

# Digital Image Processing

Dr. Mubashir Ahmad (Ph.D.)

# SOME BASIC RELATIONSHIPS BETWEEN PIXELS

- A pixel  $p$  at coordinates  $(x, y)$  has two horizontal and two vertical neighbors with coordinates.

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

- This set of pixels, called the *4-neighbors* of  $p$ , is denoted  $N_4(p)$ . The four *diagonal* neighbors of  $p$  have coordinates
- $(x + 1, y + 1), (x + 1, y - 1), (x - 1, y + 1), (x - 1, y - 1)$
- and are denoted  $N_D(p)$ . These neighbors, together with the 4-neighbors, are called the *8-neighbors* of  $p$ , denoted by  $N_8(p)$ . The set of image locations of the neighbors of a point  $p$  is called the *neighborhood* of  $p$ . The neighborhood is said to be *closed* if it contains  $p$ . Otherwise, the neighborhood is said to be *open*.

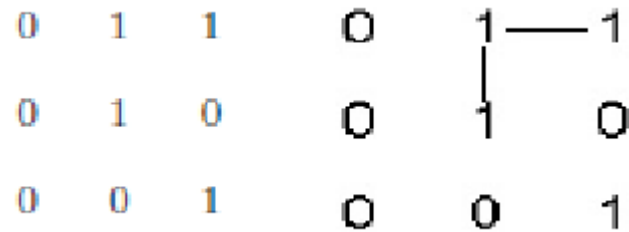
# ADJACENCY, CONNECTIVITY, REGIONS, AND BOUNDARIES

- Let  $V$  be the set of intensity values used to define adjacency. In a binary image,  $V = \{1\}$  if we are referring to adjacency of pixels with value 1. In a grayscale image, the idea is the same, but set  $V$  typically contains more elements. For example, if we are dealing with the adjacency of pixels whose values are in the range 0 to 255, set  $V$  could be any subset of these 256 values. We consider three 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* (also called *mixed adjacency*). Two pixels  $p$  and  $q$  with values from  $V$  are  $m$ -adjacent if
    - (a)  $q$  is in  $N_4(p)$ , or
    - (b)  $q$  is in  $N_D(p)$  and the set  $N_4(p) \cap N_4(q)$  has no pixels whose values are from  $V$ .

# Connectivity

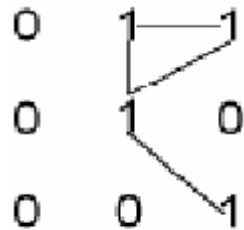
- 4-adjacency:  $v = \{1\}$

Two pixels  $p$  and  $q$  with values from  $V$  are 4-adjacent if  $q$  is in the set  $N4(p)$ .



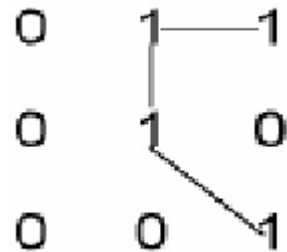
- 8-adjacency:  $v = \{1\}$

Two pixels  $p$  and  $q$  with values from  $V$  are 8-adjacent if  $q$  is in the set  $N8(p)$



# Connectivity

- m-adjacency (mixed adjacency):  
Two pixels  $p$  and  $q$  with values from  $V \setminus \{1\}$  are m-adjacent if:  
(i)  $q$  is in  $N_4(p)$ , or  
(ii)  $q$  is in  $ND(p)$  and  $N_4(p) \cap N_4(q)$  is empty



# Example

0 1 1

0 1 0

0 0 1

**Fig: An arrangement  
of pixels**

0 1—1

0 1 0

0 0 1

**Fig: 4-connectivity of  
pixels**

0 1—1

0 1 0

0 0 1

**Fig: 8-connectivity of  
pixels**

0 1—1

0 1 0

0 0 1

**Fig: m-connectivity of  
pixels**

# Connectivity?

o		o	o	o
o	o		o	o
o		o		o
o	o	o		o
o			o	o

# Question

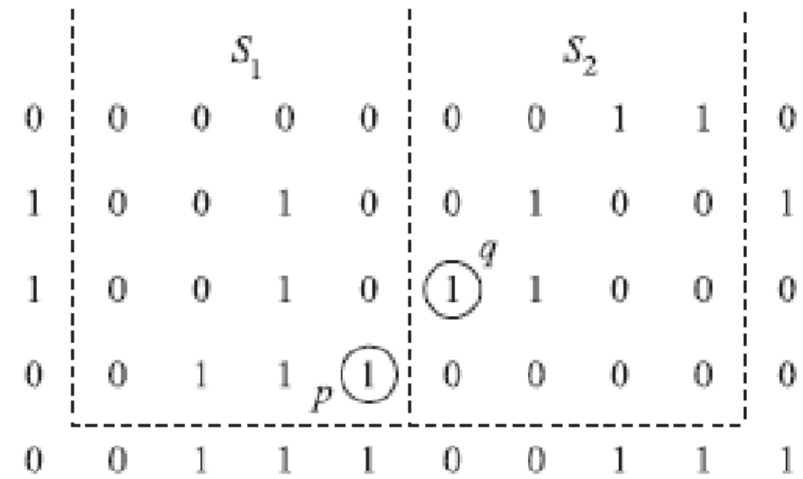
- Consider the two image subsets,  $S_1$  and  $S_2$ , shown in the following figure. For  $V=\{1\}$ , determine whether these two subsets are (a) 4-adjacent, (b) 8-adjacent, or (c) m-adjacent

	$S_1$					$S_2$				
0	0	0	0	0	0	0	0	1	1	0
1	0	0	1	0	0	0	1	0	0	1
1	0	0	1	0	1	1	1	0	0	0
0	0	1	1	1	0	0	0	0	0	0
0	0	1	1	1	0	0	0	1	1	1



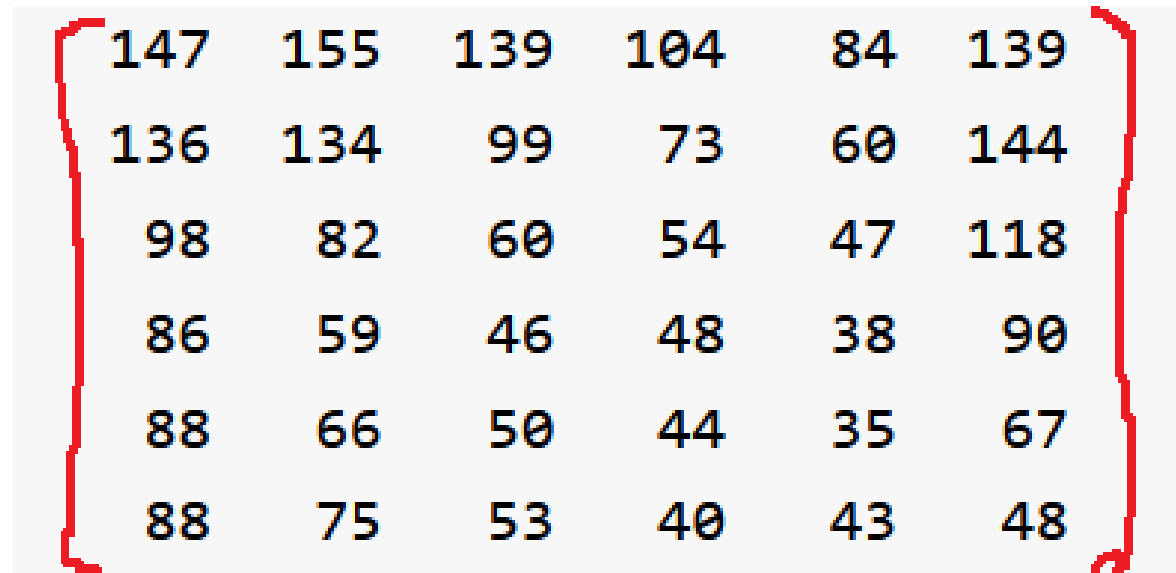
# solution

- Let  $p$  and  $q$  be as shown in Fig. Then:
  - $S_1$  and  $S_2$  are not 4-connected because  $q$  is not in the set  $N_4(p)$ ;
  - $S_1$  and  $S_2$  are 8-connected because  $q$  is in the set  $N_8(p)$ ;
  - $S_1$  and  $S_2$  are  $m$ -connected because
    - $q$  is in  $ND(p)$ , and
    - the set  $N_4(p) \cap N_4(q)$  is empty.



# Grayscale image connectivity?

- $V = \{1, 2, 3, \dots, 60\}$



147	155	139	104	84	139
136	134	99	73	60	144
98	82	60	54	47	118
86	59	46	48	38	90
88	66	50	44	35	67
88	75	53	40	43	48

# DISTANCE MEASURES

- For pixels  $p$ ,  $q$ , and  $s$ , with coordinates  $(x\ y)$ ,  $(u\ v)$ , and  $(w\ z)$ , respectively,  $D$  is a *distance function* or *metric* if
  - - (a)**  $D(p, q) \geq 0$  ( $D(p, q) = 0$  iff  $p = q$ ),
    - (b)**  $D(p, q) = D(q, p)$ , and
    - (c)**  $D(p, s) \leq D(p, q) + D(q, s)$ .

# Euclidean and city block distance

- $F(x, y), F(u, v)$
- $F(1, 3), F(5, 3)$
- $D_e = [(1-5)^2 + (3-3)^2]^{1/2}$
- $D_e = (16)^{1/2} = 4$

o		<del>P</del> o	o	o
o	o		o	o
o		o		o
o	o	o		o
o			<del>q</del> o	o

The *Euclidean distance* between  $p$  and  $q$  is defined as

$$D_e(p, q) = \left[ (x - u)^2 + (y - v)^2 \right]^{1/2} \quad (2-19)$$

For this distance measure, the 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  $(x, y)$ .

The  $D_4$  distance, (called the *city-block distance*) between  $p$  and  $q$  is defined as

$$D_4(p, q) = |x - u| + |y - v| \quad (2-20)$$

# City Block distance

The  $D_4$  distance, (called the *city-block distance*) between  $p$  and  $q$  is defined as

$$D_4(p, q) = |x - u| + |y - v| \quad (2-20)$$

In this case, pixels having a  $D_4$  distance from  $(x, y)$  that is less than or equal to some value  $d$  form a diamond centered at  $(x, y)$ . For example, the pixels with  $D_4$  distance  $\leq 2$  from  $(x, y)$  (the center point) form the following contours of constant distance:

$$\begin{array}{ccccc} & & 2 & & \\ & 2 & 1 & 2 & \\ 2 & 1 & 0 & 1 & 2 \\ & 2 & 1 & 2 & \\ & & 2 & & \end{array}$$

# Chessboard distance

The  $D_8$  distance (called the *chessboard distance*) between  $p$  and  $q$  is defined as

$$D_8(p, q) = \max(|x - u|, |y - v|) \quad (2-21)$$

In this case, the pixels with  $D_8$  distance from  $(x, y)$  less than or equal to some value  $d$  form a square centered at  $(x, y)$ . For example, the pixels with  $D_8$  distance  $\leq 2$  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

The pixels with  $D_8 = 1$  are the 8-neighbors of the pixel at  $(x, y)$ .

# ARITHMETIC OPERATIONS

Arithmetic operations between two images  $f(x, y)$  and  $g(x, y)$  are denoted as

$$s(x, y) = f(x, y) + g(x, y)$$

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

$$p(x, y) = f(x, y) \times g(x, y)$$

$$v(x, y) = f(x, y) \div g(x, y)$$

(2-24)

# EXAMPLE 2.5 : Using image addition (averaging) for noise reduction.

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

$$g(x, y) = f(x, y) + \eta(x, y) \quad (2-25)$$

where the assumption is that at every pair of coordinates  $(x, y)$  the noise is uncorrelated<sup>†</sup> and has zero average value. We assume also that the noise and image values are uncorrelated (this is a typical assumption for additive noise). The objective of the following procedure is to reduce the noise content of the output image by adding a set of noisy input images,  $\{g_i(x, y)\}$ . This is a technique used frequently for image enhancement.

If the noise satisfies the constraints just stated, it can be shown (Problem 2.26) that if an image  $\bar{g}(x, y)$  is formed by averaging  $K$  different noisy images,

$$\bar{g}(x, y) = \frac{1}{K} \sum_{i=1}^K g_i(x, y) \quad (2-26)$$

then it follows that

$$E\{\bar{g}(x, y)\} = f(x, y) \quad (2-27)$$

and

$$\sigma_{\bar{g}(x, y)}^2 = \frac{1}{K} \sigma_{\eta(x, y)}^2 \quad (2-28)$$



## EXAMPLE 2.5 : Using image addition (averaging) for noise reduction.

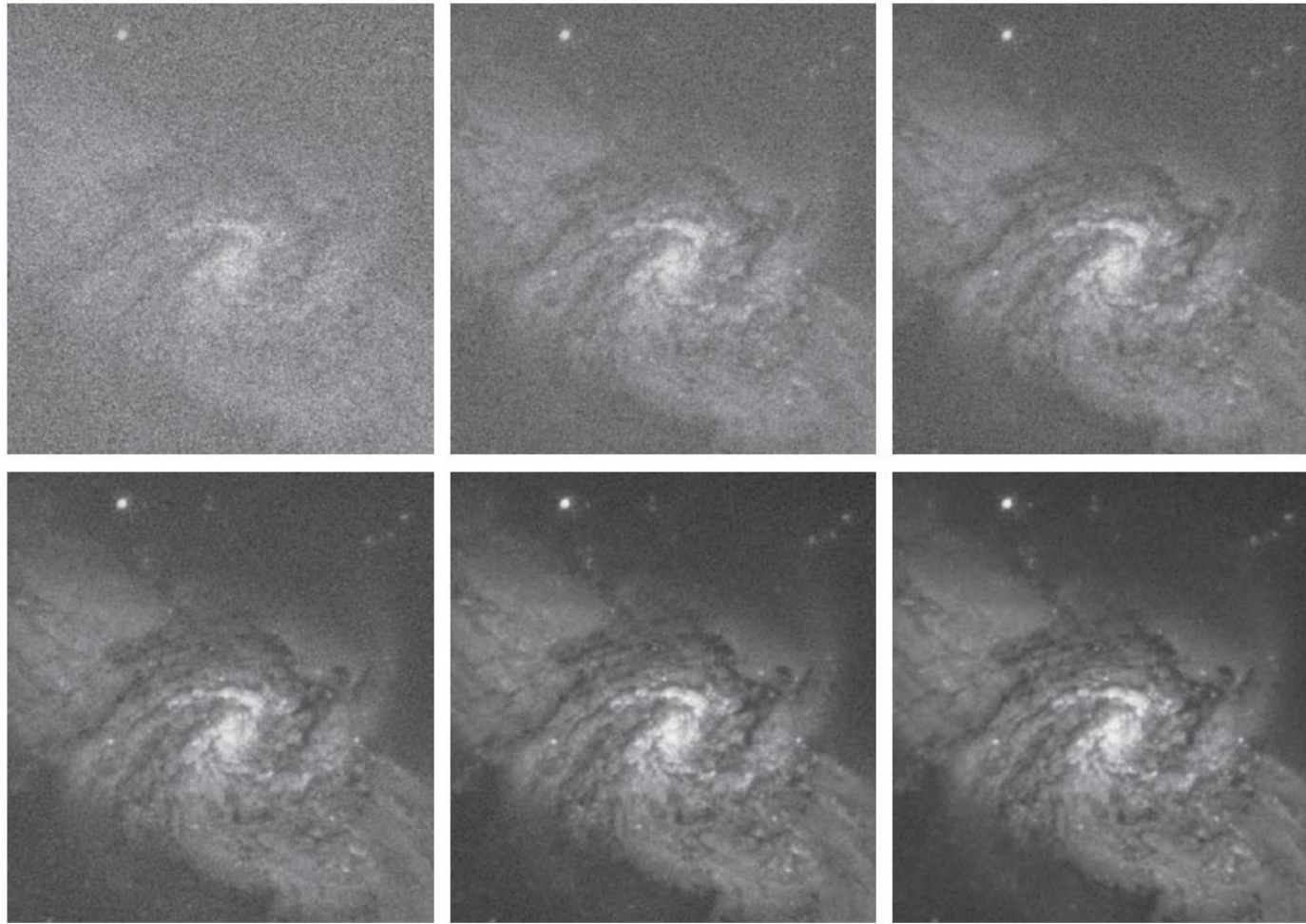
where  $E\{\bar{g}(x,y)\}$  is the expected value of  $\bar{g}(x,y)$ , and  $\sigma_{\bar{g}(x,y)}^2$  and  $\sigma_{\eta(x,y)}^2$  are the variances of  $\bar{g}(x,y)$  and  $\eta(x,y)$ , respectively, all at coordinates  $(x,y)$ . These variances are arrays of the same size as the input image, and there is a scalar variance value for each pixel location.

The standard deviation (square root of the variance) at any point  $(x,y)$  in the average image is

The standard deviation (square root of the variance) at any point  $(x,y)$  in the average image is

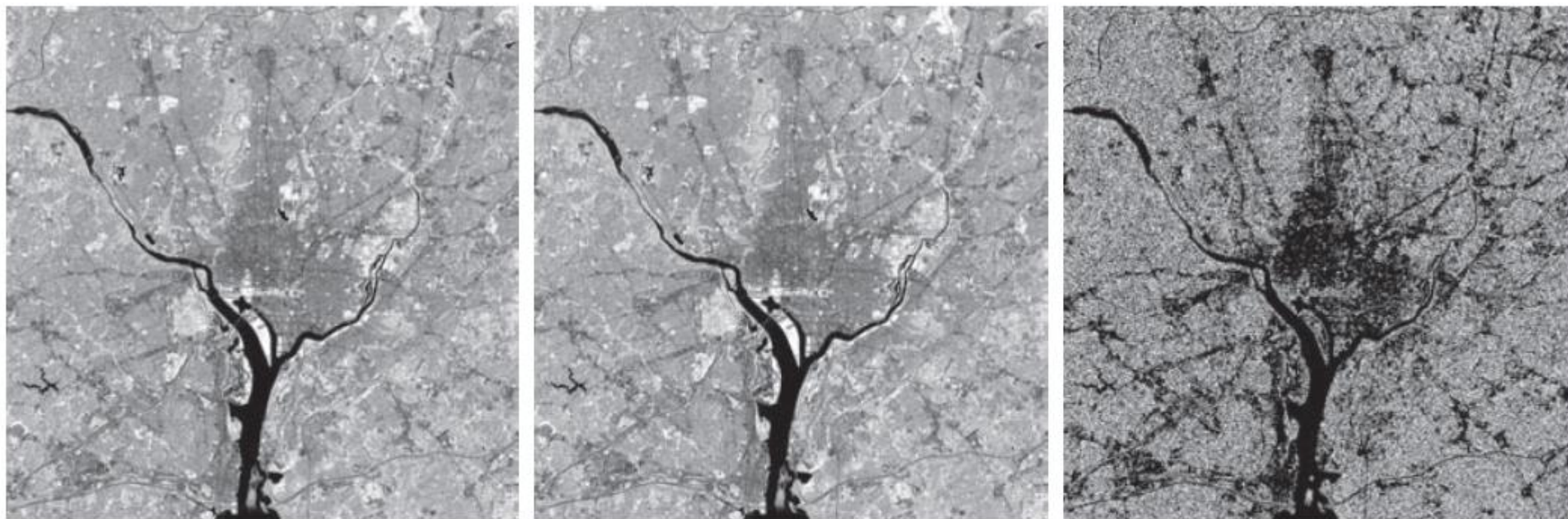
$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}} \sigma_{\eta(x,y)} \quad (2-29)$$

As  $K$  increases, Eqs. (2-28) and (2-29) indicate that the variability (as measured by the variance or the standard deviation) of the pixel values at each location  $(x,y)$  decreases. Because  $E\{\bar{g}(x,y)\} = f(x,y)$ , this means that  $\bar{g}(x,y)$  approaches the noiseless image  $f(x,y)$  as the number of noisy images used in the averaging process increases. In order to avoid blurring and other artifacts in the output (average) image, it is necessary that the images  $g_i(x,y)$  be *registered* (i.e., spatially aligned).



a	b	c
d	e	f

**FIGURE 2.29** (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)-(f) Result of averaging 5, 10, 20, 50, and 1,00 noisy images, respectively. All images are of size  $566 \times 598$  pixels, and all were scaled so that their intensities would span the full  $[0, 255]$  intensity scale. (Original image courtesy of NASA.)



a b c

**FIGURE 2.30** (a) Infrared image of the Washington, D.C. area. (b) Image resulting from 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. (Original image courtesy of NASA.)

---



# Using image addition (averaging) for noise reduction



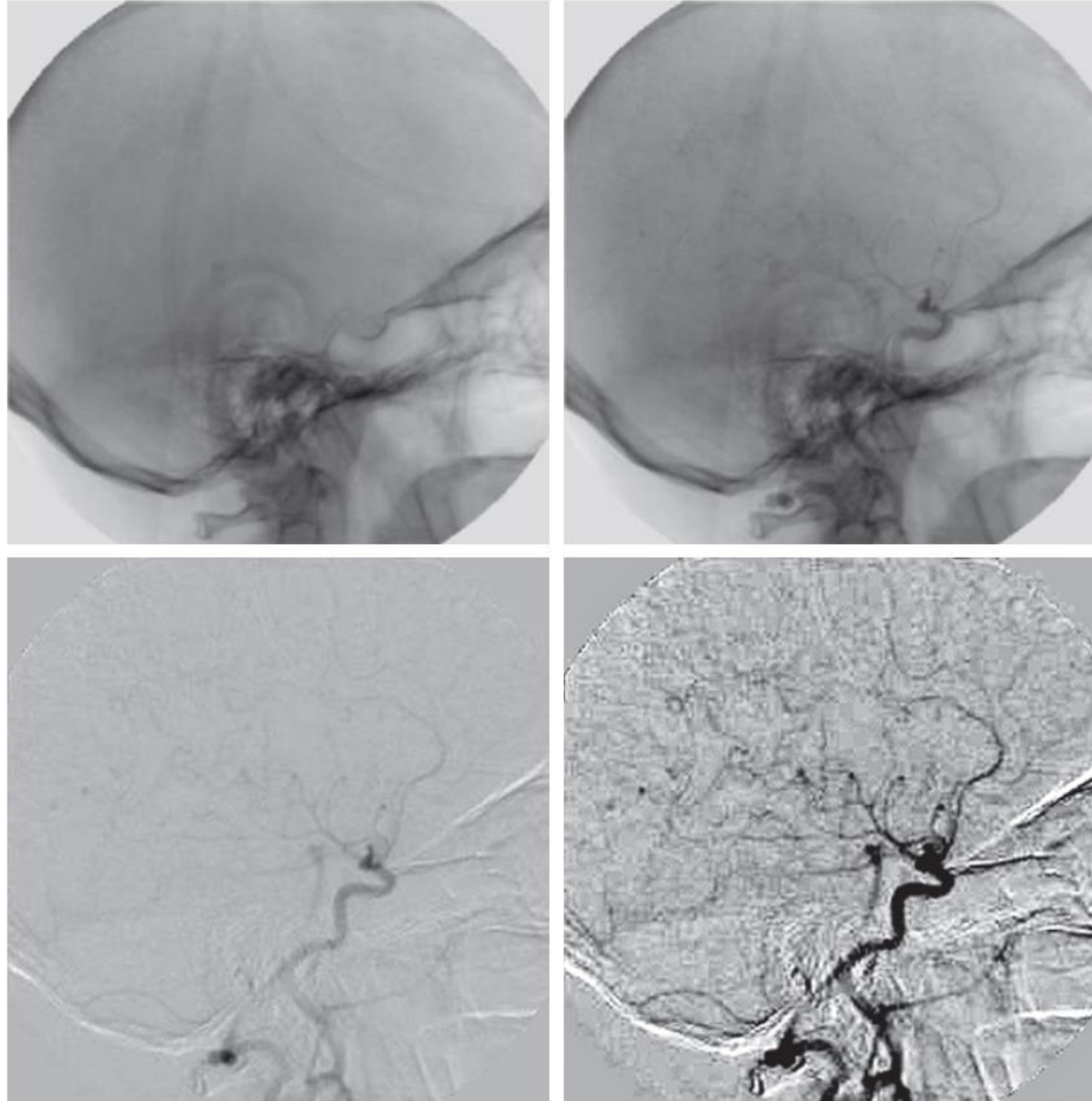
a b c

**FIGURE 2.31** (a) Difference between the 930 dpi and 72 dpi images in Fig. 2.23. (b) Difference between the 930 dpi and 150 dpi images. (c) Difference between the 930 dpi and 300 dpi images.

a	b
c	d

**FIGURE 2.32**

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.)



# Using image multiplication and division for shading correction and for masking

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

An important application of image multiplication (and division) is *shading correction*. Suppose that an imaging sensor produces images that can be modeled as the product of a “perfect image,” denoted by  $f(x, y)$ , times a shading function,  $h(x, y)$ ; that is,  $g(x, y) = f(x, y)h(x, y)$ . If  $h(x, y)$  is known or can be estimated, we can obtain  $f(x, y)$  (or an estimate of it) by multiplying the sensed image by the inverse of  $h(x, y)$  (i.e., dividing  $g$  by  $h$  using elementwise division). If access to the imaging system is possible, we can obtain a good approximation to the shading function by imaging a target of constant intensity. When the sensor is not available, we often can estimate the shading pattern directly from a shaded image using the approaches discussed in Sections 3.5 and 9.8. Figure 2.33 shows an example of shading correction using an estimate of the shading pattern. The corrected image is not perfect because of errors in the shading pattern (this is typical), but the result definitely is an improvement over the shaded image in Fig. 2.33 (a). See Section 3.5 for a discussion of how we estimated Fig. 2.33 (b). Another use of image multiplication is in *masking*, also called *region of interest* (ROI), operations. As Fig. 2.34 shows, the process consists of multiplying a given image by a mask image that has 1’s in the ROI and 0’s elsewhere. There can be more than one ROI in the mask image, and the shape of the ROI can be arbitrary.

# Shading Correction



a b c

**FIGURE 2.33** Shading correction. (a) Shaded test pattern. (b) Estimated shading pattern. (c) Product of (a) by the reciprocal of (b). (See Section 3.5 for a discussion of how (b) was estimated.)

---

# SET AND LOGICAL OPERATIONS

Union and intersection of a set with itself

$$A \cup A = A; A \cap A = A$$

Union and intersection of a set with its complement

$$A \cup A^c = \Omega; A \cap A^c = \emptyset$$

*universal set*

Commutative laws

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

Associative laws

$$(A \cup B) \cup C = A \cup (B \cup C)$$

$$(A \cap B) \cap C = A \cap (B \cap C)$$

Distributive laws

$$(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$$

$$(A \cap B) \cup C = (A \cup C) \cap (B \cup C)$$

DeMorgan's laws

$$(A \cup B)^c = A^c \cap B^c$$

$$(A \cap B)^c = A^c \cup B^c$$



# De Morgan Law

A universal set  $U$  which consists of all the natural numbers which are multiples of 3, less than or equal to 20. Let  $A$  be a subset of  $U$  which consists of all the even numbers and the set  $B$  is also a subset of  $U$  consisting of all the prime numbers. Verify De Morgan Law.

- **Solution:** We have to verify  $(A \cup B)' = A' \cap B'$  and  $(A \cap B)' = A' \cup B'$ . Given that,
- $U = \{3, 6, 9, 12, 15, 18\}$
- $A = \{6, 12, 18\}$
- $B = \{3\}$
- The union of both  $A$  and  $B$  can be given as,
- $A \cup B = \{3, 6, 12, 18\}$
- The complement of this union is given by,
- $(A \cup B)' = \{9, 15\}$
- Also, the intersection and its complement are given by:
- $A \cap B = \emptyset$

$$(A \cap B)' = \{3, 6, 9, 12, 15, 18\}$$

Now, the complement of the sets  $A$  and  $B$  can be given as:

$$A' = \{3, 9, 15\}$$

$$B' = \{6, 9, 12, 15, 18\}$$

Taking the union of both these sets, we get,

$$A' \cup B' = \{3, 6, 9, 12, 15, 18\}$$

And the intersection of the complemented sets is given as,

$$A' \cap B' = \{9, 15\}$$

We can see that:

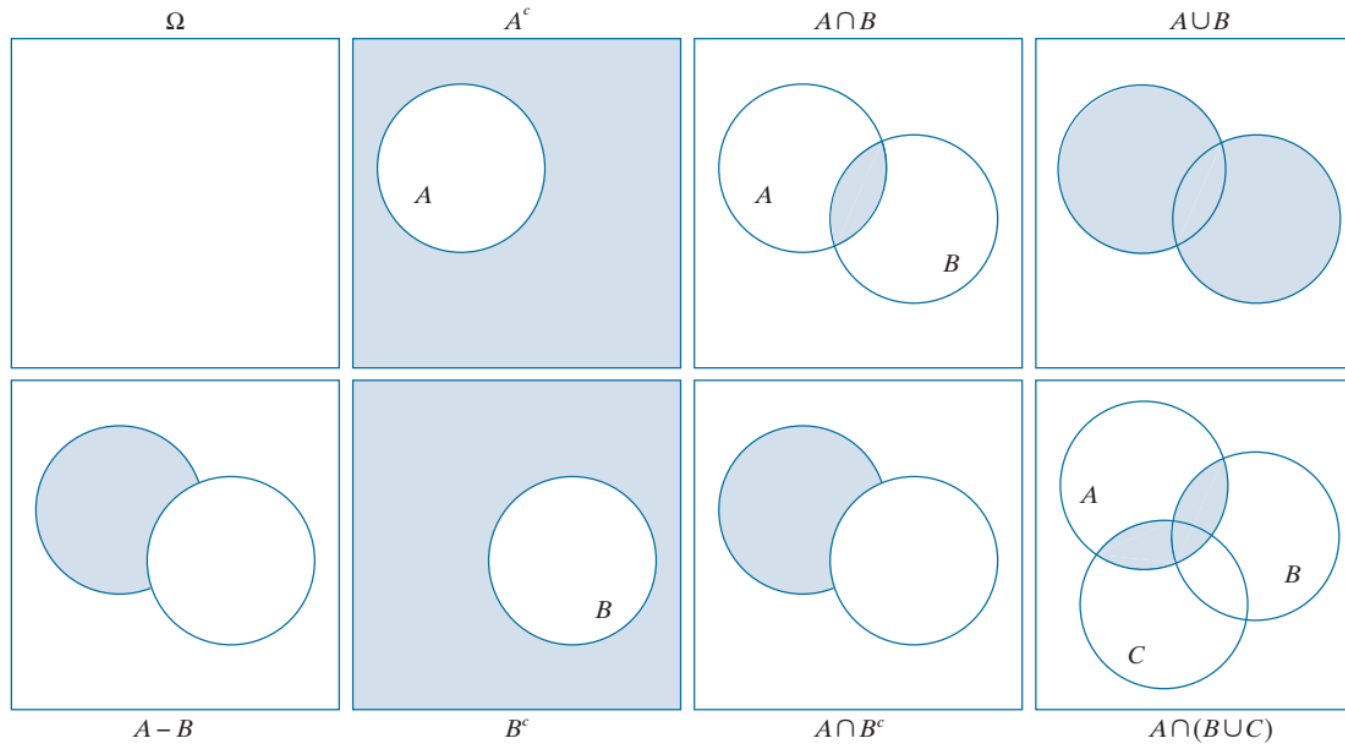
$$(A \cup B)' = A' \cap B' = \{9, 15\}$$

And also,

$$(A \cap B)' = A' \cup B' = \{3, 6, 9, 12, 15, 18\}$$

Hence, the above result is true in general and is known as De Morgan Law.

# SET AND LOGICAL OPERATIONS



a	b	c	d
e	f	g	h

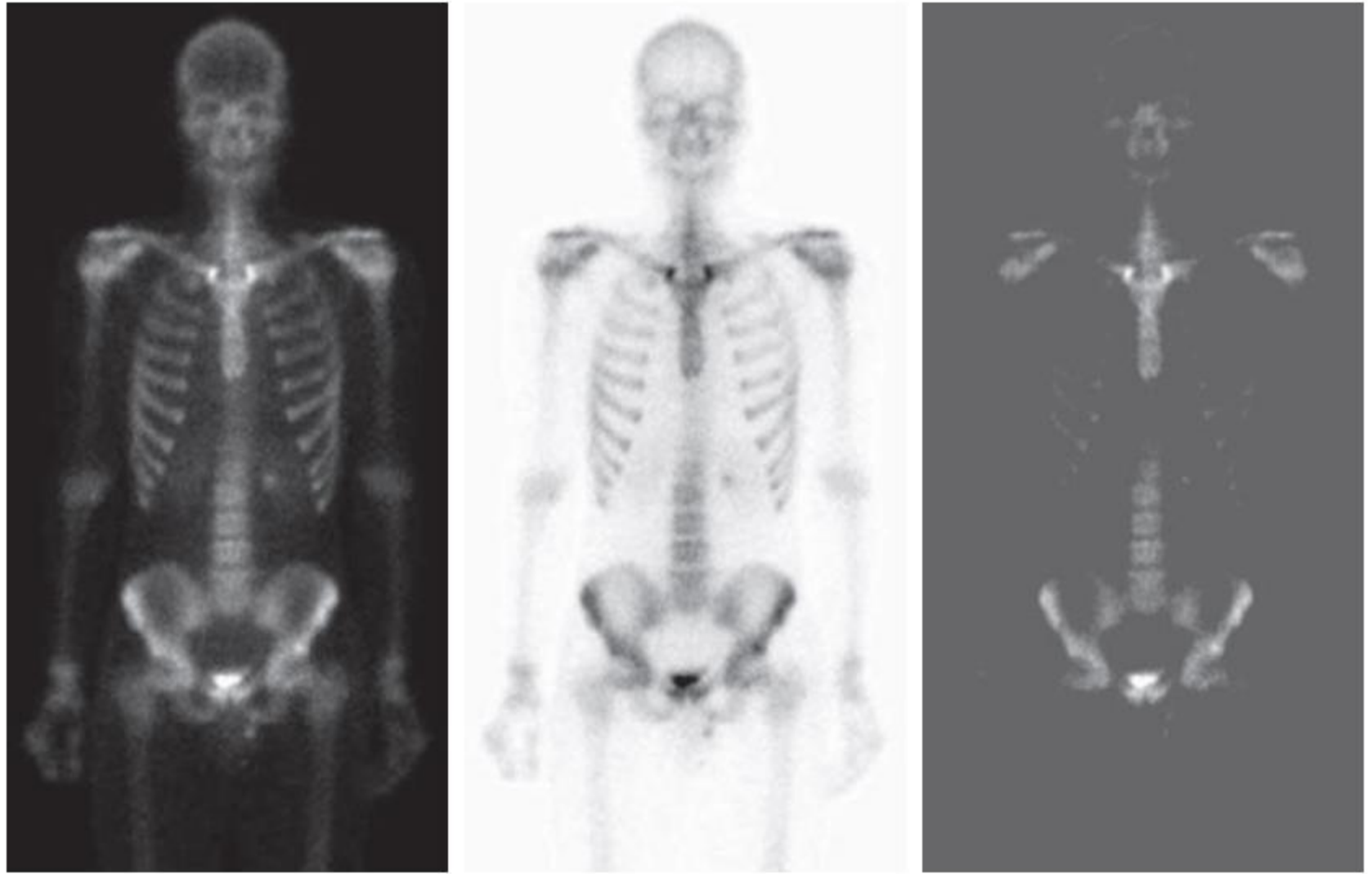
**FIGURE 2.35** Venn diagrams corresponding to some of the set operations in Table 2.1. The results of the operations, such as  $A^c$ , are shown shaded. Figures (e) and (g) are the same, proving via Venn diagrams that  $A - B = A \cap B^c$  [see Eq. (2-40)].

a b c

**FIGURE 2.36**

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

---



# SET AND LOGICAL OPERATIONS

**TABLE 2.2**

Truth table  
defining the  
logical operators  
AND( $\wedge$ ),  
OR( $\vee$ ), and  
NOT( $\sim$ ).

$a$	$b$	$a \text{ AND } b$	$a \text{ OR } b$	NOT( $a$ )
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

**FIGURE 2.37**

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

