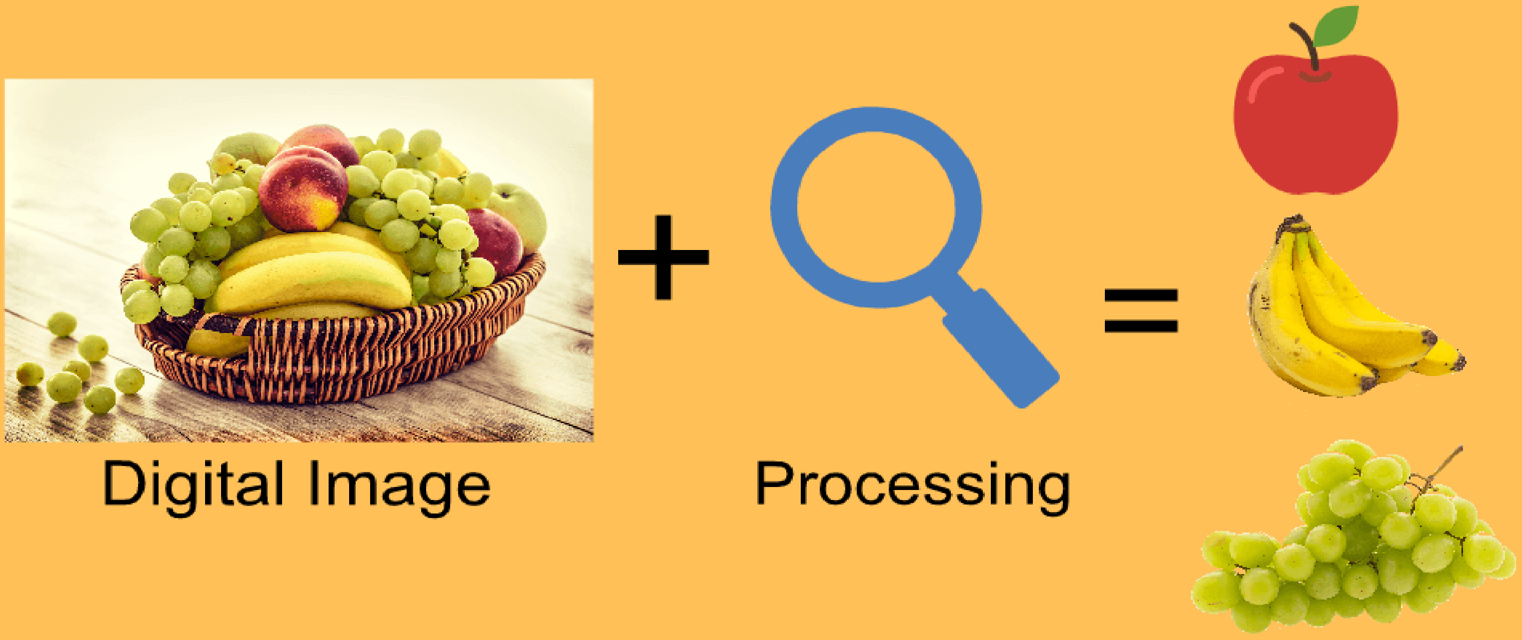
Digital Image Processing



Digital Image

+

Processing

=

Apple

Banana

Grape

Digital Image Processing

Lec. 1: Introduction

Assist. Prof. Dr. Saad Albawi

What Is an Image?

An *image* is a visual representation of an object, a person, or a scene produced by an optical device such as a mirror, a lens, or a camera. This representation is two dimensional (2D), although it corresponds to one of the infinitely many projections of a real-world, three-dimensional (3D) object or scene.

What Is a Digital Image?

A digital image is a representation of a two-dimensional image using a finite number of points, usually referred to as *picture elements*, *pels*, or *pixels*. Each pixel is represented by one or more numerical values: for monochrome (grayscale) images, a single value representing the intensity of the pixel (usually in a [0, 255] range) is enough; for color images, three values (e.g., representing the amount of red (R), green (G), and blue (B)) are usually required. Alternative ways of representing color images, such as the *indexed color image* representation.

What is Digital Image Processing

Digital image processing can be defined as the science of modifying digital images by means of a digital computer. Since both the images and the computers that process them are digital in nature, we will focus exclusively on *digital* image processing.

What Is the Levels of Image Processing?

- *Low Level*: Primitive operations (e.g., noise reduction, contrast enhancement,etc.) where both the input and the output are images.

- *Mid Level*: Extraction of attributes (e.g., edges, contours, regions, etc.) fromimages.

- *High Level*: Analysis and interpretation of the contents of a scene.

COMPONENTS OF A DIGITAL IMAGE PROCESSING SYSTEM

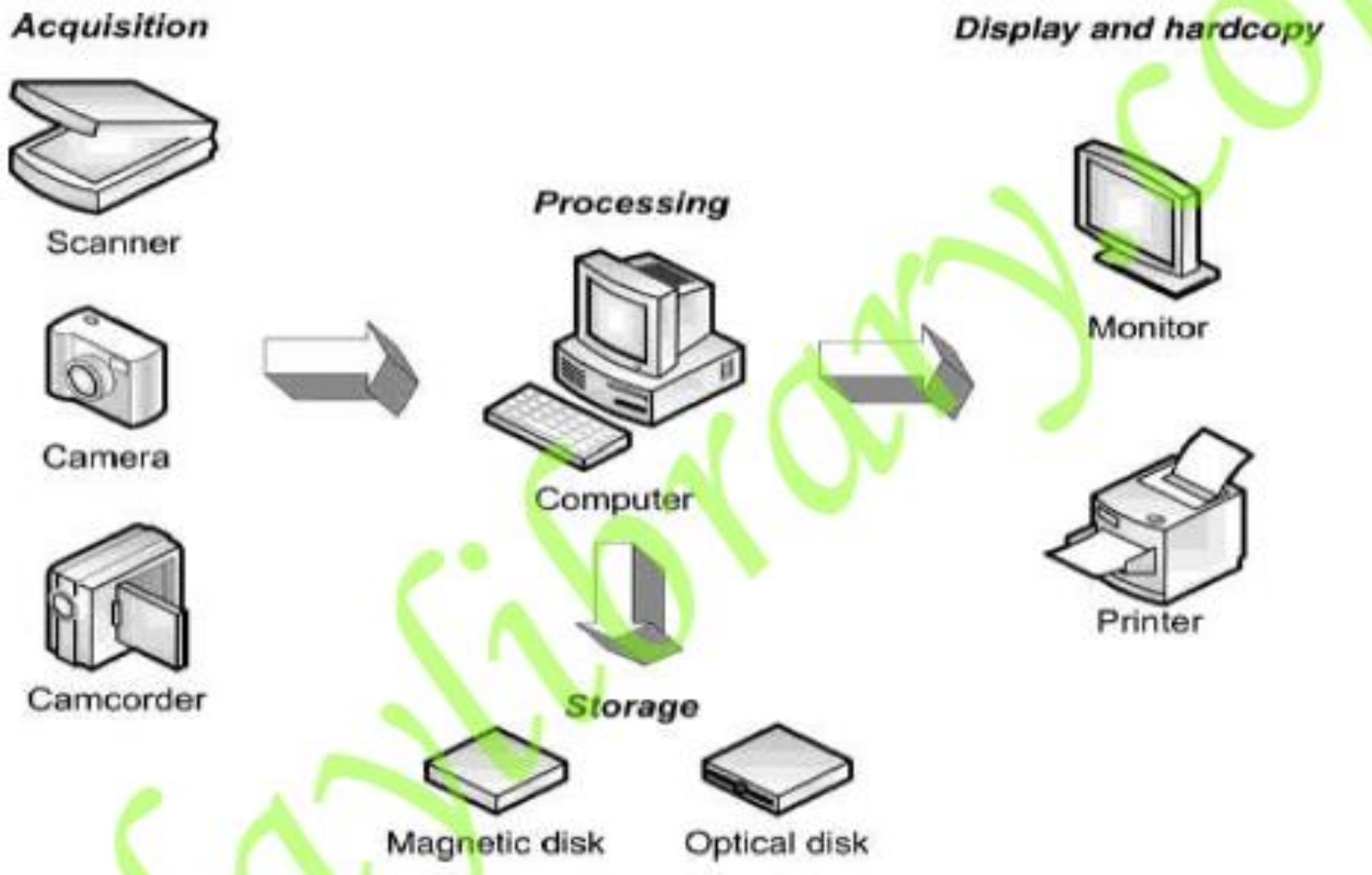


FIGURE 1.9 Components of a digital image processing system. Adapted and redrawn from [Umb05].

Hardware

The hardware components of a digital image processing system typically include the following:

- *Acquisition Devices*: Responsible for capturing and digitizing images or video sequences. Examples of general-purpose acquisition devices include scanners, cameras, and camcorders. Acquisition devices can be interfaced with the main computer in a number of ways, for example, USB, FireWire, Camera Link, or Ethernet. In cases where the cameras produce analog video output, an image digitizer—usually known as *frame grabber*—is used to convert it to digital format.
- *Processing Equipment*: The main computer itself, in whatever size, shape, or configuration. Responsible for running software that allows the processing and analysis of acquired images.
- *Display and Hardcopy Devices*: Responsible for showing the image contents for human viewing. Examples include color monitors and printers.
- *Storage Devices*: Magnetic or optical disks responsible for long-term storage of the images.

2.1 DIGITAL IMAGE REPRESENTATION

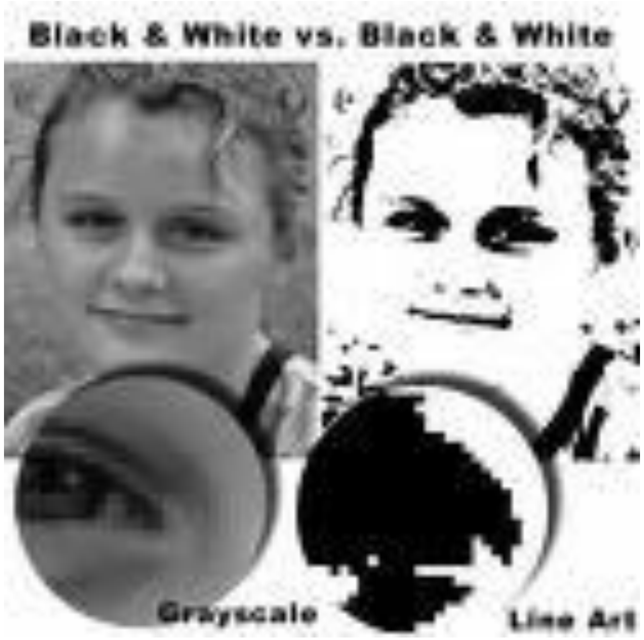
A digital image—whether it was obtained as a result of sampling and quantization of an analog image or created already in digital form—can be represented as a two-dimensional (2D) matrix of real numbers. In this book, we adopt the convention $f(x, y)$ to refer to monochrome images of size $M \times N$, where x denotes the row number (from 0 to $M - 1$) and y represents the column number (between 0 and $N - 1$) (Figure 2.1):

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

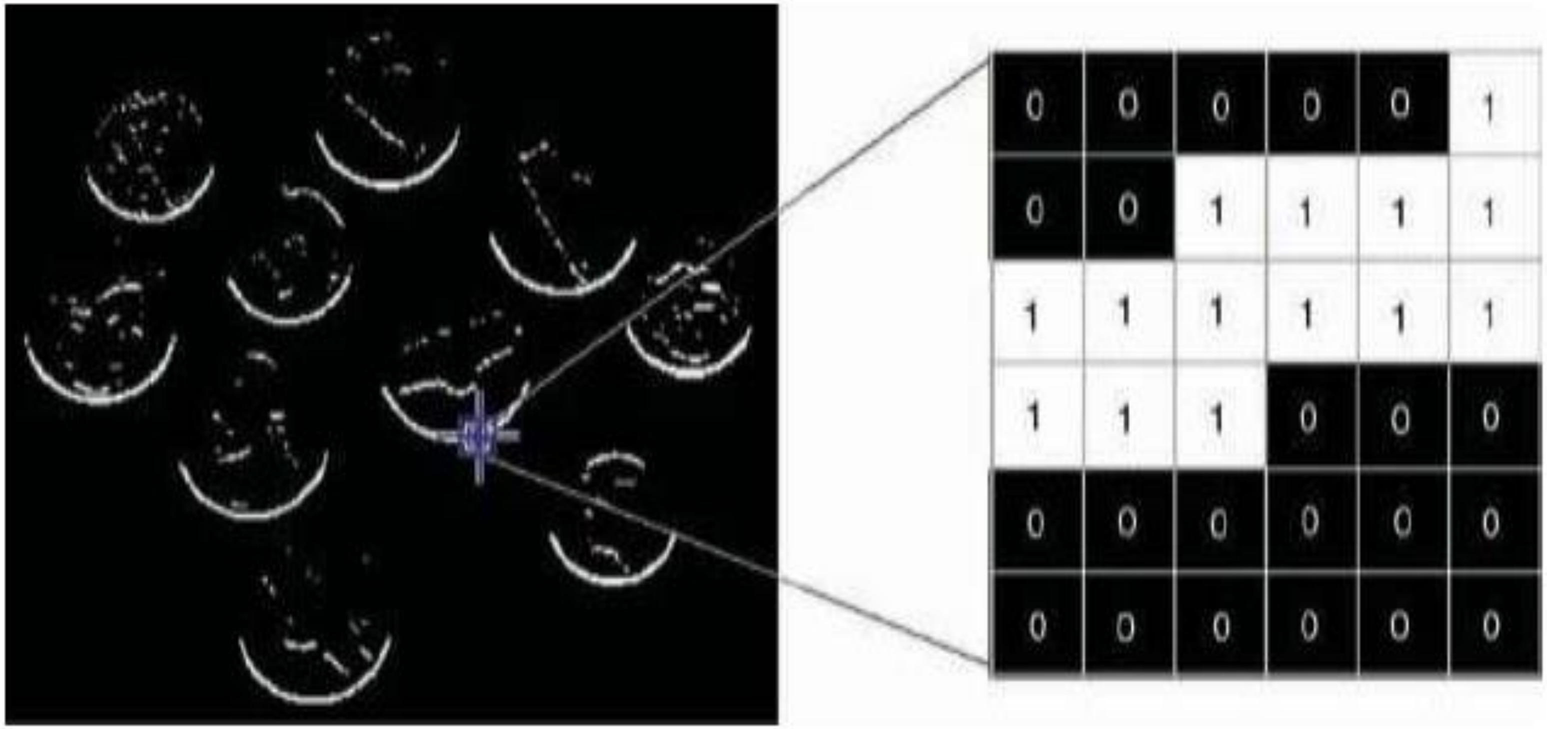
Remember: images can be: binary, grayscale, color.

“Binary Images”

<u>Binary images:</u> are images that have been quantized to two values, usually denoted 0 and 1, but often with pixel values 0 and 255, representing black and white.



Binary (1-Bit) Images

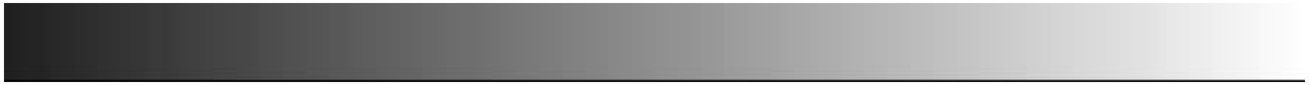


0	0	0	0	0	1
0	0	1	1	1	1
1	1	1	1	1	1
1	1	1	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

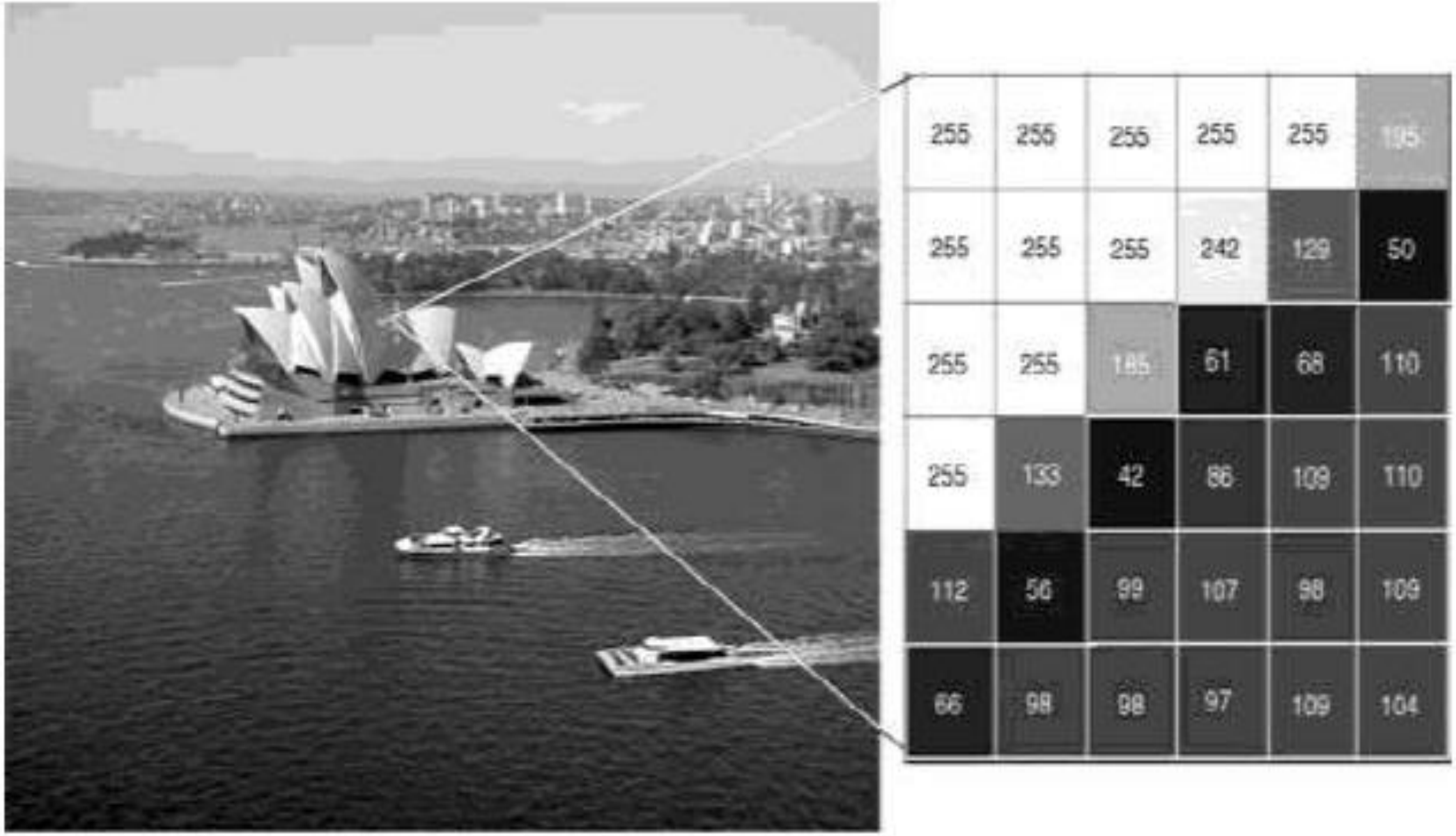
FIGURE 2.2 A binary image and the pixel values in a 6 × 6 neighborhood. Original image: courtesy of MathWorks.

“Grayscale Images”

- **A grayscale (or graylevel) image:** is simply one in which the only colors are shades of gray (0 – 255)



Gray-Level (8-Bit) Images

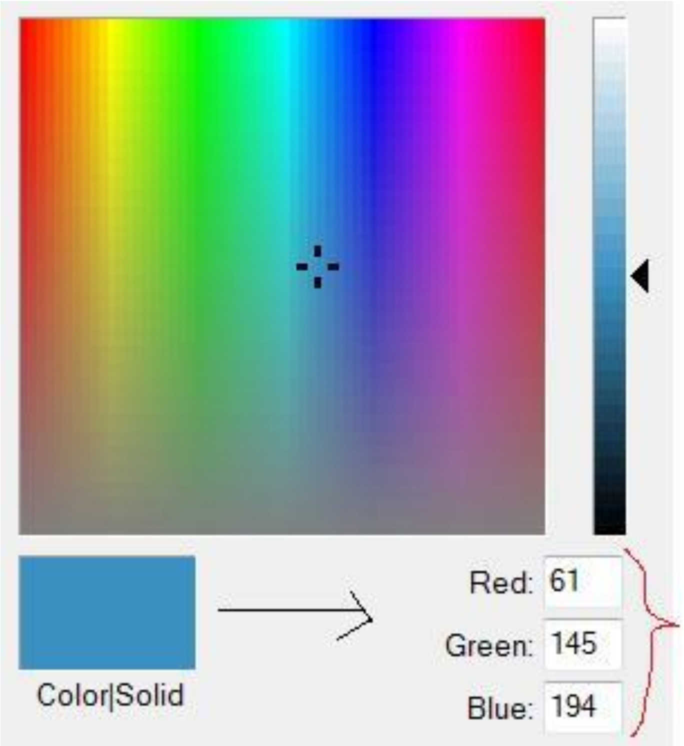


255	255	255	255	255	195
255	255	255	242	129	50
255	255	185	61	68	110
255	133	42	86	109	110
112	56	99	107	98	109
66	98	98	97	109	104

FIGURE 2.3 A grayscale image and the pixel values in a 6 × 6 neighborhood.

“Color Images”

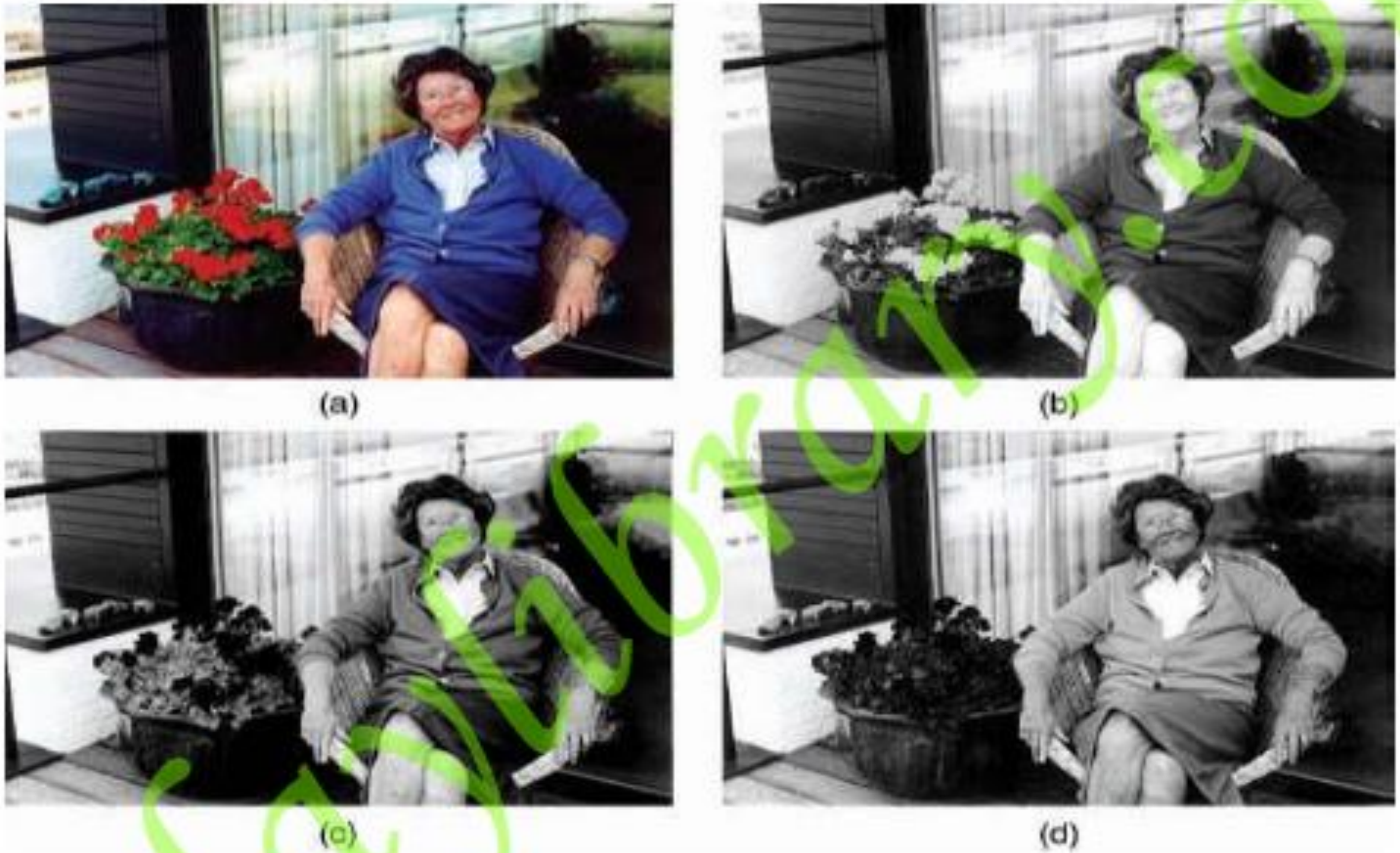
- **Color image:** A color image contains pixels each of which holds three intensity values corresponding to the red, green, and blue or(RGB)



Red: 61
Green: 145
Blue: 194

pixel color
is made
from 3
RGB
intensities

Color Images



(a) (b) (c) (d)

FIGURE 2.1 Color image (a) and its R (b), G (c), and B (d) components.

- Light reflected on an object and detected by a sensor is an ***additive (linear) combination*** of different wavelengths (i.e. Colours).

- Red, Green, and Blue are the *primary colors*. Other colors are a linear combination of R, G and B. **i.e** light color space is 3 dimentional with {R, G, B} as its base and every other colour can be expressed as:

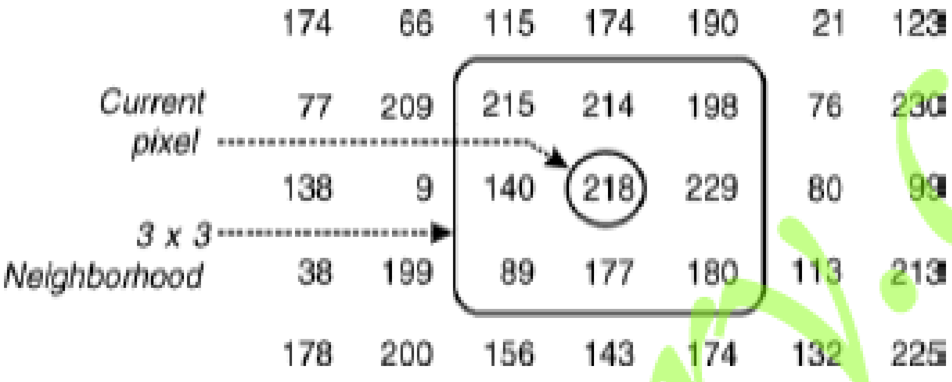
$$\mathbf{a*R + b*G + d*B,}$$

where $0 \le a,b,c \le 1$ and $a + b + c = 1$.

- ***RGB perfectly interpret of human vision***

Image Topology It involves the investigation of fundamental image properties—usually done on binary images and with the help of morphological operators (see Chapter 13)—such as number of occurrences of a particular object, number of separate (not connected) regions, and number of holes in an object, to mention but a few.

Neighborhood The pixels surrounding a given pixel constitute its *neighborhood*, which can be interpreted as a smaller matrix containing (and usually centered around) the reference pixel. Most neighborhoods used in image processing algorithms are small square arrays with an odd number of pixels, for example, the 3 × 3 neighborhood shown in Figure 2.6.



174	66	115	174	190	21	123
77	209	215	214	198	76	230
138	9	140	218	229	80	99
38	199	89	177	180	113	213
178	200	156	143	174	132	225

FIGURE 2.6 Pixels within a neighborhood.

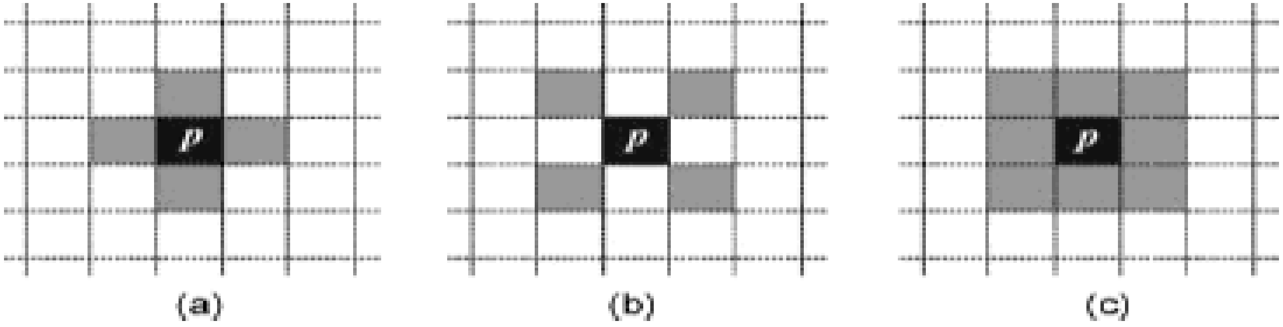


FIGURE 2.7 Concept of neighborhood of pixel p (from an image topology perspective): (a) 4-neighborhood; (b) diagonal neighborhood; (c) 8-neighborhood.

Adjacency In the context of image topology, two pixels p and q are *4-adjacent* if they are 4-neighbors of each other and *8-adjacent* if they are 8-neighbors of one another. A third type of adjacency—known as *mixed adjacency* (or simply *m-adjacency*)—is sometimes used to eliminate ambiguities (i.e., redundant paths) that may arise when 8-adjacency is used.

Paths In the context of image topology, a *4-path* between two pixels p and q is a sequence of pixels starting with p and ending with q such that each pixel in the sequence is 4-adjacent to its predecessor in the sequence. Similarly, an *8-path* indicates that each pixel in the sequence is 8-adjacent to its predecessor.

Connectivity If there is a 4-path between pixels p and q, they are said to be *4-connected*. Similarly, the existence of an 8-path between them means that they are *8-connected*.

Components A set of pixels that are connected to each other is called a *component*. If the pixels are 4-connected, the expression *4-component* is used; if the pixels are 8-connected, the set is called an *8-component*. Components are often labeled (and optionally pseudocolored) in a unique way, resulting in a *labeled image*, $L(x, y)$, whose pixel values are symbols of a chosen alphabet. The symbol value of a pixel typically denotes the outcome of a decision made for that pixel—in this case, the unique number of the component to which it belongs.

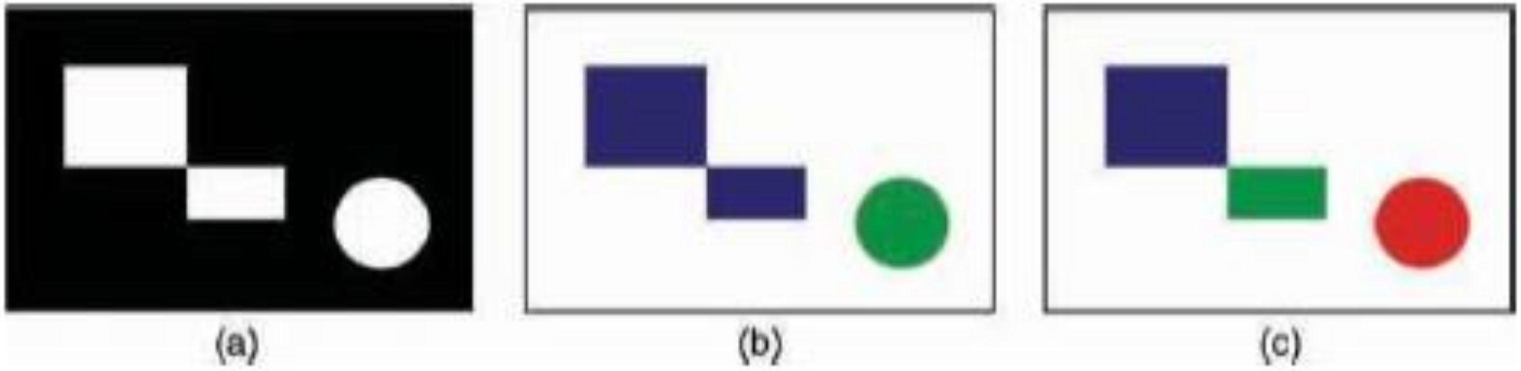


FIGURE 2.8 Connected components: (a) original (binary) image; (b) results for 8-connectivity; (c) results for 4-connectivity.

2. Distance Measures

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

a) $D(p, q) >= 0$ ($D(p,q)=0$ iff $p = q$),

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

c) $D(p,z) <= D(p,q) + D(q,z)$.

- **Euclidean distance** between *p* and *q* is defined as

$$D(p, q) = \sqrt{(x-s)^2 + (y-t)^2}$$

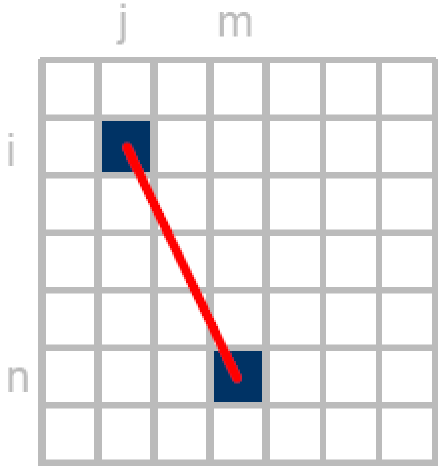
- **City-block distance** between *p* and *q* is defined as

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

- **Chessboard distance** between *p* and *q* is defined as

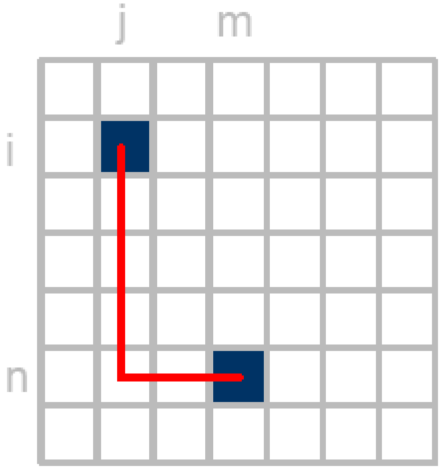
$$D(p, q) = max(|x-s|, |y-t|)$$

Distance measures



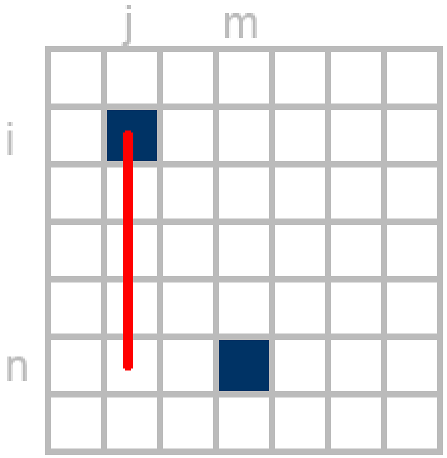
Euclidean Distance

$$=\sqrt{(i-n)^{2} + (j-m)^{2}}$$



City Block Distance

$$= |i-n| + |j-m|$$



Chessboard Distance

$$= \max[\, |i-n|, |j-m| \,]$$

Example

Compute the distance between the two pixels using the three distances :

q:(1,1)

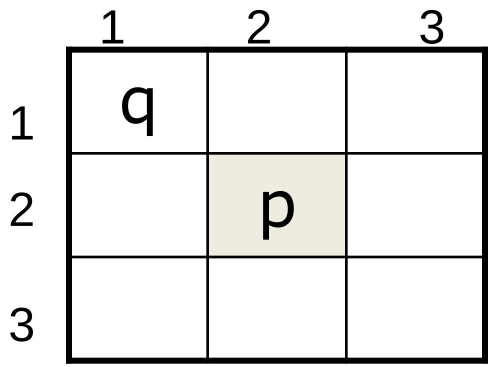
P: (2,2)

Euclidian distance : $((1\text{-}2)^2+(1\text{-}2)^2)^{1/2}$ = sqrt(2).

D4(City Block distance): |1-2| +|1-2| =2

D8(chessboard distance) : max(|1-2|,|1-2|)= 1

(because it is one of the 8-neighbors)



	1	2	3
1	q		
2		p	
3			

Source of Images

- Light
  - Visual (what the eye sees)
  - X-ray (absorption by matter)
  - Infrared (heat)
  - Radar (range)
- Ultrasound (tissue boundary)
- Synthetic (computer graphics)

Types of Images

- **Reflection images**: result of radiation that has been reflected from the surfaces of objects. Most images are reflection images

- **Emission images:** self-luminuous objects (stars)

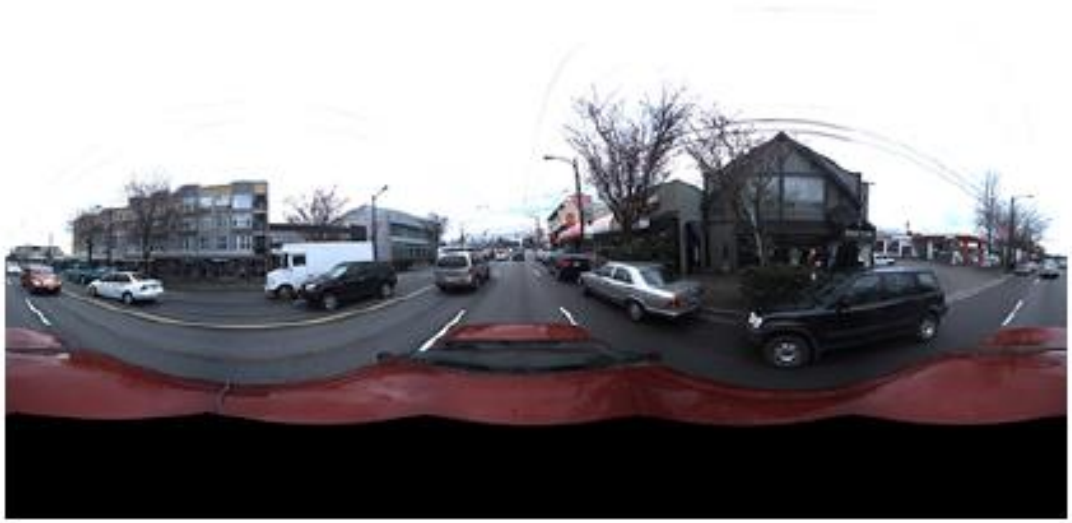
- **Absorption images:** radiation that passes through an object (X-ray)

2D Image

3D image
(Hologram)

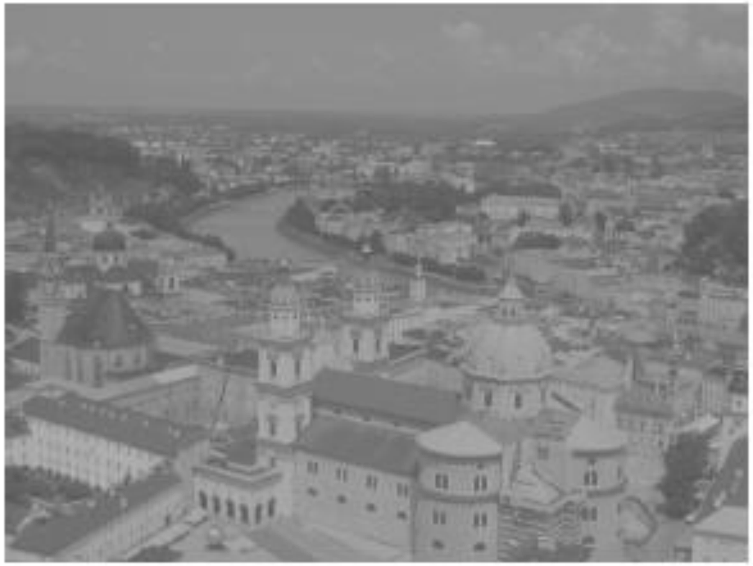
Image Sequence (Video)

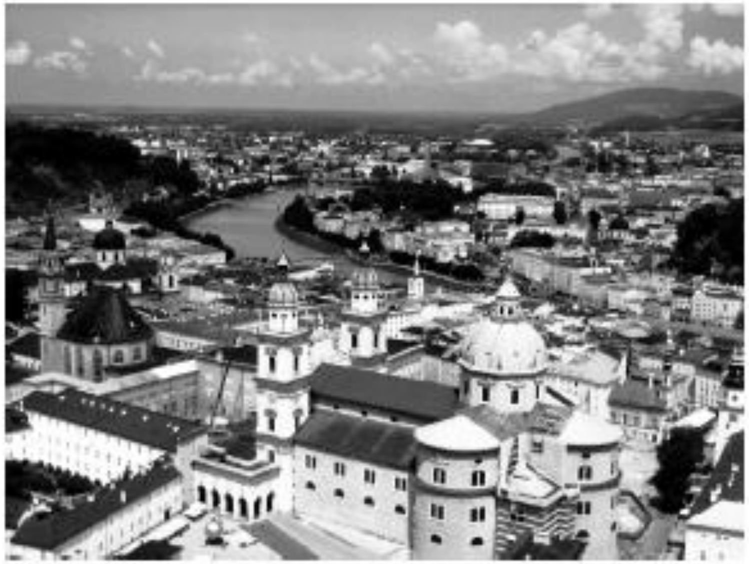
Panoramic Image



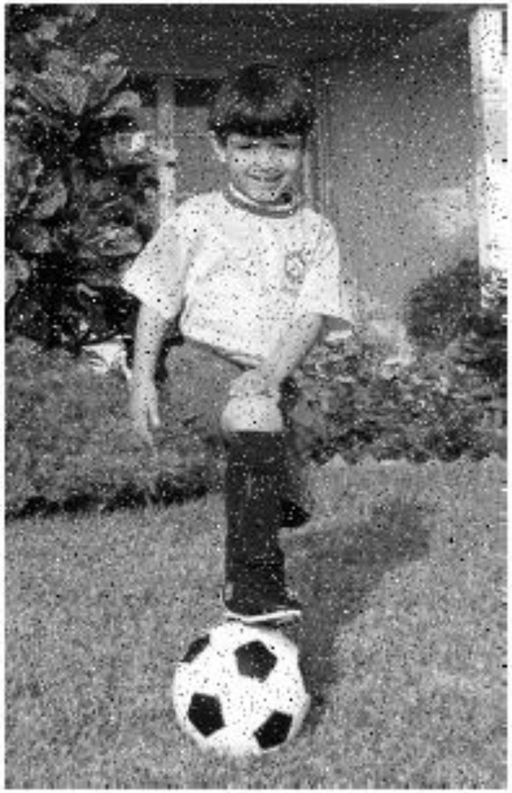
Applications of Image Processing

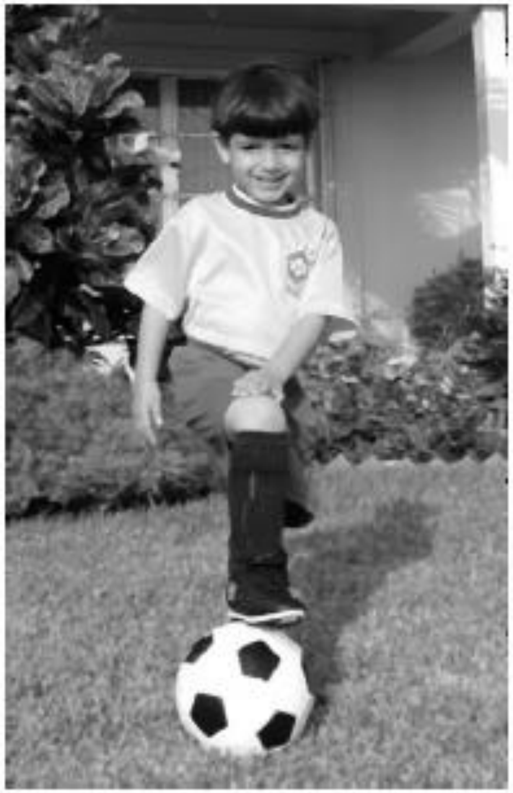
Enhancement



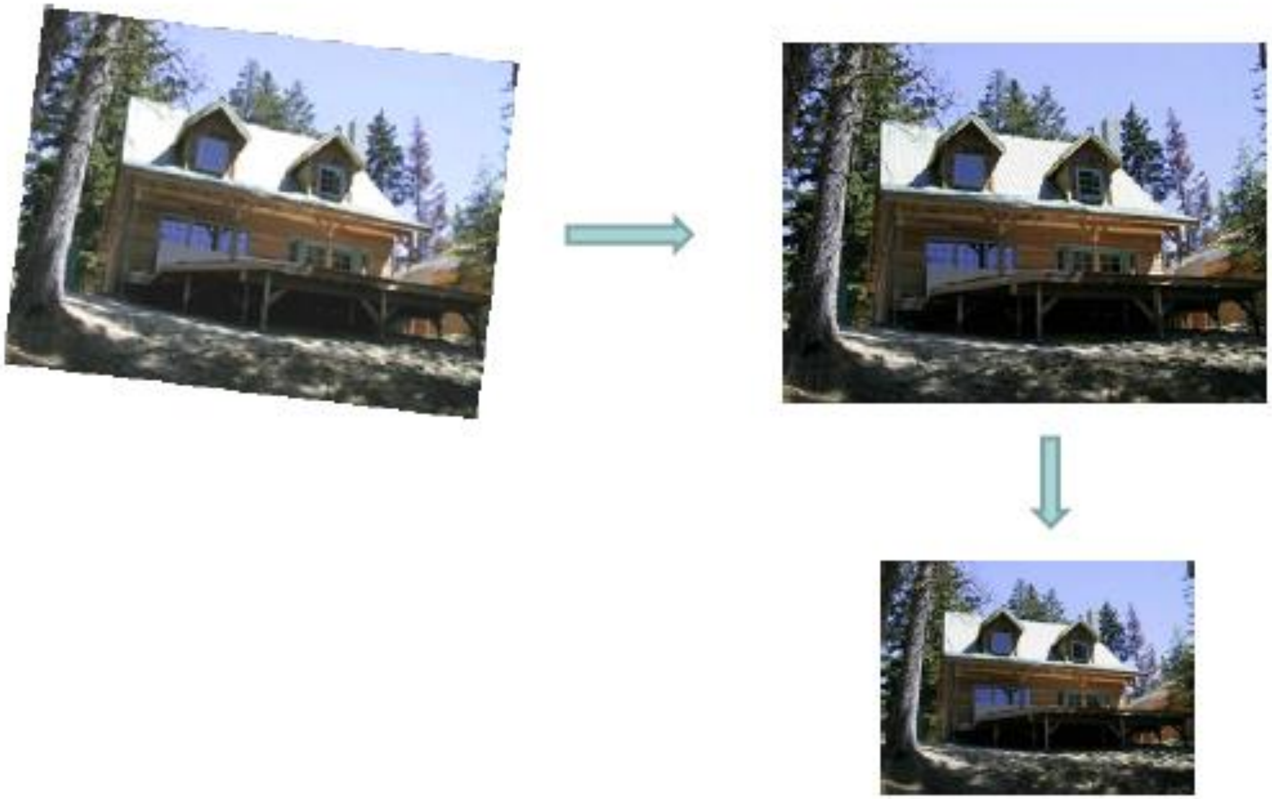


Noise Filtering

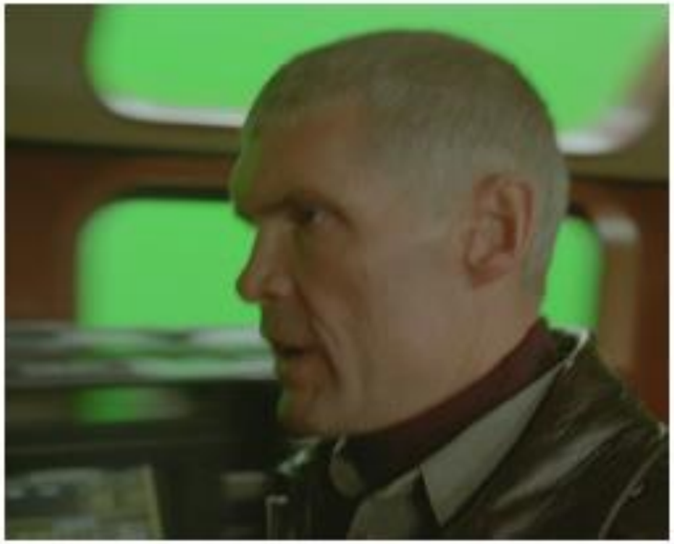


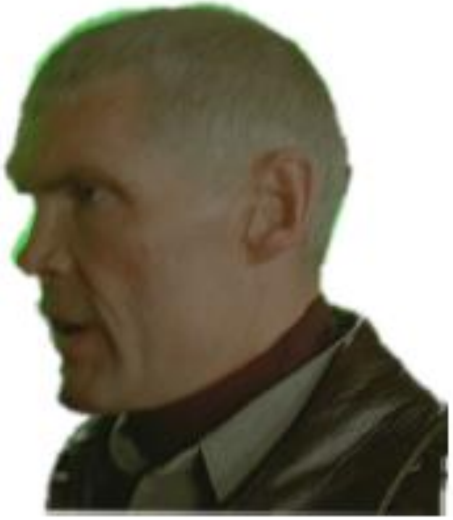


Sampling conversion

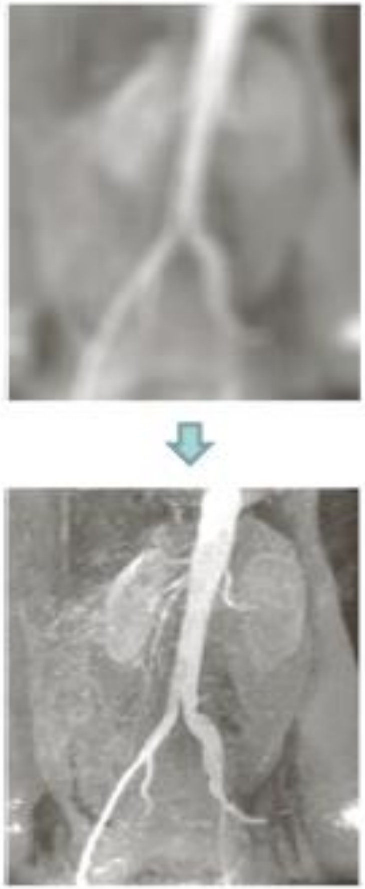


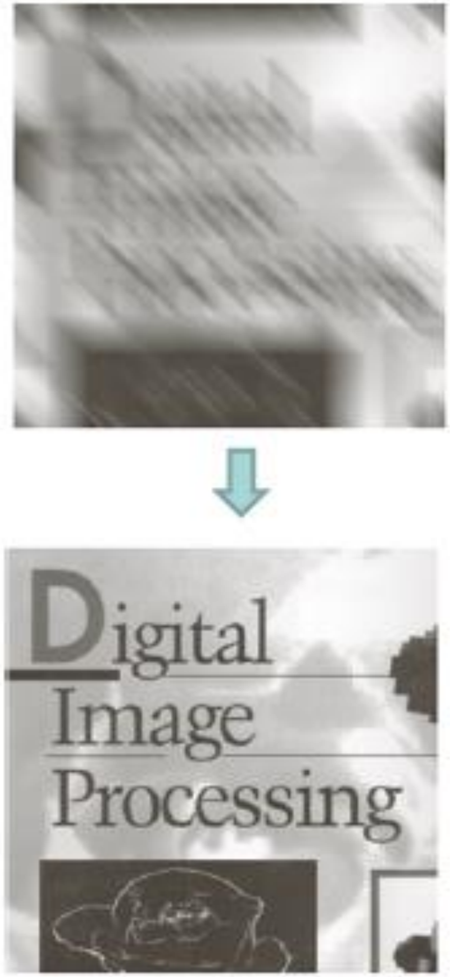
Segmentation



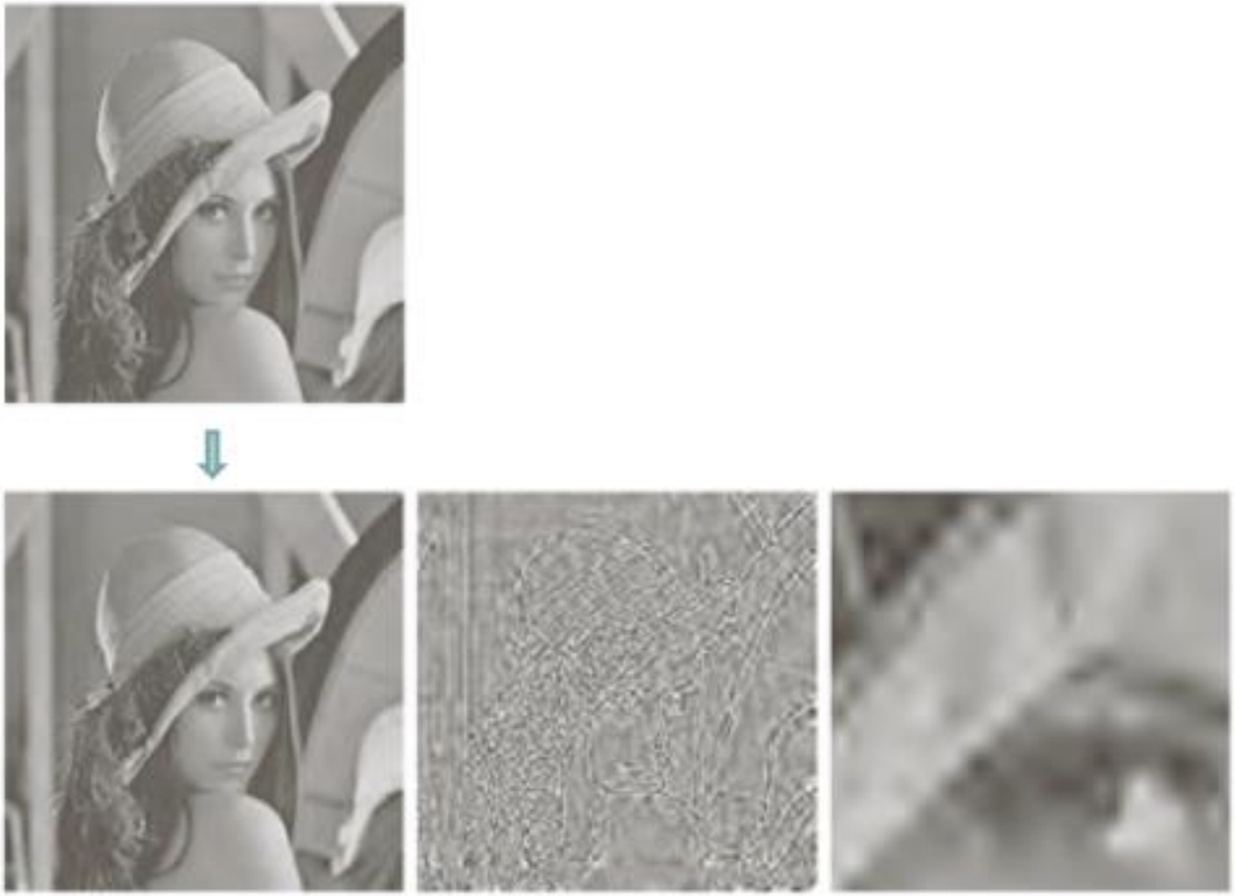


Restoration





Compression



Applications in Diverse Fields

- Medical applications
- Industrial applications
- Consumer applications
- Military applications
- Law enforcement and security
- Internet, particularly the Web.

Image Sampling and Quantization

- ❑ In order to process the image, it must be saved on computer.
- ❑ The image output of most sensors is continuous voltage waveform.
- ❑ But computer deals with digital images not with continuous images, thus: continuous images should be converted into digital form.

continuous image (in real life) → digital (computer)

Digitization: Sampling and Quantization

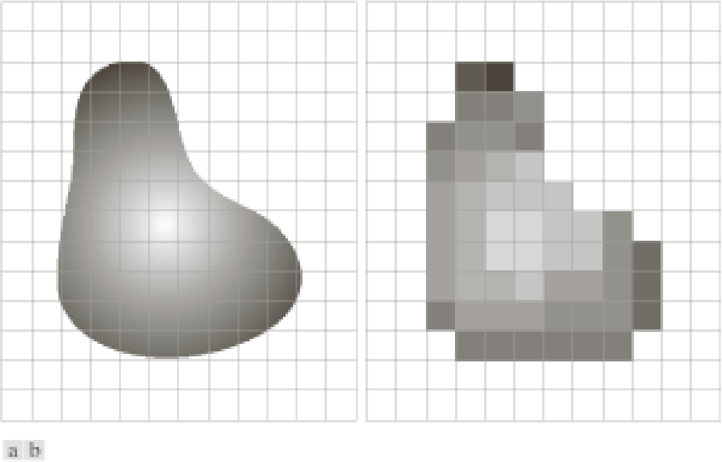
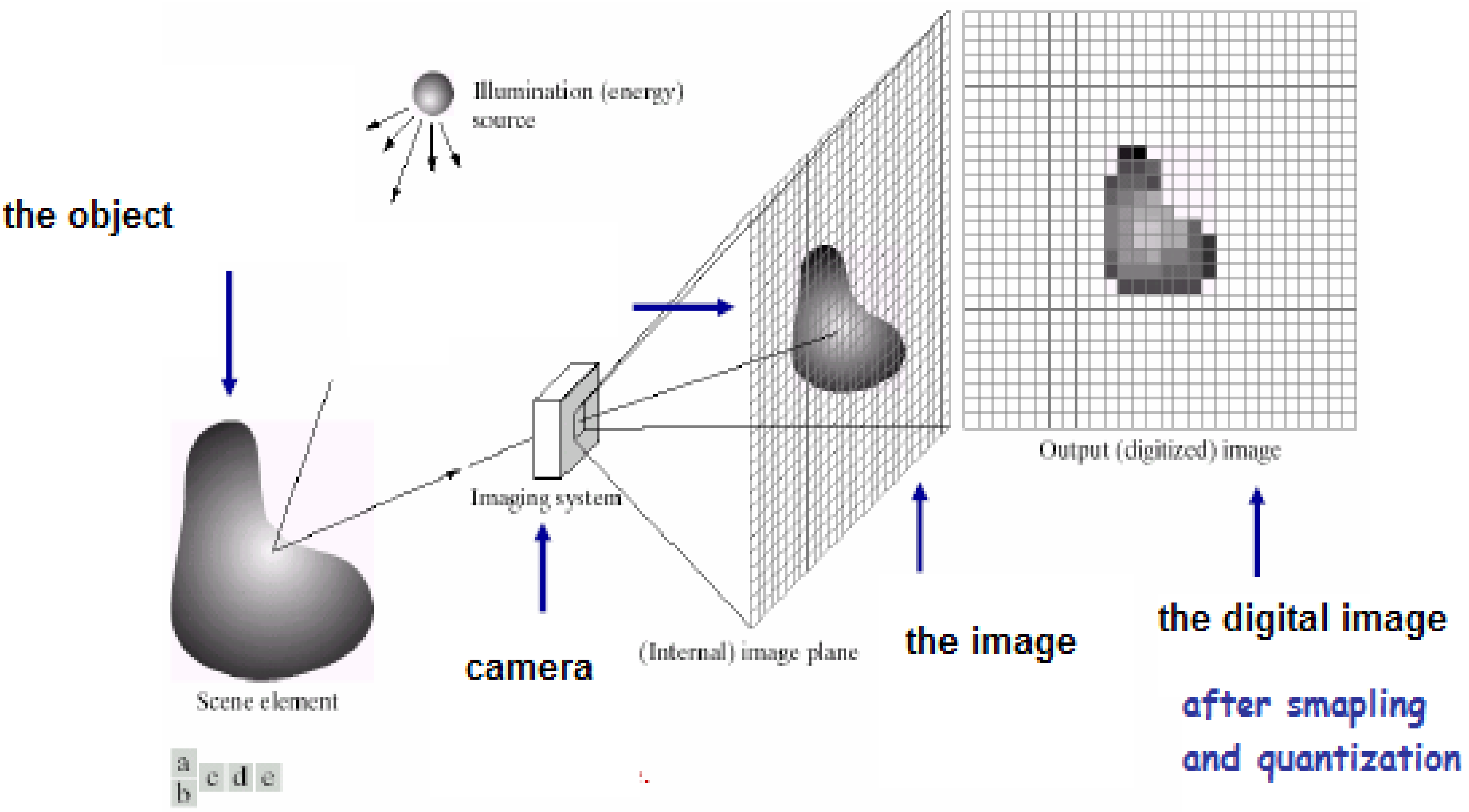


FIGURE 2.17 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.



the object

camera

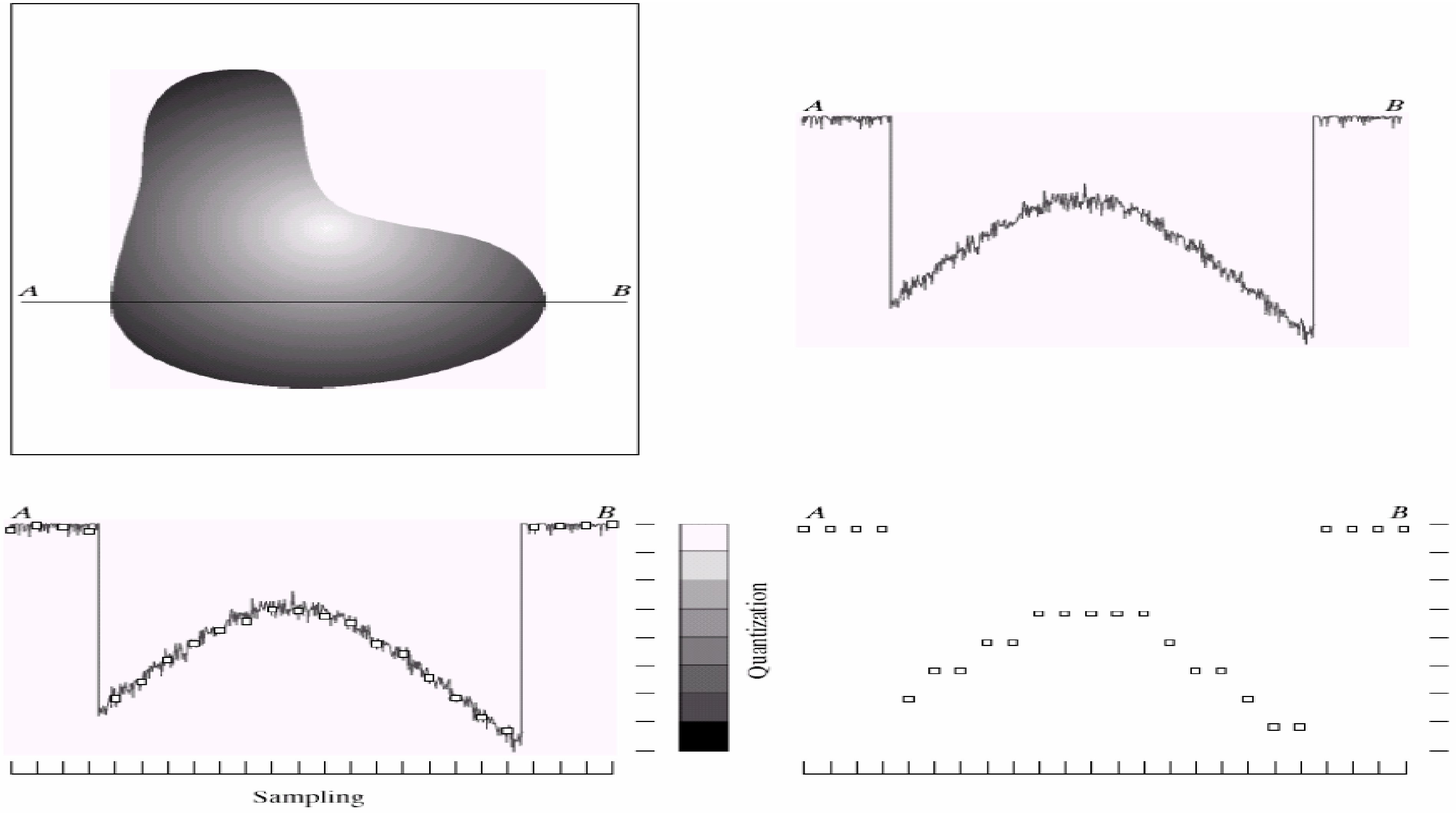
the image

the digital image

after smapling and quantization

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.

How does the computer digitize the continuous image?



a b
c d

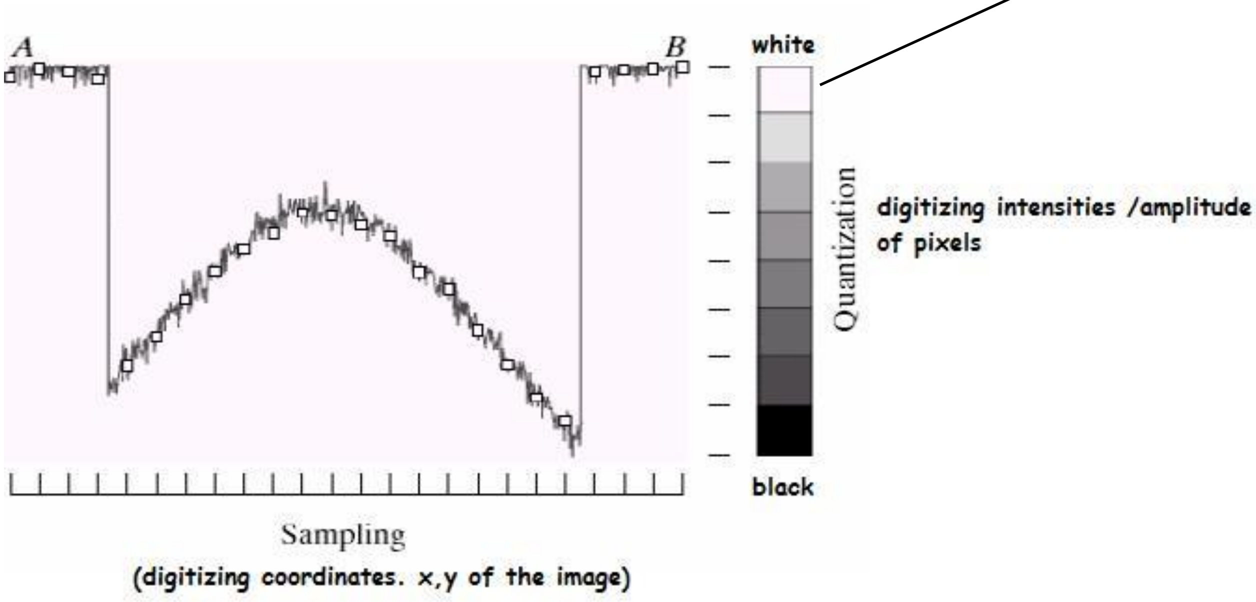
FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

How does the computer digitize the continuous image?

Sampling: digitizing coordinates

Quantization: digitizing intensities

Gray-level scale that divides gray-level into 8 discrete levels



A
B
white
black
Quantization
digitizing intensities /amplitude of pixels
Sampling
(digitizing coordinates. x,y of the image)

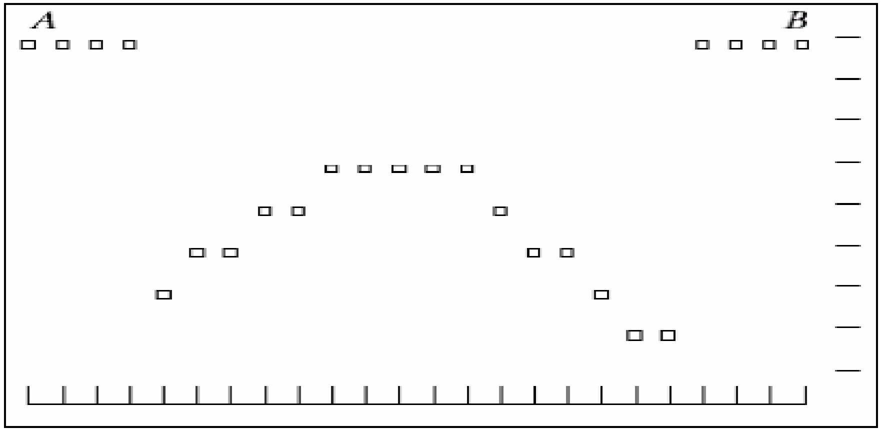
Quantization: converting each sample gray-level value into discrete digital quantity.

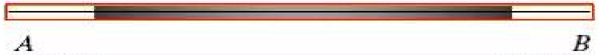
sample is a small white square, located by a vertical tick mark as a point x,y

How does the computer digitize the continuous image?

Now:

the digital scanned line AB representation on computer: →





The continuous image VS the result of digital image after sampling and quantization

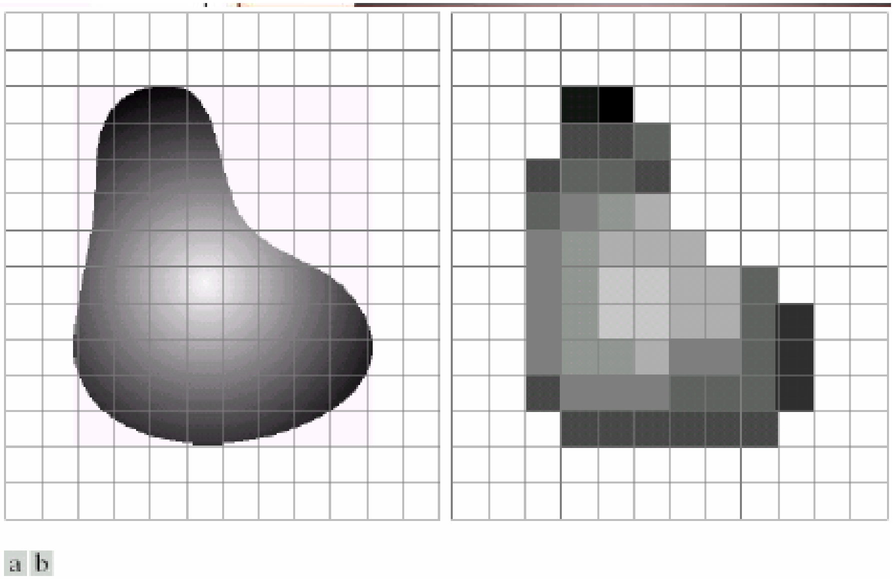
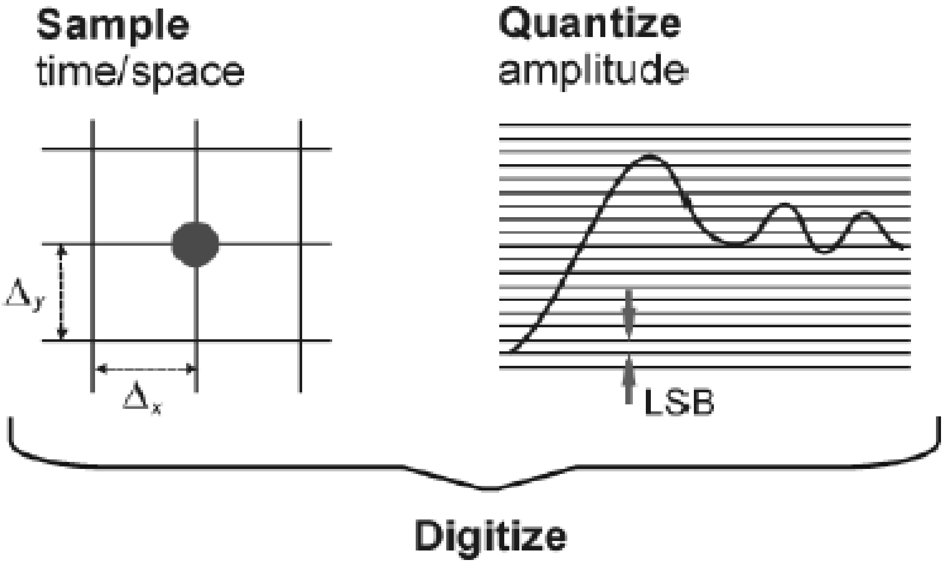


FIGURE 2.17 (a) Continuos image projected onto a sensor array. (b) Result of image sampling and quantization.

Digitization = Sampling + Quantization

- Sampling selects a finite number of points within an interval.
- Quantization assigns an amplitude to one of finite values.



Number of storage of bits:

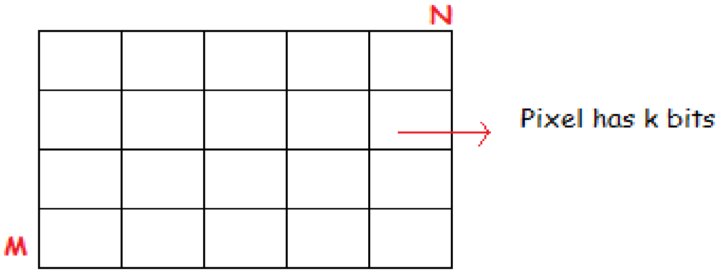
N * M: the no. of pixels in all the image.

K: no. of bits in each pixel

L: grayscale levels the pixel can represent

$L= 2^K$

all bits in image= N*N*k



NO of pixels= N *M
NO of bits= N*M*k

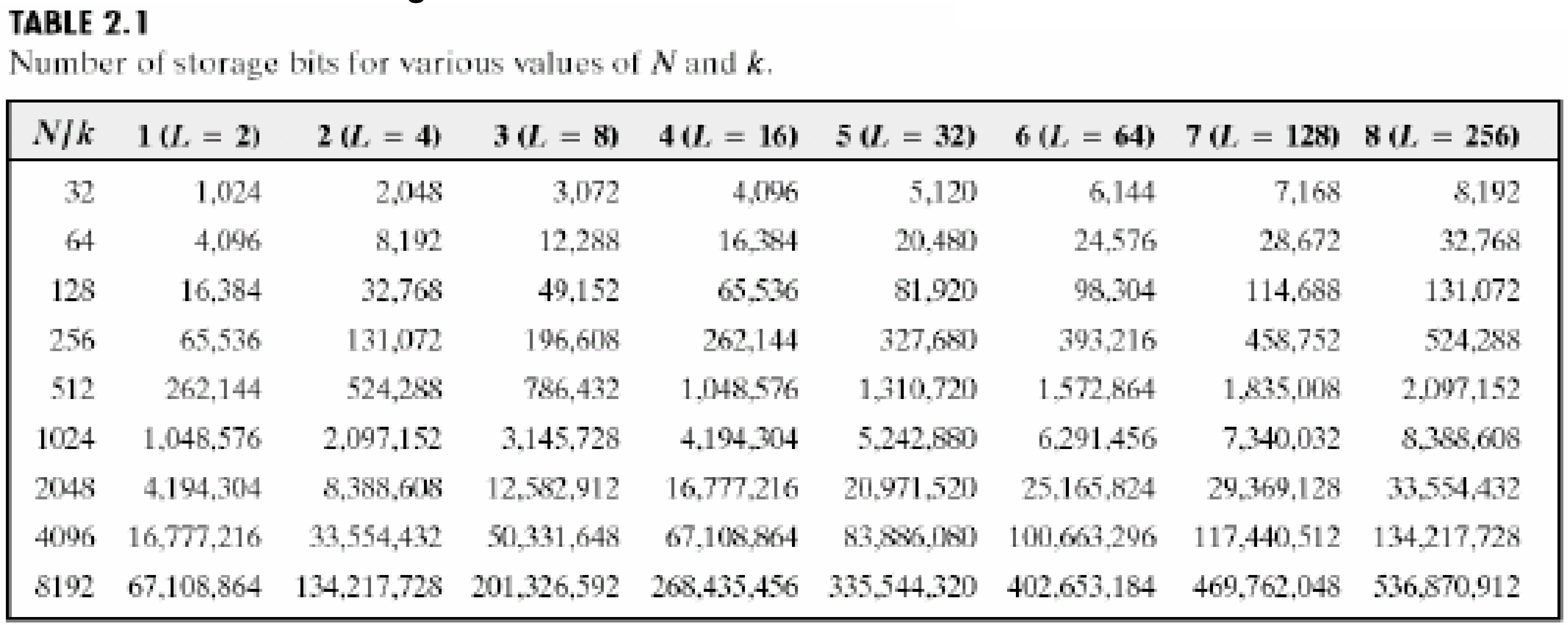


TABLE 2.1
Number of storage bits for various values of N and k.

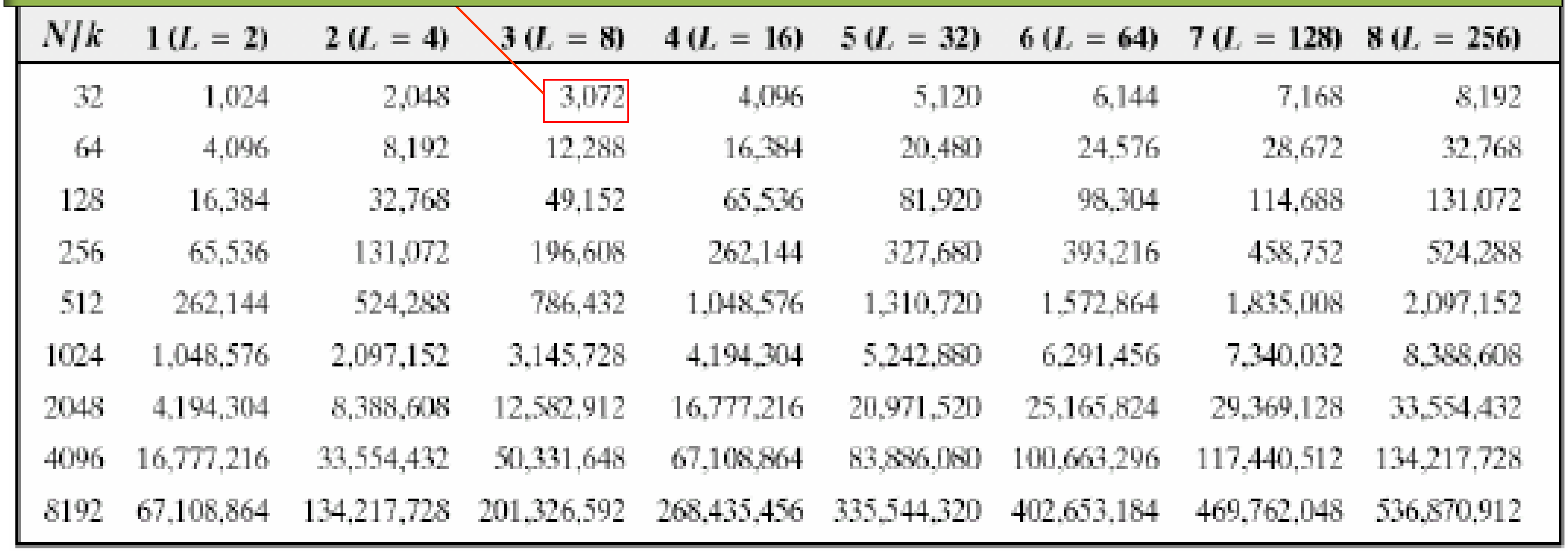
N/k	1 (L = 2)	2 (L = 4)	3 (L = 8)	4 (L = 16)	5 (L = 32)	6 (L = 64)	7 (L = 128)	8 (L = 256)
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	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

Number of storage of bits:

EX: Here: N=32, K=3, L = 2^3 =8

# of pixels=N*N = 1024 . (because in this example: M=N)

# of bits = N*N*K = 1024*3= 3072



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

N=M in this table, which means no. of horizontal pixels= no. of vertical pixels. And thus:

# of pixels in the image= N*N

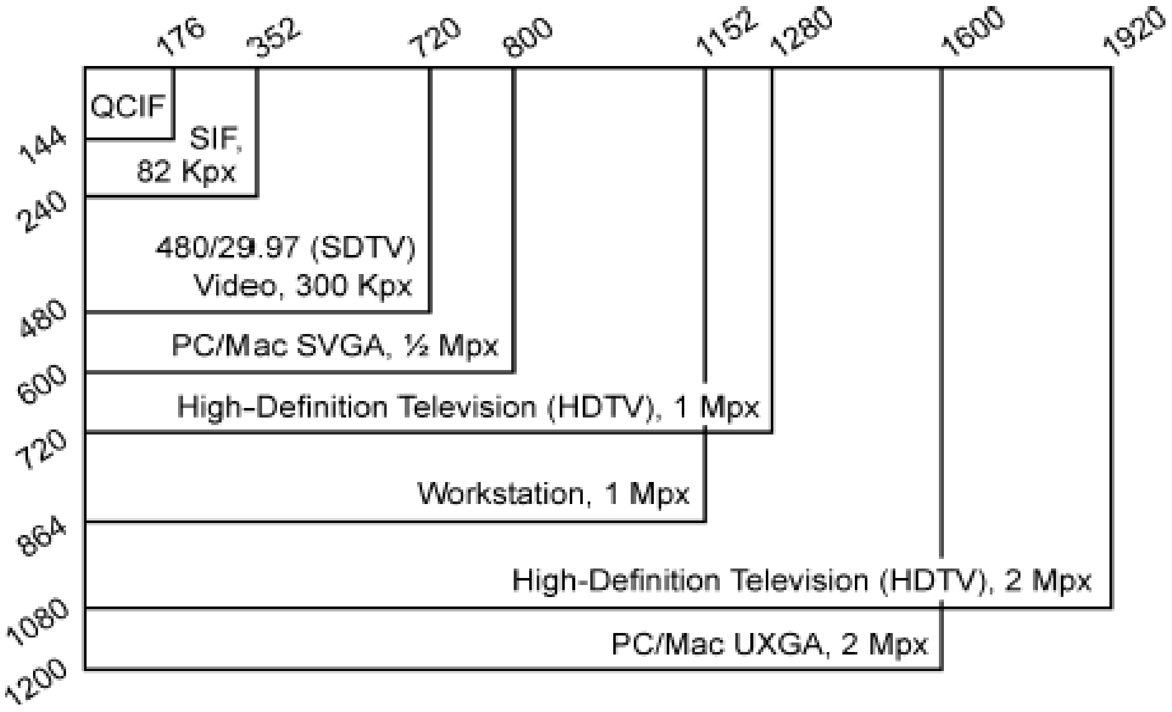
Spatial resolution

- A way of expressing the density of pixels in an image using units such as dots per inch (dpi).
- The greater the spatial resolution, the more pixels are used to display the image within a certain fixed physical size.

Gray Level Resolution

- Gray-level resolution is the smallest change in intensity level that the HVS can discern.
- Adoption of 8 bits per pixel is a good compromise between subjective quality and practical implementation.
- Higher end imaging applications may require 12 or 16 bits per pixel.

Sampling Examples



By Oge Marques

Spatial and gray-level resolution

Sub sampling

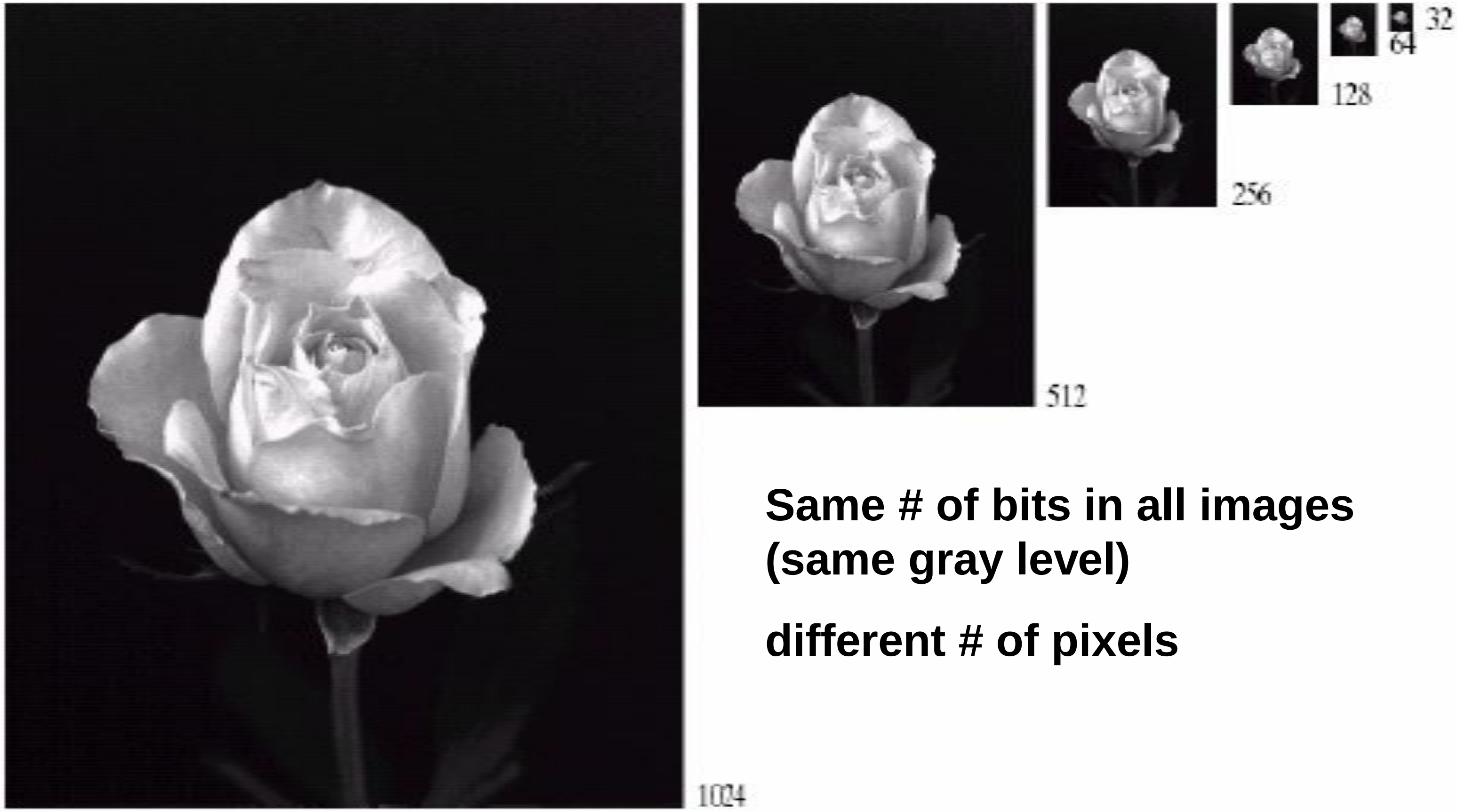


FIGURE 2.19 A 1024 × 1024, 8-bit image subsampled down to size 32 × 32 pixels. The number of allowable gray levels was kept at 256.

Same # of bits in all images (same gray level)

different # of pixels

subSampling is performed by deleting rows and columns from the original image.

Spatial and gray-level resolution

Re sampling

(pixel replication)

A special case of nearest neighbor zooming.

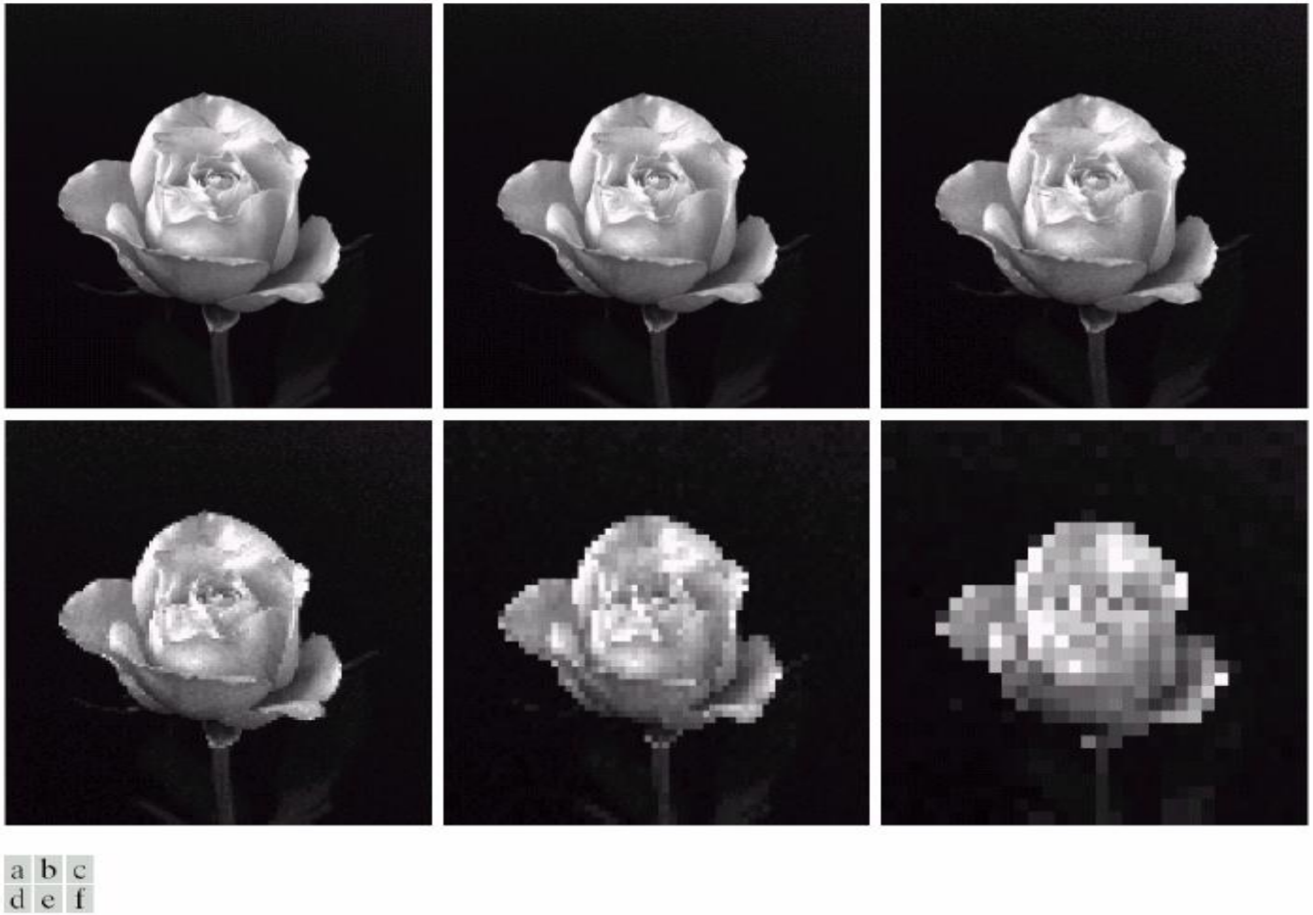


FIGURE 2.20 (a) 1024 × 1024, 8-bit image. (b) 512 × 512 image resampled into 1024 × 1024 pixels by row and column duplication. (c) through (f) 256 × 256, 128 × 128, 64 × 64, and 32 × 32 images resampled into 1024 × 1024 pixels.

Resampling is performed by row and column duplication

Effect of grey level resolution

	123	162	200	147	93
	137	157	165	232	189
Image f =	151	155	152	141	130
	205	101	100	193	115
	250	50	75	88	100
			8 bits		

f(i,j)← int(f(i,j)/2)

61	80	100	73	46
68	78	82	116	94
75	77	76	70	65
102	50	50	96	57
125	25	37	43	50
		7 bits		

30	40	50	36	23
34	39	41	58	47
37	38	38	35	32
51	25	25	48	28
62	12	18	21	25
		6 bits		

15	20	25	18	11
17	19	20	29	23
18	19	19	17	16
25	12	12	24	14
31	6	9	10	12
		5 bits		

7	10	12	9	5
8	9	10	14	11
9	9	9	8	8
12	6	6	12	7
15	3	4	5	6
		4 bits		

3	5	6	4	2
4	4	5	7	5
4	4	4	4	4
6	3	3	6	3
7	1	2	2	3
		3 bits		

1	2	3	2	1
2	2	2	3	2
2	2	2	2	2
3	1	1	3	1
3	0	1	1	1
		2 bits		

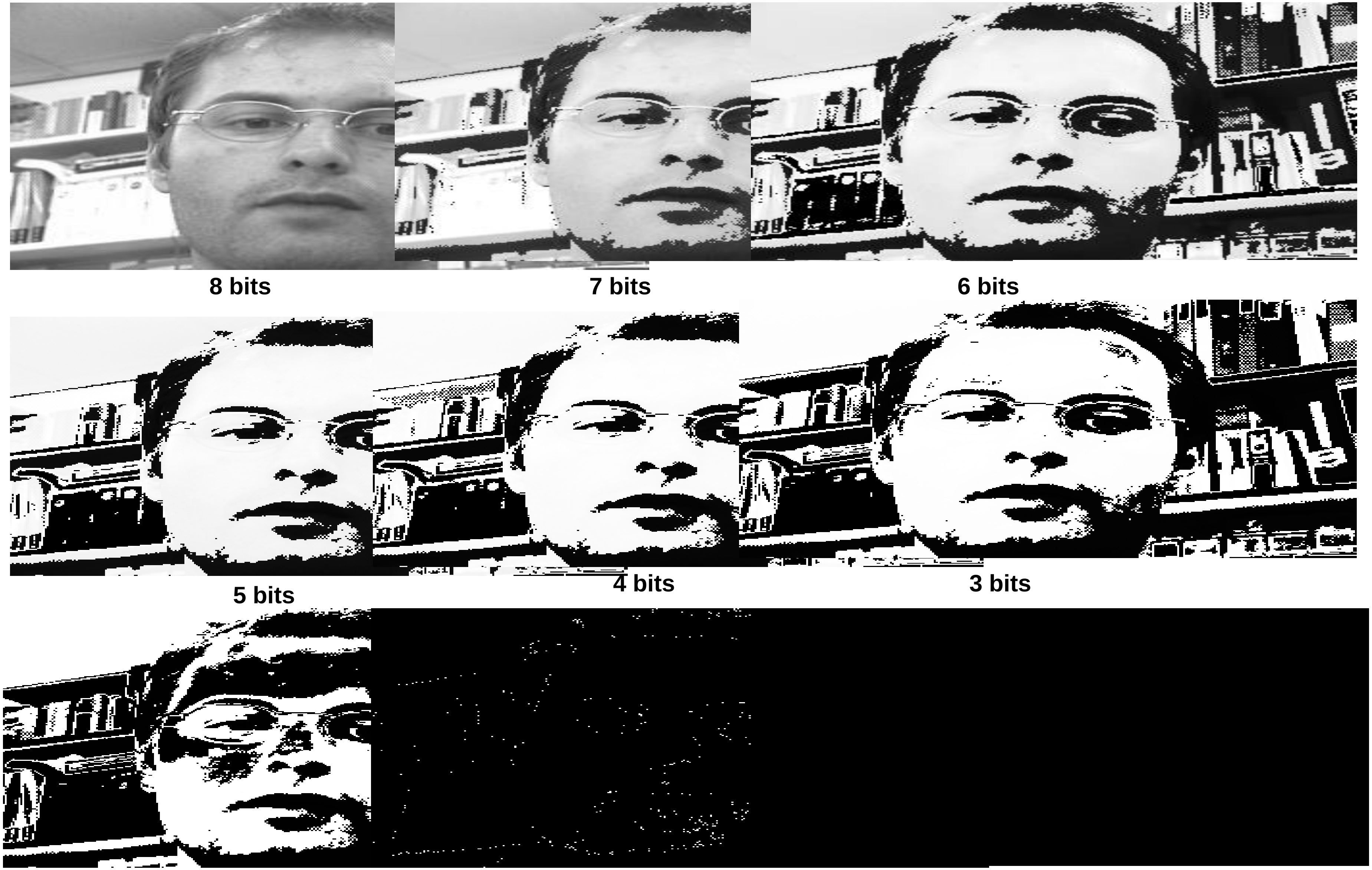
0	1	1	1	0
1	1	1	1	1
1	1	1	1	1
1	0	0	1	0
1	0	0	0	0
		1 bits		

Original image f is reasonably bright, but gradually the pixels get darker as the Grey-level resolution decreases.

Quantization Examples

- Bi-level (black & white) image (fax)
  - $s = 0$ or 1
- 8-bit color image (photograph)
  - $0 \leq r,g,b \leq 255$
- 10-bit color image (movie)
  - $0 \leq r,g,b \leq 1023$
- 12-bit intensity image (X-ray)
  - $0 \leq s \leq 4095$
- Multi-spectral image (satellite)
  - $0 \leq c1,c2,\ldots,c7 \leq 255$

Effect of grey level resolution



8 bits

7 bits

6 bits

5 bits

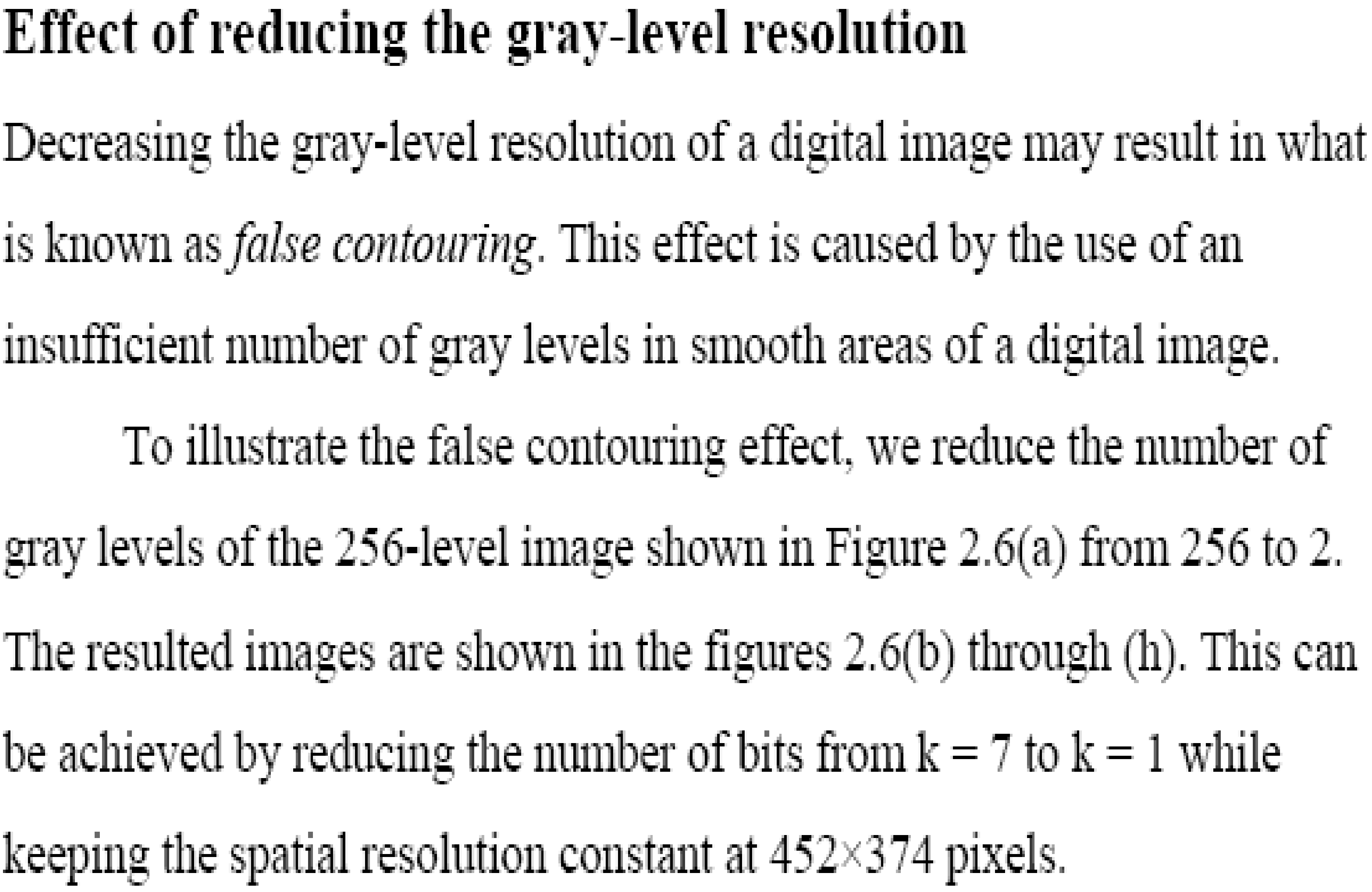
4 bits

3 bits

2 bits

1 bit

0 bits !!!



Effect of reducing the gray-level resolution

Decreasing the gray-level resolution of a digital image may result in what is known as *false contouring*. This effect is caused by the use of an insufficient number of gray levels in smooth areas of a digital image.

To illustrate the false contouring effect, we reduce the number of gray levels of the 256-level image shown in Figure 2.6(a) from 256 to 2. The resulted images are shown in the figures 2.6(b) through (h). This can be achieved by reducing the number of bits from k = 7 to k = 1 while keeping the spatial resolution constant at 452×374 pixels.

We can clearly see that the 256-, 128-, and 64-level images are visually identical. However, the 32-level image shown in Figure 2.6(d) has an almost imperceptible set of very fine ridgelike structures in areas of smooth gray levels (particularly in the skull).False contouring generally is quite visible in images displayed using 16 or less uniformly spaced gray levels, as the images in Figures 2.6(e) through (h) show.

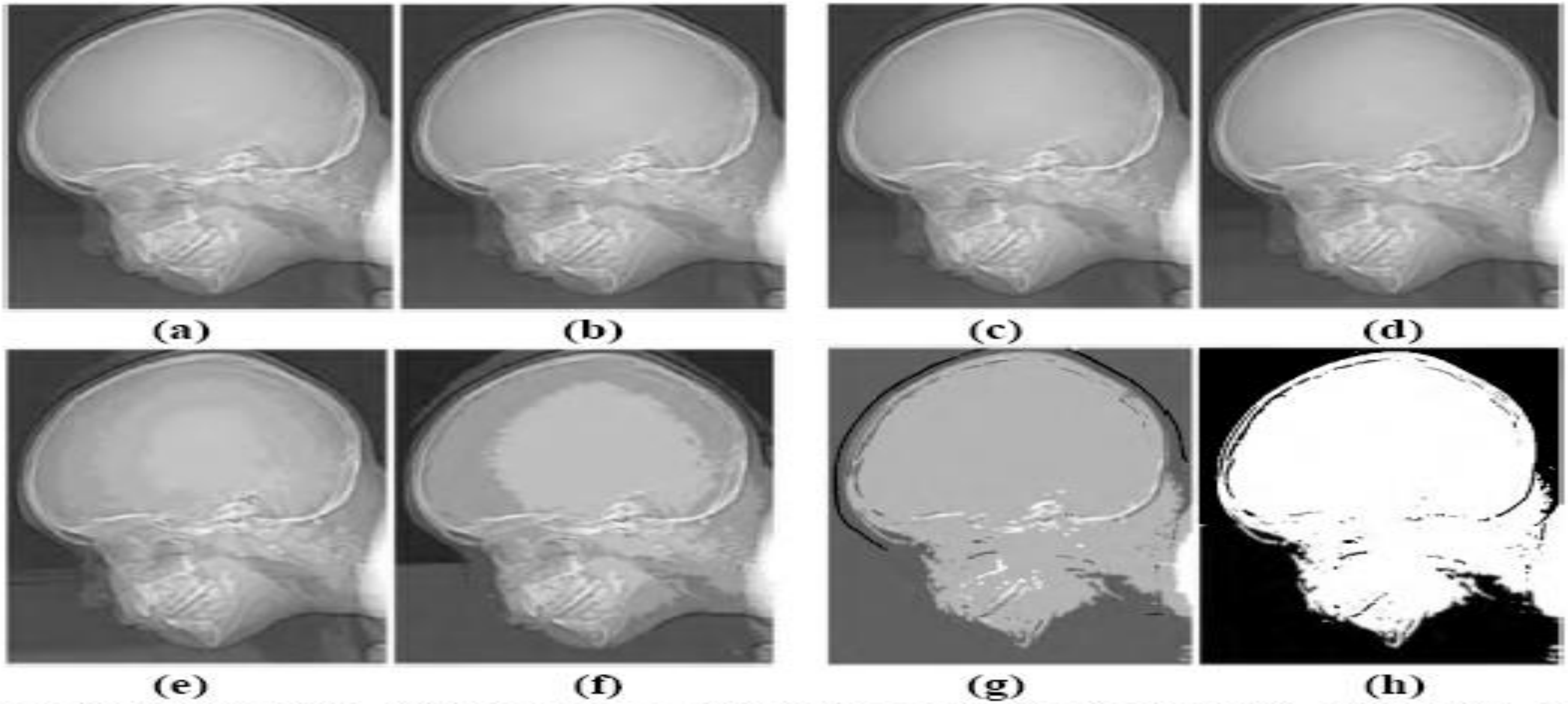


Figure 2.6 (a) 452×374, 256-level image. (b)-(h) Image displayed in 128, 64, 32, 16, 8, 4, and 2 gray levels, while keeping the spatial resolution constant.

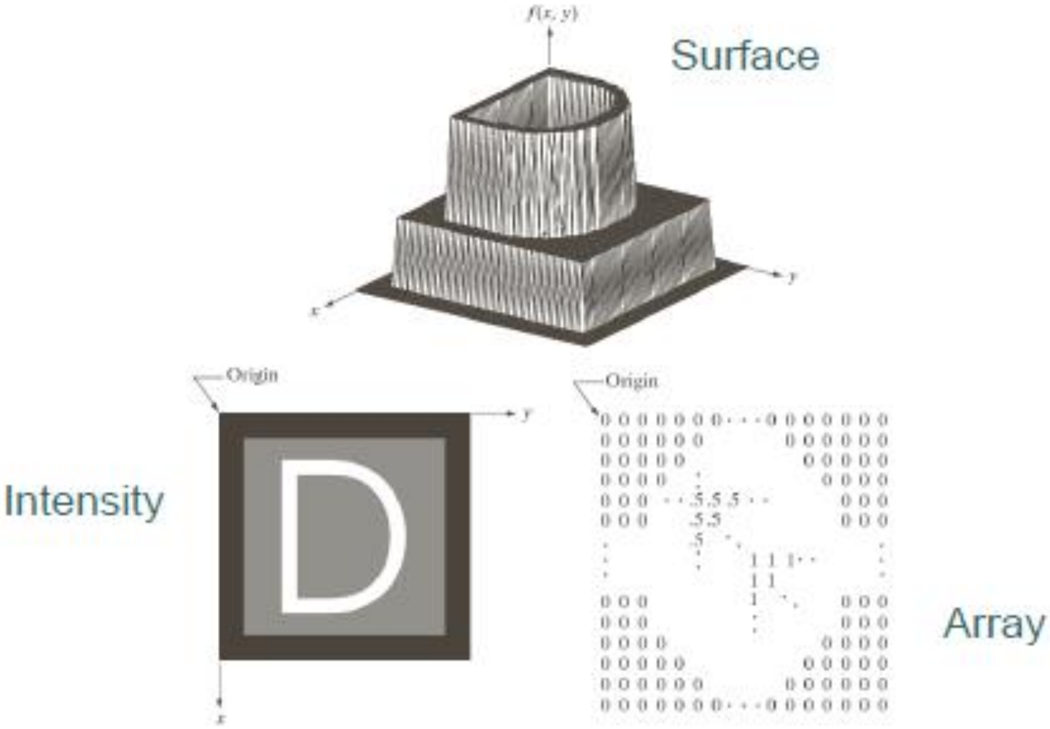
Images as Functions

- We can think of an **image** as a function, *f*, from R^2 to R:
  - $f(x, y)$ gives the **intensity** at position (x, y)
  - Realistically, we expect the image only to be defined over a rectangle, with a finite range:
    - $f: [a,b] \mathrm{x} [c,d] \rightarrow [0,1]$

- A color image is just three functions pasted together. We can write this as a "vector-valued" function:

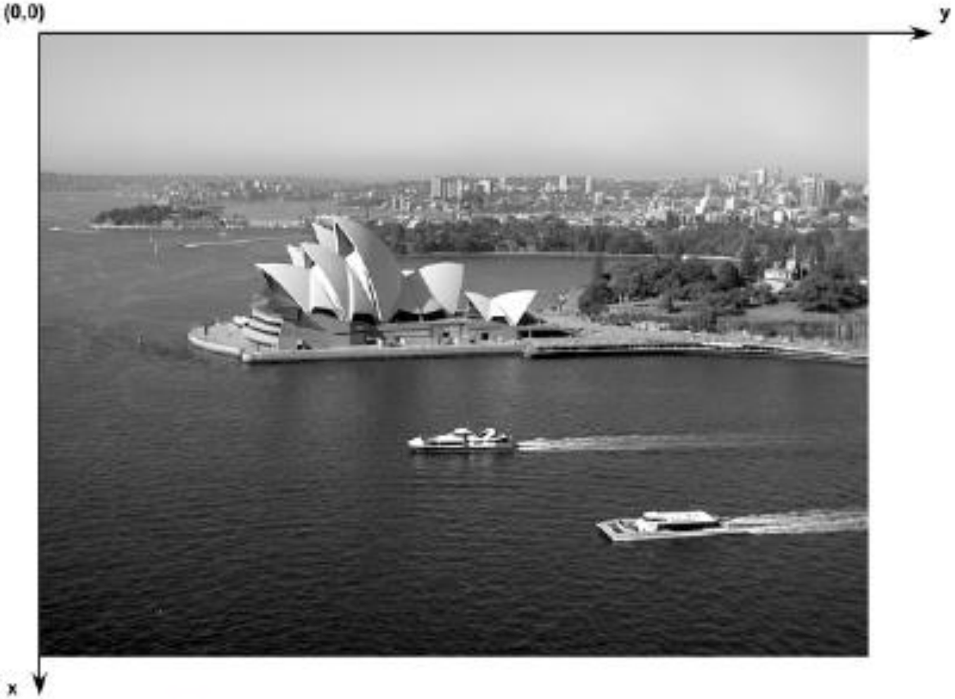
$$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$

Image Representation



```
0 0 0 0 0 0 0 · · · 0 0 0 0 0 0 0
0 0 0 0 0 0               0 0 0 0 0 0
0 0 0 0 0                   0 0 0 0 0
0 0 0 0    :                  0 0 0 0
0 0 0  · · .5 .5 .5 · ·         0 0 0
0 0 0     .5 .5                 0 0 0
:          .5  ·  ·                 :
:           :       1 1 1 · ·       :
:           :       1 1             :
0 0 0               1   ·  ·      0 0 0
0 0 0               :             0 0 0
0 0 0 0             :           0 0 0 0
0 0 0 0 0                     0 0 0 0 0
0 0 0 0 0 0                 0 0 0 0 0 0
0 0 0 0 0 0 0 · · · 0 0 0 0 0 0 0
```

Intensity Representation



NB: There is no universally accepted convention or notation. Always check carefully!

Array Representation

- Mathematical notation

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

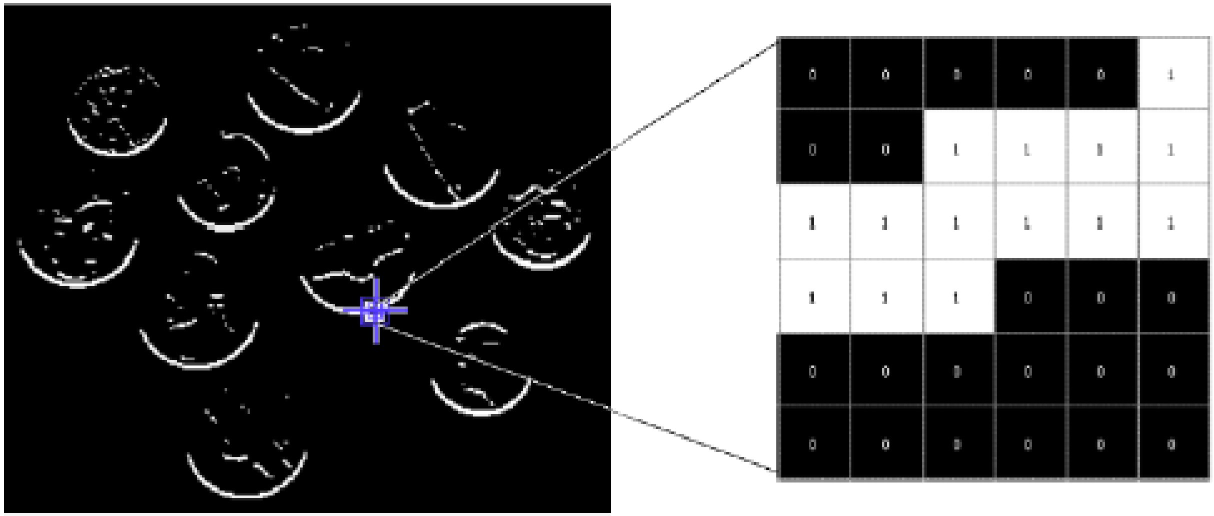
Array Representation

- MATLAB notation

$$\texttt{f(p,q)} = \begin{bmatrix} \texttt{f(1,1)} & \texttt{f(1,2)} & \cdots & \texttt{f(1,N)} \\ \texttt{f(2,1)} & \texttt{f(2,2)} & \cdots & \texttt{f(2,N)} \\ \vdots & \vdots & & \vdots \\ \texttt{f(M,1)} & \texttt{f(M,2)} & \cdots & \texttt{f(M,N)} \end{bmatrix}$$

Digital image representation

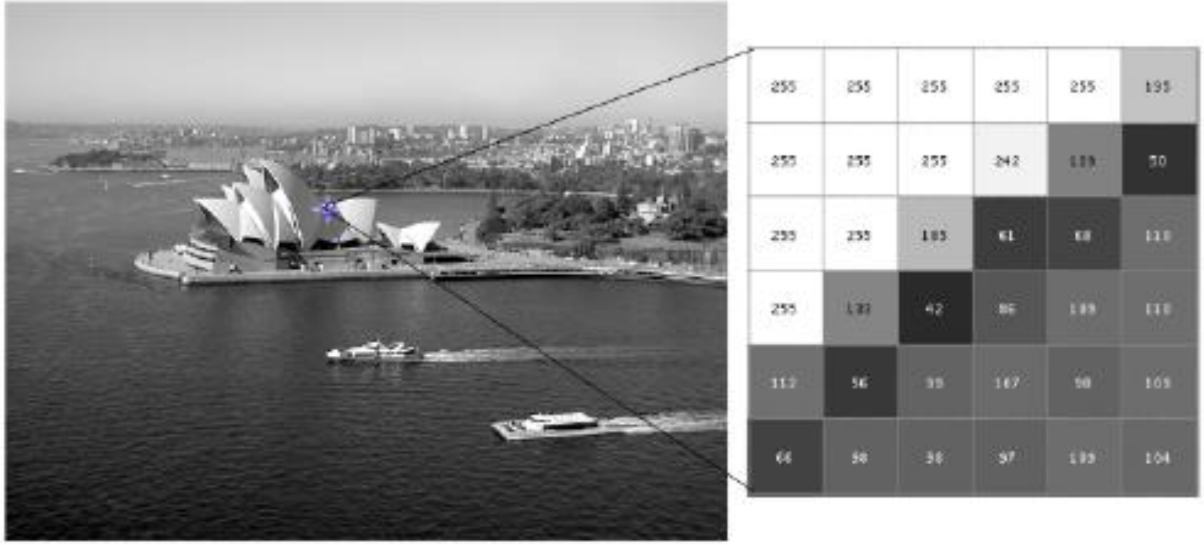
- Binary (1-bit) images
  - 2D array, one bit per pixel, a 0 *usually* means "black" and a 1 means "white".
  - In MATLAB: binary images are represented using a `logical` array of 0s and 1s.



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

Digital image representation

- Gray-level (8-bit) images
  - 2D array, 8 bits per pixel, a 0 usually means "black" and a 255 means "white".
  - In MATLAB: intensity images can be represented using different data types (or classes): **`uint8`**, **`uint16`**, or



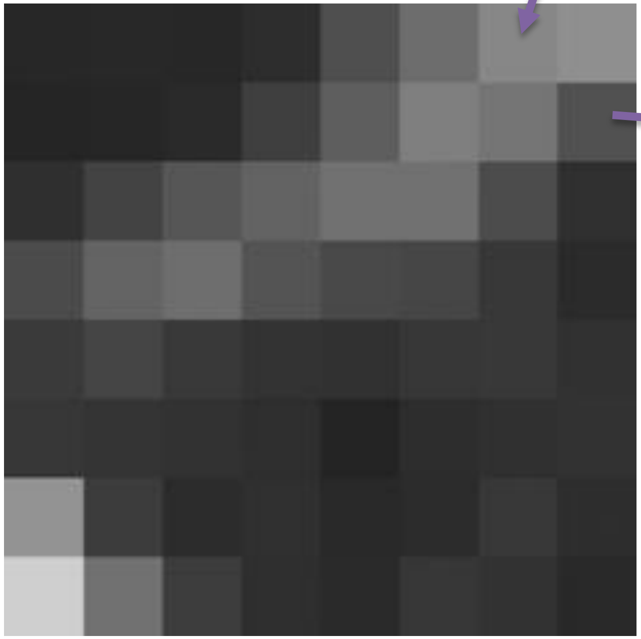
255	255	255	255	255	195
255	255	255	242	135	50
255	255	185	61	68	110
255	133	42	86	139	110
112	56	99	107	98	105
66	98	98	97	133	104

By Oge Marques Copyright © 2011 by John Wiley & Sons, Inc. All rights reserved.

What is digital image?

The image consists of finite number of pixels (f(x,y))

Every pixel Is an intersection تقاطع between a row and a column.



every pixel has intensity كثافة

☝ **Ex:**

f(4,3)= 123

Refers to a pixel existing on the intersection between row 4 with column 3, and its intensity is 123.

Digital image representation

- Color images
  - **RGB representation**: each pixel is usually represented by a 24-bit number containing the amount of its Red (R), Green (G), and Blue (B) components (8 bits per component)
  - **Indexed representation**: a 2D array contains indices to a color palette (or look-up table, LUT).

Digital image representation

(a) 24-bit (true color) RGB image

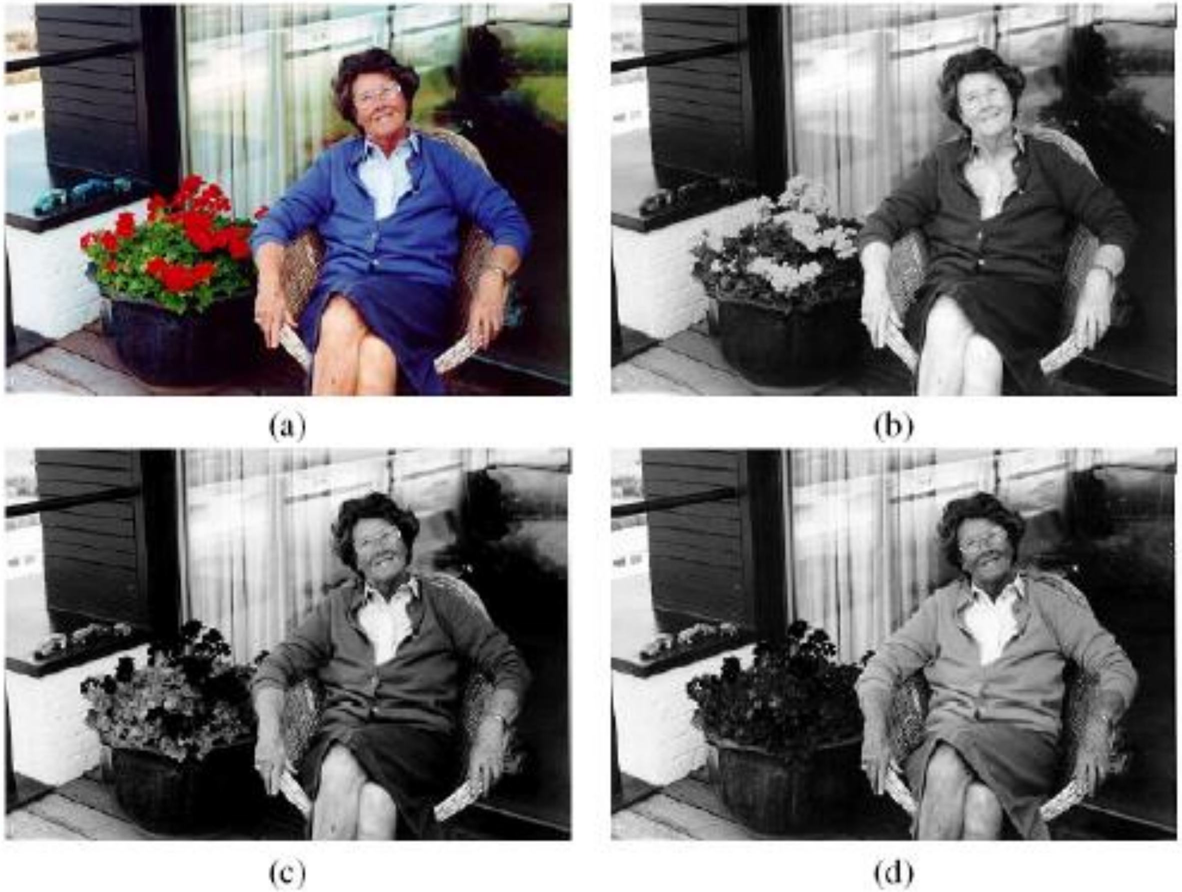
(b) R

(c) G

(d) B

2^24=

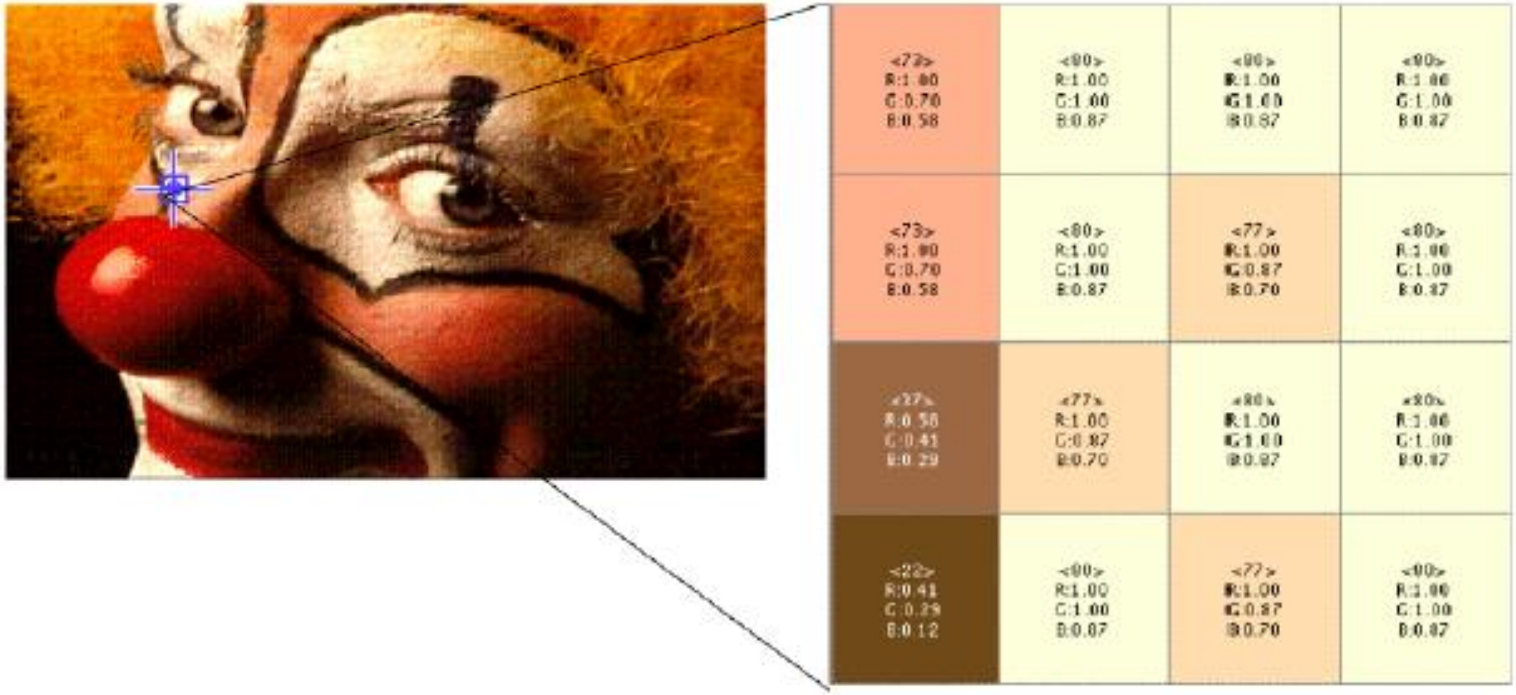
16M colors


(a)
(b)
(c)
(d)

By Oge Marques

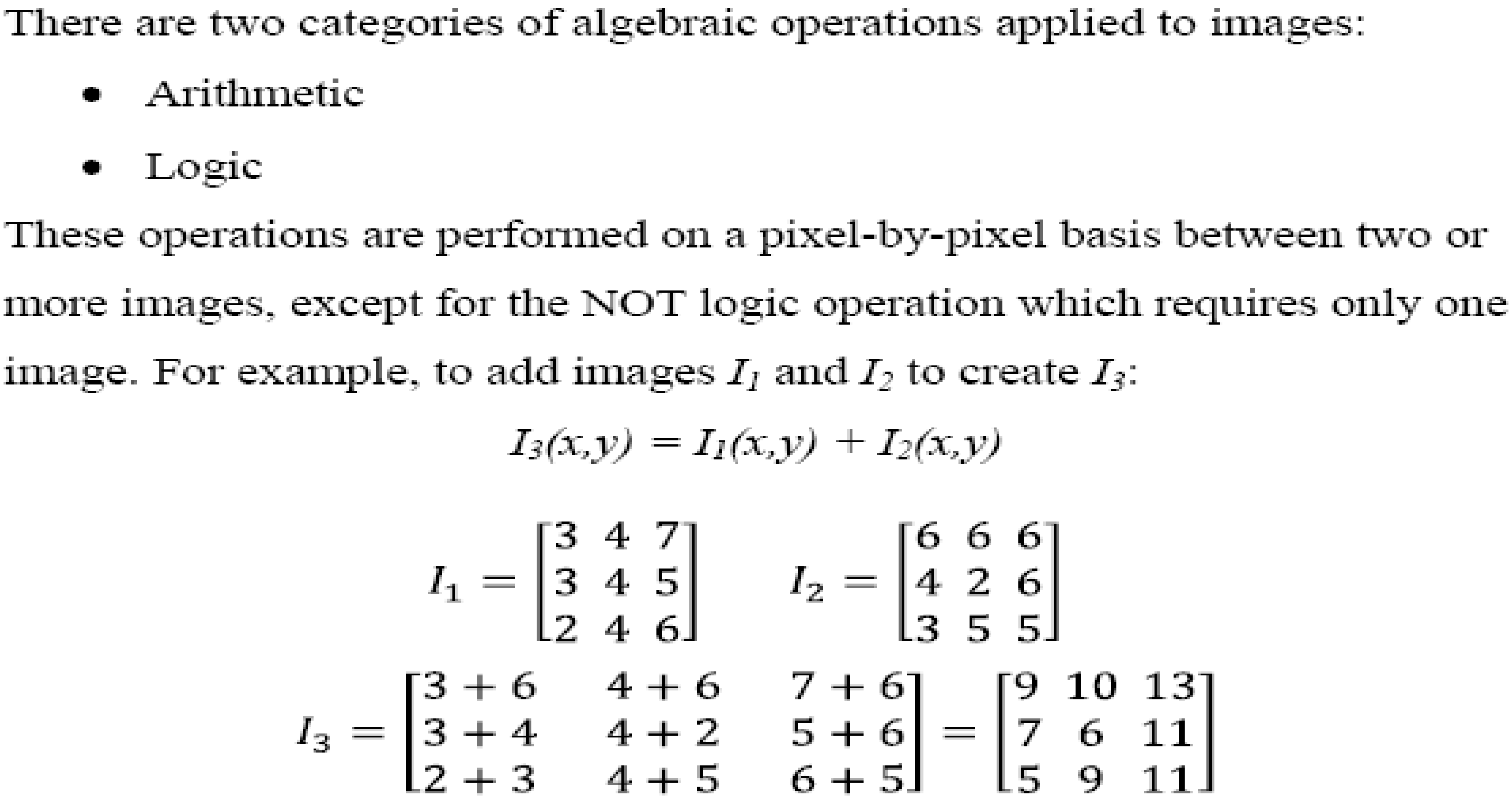
Digital image representation

- Indexed color images: old hardware can not display 16M colors. Use a pointer to a color palette (color map) . Typically 256 colors.



<73> R:1.00 G:0.70 B:0.58	<80> R:1.00 G:1.00 B:0.87	<80> R:1.00 G:1.00 B:0.87	<80> R:1.00 G:1.00 B:0.87
<73> R:1.00 G:0.70 B:0.58	<80> R:1.00 G:1.00 B:0.87	<77> R:1.00 G:0.87 B:0.70	<80> R:1.00 G:1.00 B:0.87
<37> R:0.58 G:0.41 B:0.29	<77> R:1.00 G:0.87 B:0.70	<80> R:1.00 G:1.00 B:0.87	<80> R:1.00 G:1.00 B:0.87
<22> R:0.41 G:0.29 B:0.12	<80> R:1.00 G:1.00 B:0.87	<77> R:1.00 G:0.87 B:0.70	<80> R:1.00 G:1.00 B:0.87

Mathematical representation of Digital Images



There are two categories of algebraic operations applied to images:

- Arithmetic
- Logic

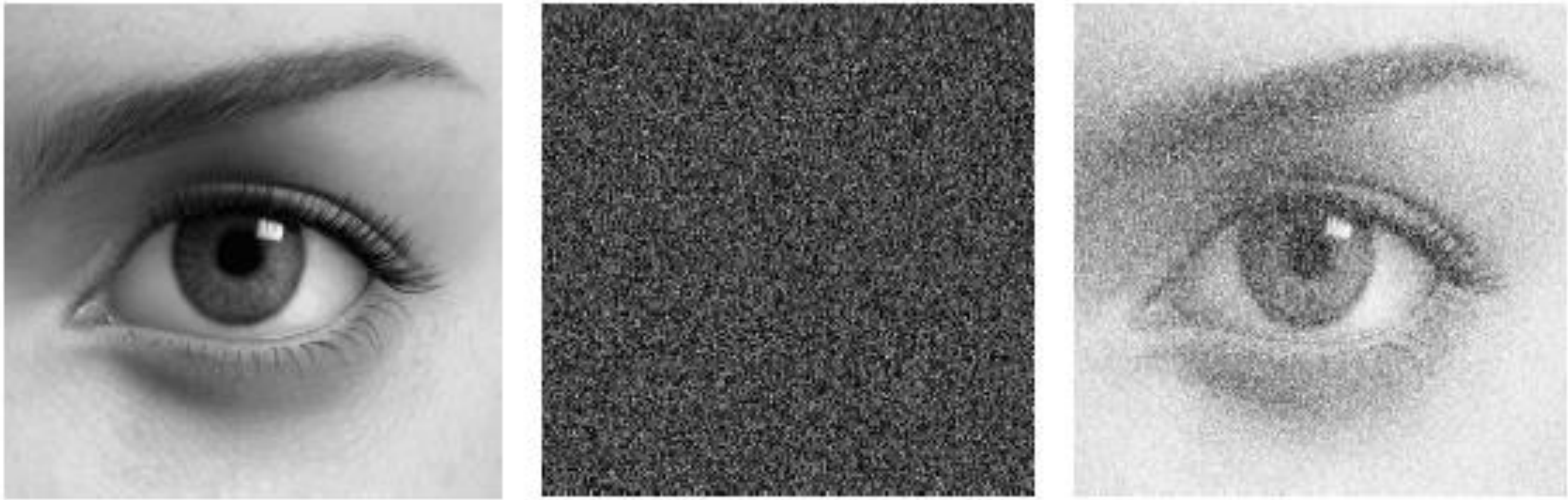
These operations are performed on a pixel-by-pixel basis between two or more images, except for the NOT logic operation which requires only one image. For example, to add images I_1 and I_2 to create I_3:

$$I_3(x,y) = I_1(x,y) + I_2(x,y)$$

$$I_1 = \begin{bmatrix} 3 & 4 & 7 \\ 3 & 4 & 5 \\ 2 & 4 & 6 \end{bmatrix} \qquad I_2 = \begin{bmatrix} 6 & 6 & 6 \\ 4 & 2 & 6 \\ 3 & 5 & 5 \end{bmatrix}$$

$$I_3 = \begin{bmatrix} 3+6 & 4+6 & 7+6 \\ 3+4 & 4+2 & 5+6 \\ 2+3 & 4+5 & 6+5 \end{bmatrix} = \begin{bmatrix} 9 & 10 & 13 \\ 7 & 6 & 11 \\ 5 & 9 & 11 \end{bmatrix}$$

- **Addition** is used to combine the information in two images. Applications include development of image restoration algorithms for modeling additive noise and special effects such as image morphing in motion pictures as shown in the figures below.

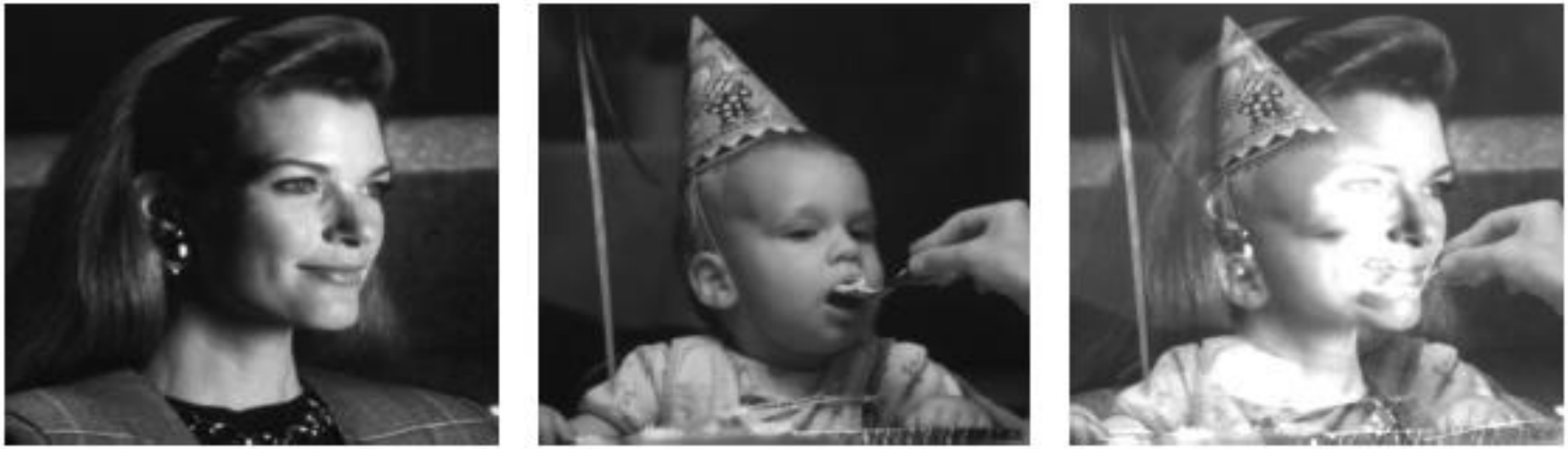


(a) Original image

(b) Gaussian noise

(c) Addition of images (a) and (b)

Image addition (adding noise to the image)



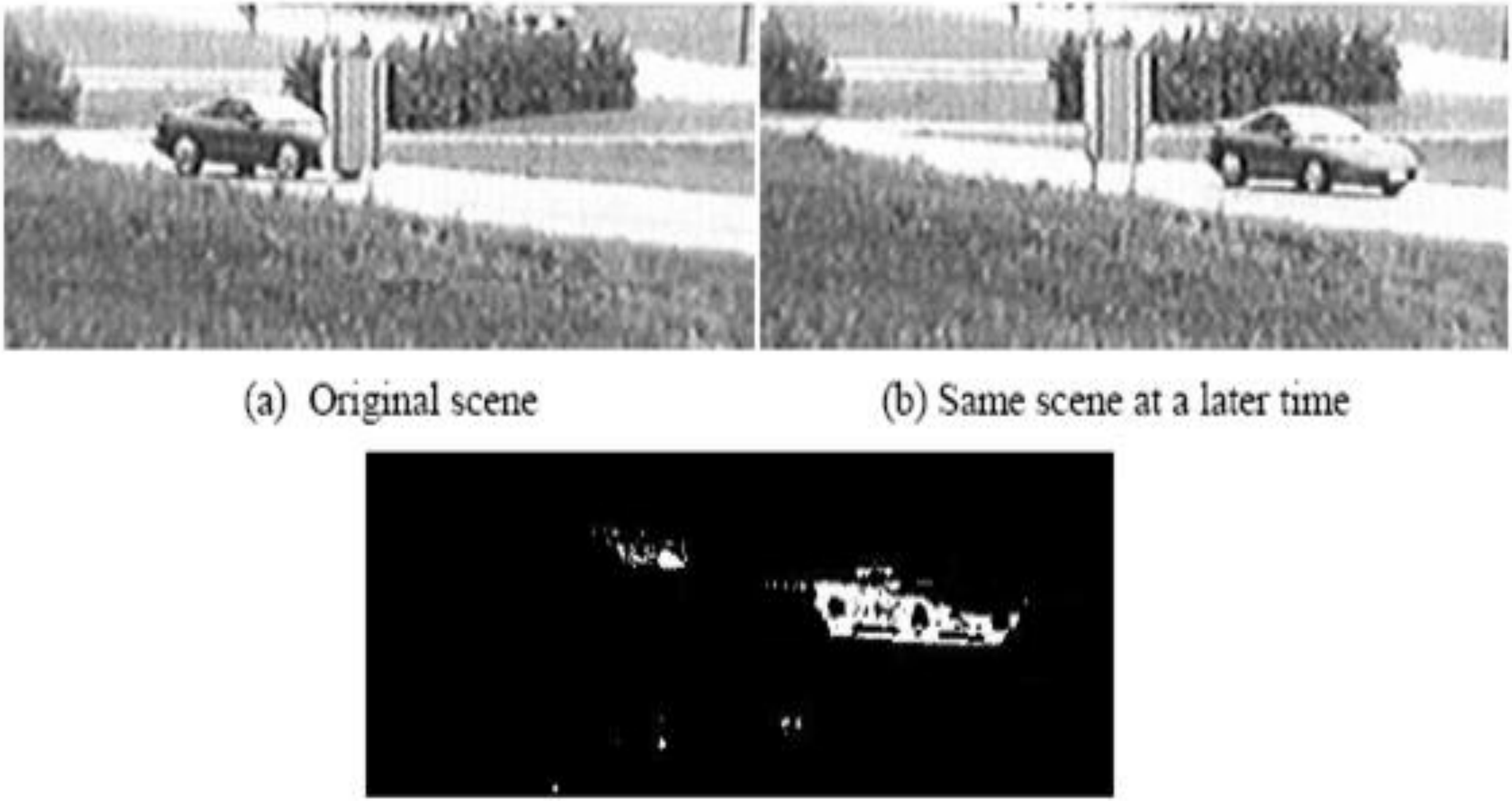
(a) First Original

(b) Second Original

(c) Addition of images (a) and (b)

Figure 3.3 Image addition (image morphing example)

- **Subtraction** of two images is often used to detect motion. For example, in a scene when nothing has changed, the image resulting from the subtraction is filled with zeros(black image). If something has changed in the scene, subtraction produces a nonzero result at the location of movement as shown in the figure below.



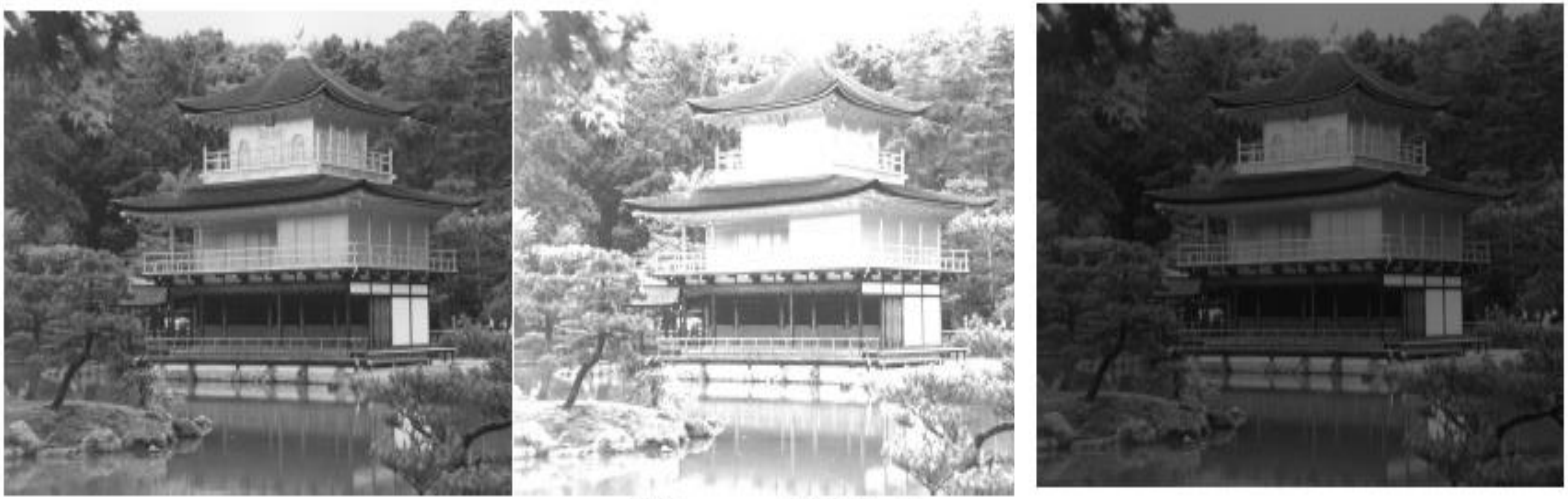
(a) Original scene

(b) Same scene at a later time

(c) Subtracting image (b) from (a). Only moving objects appear in the resulting image

Image subtraction

- **Multiplication and division** are used to adjust the brightness of an image. Multiplying the pixel values by a number greater than one will brighten the image, and dividing the pixel values by a factor greater than one will darken the image. An example of brightness adjustment is shown in the figure below.



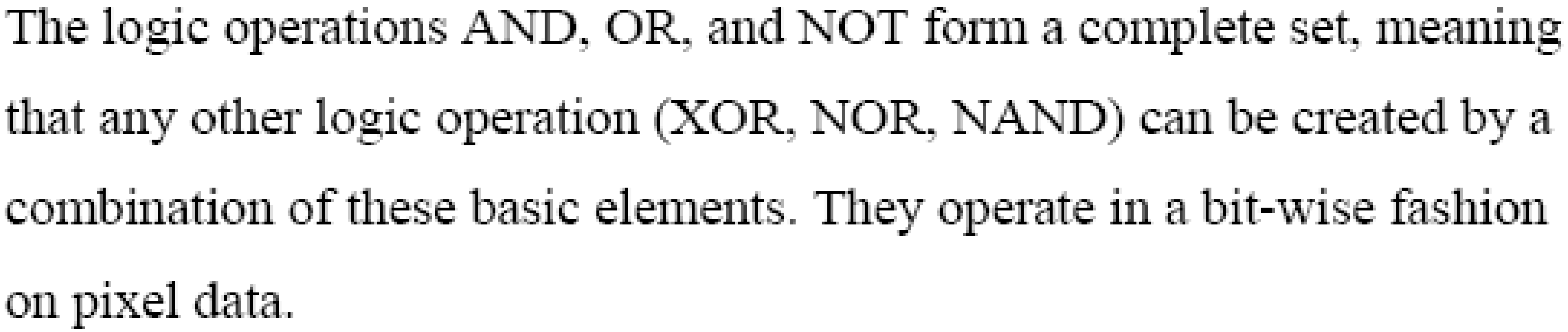
(a) Original image

(b) Image multiplied by 2

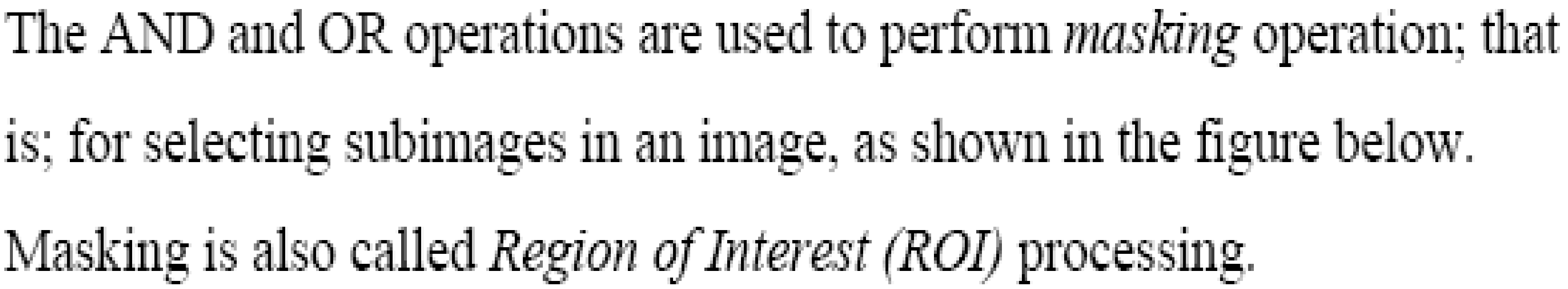
(c) Image divided by 2

Image multiplication and division

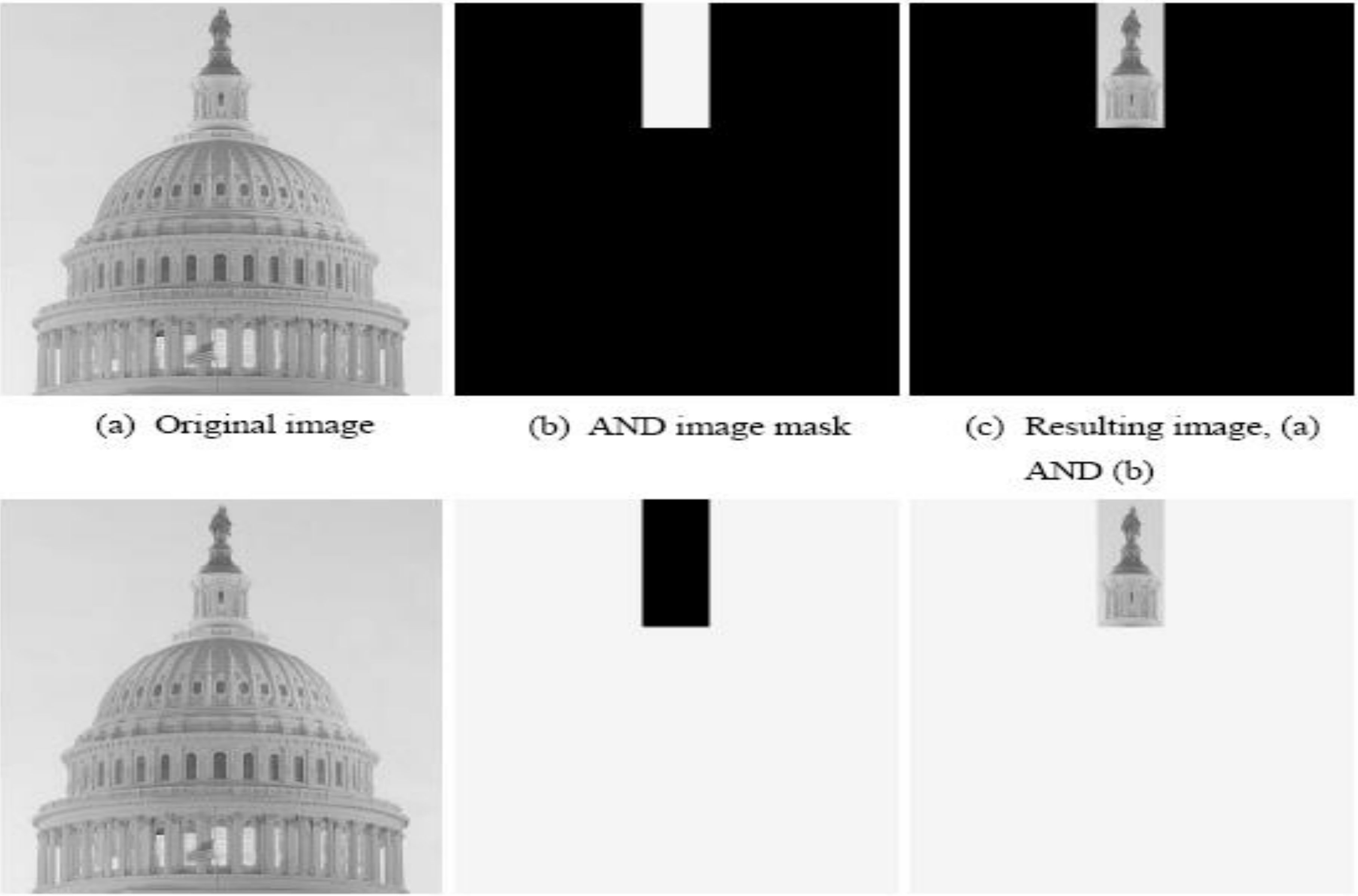
Logical Operations



The logic operations AND, OR, and NOT form a complete set, meaning that any other logic operation (XOR, NOR, NAND) can be created by a combination of these basic elements. They operate in a bit-wise fashion on pixel data.



The AND and OR operations are used to perform *masking* operation; that is; for selecting subimages in an image, as shown in the figure below. Masking is also called *Region of Interest (ROI)* processing.



(a) Original image

(b) AND image mask

(c) Resulting image, (a) AND (b)

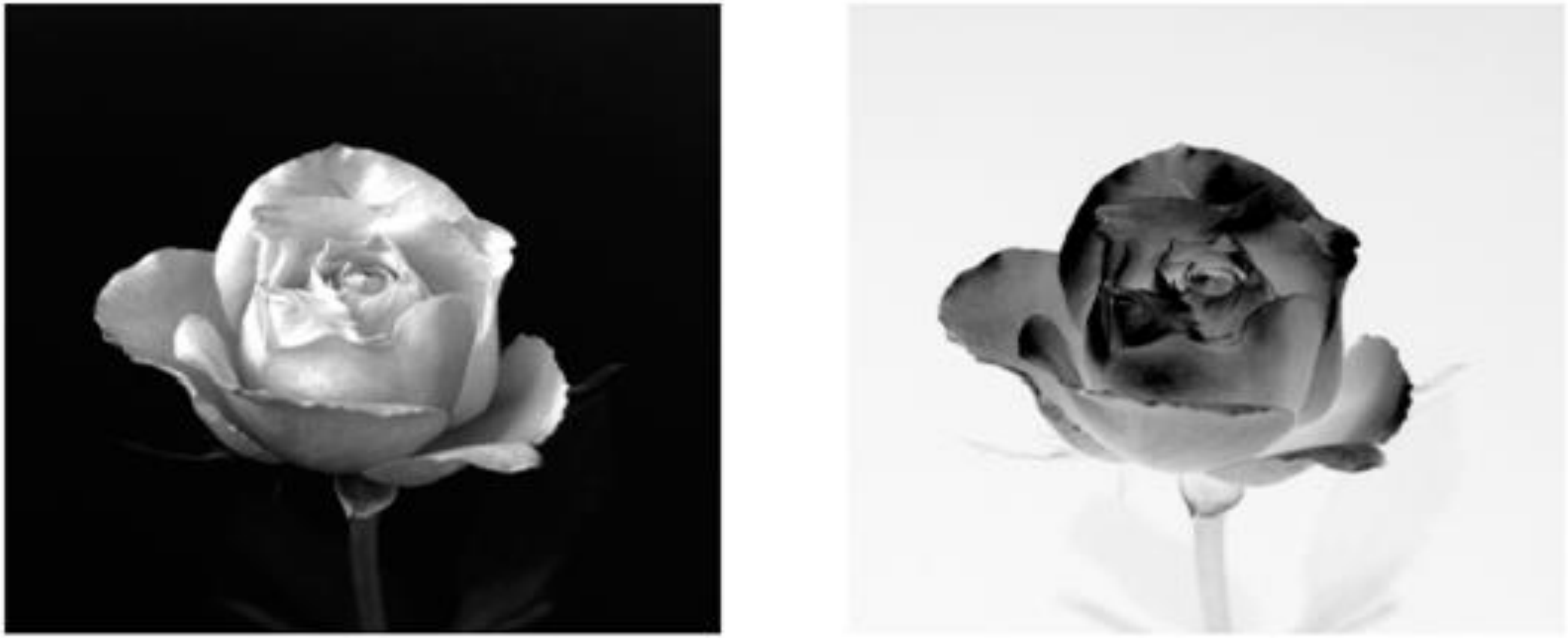
(d) Original image

(e) OR image mask

(f) Resulting image, (d) OR (e)

Image masking

The NOT operation creates a negative of the original image ,as shown in the figure below. by inverting each bit within each pixel value.



(a) Original image

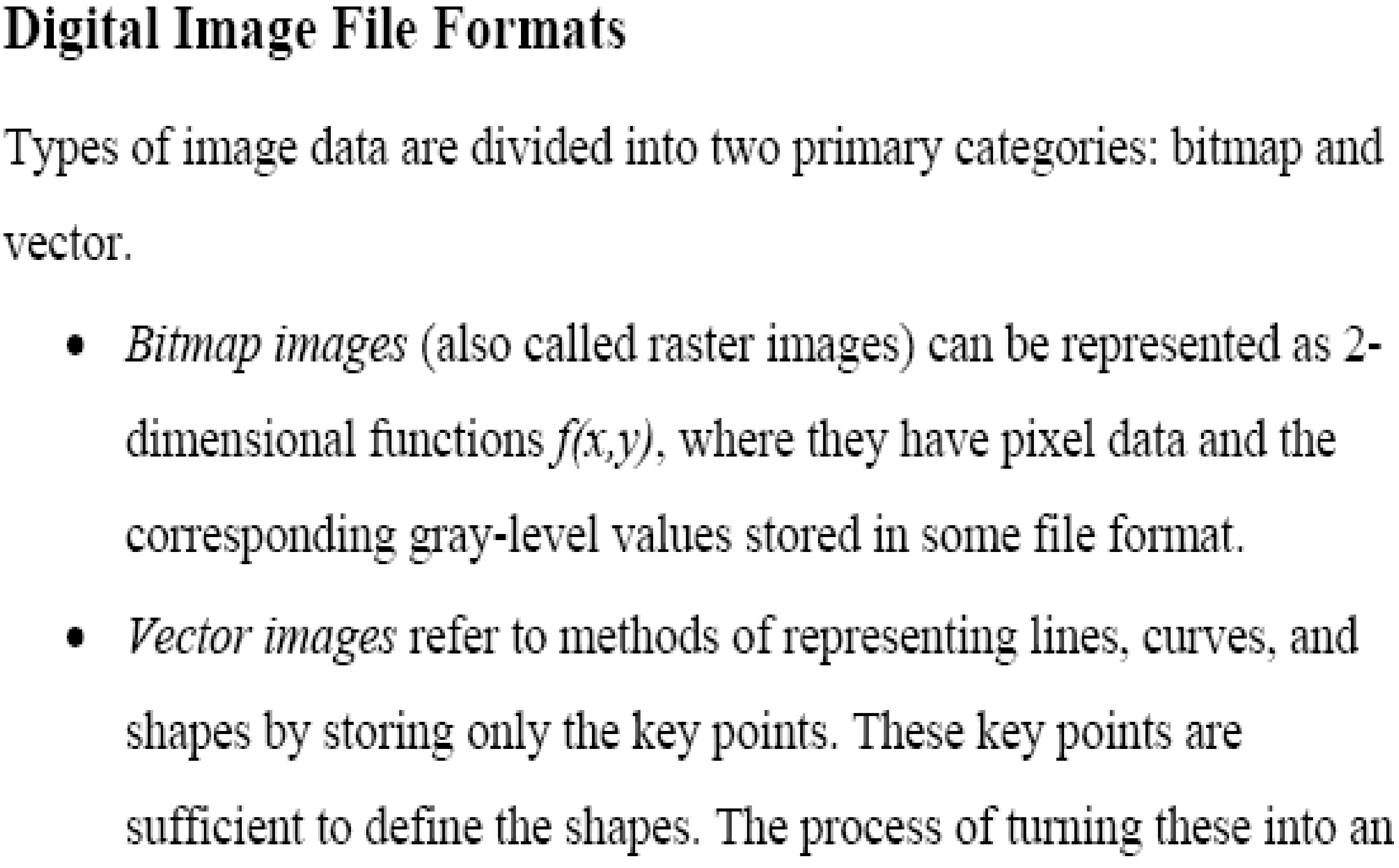
(b) NOT operator applied to image (a)

Complement image

Image files Format

Image files consists of two parts:

- A ***header*** found at the start of the file and consisting of parameters regarding:
  - ✓ Number of rows (height)
  - ✓ Number of columns (width)
  - ✓ Number of bands (i.e. colors)
  - ✓ Number of bits per pixel (bpp)
  - ✓ File type
- Image ***data*** which lists all pixel values (vectors) on the first row, followed by 2nd row, and so on.



Digital Image File Formats

Types of image data are divided into two primary categories: bitmap and vector.

- *Bitmap images* (also called raster images) can be represented as 2-dimensional functions *f(x,y)*, where they have pixel data and the corresponding gray-level values stored in some file format.
- *Vector images* refer to methods of representing lines, curves, and shapes by storing only the key points. These key points are sufficient to define the shapes. The process of turning these into an

Most of the types of file formats fall into the category of bitmap images, for example:

- PPM (Portable Pix Map) format
- TIFF (Tagged Image File Format)
- GIF (Graphics Interchange Format)
- JPEG (Joint Photographic Experts Group) format
- BMP (Windows Bitmap)
- PNG (Portable Network Graphics)
- XWD (X Window Dump)

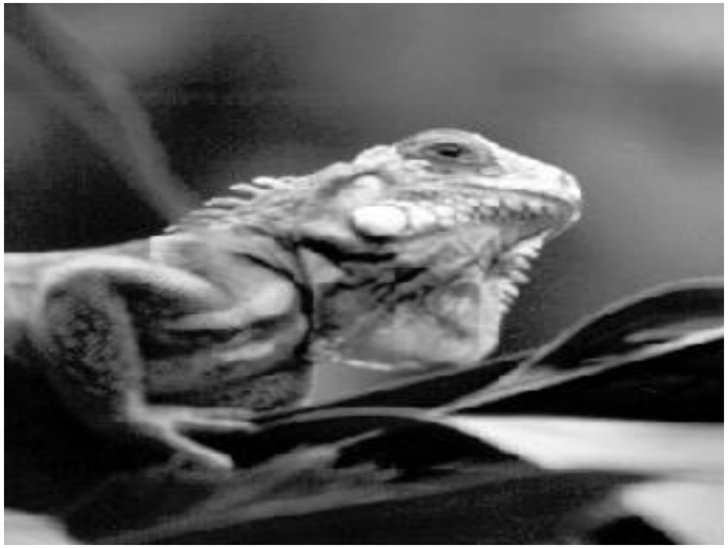
Image Histogram

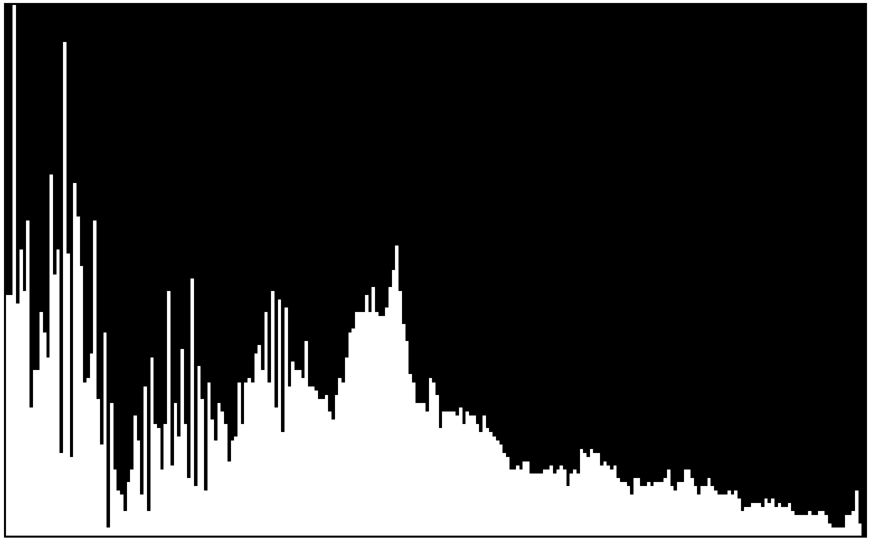
- The distribution of gray levels in an image convey some useful information on the image content.
- For any image f of size mxn and Gray Level resolution k, the histogram of h is a discrete function defined on the set {0, 1, …, 2^k-1} of gray values such that h(i) is th number of pixels in the image f which have the gray value i.
- It is customary to "normalise" a histogram by dividing h(i) by the total number of pixels in the image, i.e. use the probability distribution:

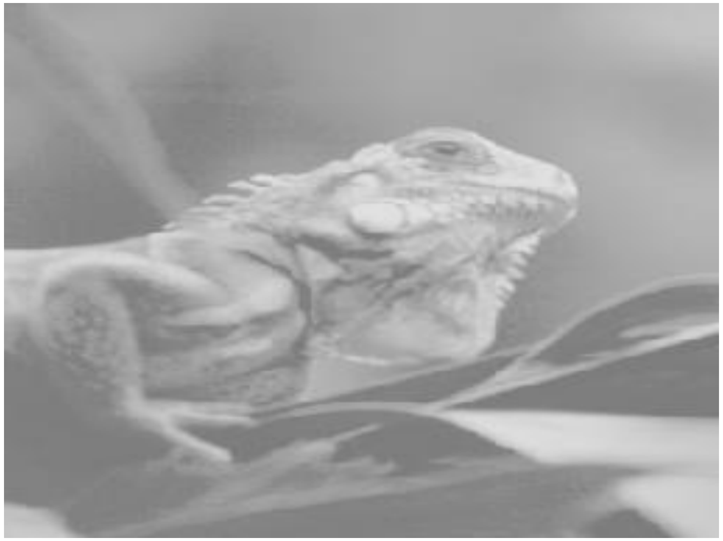
$$p(i) = h(i)/mn.$$

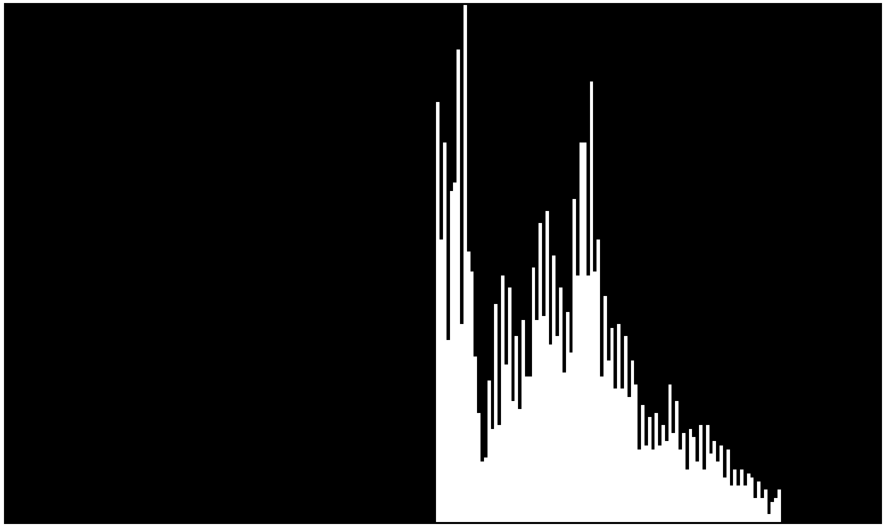
- Histograms are used in numerous processing operations.

Histograms - Examples

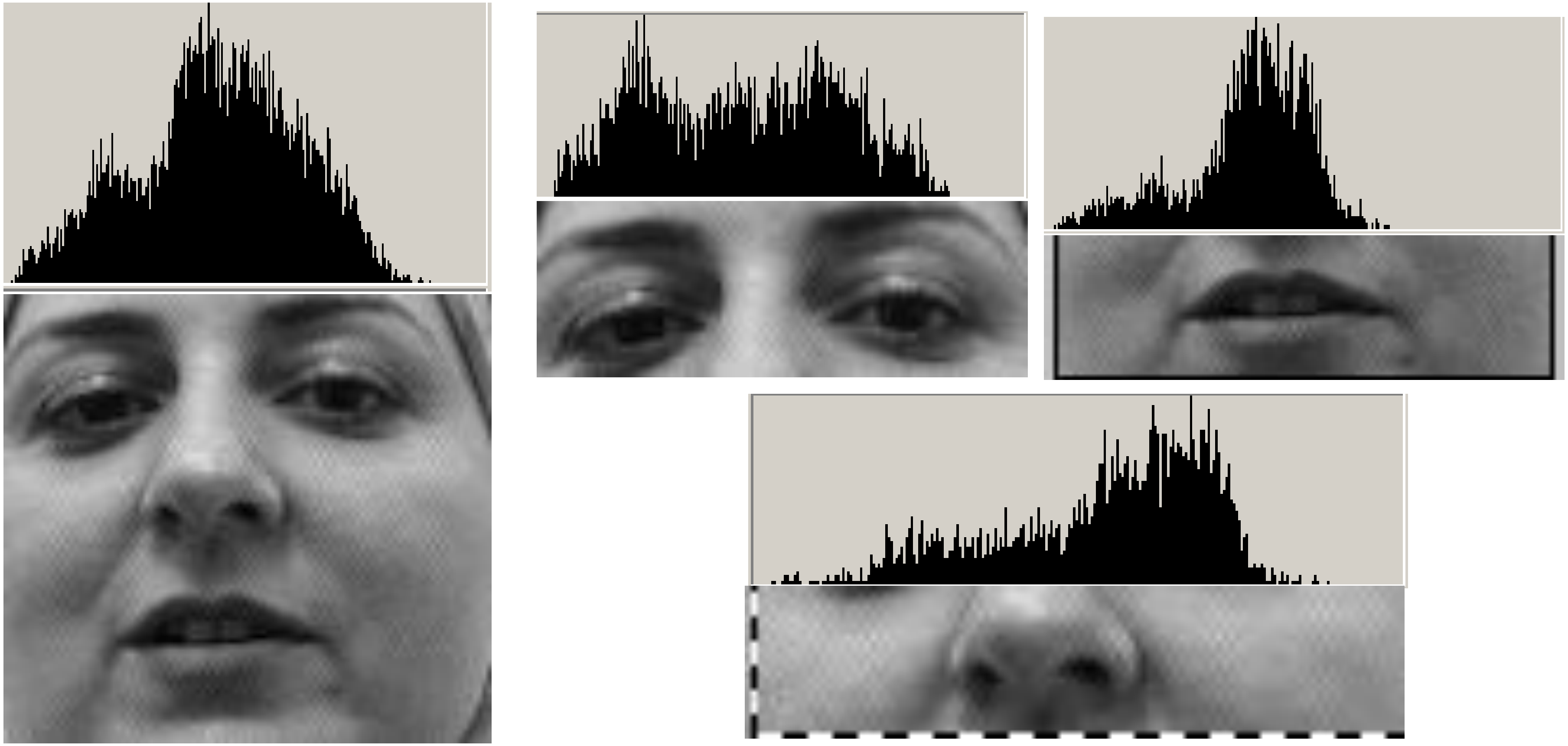








Local Vs. Global Histograms – Image Features



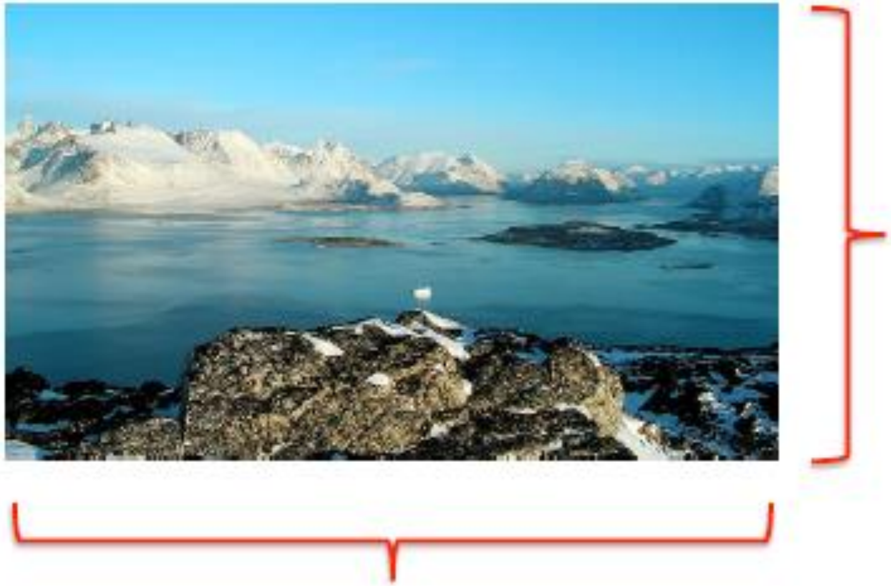
❑ **Histograms for parts of an image provide useful tools for feature analysis.**

❑ **Local Histograms provide more information on image content than the global histogram.**

Compression

- Most image file formats employ some type of compression.
- Compression methods can be:
  - **Lossy**: a tolerable degree of deterioration is introduced in visual quality.
  - **Lossless**: image is encoded in its full quality.
- As a general guideline:
  - **Lossy** compression should be used for general purpose photographic images
  - **Lossless** compression should be used for images in which no loss of detail may be tolerable (e.g., space images and medical images).

Image Aspect Ratio

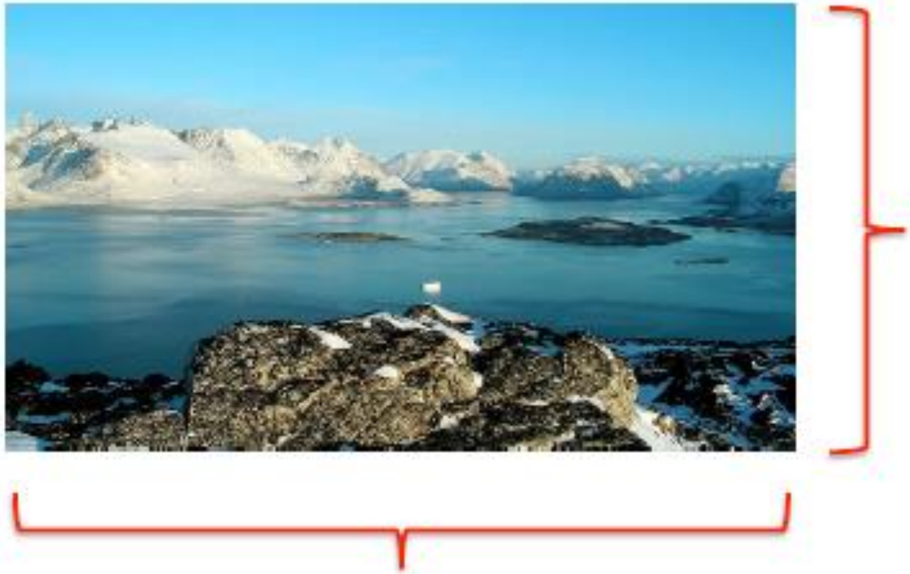


Height in cm (H)

Width in cm (W)

Image Aspect Ratio = image width/image height = W/H

Pixel Aspect Ratio



Number of lines (N)

Number of pixels per line (M)

Pixel Aspect Ratio = pixel width/pixel height = WN/HM