Introduction to Mathematical Operations in DIP

- **Array vs. Matrix Operation**

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

Array product operator

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

Array product

Matrix product operator

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

Matrix product

Introduction to Mathematical Operations in DIP

- **Linear vs. Nonlinear Operation**

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

$$H\left[a_i f_i(x,y) + a_j f_j(x,y)\right]$$

$$= H\left[a_i f_i(x,y)\right] + H\left[a_j f_j(x,y)\right]$$

Additivity

$$= a_i H\left[f_i(x,y)\right] + a_j H\left[f_j(x,y)\right]$$

Homogeneity

$$= a_i g_i(x,y) + a_j g_j(x,y)$$

H is said to be a **linear operator;**

H is said to be a **nonlinear operator** if it does not meet the above qualification.

Arithmetic Operations

- Arithmetic operations between images are array operations. The four arithmetic operations are denoted as

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

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

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

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

Example: Addition of Noisy Images for Noise Reduction

Noiseless image: f(x,y)

Noise: n(x,y) (at every pair of coordinates (x,y), the noise is uncorrelated and has zero average value)

Corrupted image: g(x,y)

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

Reducing the noise by adding a set of noisy images, $\{g_i(x,y)\}$

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

Example: Addition of Noisy Images for Noise Reduction

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

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

$$= E\left\{\frac{1}{K}\sum_{i=1}^{K}[f(x,y)+n_i(x,y)]\right\}$$

$$= f(x,y) + E\left\{\frac{1}{K}\sum_{i=1}^{K} n_i(x,y)\right\}$$

$$= f(x,y)$$

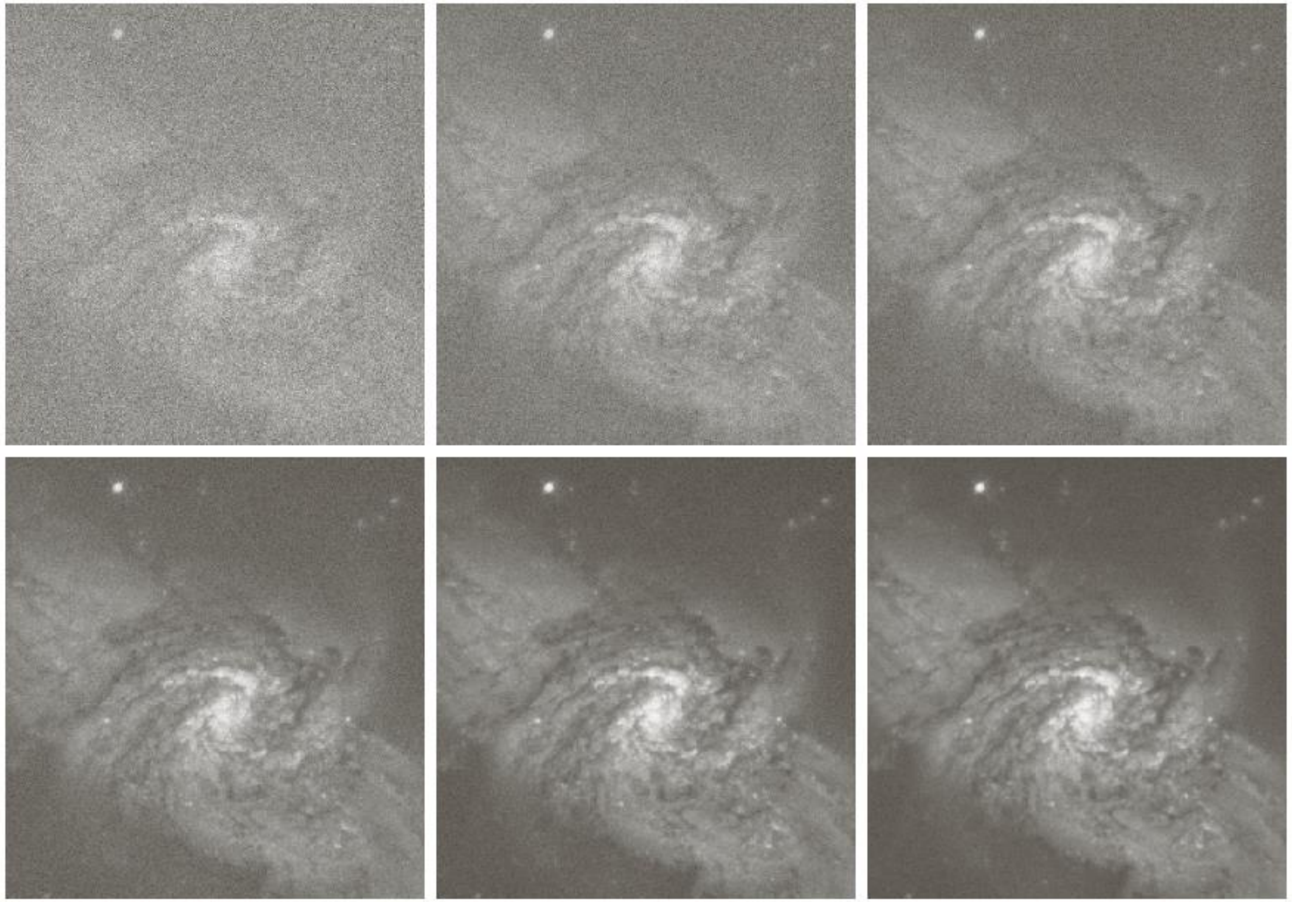
$$\sigma^2_{\bar{g}(x,y)} = \sigma^2_{\frac{1}{K}\sum_{i=1}^{K} g_i(x,y)}$$

$$= \sigma^2_{\frac{1}{K}\sum_{i=1}^{K} n_i(x,y)} = \frac{1}{K}\sigma^2_{n(x,y)}$$

Example: Addition of Noisy Images for Noise Reduction

- In astronomy, imaging under very low light levels frequently causes sensor noise to render single images virtually useless for analysis.

- In astronomical observations, similar sensors for noise reduction by observing the same scene over long periods of time. Image averaging is then used to reduce the noise.



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.)

An Example of Image Subtraction: Mask Mode Radiography

Mask h(x,y): an X-ray image of a region of a patient's body

Live images f(x,y): X-ray images captured at TV rates after injection of the contrast medium

Enhanced detail g(x,y)

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

The procedure gives a movie showing how the contrast medium propagates through the various arteries in the area being observed.

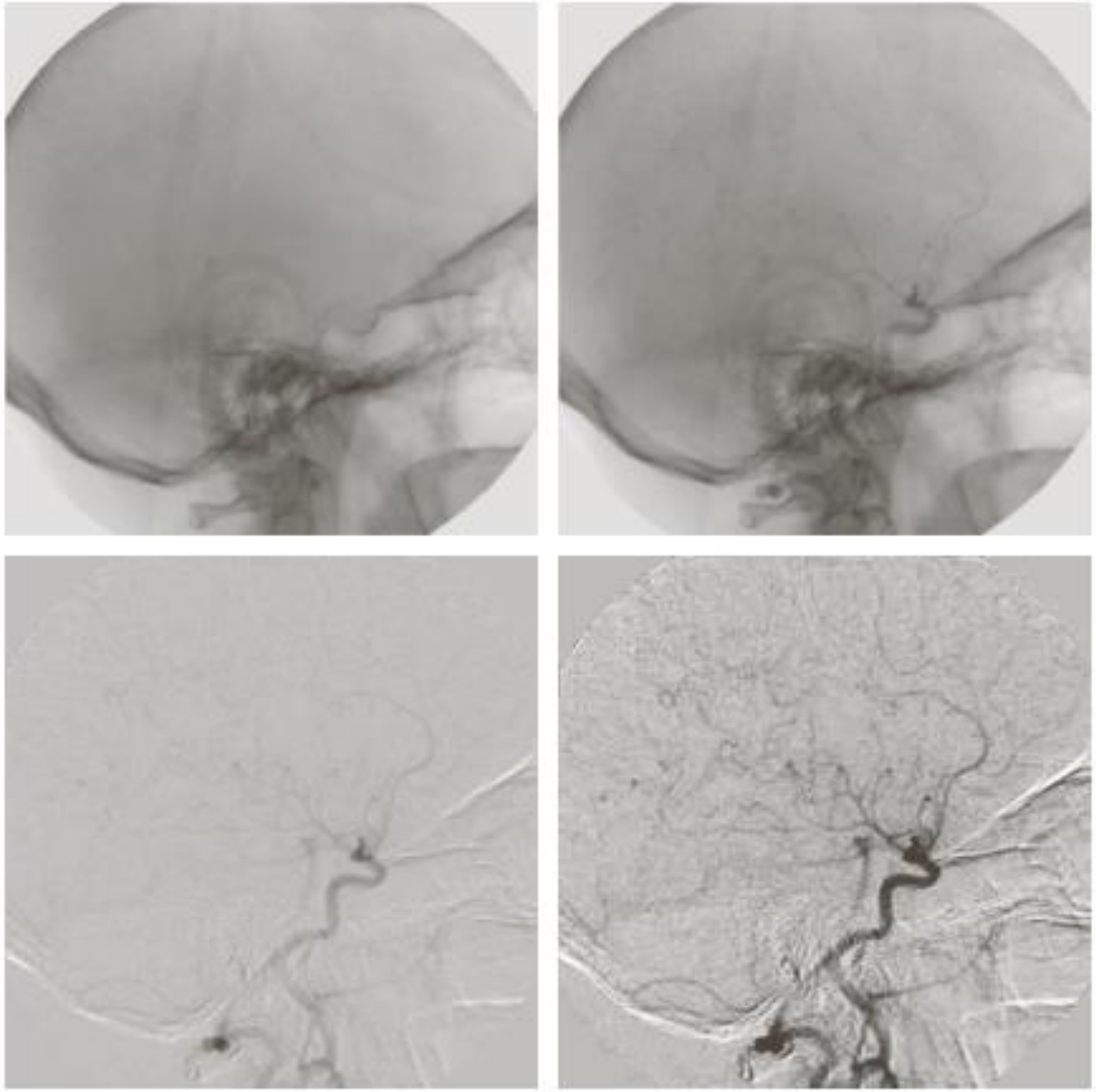


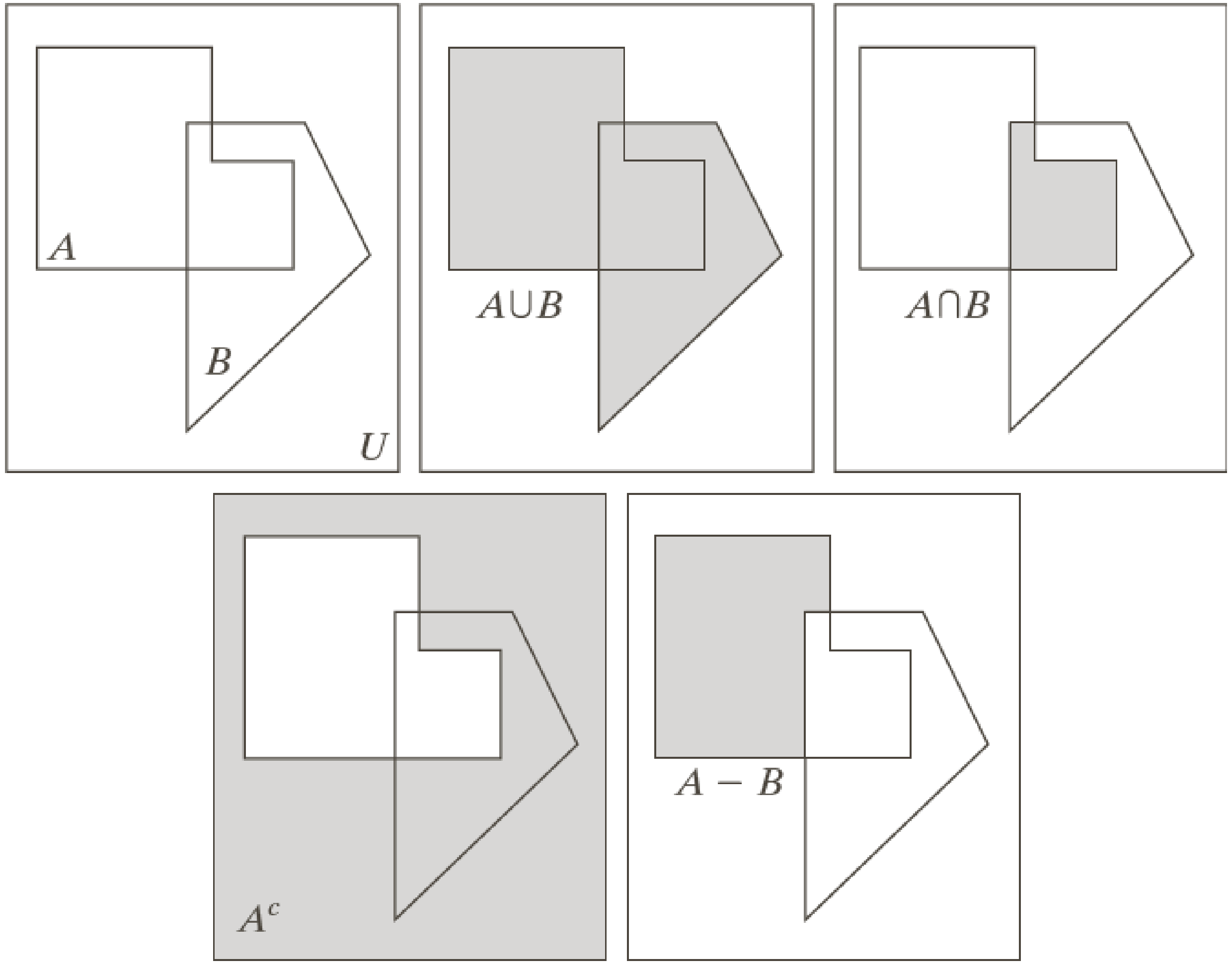
FIGURE 2.28 Digital subtraction angiography. (a) Mask image. (b) A live image. (c) Difference between (a) and (b). (d) Enhanced difference image. (Figures (a) and (b) courtesy of The Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)

An Example of Image Multiplication

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



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.

Set and Logical Operations

- Let A be the elements of a gray-scale image

  The elements of A are triplets of the form (x, y, z), where x and y are spatial coordinates and z denotes the intensity at the point (x, y).

$$A = \{(x, y, z) \mid z = f(x, y)\}$$

- The complement of A is denoted A^c

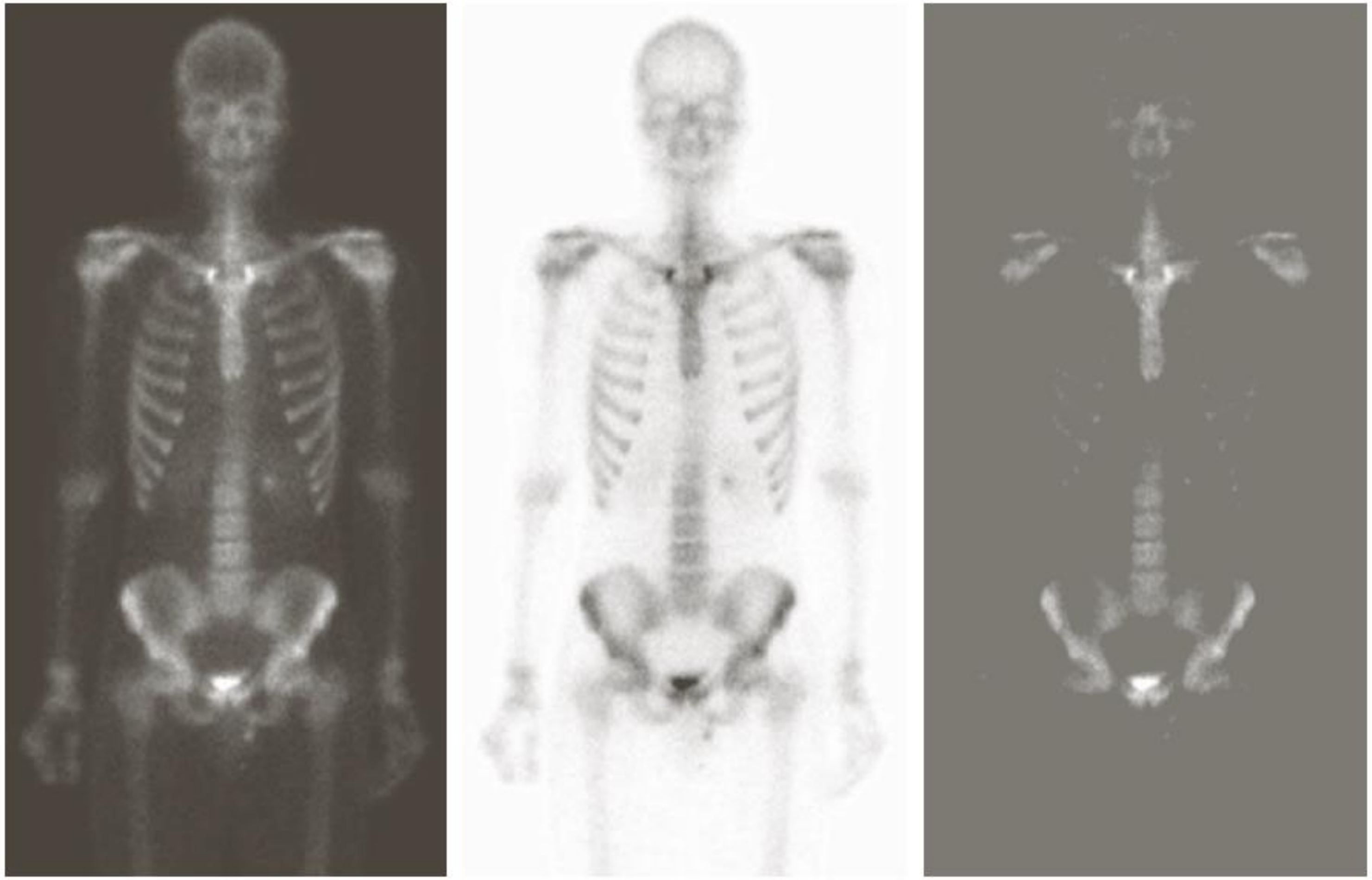
$$A^c = \{(x, y, K - z) \mid (x, y, z) \in A\}$$

$$K = 2^k - 1; k \text{ is the number of intensity bits used to represent } z$$

Set and Logical Operations

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

Set and Logical Operations



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.)

Set and Logical Operations

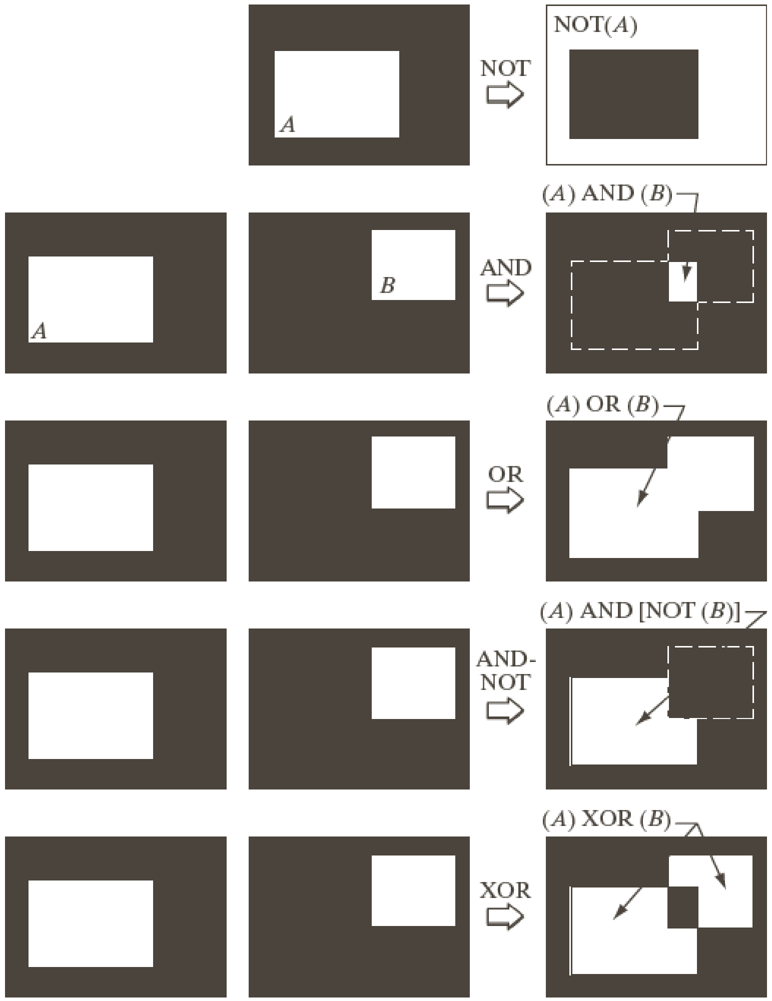


FIGURE 2.33 Illustration of logical operations involving foreground (white) pixels. Black represents binary 0s and white binary 1s. The dashed lines are shown for reference only. They are not part of the result.

Spatial Operations

- Single-pixel operations

Alter the values of an image's pixels based on the intensity.

$$s = T(z)$$

e.g.,

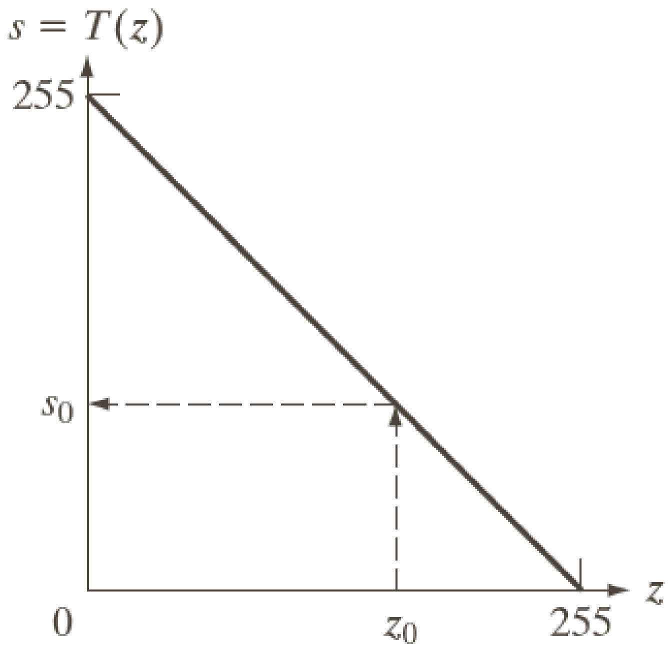
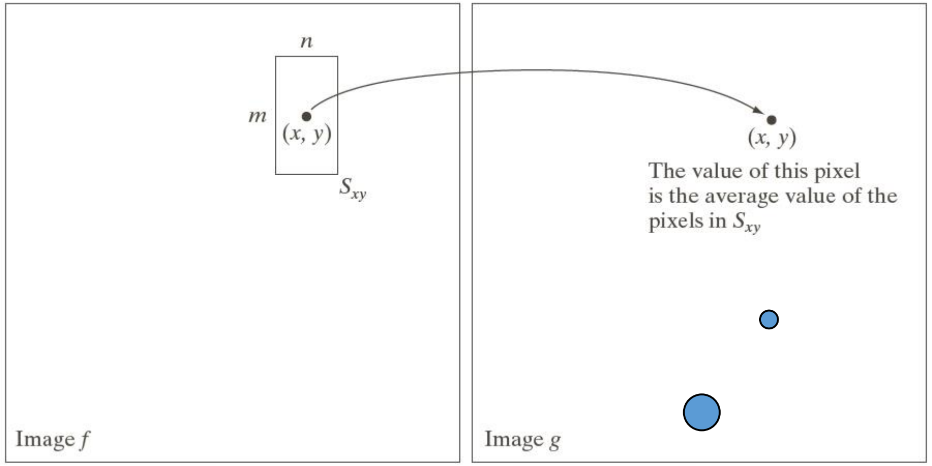


FIGURE 2.34 Intensity transformation function used to obtain the negative of an 8-bit image. The dashed arrows show transformation of an arbitrary input intensity value z_0 into its corresponding output value s_0.

Spatial Operations

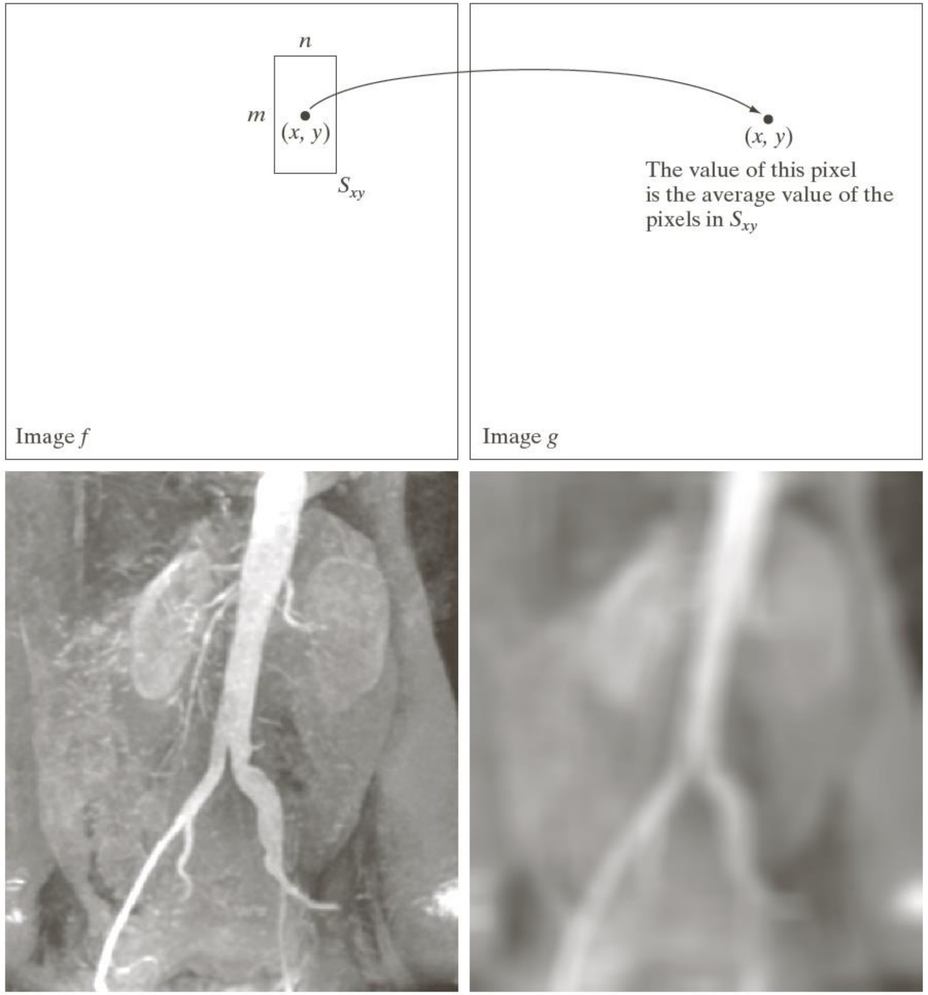
- Neighborhood operations



The value of this pixel is determined by a specified operation involving the pixels in the input image with coordinates in S_{xy}

Spatial Operations

- Neighborhood operations



Geometric Spatial Transformations

- Geometric transformation (rubber-sheet transformation)

  — A spatial transformation of coordinates

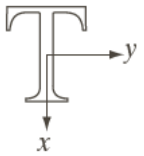
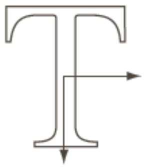
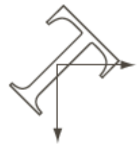
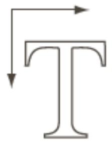
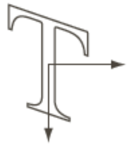
$$(x, y) = T\{(v, w)\}$$

  — intensity interpolation that assigns intensity values to the spatially transformed pixels.

- Affine transform

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

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	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\cos\theta + w\sin\theta$	
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)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

Intensity Assignment

- Forward Mapping

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

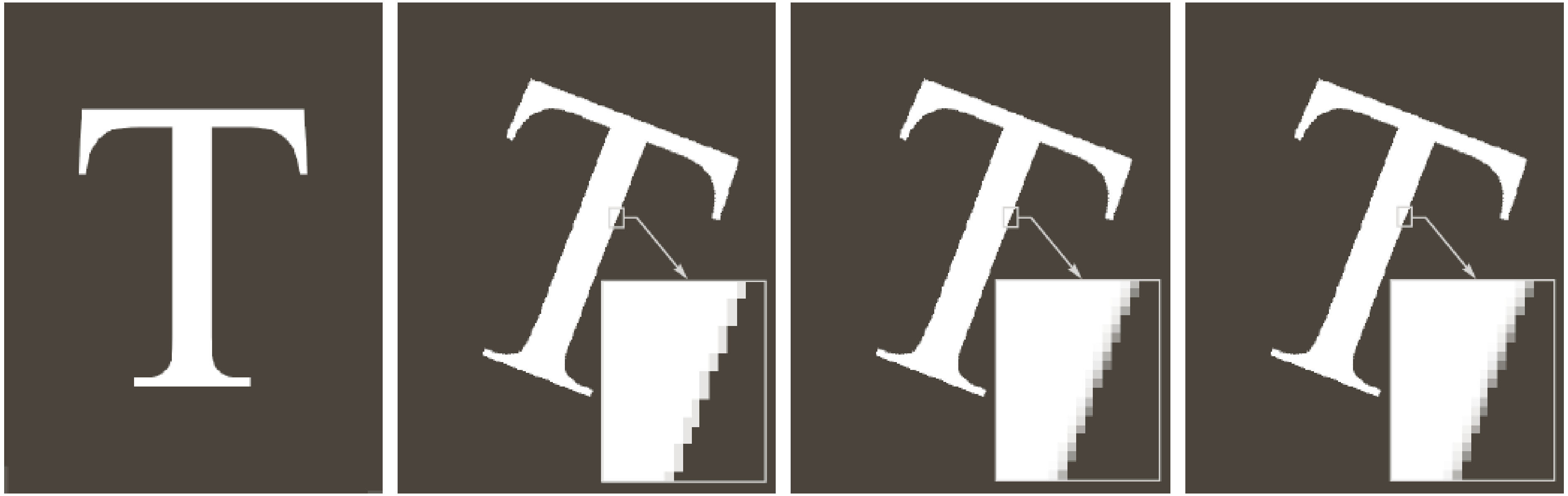
It's possible that two or more pixels can be transformed to the same location in the output image.

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

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

Inverse mappings are more efficient to implement than forward mappings.

Example: Image Rotation and Intensity Interpolation



a b c d

FIGURE 2.36 (a) A 300 dpi image of the letter T. (b) Image rotated 21° clockwise 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.

Image Registration

- Input and output images are available but the transformation function is unknown.

  Goal: estimate the transformation function and use it to register the two images.

- One of the principal approaches for image registration is to use ***tie points*** (also called ***control points***)

➢ The corresponding points are known precisely in the input and output (**reference**) images.

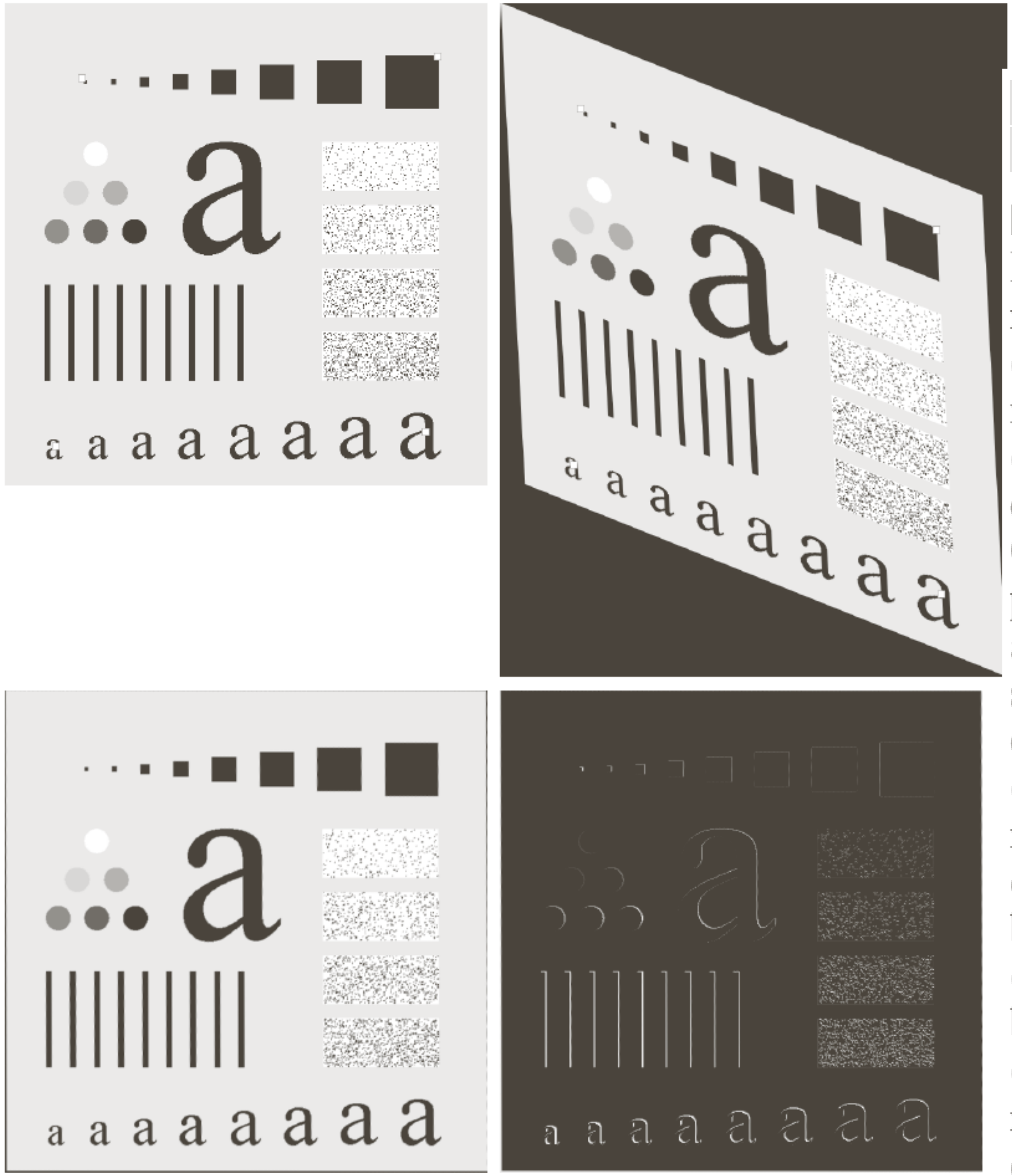
Image Registration

- A simple model based on bilinear approximation:

$$\begin{cases} x = c_1 v + c_2 w + c_3 vw + c_4 \\ y = c_5 v + c_6 w + c_7 vw + c_8 \end{cases}$$

Where (v, w) and (x, y) are the coordinates of tie points in the input and reference images.

Image Reg



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.

Image Transform

- A particularly important class of 2-D linear transforms, denoted T(u, v)

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

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

$r(x,y,u,v)$ is the *forward transformation* kernel,

variables u and v are the transform variables,

u = 0, 1, 2, ..., M-1 and v = 0, 1, ..., N-1.

Image Transform

- Given T(u, v), the original image f(x, y) can be recoverd using the inverse tranformation of T(u, v).

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

where $s(x,y,u,v)$ is the *inverse transformation kernel*,

x = 0, 1, 2, ..., M-1 and y = 0, 1, ..., N-1.

Image Transform

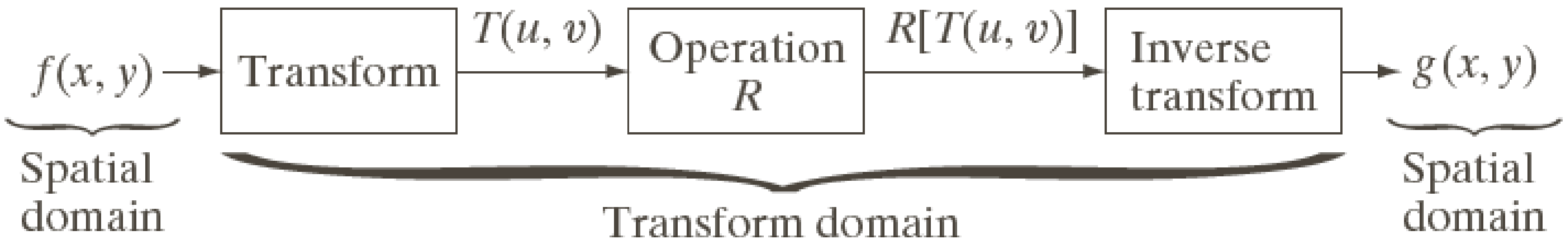
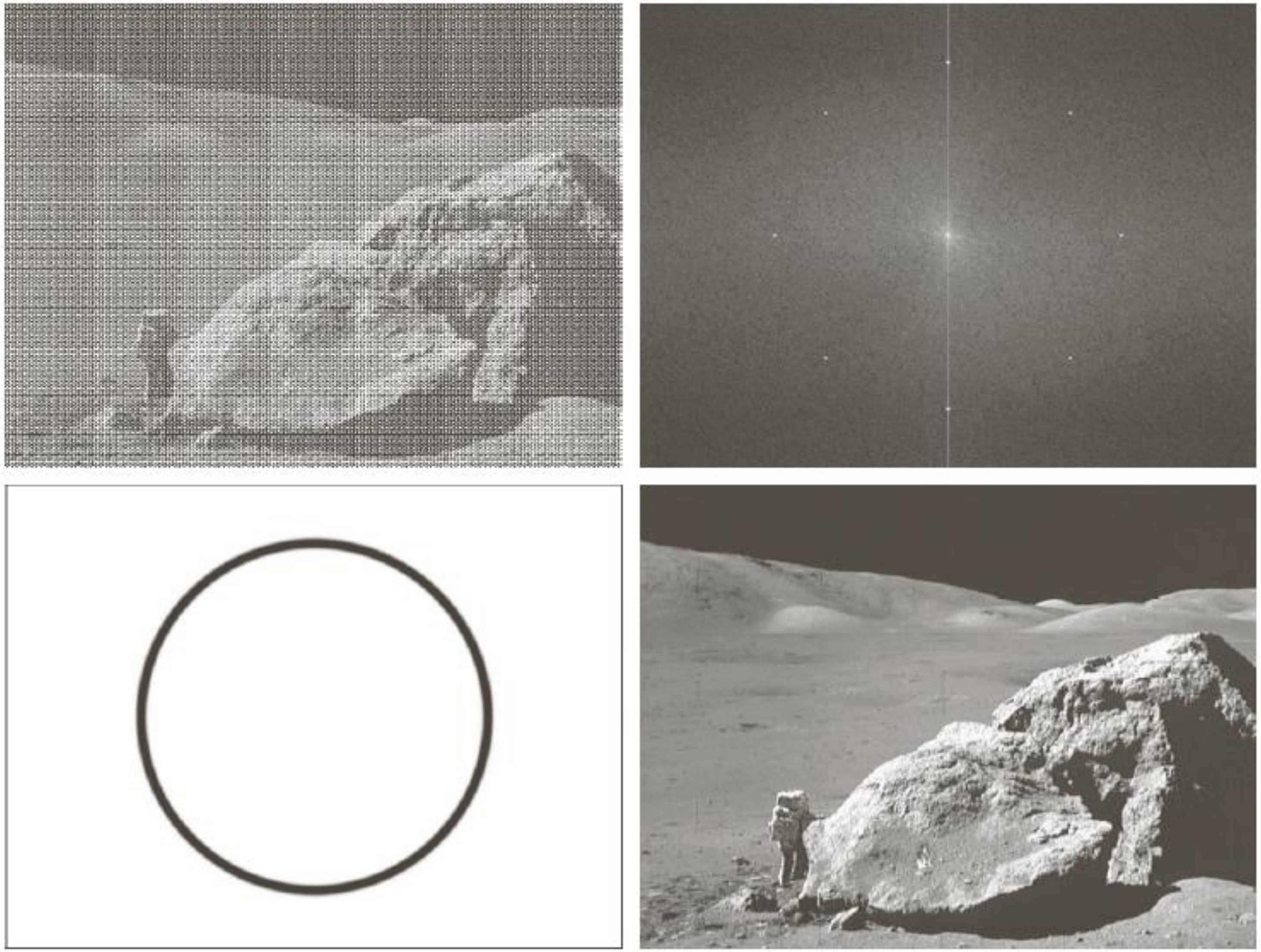


FIGURE 2.39
General approach for operating in the linear transform domain.

Example: Image Denoising by Using DCT Transform



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.)

Forward Transform Kernel

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

The kernel $r(x,y,u,v)$ is said to be SEPERABLE if

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

In addition, the kernel is said to be SYMMETRIC if

$r_1(x,u)$ is functionally equal to $r_2(y,v)$, so that

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

The Kernels for 2-D Fourier Transform

The *forward* kernel

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

Where $j = \sqrt{-1}$

The *inverse* kernel

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

2-D Fourier Transform

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

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

Probabilistic Methods

Let $z_i,\ i = 0,\ 1,\ 2,\ \ldots,\ L\text{-}1$, denote the values of all possible intensities in an $M \times N$ digital image. The probability, $p(z_k)$, of intensity level z_k occurring in a given image is estimated as

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

where n_k is the number of times that intensity z_k occurs in the image.

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

The mean (average) intensity is given by

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

Probabilistic Methods

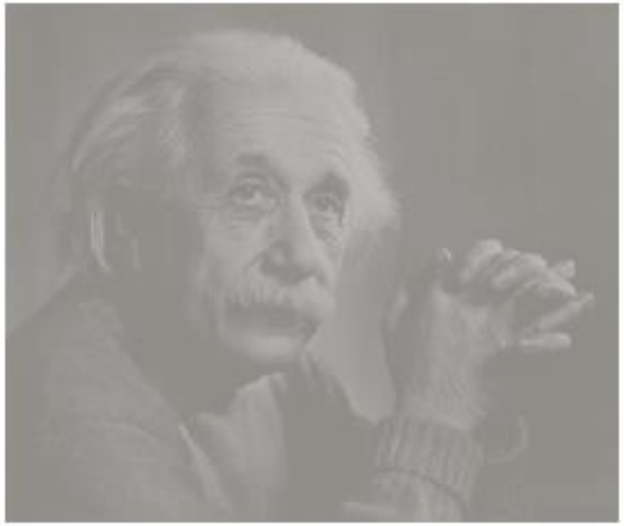
The variance of the intensities is given by

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

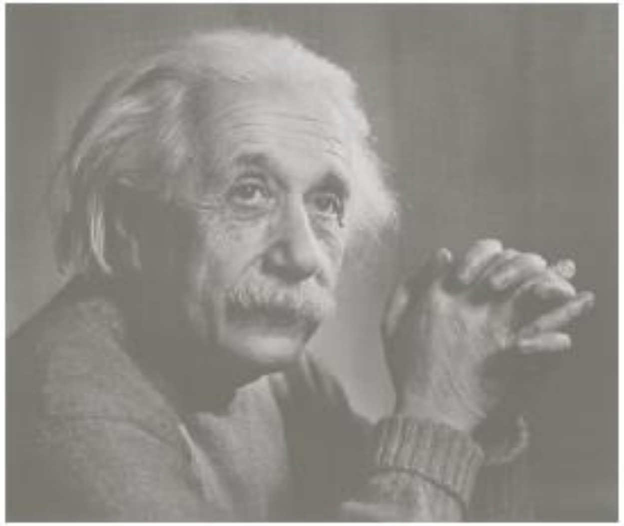
The n^{th} moment of the intensity variable z is

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

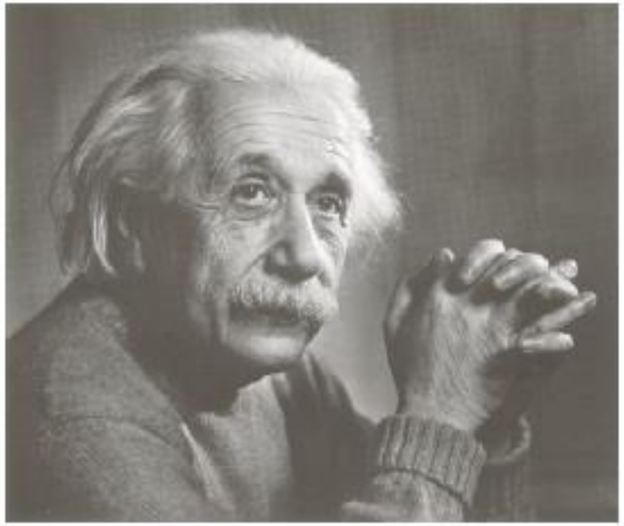
Example: Comparison of Standard Deviation Values



$\sigma = 14.3$



$\sigma = 31.6$



$\sigma = 49.2$