Image Interpolation

Interpolation is a basic tool used extensively in tasks such as zooming, shrinking, rotating, and geometric corrections.

Fundamentally, interpolation is the process of using known data to estimate values at unknown locations.

For example, we want to resize an image of size 500×500 pixels to 750×750 pixels. To perform intensity-level assignment for any point in the overlay, we look for its closest pixel in the original image and assign its intensity to the new pixel in the 750×750 grid. This method is called nearest neighbour interpolation.

A more suitable approach is bilinear interpolation, in which we use the four nearest neighbours to estimate the intensity at a given location.

Let (x, y) denote the coordinates of the location to which we want to assign an intensity value, and let $v(x, y)$ denote that intensity value. For bilinear interpolation, the assigned value is obtained using the equation

$$v(x, y) = ax + by + cxy + d \,, \tag{2.4-6}$$

where the four coefficients are determined from the four equations in four unknowns that can be written using the four nearest neighbours of point (x, y).

The next level of complexity is bicubic interpolation, which involves the sixteen nearest neighbours of a point:

$$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \,, \tag{2.4-7}$$

where the sixteen coefficients are determined from the sixteen equations in sixteen unknowns that can be written using the sixteen nearest neighbours of point (x, y).

Example 2.4: Comparison of interpolation approaches for image shrinking and zooming.

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).

It is possible to use more neighbours in interpolation, and there are more complex techniques.

2.5 Some Basic Relationships between Pixels

Here, we consider some important relationships between pixels in a digital image.

Neighbours of a Pixel

A pixel p at coordinates (x, y) has four horizontal and vertical neighbours:

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

This set of pixels is called the 4-neighbuors of p, and denoted by $N_4(p)$.

The four diagonal neighbours of p are

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

and are denoted by $N_D(p)$.

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 gray-scale image, set V typically contains more elements. For example, with a range of possible intensity values 0 to 255, set V could be any subset of these 256 values.

Consider three types of adjacency:

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

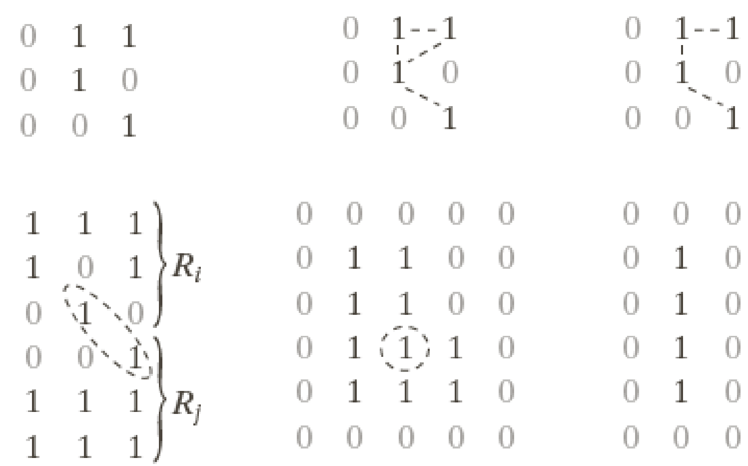
(b) 8-adjacency. Two pixels p and q with values from V are 8-adjacency if q is in the set $N_8(p)$.

(c) m-adjacency (mixed adjacency). Two pixels p and q with values from V are m-adjacency if

(i) q is in the $N_4(p)$, or

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

Mixed adjacency is a modified of 8-adjacency.



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 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.

For example, consider the arrangement shown in Figure 2.25 (a) for $V = \{1\}$.

The three pixels at the top of Figure 2.25 (b) show ambiguous 8-adjacency, which is removed by using m-adjacency, as shown in Figure 2.25 (c).

A path from pixel p with coordinates (x, y) to pixel q with coordinates (s, t) is a sequence of distinct pixels with coordinates

$$(x_0, y_0),\ (x_1, y_1),\ \cdots,\ (x_n, y_n),$$

where $(x_0, y_0) = (x, y)$, $(x_n, y_n) = (s, t)$, 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 a closed path.

We can define 4-, 8-, or *m*-paths depending on the type of adjacency. The path shown in Figure 2.25 (b) between the top right and bottom right points are 8-paths, and the path in Figure 2.25 (c) is an *m*-path.

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. If it only has one connected component, set S is called a connected set.

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 and R_j are said to be adjacent if their union forms a connected set. Regions that are not adjacent are said to be disjoint.

The two regions (of 1s) in Figure 2.25 (d) are adjacent only if 8-adjacency is used.

Suppose that an image contains K disjoint regions, R_k, $k = 1, 2, \ldots, K$, and none of which touches the image border. Let R_u denote the union of all the K regions, and let $(R_u)^c$ denote its complement. We call all the points in R_u the foreground, and all the points in $(R_u)^c$ the background of the image.

The boundary (also called the border or contour) of a region R is the set of points that are adjacent to points in the complement of R.

Again, we must specify the connectivity being used to define adjacency. For example, the point circled in Figure 2.25 (e) is not a member of the border of the 1-valued region if 4-connectivity is used between the region and its background.

As a rule, adjacency between points in a region and its background is defined in terms of 8-adjacency to handle situations like above.

The preceding definition is referred to as the inner border of the region to distinguish it from its outer border, which is the corresponding border in the background.

This issue is important in the development of border-following algorithms. Such algorithms usually are formulated to follow the outer boundary in order to guarantee that the result will form a closed path.

For example, the inner border of the 1-valued region in Figure 2.25 (f) is the region itself.

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 of the image. This extra definition is required because an image has no neighbours beyond its border.

Distance Measures

For pixels p, q, and z, with coordinates (x, y), (s, t), and (v, w), 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) $D(p,z) \leq D(p,q) + D(q,z)$.

The Euclidean distance between p and q is defined as

$$D_e(p,q) = \left[(x-s)^2 + (y-t)^2\right]^{1/2}. \qquad (2.5\text{-}1)$$

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

$$D_4(p,q) = |x-s| + |y-t|. \qquad (2.5\text{-}2)$$

Example: the pixels with D_4 distance ≤ 2 from (x, y) form the following contours of constant distance:

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

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

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

$$D_8(p,q) = \max(|x-s|,|y-t|). \qquad (2.5\text{-}3)$$

Example: the pixels with D_8 distance ≤ 2 from (x,y) form the following contours of constant distance:

$$\begin{matrix} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{matrix}$$

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

2.6 An Introduction to the Mathematical Tools Used in Digital Image Processing

Two principal objectives for this section: (1) to introduce the various mathematical tools we will use in the following chapters; (2) to develop a "feel" for how these tools are used by applying them to a variety of basic image processing tasks.

Array versus Matrix Operations

An array operation involving one or more images is carried out on a pixel-by-pixel basis.

There are many situations in which operations between images are carried out using matrix theory.

Consider the following 2×2 images:

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

The array product of these two images is

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

while the matrix product is given by

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

We assume array operations throughout this course, unless stated otherwise.

Linear versus Nonlinear Operations

One of the most important classifications of an image processing method is whether it is linear or nonlinear.

Consider a general operator, H, that produces an output image, $g(x,y)$, for a given input image, $f(x,y)$:

$$H[f(x,y)] = g(x,y). \qquad (2.6\text{-}1)$$

H is said to be a linear operator if

$$\begin{aligned} H\left[a_i f_i(x,y) + a_j f_j(x,y)\right] &= a_i H[f_i(x,y)] + a_j H[f_j(x,y)] \\ &= a_i g_i(x,y) + a_j g_j(x,y), \end{aligned} \qquad (2.6\text{-}2)$$

where a_i, a_j, $f_i(x,y)$, and $f_j(x,y)$ are arbitrary constants and images.

As indicated in (2.6-2), the output of a linear operation due to the sum of two inputs is the same as performing the operation on the input individually and then summing the results.

Example: a nonlinear operation.

Consider the following two images for the max operation:

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

and we let $a_1 = 1$ and $a_2 = -1$. To test for linearity, we start with the left side of (2.6-2):

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

Then, we work with the right side:

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

The left and right sides of (2.6-2) are not equal in this case, so we have proved that in general the max operator is nonlinear.

Arithmetic Operations

The four arithmetic operations, which are array operations, are denoted as

$$\begin{aligned} s(x,y) &= f(x,y) + g(x,y) \\ d(x,y) &= f(x,y) - g(x,y) \\ p(x,y) &= f(x,y) \times g(x,y) \\ v(x,y) &= f(x,y) \div g(x,y) \end{aligned} \qquad (2.6\text{-}3)$$

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 $f(x,y)$:

$$g(x,y) = f(x,y) + \eta(x,y) \qquad (2.6\text{-}4)$$

where $\eta(x,y)$ is assumed to be uncorrelated with zero average value.

The objective of the following procedure is to reduce the noise content by adding a set of noisy images, $\{g_i(x,y)\}$. With the above assumption, it can be shown that if an image $\overline{g}(x,y)$ is formed by averaging K different noisy images,

$$\overline{g}(x,y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x,y)\,, \qquad (2.6\text{-}5)$$

then it follows that

$$E\{\overline{g}(x,y)\} = f(x,y)\,, \qquad (2.6\text{-}6)$$

and

$$\sigma^2_{\overline{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)}\,, \qquad (2.6\text{-}7)$$

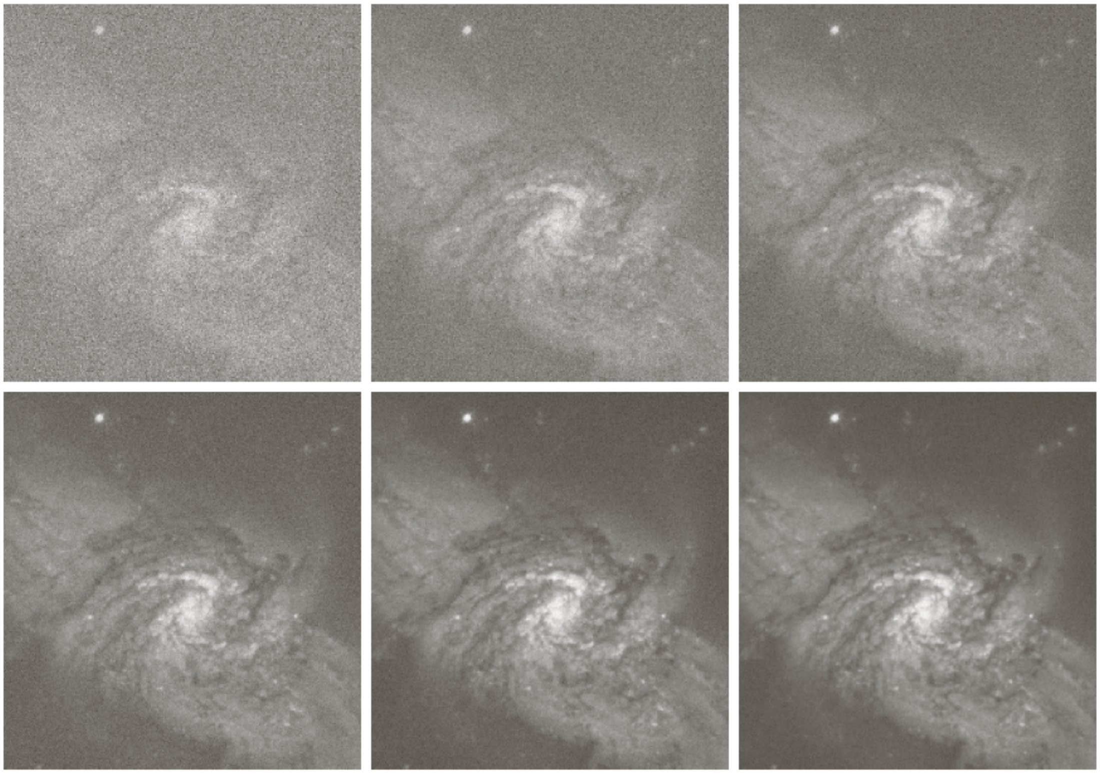
where $E\{\overline{g}(x,y)\}$ is the expected value of $\overline{g}$, and $\sigma^2_{\overline{g}(x,y)}$ and $\sigma^2_{\eta(x,y)}$ are the variances of $\overline{g}$ and η, all at coordinate (x,y).

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

$$\sigma_{\overline{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)}\,. \qquad (2.6\text{-}8)$$

As K increases, (2.6-7) and (2.6-8) indicate that the variability of the pixel values at each location (x,y) decreases. This means that $\overline{g}(x,y)$ approaches $f(x,y)$ as the number of noisy images used in the averaging process increases.

Figure 2.26 shows results of averaging different number of noisy images to image of Galaxy Pair NGC 3314.

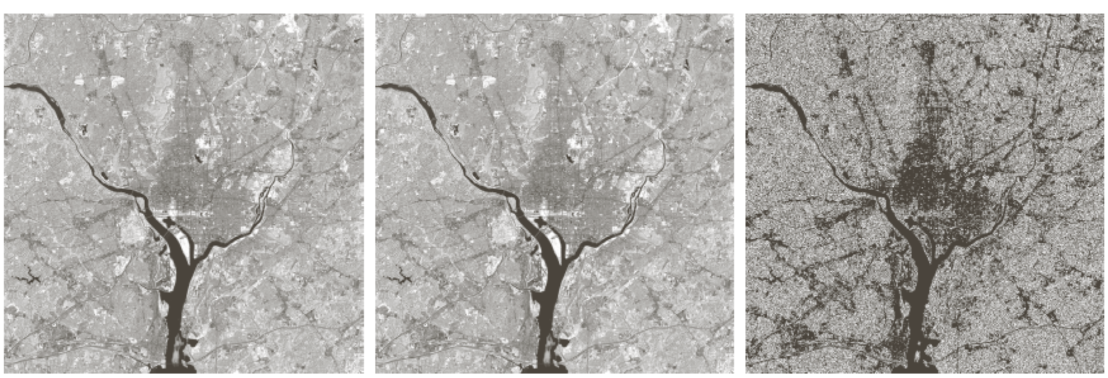


a b c
d e f

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.

A frequent application of image subtracting is in the enhancement of differences between images.



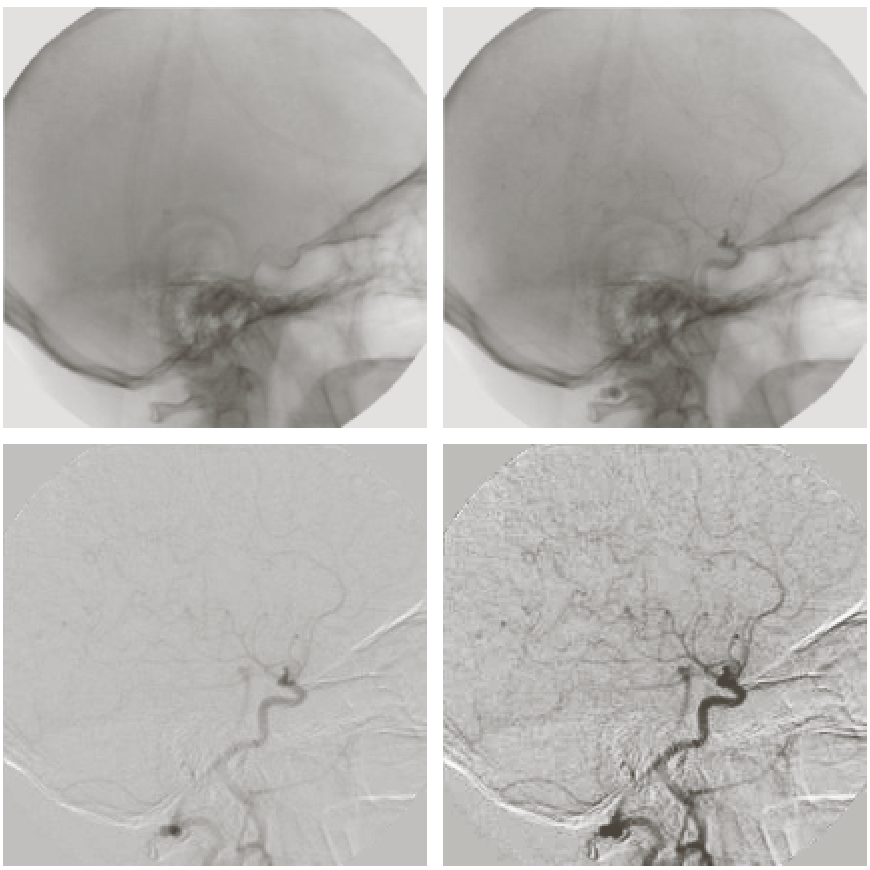
a b c

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.

As another illustration, we discuss an area of medical imaging called mask mode radiography. Consider image differences of the form

$$g(x,y) = f(x,y) - h(x,y)\,. \qquad (2.6\text{-}9)$$

where $h(x,y)$, the mask, is an X-ray image of a region of a patient's body captured by an intensified TV camera located opposite an X-ray source.



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.

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", $f(x,y)$, times a shading function, $h(x,y)$:

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

If $h(x,y)$ is known, we can obtain $f(x,y)$ by

$$f(x,y) = g(x,y)\,/\,h(x,y).$$

If $h(x,y)$ is not known, but access to the imaging system is possible, we can obtain an approximation to the shading function by imaging a target of constant intensity.

Figure 2.29 shows an example of shading correction.

a b c

FIGURE 2.29 Shading correction. (a) Shaded SEM image of a tungsten filament and support, magnified approximately 130 times. (b) The shading pattern. (c) Product of (a) by the reciprocal of (b). (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

Set and Logical Operations

Basic set operations

Let A be a set composed of ordered pairs of real numbers. If $a = (a_1, a_2)$ is an element of A, then we write

$$a \in A \tag{2.6-12}$$

Similarly, if a is not an element of A, we write

$$a \notin A \tag{2.6-13}$$

The set with no elements is called the null or empty set and is denoted by the symbol $\varnothing$.

A set is specified by the contents of two braces: $\{\bullet\}$. For example, an expression of the form

$$C = \{w \mid w = -d, d \in D\}$$

means that set C is the set of elements, w, such that w is formed by multiplying each of the elements of set D by -1.

If every element of a set A is also an element of a set B, then A is said to be a subset of B, denoted as

$$A \subseteq B \tag{2.6-14}$$

The union of two sets A and B, denoted by

$$C = A \cup B \tag{2.6-15}$$

is the set of elements belonging to either A, B, or both.

Similarly, the intersection of two sets A and B, denoted by

$$D = A \cap B \tag{2.6-16}$$

is the set of elements belonging to both A and B.

Two sets A and B are said to be disjoint or mutually exclusive if they have no common elements

$$A \cap B = \varnothing \tag{2.6-17}$$

The set universe, U, is the set of all elements in a given application.

The complement of a set A is the set of elements that are not in A:

$$A^c = \{w \mid w \notin A\} \tag{2.6-18}$$

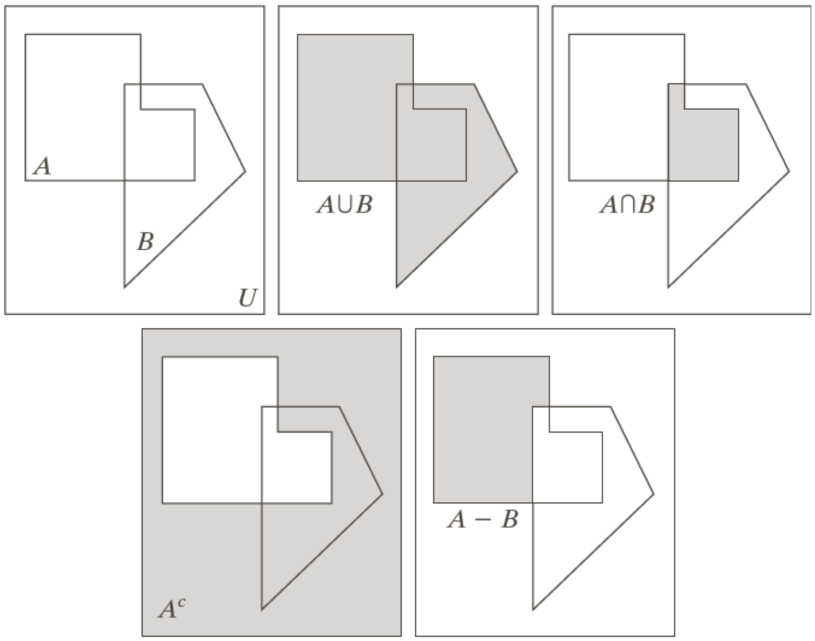
The difference of two sets A and B, $A - B$, is defined as

$$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c \tag{2.6-19}$$

As an example, we would define A^c in terms of U and the set difference operation:

$$A^c = U - A$$

Figure 2.31 illustrates the preceding concepts.



a b c
d e

FIGURE 2.31 (a) Two sets of coordinates, A and B, in 2-D space. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B. In (b)–(e) the shaded areas represent the member of the set operation indicated.

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 special coordinates and z denoted intensity.

We can define the complement of A as the set

$$A^c = \{(x, y, K - z) \mid (x, y, z) \in A\},$$

which denotes the set of pixels of A whose intensities have been subtracted from a constant K. The constant K 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 Figure 2.32 (a).

To form the negative of A using set operations, we can form

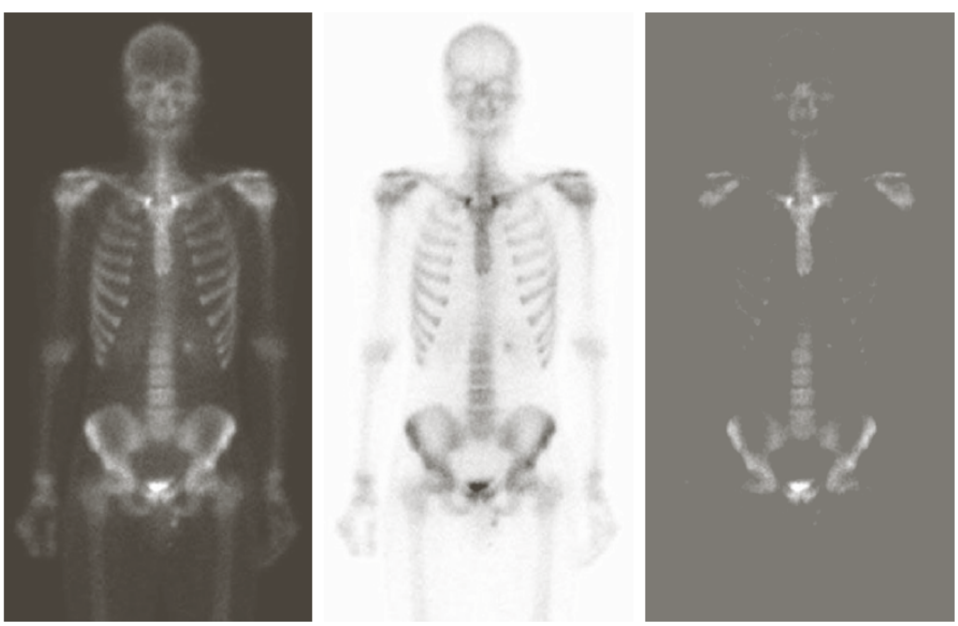
$$A_n = A^c = \{(x, y, 255 - z) \mid (x, y, z) \in A\}$$

This image is shown in Figure 2.32 (b).

The union of two gray-scale set A and B may be defined as the set

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

For example, assume A represents the image in Figure 2.32 (a), and let B denote an array of the same size as A, but in which all values of z are equal to 3 times the mean intensity, m, of the elements of A. Figure 2.32 (c) shows the result.



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

When dealing with binary images, it is common practice to refer to union, intersection, and complement as the OR, AND, and NOT logical operations.

Figure 2.33 illustrates some logical operations.

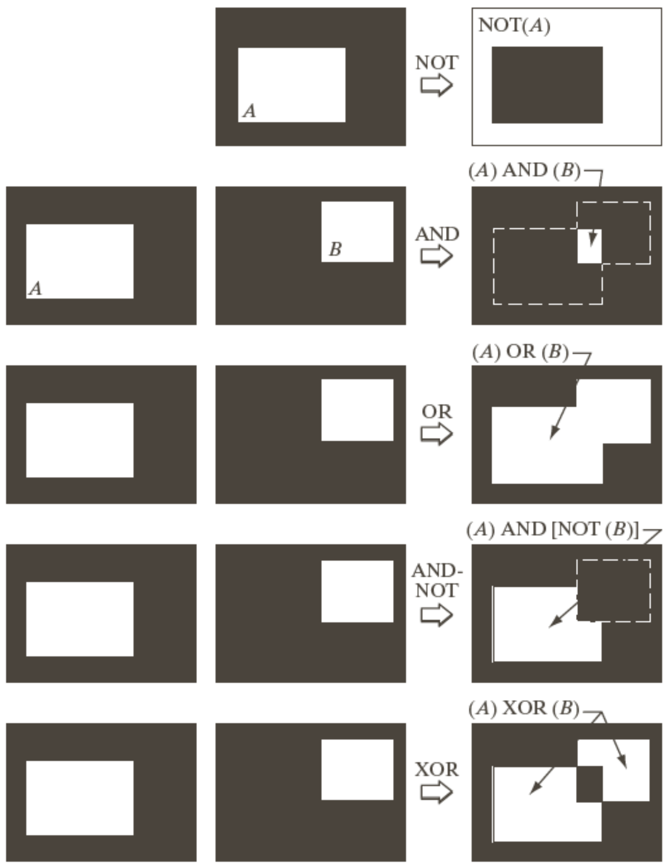


FIGURE 2.33 Illustration of logical operations involving foreground (white) pixels. Black represents binary 0s and white binary 1s. The dashed lines are shown for reference only. They are not part of the result.

The fourth row of Figure 2.33 shows the result of operation that the set of foreground pixels belonging to A but not to B, which is the definition of set difference in

$$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c \qquad (2.6\text{-}19)$$

The last row shows the XOR (exclusive OR) operation, which is the set of foreground pixels belonging to A or B, but not both.

The three operators, OR, AND, and NOT, are functionally complete.