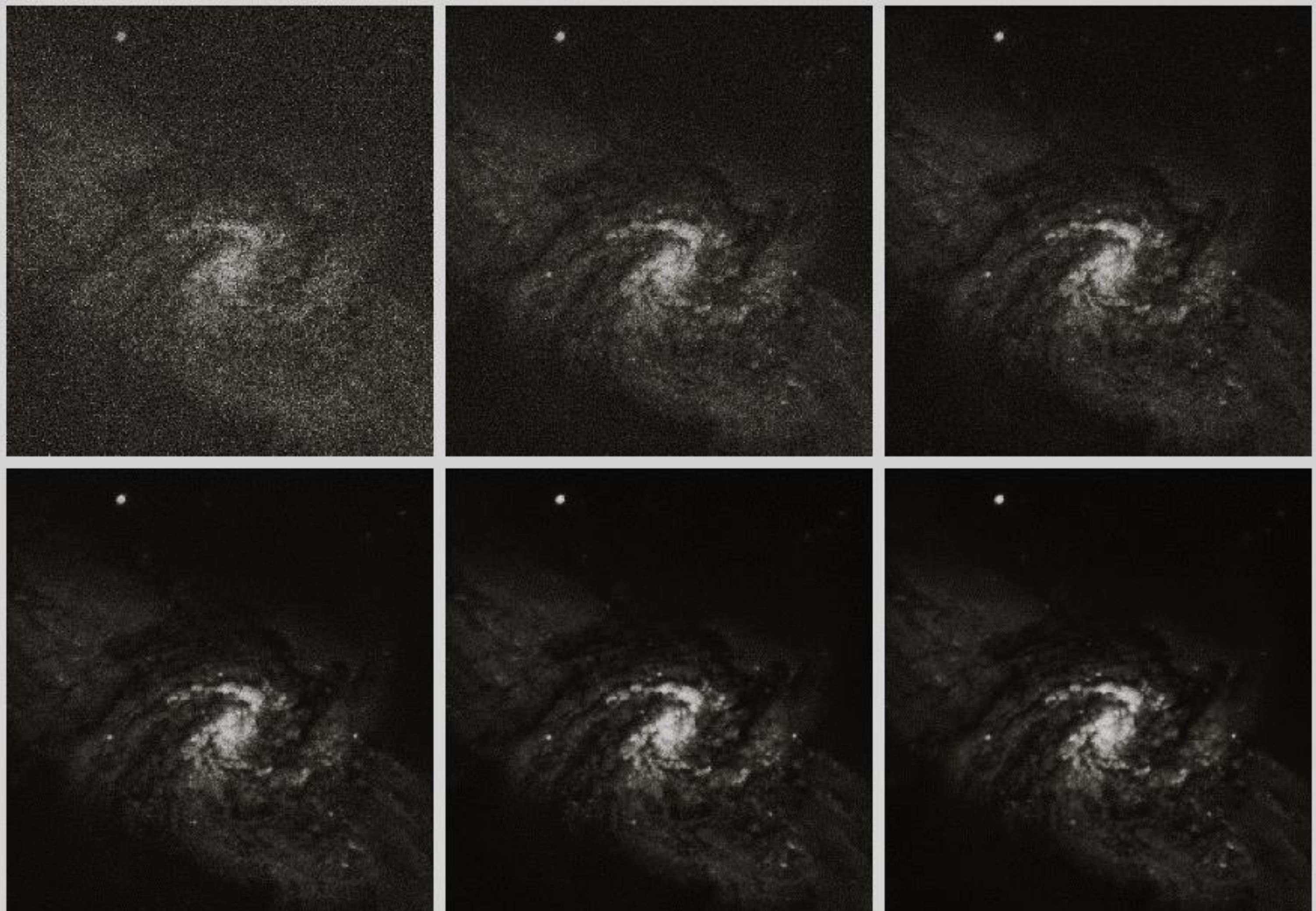
$$g(x, y) = f(x, y) + \eta(x, y)$$

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

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

$$E\{\bar{g}(x, y)\} = f(x, y)$$

$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)}$$

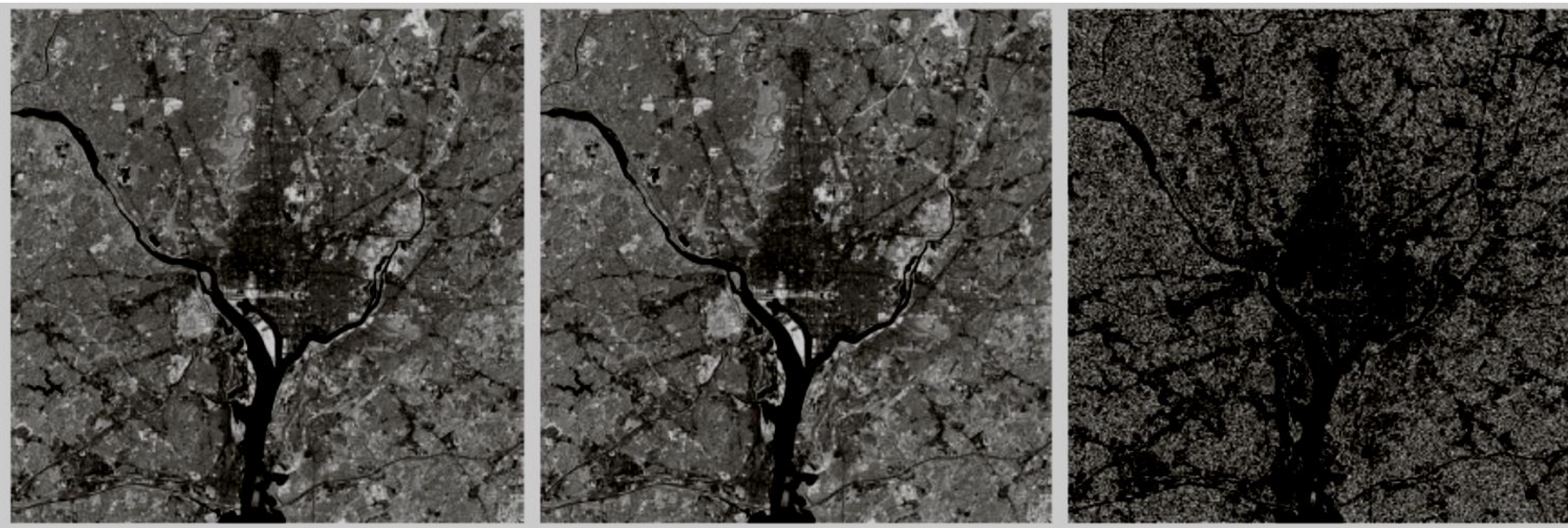


a b c
d e f

FIGURE 2.26 (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)–(f) Results of averaging 5, 10, 20, 50, and 100 noisy images, respectively. (Original image courtesy of NASA.)

EXAMPLE 2.6:
Image subtraction for enhancing differences.

$$g(x, y) = f(x, y) - h(x, y)$$

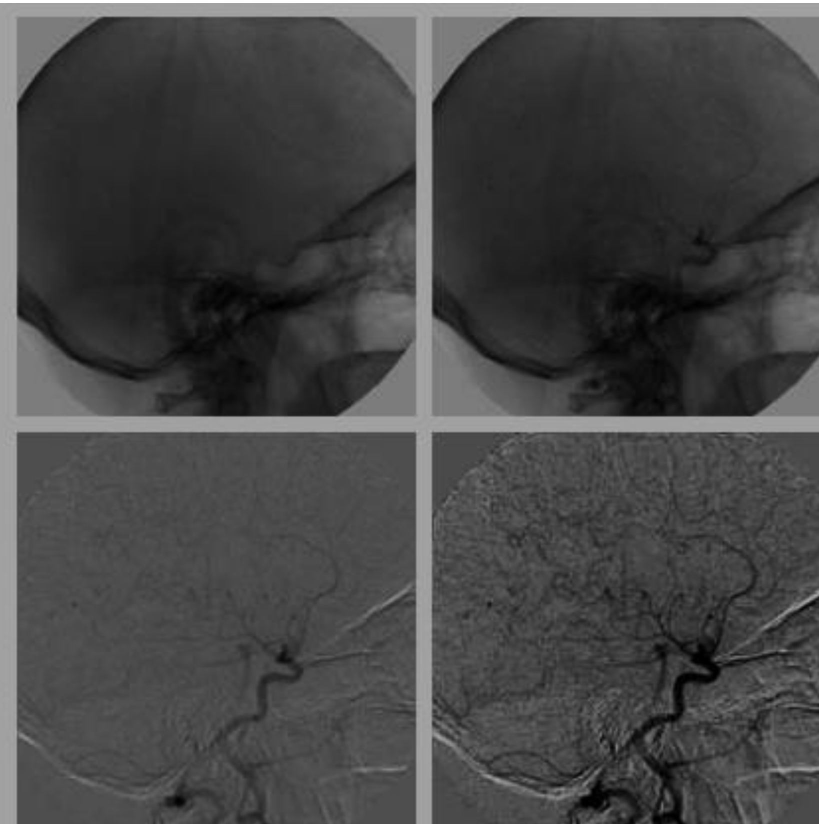


a b c

FIGURE 2.27 (a) Infrared image of the Washington, D.C. area. (b) Image obtained by setting to zero the least significant bit of every pixel in (a). (c) Difference of the two images, scaled to the range [0, 255] for clarity.

EXAMPLE 2.6: Image subtraction for enhancing differences.

$$g(x, y) = f(x, y) - h(x, y)$$

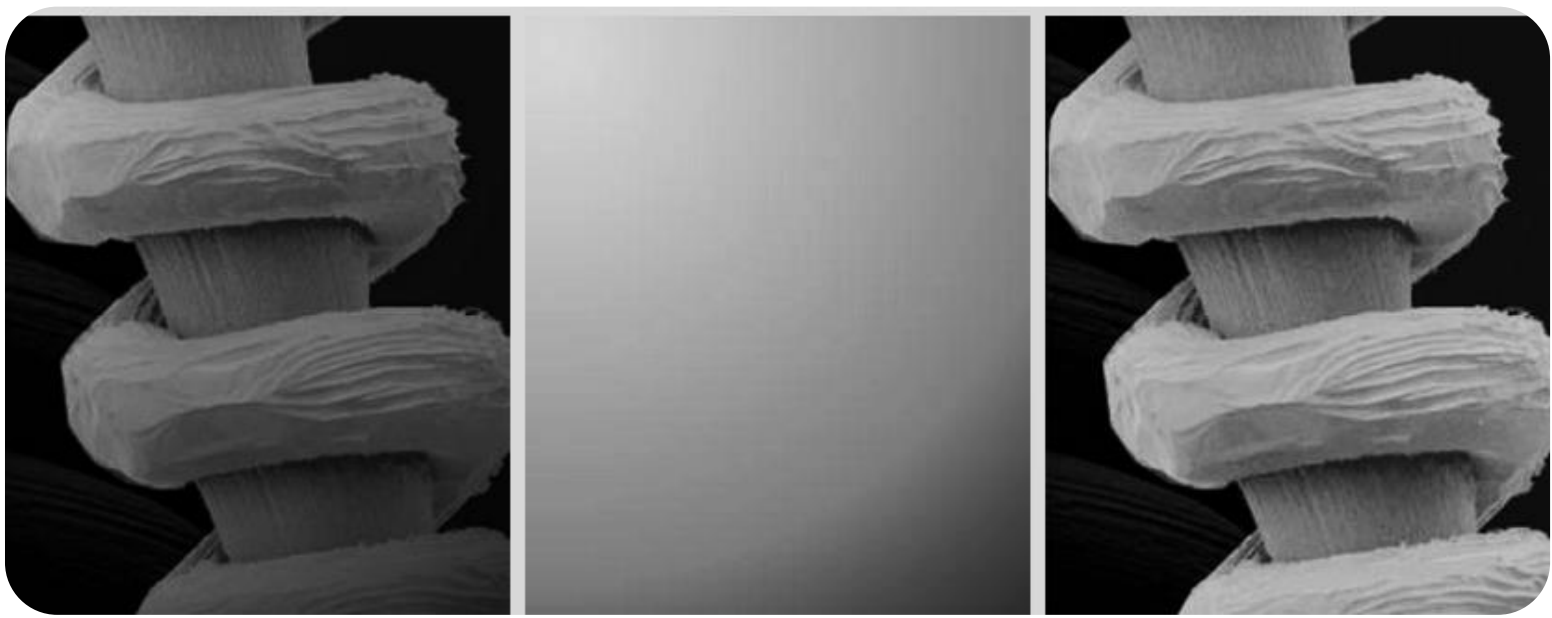


a b
c d

FIGURE 2.28 Digital subtraction angiography. (a) Mask image. (b) A live image. (c) Difference between (a) and (b). (d) Enhanced difference image. (Figures (a) and (b) courtesy of The Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)

EXAMPLE 2.7: Using image multiplication and division for shading correction.

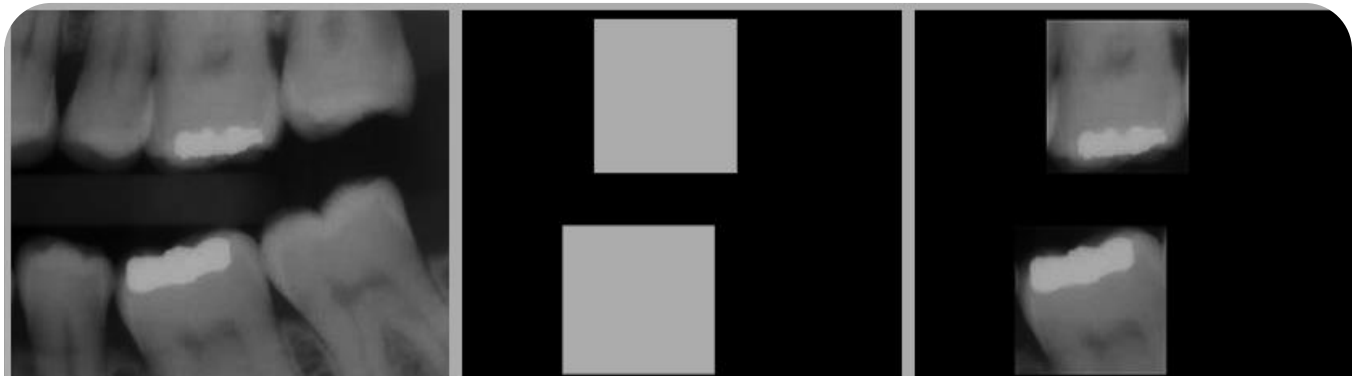
$$g(x, y) = f(x, y)h(x, y)$$



EXAMPLE 2.7: Using image multiplication and division for shading correction.

Another common use of image multiplication is in *masking*, also called *region of interest* (ROI), operations. The process, illustrated in Fig. 2.30, consists simply of multiplying a given image by a mask image that has 1s in the ROI and 0s elsewhere. There can be more than one ROI in the mask image, and the shape of the ROI can be arbitrary, although rectangular shapes are used frequently for ease of implementation.

$$g(x, y) = f(x, y)h(x, y)$$



a b c

FIGURE 2.30 (a) Digital dental X-ray image. (b) ROI mask for isolating teeth with fillings (white corresponds to 1 and black corresponds to 0). (c) Product of (a) and (b).

- For a given image *f*, an approach that guarantees the full range of an arithmetic operation between captured images into a fixed number of bits is as follows:
- First, we perform the operation $f_m = f - \min(f)$, which creates an image whose minimum value is 0
- Then, we perform the operation $f_s = K\left[f_m/\max(f_m) \right]$, which creates a scaled image, f_s, whose values are in the range [0, *K*]
- When working with 8-bit images, setting K=255 gives us a scaled image whose intensities span the full 8-bit scale from 0 to 255
- When performing division, we have the extra requirement that a small number should be added to the pixels of the divisor image to avoid division by 0

SET & LOGICAL OPERATIONS

Basic Set Operations

$a \in A$

$a \notin A$

$A \subseteq B$

$C = A \cup B$

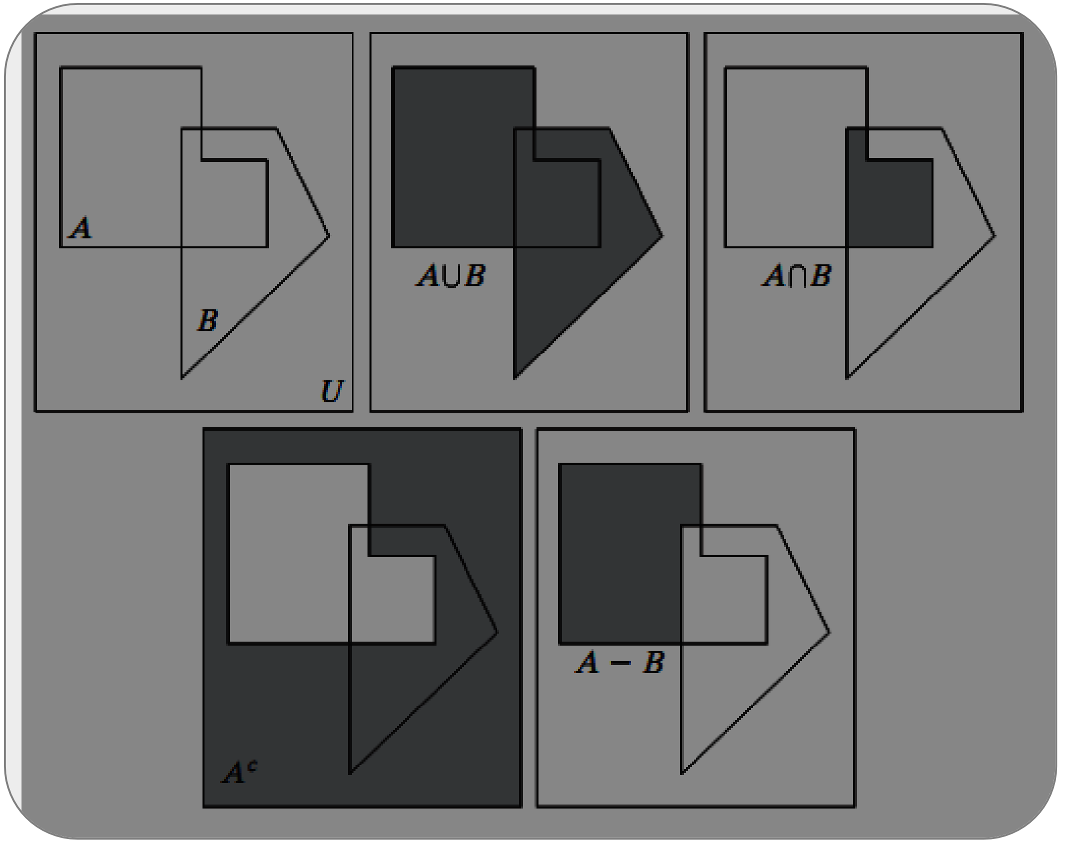
$D = A \cap B$

$A \cap B = \varnothing$

$A^c = \{w \mid w \notin A\}$

$A^c = U - A$

$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c$



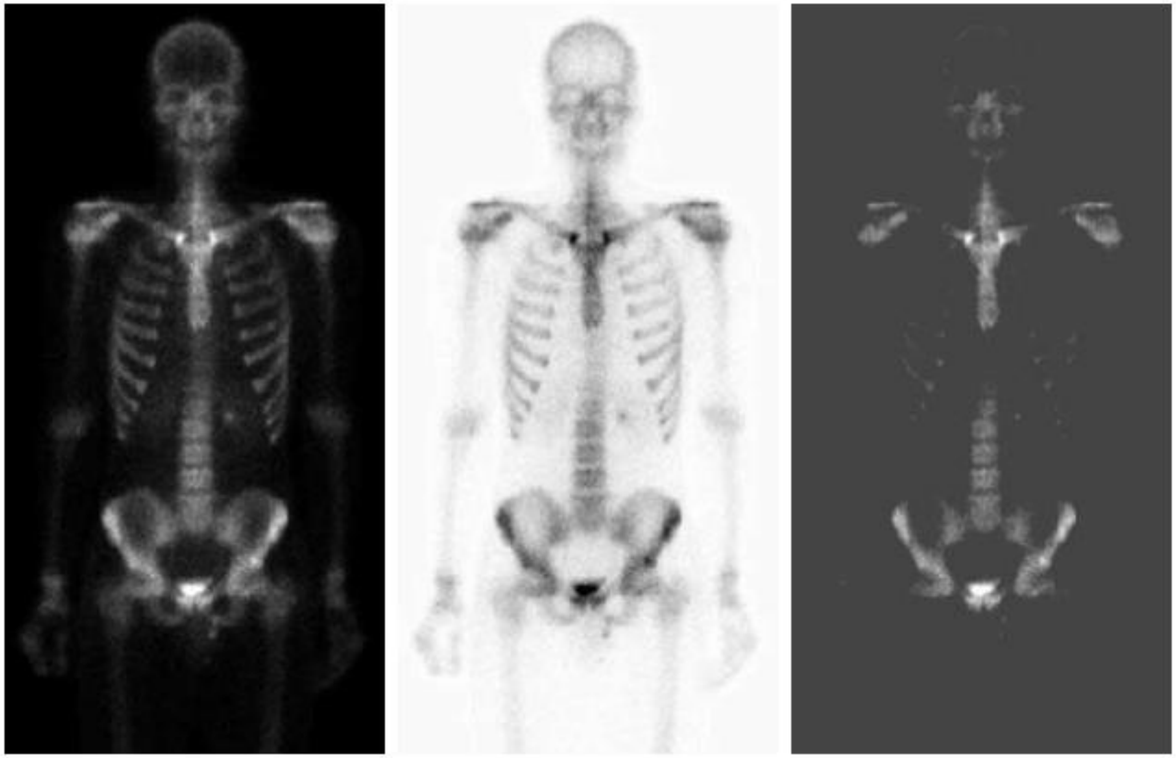
EXAMPLE 2.8: Set operations involving image intensities

Let the elements of a gray scale image be represented by a set *A* whose elements are triplets of the form (x, y, z), where *x* and *y* are spatial coordinates and *z* denotes intensity. We can define the *complement* of *A* as the set $A^c = \{(x, y, K\text{-}z) \mid (x, y, z) \in A\}$ which simply denotes the set of pixels of *A* whose intensities have been subtracted from a constant *K*. This constant is equal to $2^k - 1$ where *k* is the number of intensity bits used to represent *z*. Let *A* denote the 8-bit gray scale image in Fig. 2.32(a), and suppose that we want to form the negative of *A* using set operations. Figure 2.32(b) shows the result. The union of two gray-scale sets *A* and *B* may be defined as the set

$$A \cup B = \{\max_z (a, b) \mid a \in A, b \in B\}$$

a b c

FIGURE 2.32 Set operations involving gray-scale images. (a) Original image. (b) Image negative obtained using set complementation. (c) The union of (a) and a constant image. (Original image courtesy of G.E. Medical Systems.)



Logical Operations

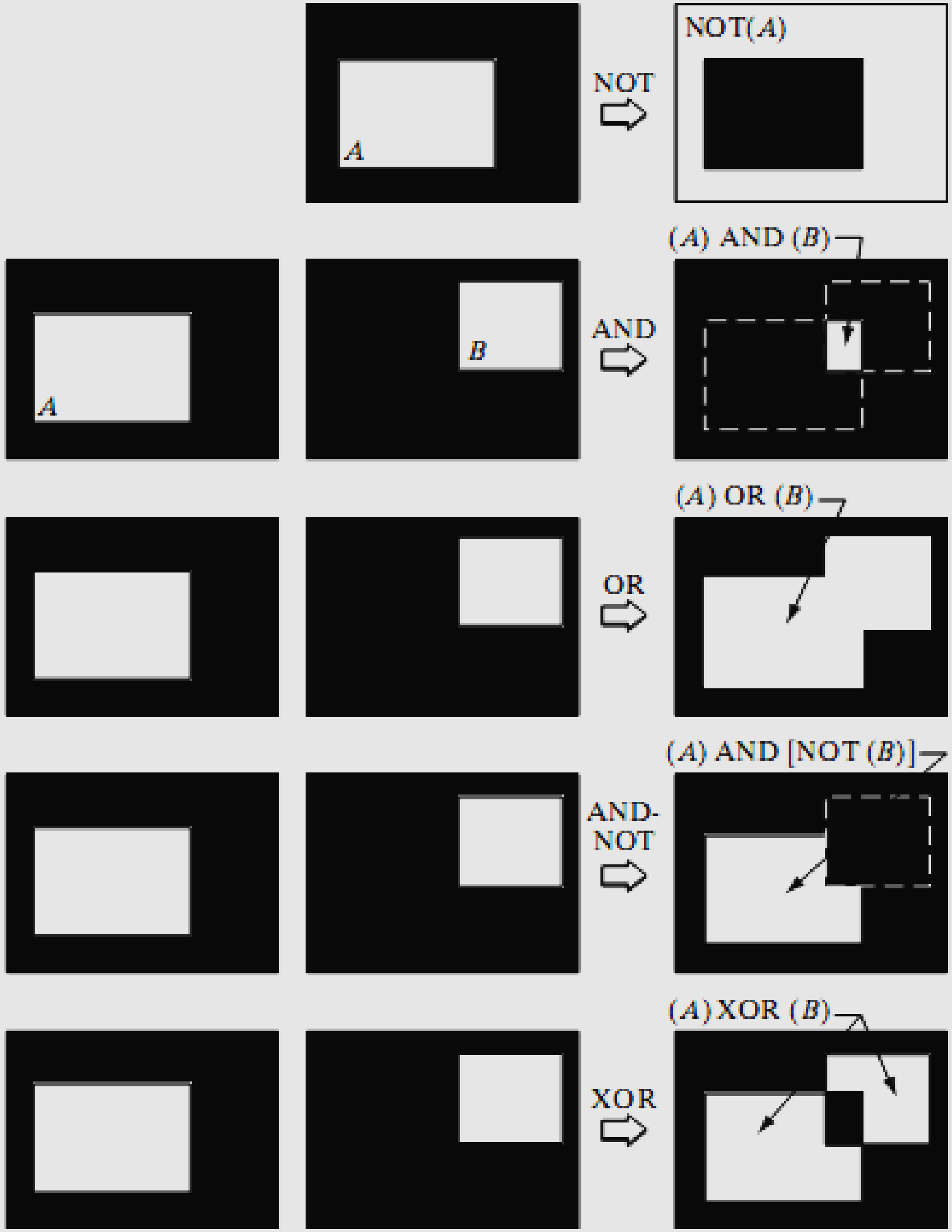


FIGURE 2.33 Illustration of logical operations involving foreground (white) pixels. Black represents binary 0s and white binary 1s. The dashed lines are shown for reference only. They are not part of the result.

Spatial Operations

- Spatial operations are performed directly on the pixels of a given image
- We classify spatial operations into three broad categories:
- (1) single-pixel operations, (2) neighborhood operations, and (3) geometric spatial transformations

$$s = T(z)$$

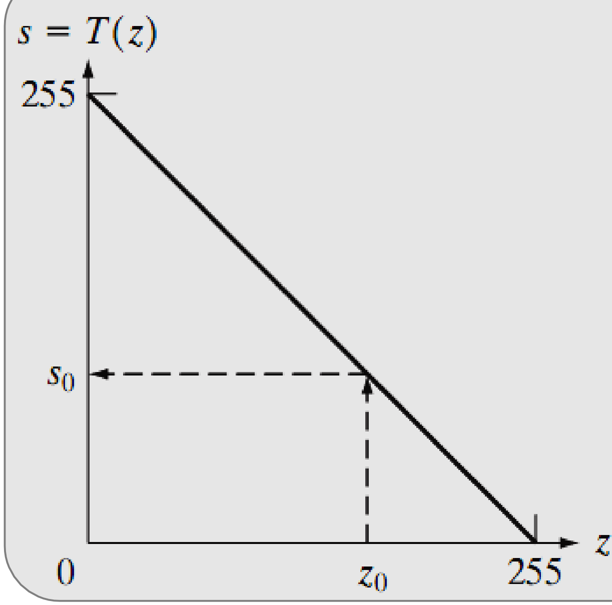
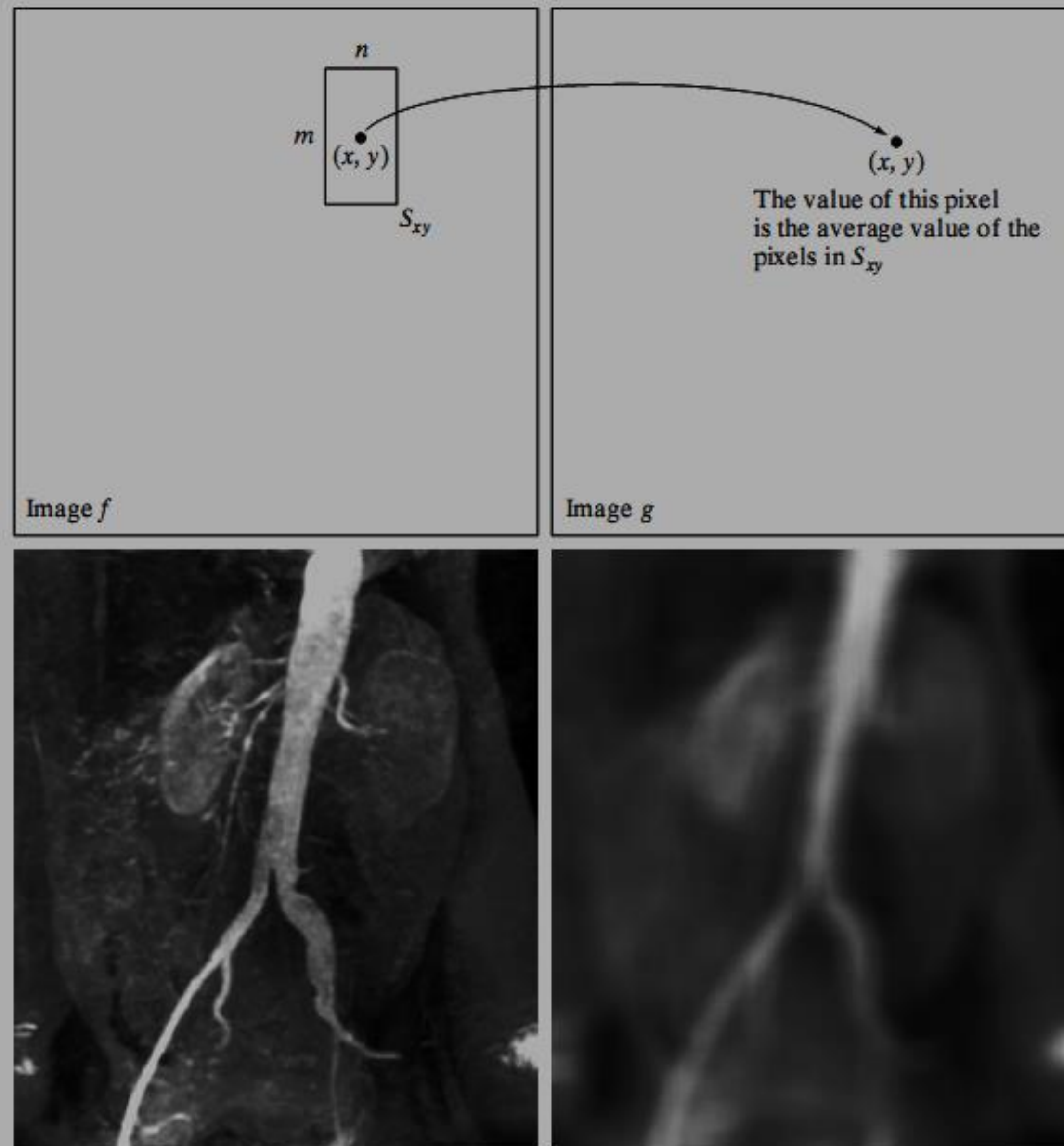


FIGURE 2.34 Intensity transformation function used to obtain the negative of an 8-bit image. The dashed arrows show transformation of an arbitrary input intensity value z_0 into its corresponding output value s_0.

Spatial Operations

$$g(x, y) = \frac{1}{mn} \sum_{(r,c) \in S_{xy}} f(r, c)$$



a b
c d

FIGURE 2.35 Local averaging using neighborhood processing. The procedure is illustrated in (a) and (b) for a rectangular neighborhood. (c) The aortic angiogram discussed in Section 1.3.2. (d) The result of using Eq. (2.6-21) with $m = n = 41$. The images are of size 790×686 pixels.

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

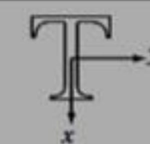
$$(x, y) = T\{(v, w)\} = (v/2, w/2)$$

Affine Transform

$$[x \;\; y \;\; 1] = [v \;\; w \;\; 1]\,\mathbf{T} = [v \;\; w \;\; 1]\begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix}$$

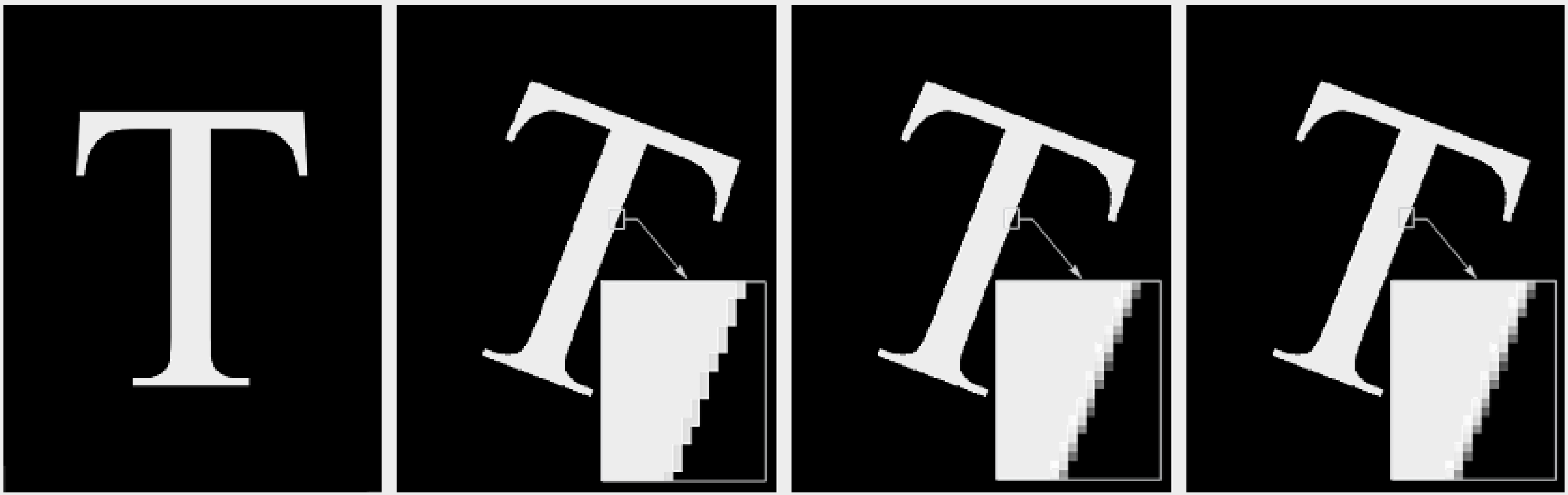
- For assigning intensity levels, we consider nearest neighbour, bilinear, and bicubic interpolation techniques when working with these transformations

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

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

EXAMPLE 2.9: Image rotation and intensity interpolation

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



a b c d

FIGURE 2.36 (a) A 300 dpi image of the letter T. (b) Image rotated 21° using nearest neighbor interpolation to assign intensity values to the spatially transformed pixels. (c) Image rotated 21° using bilinear interpolation. (d) Image rotated 21° using bicubic interpolation. The enlarged sections show edge detail for the three interpolation approaches.

EXAMPLE 2.10: Image Registration

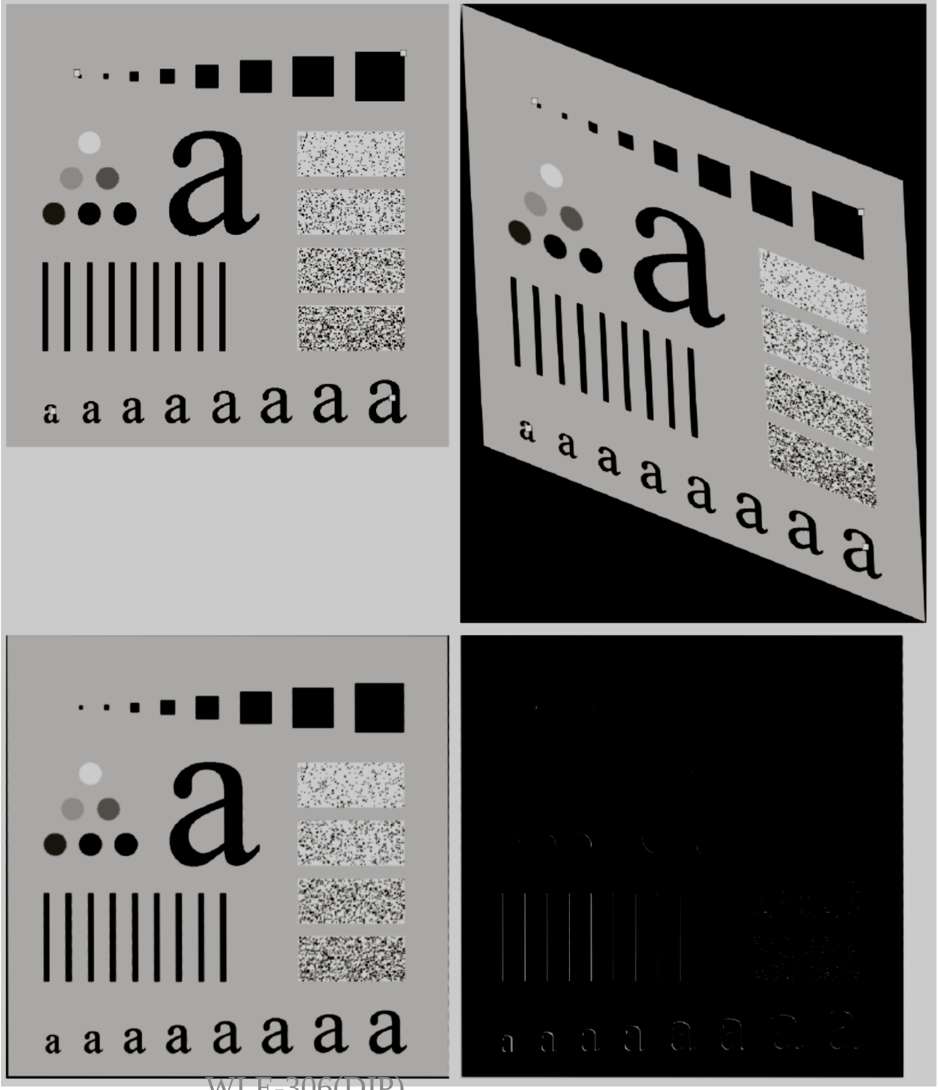
- Image registration is an important application of digital image processing used to align two or more images of the same scene
- In image registration, we have available the input and output images, but the specific transformation that produced the output image from the input generally is unknown
- the input image is the image that we wish to transform, and what we call the *reference* image is the image against which we want to register the input

- The distortion may be caused by the images were taken at different times using the same instrument, such as satellite images of a given location taken several days, months, or even years apart
- In either case, combining the images or performing quantitative analysis and comparisons between them requires compensating for geometric distortions caused by differences in viewing angle, distance, and orientation; sensor resolution; shift in object positions; and other factors.

EXAMPLE 2.10: Image Registration

$$x = c_1 v + c_2 w + c_3 vw + c_4$$

$$y = c_5 v + c_6 w + c_7 vw + c_8$$



a b
c d

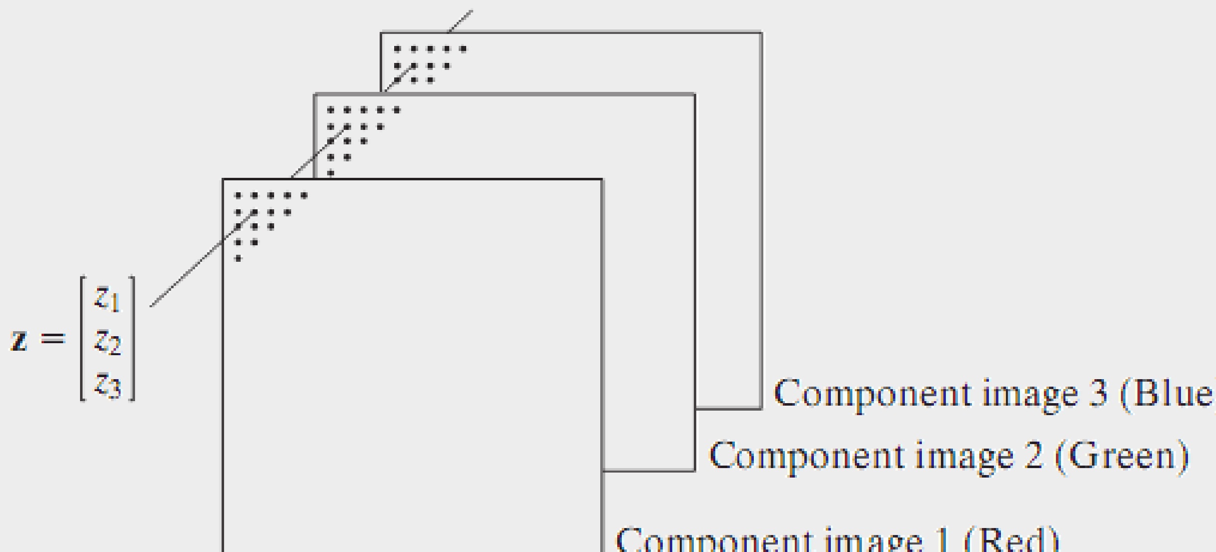
FIGURE 2.37 Image registration. (a) Reference image. (b) Input (geometrically distorted image). Corresponding tie points are shown as small white squares near the corners. (c) Registered image (note the errors in the borders). (d) Difference between (a) and (c), showing more registration errors.

Vector and Matrix Operations

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$$

$$D(\mathbf{z}, \mathbf{a}) = \left[(\mathbf{z} - \mathbf{a})^T(\mathbf{z} - \mathbf{a})\right]^{\frac{1}{2}}$$
$$= \left[(z_1 - a_1)^2 + (z_2 - a_2)^2 + \cdots + (z_n - a_n)^2\right]^{\frac{1}{2}}$$

FIGURE 2.38 Formation of a vector from corresponding pixel values in three RGB component images.



$$\mathbf{w} = \mathbf{A}(\mathbf{z} - \mathbf{a})$$

$$\mathbf{g} = \mathbf{H}\mathbf{f} + \mathbf{n}$$

Image Transforms

$$T(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) r(x,y,u,v)$$

for u=0,1,2, ... , M-1 and v=0,1,2, ..., N-1, where r(x,y,u,v) is called the *forward transformation kernel*.

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u,v) s(x,y,u,v)$$

for x=0,1,2, ... , M-1 and y=0,1,2, ..., N-1, where s(x,y,u,v) is called the *inverse transformation kernel*.

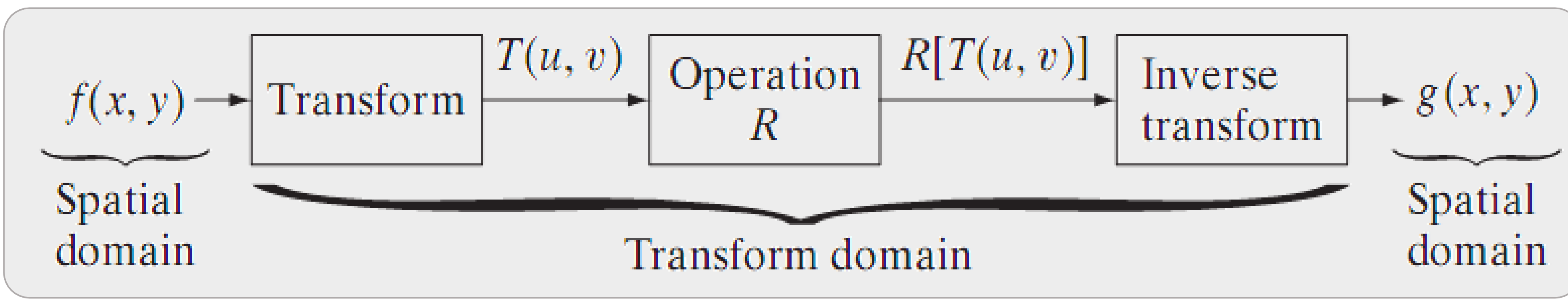


Image Transforms

- The forward transformation kernel is said to be *separable* if

$$r(x, y, u, v) = r_1(x, u)\, r_2(y, v)$$

- In Addition, the kernel $r_1(x, y)$ is said to be *symmetric* if is functionally equal to $r_2(x, y)$ so that

$$r(x, y, u, v) = r_1(x, u)\, r_1(y, v)$$

$$r(x, y, u, v) = e^{-j2\pi(ux/M + vy/N)}$$

$$s(x, y, u, v) = \frac{1}{MN} e^{j2\pi(ux/M + vy/N)}$$

$$T(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y)\, e^{-j2\pi(ux/M + vy/N)}$$

$$f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u, v)\, e^{j2\pi(ux/M + vy/N)}$$

- T=AFA
- BTB = BAFAB
- If $B=A^{-1}$, then F=BTB

When the forward and inverse kernels of a transform pair satisfy the conditions of separability and symmetry, and f(x,y) is a square image of size MxM, then transform and inverse transform can be expressed in matrix form:

$$T=AFA$$

where **F** is an M x M matrix containing the elements of f(x,y),
A is an MxM matrix with elements $a_{ij} = r_1(i, j)$, and **T** is the resulting MxM transform, with values T(u,v) for u,v =0,1,2, …, M-1
To obtain inverse transform, we pre- and post-multiply the above eqn. by an inverse transformation matrix B

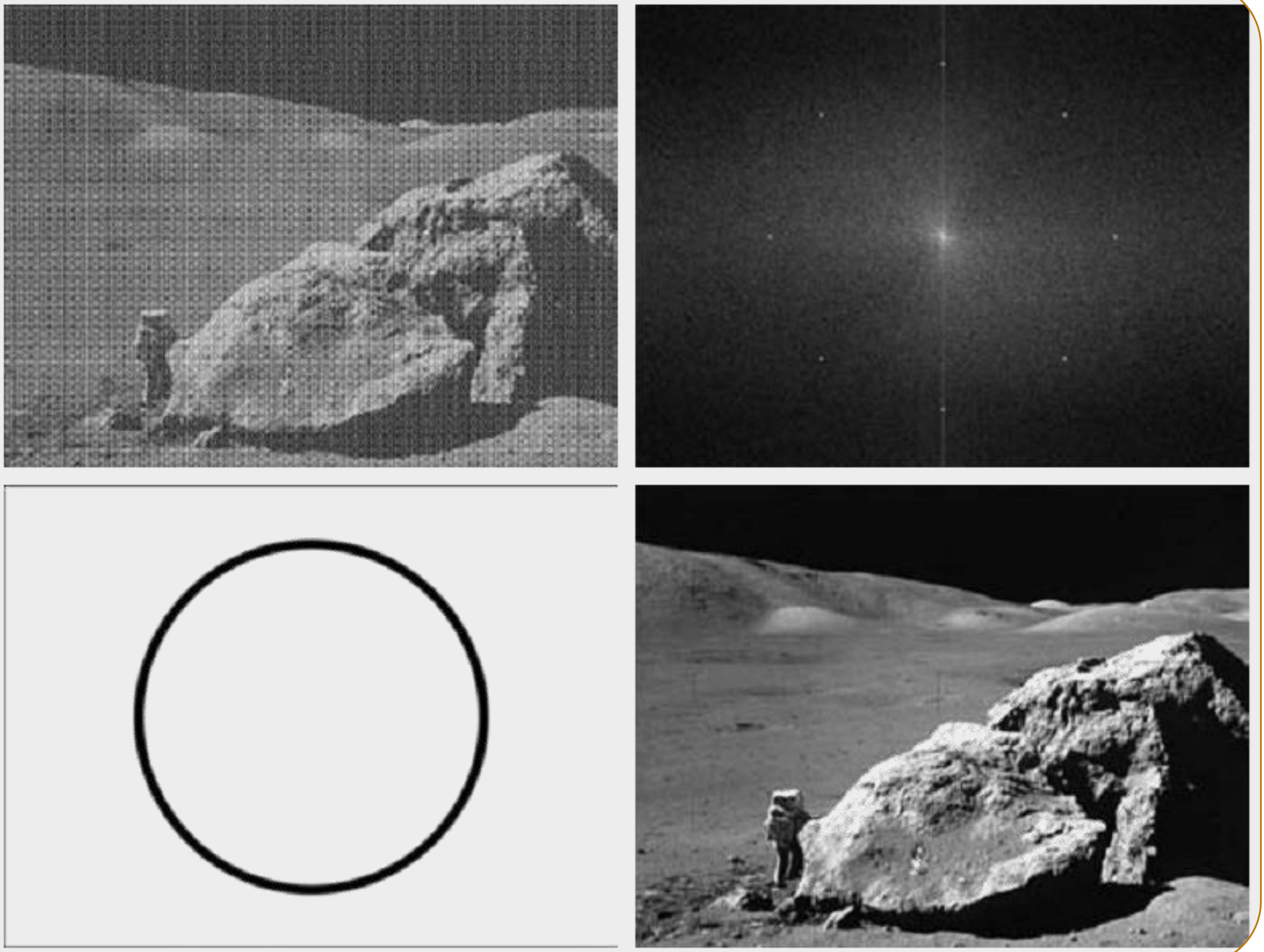
$$BTB = BAFAB$$

If $B=A^{-1}$,

then $F=BTB$

$$\hat{\mathbf{F}} = \mathbf{BAFAB}$$

EXAMPLE 2.11: Image processing in the transform domain



a b
c d

FIGURE 2.40
(a) Image corrupted by sinusoidal interference. (b) Magnitude of the Fourier transform showing the bursts of energy responsible for the interference. (c) Mask used to eliminate the energy bursts. (d) Result of computing the inverse of the modified Fourier transform. (Original image courtesy of NASA.)

Probabilistic Methods

$$p(z_k) = \frac{n_k}{MN}$$

$$\sigma^2 = \sum_{k=0}^{L-1} (z_k - m)^2 p(z_k)$$

$$\sum_{k=0}^{L-1} p(z_k) = 1$$

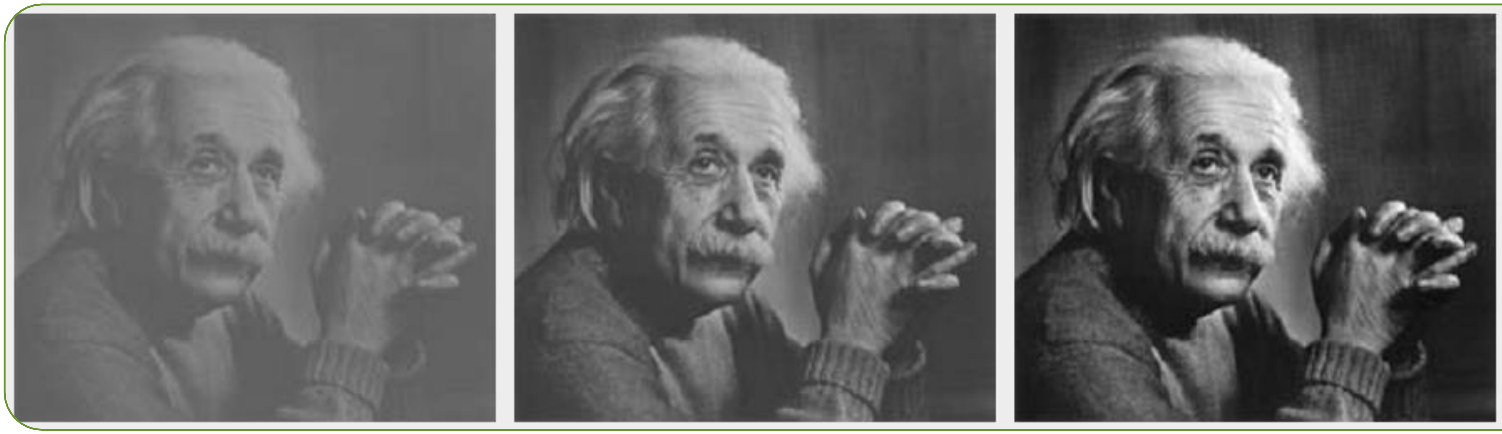
$$\mu_n(z) = \sum_{k=0}^{L-1} (z_k - m)^n p(z_k)$$

$$m = \sum_{k=0}^{L-1} z_k p(z_k)$$

We see that $\mu_0(z) = 1$, $\mu_1(z) = 0$, and $\mu_2(z) = \sigma^2$. Whereas the mean and variance have an immediately obvious relationship to visual properties of an image, higher-order moments are more subtle. For example, a positive third moment indicates that the intensities are biased to values higher than the mean, a negative third moment would indicate the opposite condition, and a zero third moment would tell us that the intensities are distributed approximately equally on both sides of the mean. These features are useful for computational purposes, but they do not tell us much about the appearance of an image in general.

EXAMPLE 2.12: Comparison of standard deviation values as measures of image intensity contrast.

Figure 2.41 shows three 8-bit images exhibiting low, medium, and high contrast, respectively. The standard deviations of the pixel intensities in the three images are 14.3, 31.6, and 49.2 intensity levels, respectively. The corresponding variance values are 204.3, 997.8, and 2424.9, respectively. Both sets of values tell the same story but, given that the range of possible intensity values in these images is [0, 255], the standard deviation values relate to this range much more intuitively than the variance.



a b c

FIGURE 2.41 Images exhibiting (a) low contrast, (b) medium contrast, and (c) high contrast.

Neighbors of a Pixel

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & f(0,3) & f(0,4) & ----- \\ f(1,0) & f(1,1) & f(1,2) & f(1,3) & f(1,4) & ----- \\ f(2,0) & f(2,1) & f(2,2) & f(2,3) & f(2,4) & ----- \\ f(3,0) & f(3,1) & f(3,2) & f(3,3) & f(3,4) & ----- \\ | & | & | & | & | & ----- \\ | & | & | & | & | & ----- \end{bmatrix}$$

- A Pixel p at coordinates (x, y) has 4 horizontal and vertical neighbors.

- Their coordinates are given by:

(x+1, y)	(x-1, y)	(x, y+1)	&	(x, y-1)
f(2,1)	f(0,1)	f(1,2)		f(1,0)

- This set of pixels is called the ***4-neighbors*** of p denoted by $N_4(p)$.

- Each pixel is unit distance from (x ,y).

Neighbors of a Pixel

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & f(0,3) & f(0,4) & ----- \\ f(1,0) & f(1,1) & f(1,2) & f(1,3) & f(1,4) & ----- \\ f(2,0) & f(2,1) & f(2,2) & f(2,3) & f(2,4) & ----- \\ f(3,0) & f(3,1) & f(3,2) & f(3,3) & f(3,4) & ----- \\ | & | & | & | & | & ----- \\ | & | & | & | & | & ----- \end{bmatrix}$$

- A Pixel p at coordinates (x, y) has 4 diagonal neighbors.
- Their coordinates are given by:

  (x+1, y+1) (x+1, y-1) (x-1, y+1) & (x-1, y-1)

  f(2,2) f(2,0) f(0,2) f(0,0)
- This set of pixels is called the ***diagonal-neighbors*** of p denoted by $N_D(p)$.
- diagonal neighbors + 4-neighbors = 8-neighbors of p.
- They are denoted by $N_8(p)$. So, $N_8(p) = N_4(p) + N_D(p)$

Adjacency, Connectivity

Adjacency: Two pixels are adjacent if they are neighbors and their intensity level 'V' satisfy some specific criteria of similarity.

e.g. V = {1}
V = { 0, 2}
Binary image = { 0, 1}
Gray scale image = { 0, 1, 2, ------, 255}

In binary images, 2 pixels are adjacent if they are neighbors & have some intensity values either 0 or 1.

In gray scale, image contains more gray level values in range 0 to 255.

Adjacency, Connectivity

4-adjacency: Two pixels p and q with the values from set 'V' are 4-adjacent if q is in the set of $N_4(p)$.

e.g. V = { 0, 1}

1	1	0
1	1	0
1	0	1

p in **RED** color
q can be any value in **GREEN** color.

Adjacency, Connectivity

8-adjacency: Two pixels p and q with the values from set 'V' are 8-adjacent if q is in the set of $N_8(p)$.

e.g. V = { 1, 2}

0	1	1
0	2	0
0	0	1

p in **RED** color
q can be any value in **GREEN** color

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$ OR

(ii) q is in $N_D(p)$ & the set $N_4(p) \cap N_4(q)$ have no pixels whose values are from 'V'.

e.g. V = { 1 }

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 i

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$

e.g. V = { 1 }

(i) b & c

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$

e.g. V = { 1 }

(i) b & c

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Soln: b & c are m-adjacent.

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$

e.g. V = { 1 }

(ii) b & e

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$

e.g. V = { 1 }

(ii) b & e

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Soln: b & e are m-adjacent.

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$ OR

e.g. V = { 1 }

(iii) e & i

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 i

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set ‘V’ are m-adjacent if

(i) q is in $N_D(p)$ & the set $\underline{N_4(p) \cap N_4(q)}$ have no pixels whose values are from ‘V’.

e.g. V = { 1 }

(iii) e & i

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_D(p)$ & the set $N_4(p) \cap N_4(q)$ have no pixels whose values are from 'V'.

e.g. V = { 1 }

(iii) e & i

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Soln: e & i are m-adjacent.

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$ OR

(ii) q is in $N_D(p)$ & the set $N_4(p) \cap N_4(q)$ have no pixels whose values are from 'V'.

e.g. V = { 1 }

(iv) e & c

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Adjacency, Connectivity

m-adjacency: Two pixels p and q with the values from set 'V' are m-adjacent if

(i) q is in $N_4(p)$ OR

(ii) q is in $N_D(p)$ & the set $N_4(p) \cap N_4(q)$ have no pixels whose values are from 'V'.

e.g. V = { 1 }

(iv) e & c

0 a	1 b	1 c
0 d	1 e	0 f
0 g	0 h	1 I

Soln: e & c are NOT m-adjacent.

Adjacency, Connectivity

Connectivity: 2 pixels are said to be connected if their exists a path between them.

Let ‘S’ represent subset of pixels in an image.

Two pixels p & q are said to be connected in ‘S’ if their exists a path between them consisting entirely of pixels in ‘S’.

For any pixel p in S, the set of pixels that are connected to it in S is called a **connected component of S**.

Path/Curve

Path: A (digital) path/curve from pixel p with coordinate (x, y) with pixel q with coordinate (s, t) is a sequence of distinct sequence with coordinates (x_o, y_o), (x_1, y_1), ….., (x_n, y_n) where

$(x, y) = (x_o, y_o)$
& $(s, t) = (x_n, y_n)$

Closed path: $(x_o, y_o) = (x_n, y_n)$

Path

<u>Example # 1</u>: Consider the image segment shown in figure. Compute length of the ***shortest-4, shortest-8 & shortest-m paths*** between pixels p & q where,
V = {1, 2}.

$$
\begin{array}{cccccc}
 & 4 & 2 & 3 & 2 & q \\
 & 3 & 3 & 1 & 3 & \\
 & 2 & 3 & 2 & 2 & \\
p & 2 & 1 & 2 & 3 &
\end{array}
$$

Path

Example # 1:

Shortest-4 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1	2	3	

Path

<u>Example # 1</u>:

<u>Shortest-4 path:</u>

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1 →	2	3	

Path

Example # 1:

Shortest-4 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1 →	2 ↑	3	

Path

Example # 1:

Shortest-4 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

Example # 1:

Shortest-4 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

Example # 1:

Shortest-4 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

So, Path does not exist.

Path

Example # 1:

Shortest-8 path:

V = {1, 2}.

	4	2	3	2 q
	3	3	1	3
	2	3	2	2
p	2	1	2	3

Path

Example # 1:

Shortest-8 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1	2	3	

Path

<u>Example # 1</u>:

<u>Shortest-8 path:</u>

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

Example # 1:

Shortest-8 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

Example # 1:

Shortest-8 path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

<u>Example # 1</u>:

<u>Shortest-8 path:</u>

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

So, shortest-8 path = 4

Path

Example # 1:

Shortest-m path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

<u>Example # 1</u>:

<u>Shortest-m path:</u>

V = {1, 2}.

$$\begin{array}{cccccc} & 4 & 2 & 3 & 2 & q \\ & 3 & 3 & 1 & 3 & \\ & 2 & 3 & 2 & 2 & \\ p & 2 \rightarrow & 1 & 2 & 3 & \end{array}$$

Path

Example # 1:

Shortest-m path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1 →	2	3	

Path

Example # 1:

Shortest-m path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

Example # 1:

Shortest-m path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2 →	1 →	2	3	

Path

Example # 1:

Shortest-m path:

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

Path

<u>Example # 1</u>:

<u>Shortest-m path:</u>

V = {1, 2}.

	4	2	3	2	q
	3	3	1	3	
	2	3	2	2	
p	2	1	2	3	

So, shortest-m path = 5

Region & Boundary

Region: Let R be a subset of pixels in an image. Two regions Ri and Rj are said to be adjacent if their union form a connected set.

Regions that are not adjacent are said to be disjoint.

We consider 4- and 8- adjacency when referring to regions.

Below regions are adjacent only if 8-adjacency is used.

1	1	1	
1	0	1	R_i
0	1	0	
0	0	1	
1	1	1	R_j
1	1	1	

Regions & Boundaries

<u>Boundaries (border or contour)</u>: The boundary of a region R is the set of points that are adjacent to points in the compliment of R.

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

RED colored 1 is NOT a member of border if 4-connectivity is used between region and background. It is if 8-connectivity is used.

Distance Measures

Distance Measures: Distance between pixels p, q & z with co-ordinates (x, y), (s, t) & (v, w) resp. is given by:

a) D(p, q) ≥ o [D(p, q) = o if p = q] ....called reflexivity
b) D(p, q) = D(q, p) ....called symmetry
c) D(p, z) ≤ D(p, q) + D(q, z) ....called transmitivity

Euclidean distance between p & q is defined as-

$$D_e(p, q) = [(x- s)^2 + (y - t)^2]^{1/2}$$

Distance Measures

City Block Distance: The D_4 distance between p & q is defined as

$$D_4(p, q) = |x - s| + |y - t|$$

In this case, pixels having D_4 distance from (x, y) less than or equal to some value r form a diamond centered at (x, y).

		2		
	2	1	2	
2	1	0	1	2
	2	1	2	
		2		

Pixels with D_4 distance $\leq$ 2 forms the following contour of constant distance.

Distance Measures

Chess-Board Distance: The D_8 distance between p & q is defined as

$$D_8(p, q) = \max(|x - s|, |y - t|)$$

In this case, pixels having D_8 distance from (x, y) less than or equal to some value r form a square centered at (x, y).

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Pixels with D_8 distance ≤ 2 forms the following contour of constant distance.