Text slides to

Digital Image Processing- A Signal Processing and Algorithmic Approach

D. Sundararajan

Second Edition

©Springer 2026

Chapter 1 Digital Image

The 2-D discrete impulse signal is defined as

$$\delta(m,n) = \begin{cases} 1 & \text{for } m = n = 0 \\ 0 & \text{otherwise} \end{cases}$$

The unit-impulse signal has a value of 1, when both of its arguments are zero and zero otherwise.

A 2-D sequence $x(m,n)$ is periodic with period $M \times N$, if

$x(m,n) = x(m+M,n) = x(m,n+N)$, for all (m,n)

For example, $\cos(\frac{2\pi}{8}(m+n))$ is periodic with period 8×8

$$\cos(\frac{2\pi}{8}(m+n))$$

$$\sin(\frac{2\pi}{8}(m+n)) \text{ and}$$

$$e^{j\frac{2\pi}{8}(m+n)} = \cos(\frac{2\pi}{8}(m+n)) + j\sin(\frac{2\pi}{8}(m+n))$$

are also periodic with period 8×8

Electromagnetic Spectrum

Cosmic rays	Gamma rays	X-rays	Ultra violet	Visible Spectrum	Infra-red

Microwaves	TV	Radio

A $M \times N$ image, $x(m,n)$, with M rows and N columns is given by

$$
\begin{array}{c} \\ m \\ \downarrow \end{array}
\begin{array}{c}
\begin{array}{cccc} & & n \rightarrow & \end{array} \\
\left[\begin{array}{cccc}
x(0,0) & x(0,1) & \dots & x(0,N-1) \\
x(1,0) & x(1,1) & \dots & x(1,N-1) \\
 & & \vdots & \\
x(M-1,0) & x(M-1,1) & \dots & x(M-1,N-1)
\end{array}\right]
\end{array}
$$

Pixel values of a 8×8 8-bit sub-image

173	185	189	186	199	195	195	192
177	187	189	192	197	195	189	177
188	190	196	197	199	193	171	124
191	192	197	198	192	158	111	110
196	199	99	189	149	108	110	113
202	200	182	130	100	98	108	114
204	178	117	85	100	96	104	108
173	100	85	87	95	98	96	100

Storage requirements

(i) 512×512 binary image,

$$512 \times 512 \times 1 = 262144 \text{ bits} = 32768 \text{ bytes}$$

(ii) 512×512 8-bit gray level image,

$$512 \times 512 \times 1 = 262144 \text{ bytes}$$

(iii) 512×512 color image, with a byte of storage for each of the 3 color components of a pixel,

$$512 \times 512 \times 3 = 786432 \text{ bytes}$$

Bit-plane components from MSB to LSB

$$\begin{bmatrix} 8 & 1 & 7 & 3 \\ 1 & 11 & 15 & 12 \\ 0 & 11 & 7 & 13 \\ 2 & 10 & 9 & 6 \end{bmatrix} = 2^3 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} +$$

$$2^2 \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} + 2 \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Two sinusoidal surfaces produce oscillations with the same frequency

$$
\begin{aligned}
x(k,l) &= \cos(\frac{2\pi}{32}30k + \frac{2\pi}{32}31l - \frac{\pi}{2}) \\
&= \cos(\frac{2\pi}{32}(32-2)k + \frac{2\pi}{32}(32-1)l - \frac{\pi}{2}) \\
&= \cos(\frac{2\pi}{32}2k + \frac{2\pi}{32}1l + \frac{\pi}{2})
\end{aligned}
$$

Applications of Digital image processing

Widely used in entertainment, business, science and engineering applications. Some of the areas of applications are:

1. Image sharpening and restoration.
2. Medical Applications.
3. Remote sensing.
4. Image compression and transmission.
5. Robots.
6. Automatic inspection of components.
7. Security.

Chapter 2 Image Enhancement in the Spatial Domain

An image is enhanced to increase the amount of information that can be interpreted visually.

Point Operations

$$
\begin{aligned}
z(m,n) &= x(m,n) + y(m,n) \\
z(m,n) &= x(m,n) - y(m,n) \\
z(m,n) &= x(m,n) * y(m,n) \\
z(m,n) &= x(m,n)/y(m,n)
\end{aligned}
$$

One of the operands in these operations can be a constant. For example, $z(m,n) = Cx(m,n)$ and $z(m,n) = C + x(m,n)$, where C is a constant. Logical operations AND (&), OR (|) and NOT (~) are also used in a similar way on binary images.

Image Complement

In a 8-bit gray-level image, the complement, $\tilde{x}(m,n)$, of the image $x(m,n)$ is given by

$$\tilde{x}(m,n) = 255 - x(m,n)$$

For a binary image, the complement is given by

$$\tilde{x}(m,n) = 1 - x(m,n)$$

Gamma Correction

$$inew = i^{\gamma}$$

i	0	0.1	0.2	0.3	0.4
$i^{0.6}$	0	0.2512	0.3807	0.4856	0.5771
$i^{1.6}$	0	0.0251	0.0761	0.1457	0.2308

0.5	0.6	0.7	0.8	0.9	1
0.6598	0.7360	0.8073	0.8747	0.9387	1
0.3299	0.4416	0.5651	0.6998	0.8449	1

Histogram

The histogram depicts the number of occurrences of each possible gray level in an image.

Pixel values of a 4×4 8-bit image (left) and its contrast-stretched version (right)

249	108	110	113
10	98	108	114
85	100	96	104
85	87	95	98

255	201	219	245
0	114	201	254
1	131	96	166
1	18	88	114

Contrast Stretching

$$inew = \left\lfloor \frac{(I_{max} - I_{min} - 2)}{(M - L)}(i - L) \right\rfloor + 1, \; L \leq i \leq M$$

$$inew = I_{min}, \quad i < L, \quad inew = I_{max}, \quad i > M$$

Gray level	10	85	87	95	96	98	100	104
Count	1	2	1	1	1	2	1	1
Gray level	0	1	18	88	96	114	131	166

108	110	113	114	249
2	1	1	1	1
201	219	245	254	255

Histogram Equalization

In both contrast stretching and histogram equalization, the objective is to spread the gray levels over the entire allowable gray level range. While stretching is a linear process and is reversible, equalization is a nonlinear process and is irreversible. Histogram equalization tries to redistribute about the same number of pixels for each gray level and it is automatic.

A 4×4 4-bit image (left) and its histogram-equalized version (right)

13	14	2	14
10	2	5	9
15	15	3	15
15	8	13	1

9	11	3	11
8	3	5	7
15	15	4	15
15	6	9	1

The equalization process for a gray level u of the input image is given by

$$v = (L-1)\sum_{n=0}^{u} p(n), \quad u = 0, 1, \ldots, L-1$$

where v is the corresponding gray level in the histogram equalized image.

Normalized histogram of the image
$\{0, 0.0625, 0.125, 0.0625, 0, 0.0625, 0, 0, 0.0625,$
$0.0625, 0.0625, 0, 0, 0.125, 0.125, 0.25\}$

The cumulative distribution
$\{0, 0.0625, 0.1875, 0.25, 0.25, 0.3125, 0.3125, 0.3125,$
$0.375, 0.4375, 0.5, 0.5, 0.5, 0.625, 0.75, 1\}$

These values, multiplied by $L - 1 = 15$, are
$\{0, 0.9375, 2.8125, 3.75, 3.75, 4.6875, 4.6875, 4.6875,$
$5.625, 6.5625, 7.5, 7.5, 7.5, 9.375, 11.25, 15\}$

Rounding of these values yields
$\{0, 1, 3, 4, 4, 5, 5, 5, 6, 7, 8, 8, 8, 9, 11, 15\}$

Histogram of the image and its equalized version

Gray level	0	1	2	3	4	5	6	7	8	9	10
$count_in$	0	1	2	1	0	1	0	0	1	1	1
$count_eq$	0	1	0	2	1	1	1	1	1	2	0

11	12	13	14	15
0	0	2	2	4
2	0	0	0	4

Histogram Specification

The histogram $a(n)$ of a reference image A is specified and the histogram $b(n)$ of the input image B is to be modified to produce an image C so that its distribution of pixels (histogram $c(n)$) is as similar to that of image A as possible.

The steps of the algorithms are:

1. Compute the cumulative distribution, $cum_a(n)$, of the reference image A.
2. Compute the cumulative distribution, $cum_b(k)$, of the input image B.
3. For each value in $cum_b(k)$, find the minimum value in $cum_a(n)$ that is greater than or equal to the current value in $cum_b(k)$. That n is the new gray level in the image C corresponding to k in image B.

4×4 reference, input and output images, respectively, from left

13	14	2	14
10	2	5	9
15	15	3	15
15	8	13	1

11	13	0	13
7	0	2	5
15	15	1	15
15	4	11	0

13	14	2	14
10	2	5	9
15	15	3	15
15	8	13	2

The cumulative distribution, $cum_a(n)$, of the reference image is
$\{0, 0.0625, 0.1875, 0.25, 0.25, 0.3125, 0.3125, 0.3125,$
$0.3750, 0.4375, 0.5, 0.5, 0.5, 0.6250, 0.75, 1\}$

The cumulative distribution, $cum_b(k)$, of the input image is
$\{0.1875, 0.25, 0.3125, 0.3125, 0.3750, 0.4375, 0.4375,$
$0.5, 0.5, 0.5, 0.6250, 0.6250, 0.75, 0.75, 1\}$

Pixels of the input image 0–15 are mapped to
$\{2, 3, 5, 5, 8, 9, 9, 10, 10, 10, 10, 13, 13, 14, 14, 15\}$

in the output image

Thresholding

A threshold indicates an intensity level of some significance.

$$g_b(x) = \begin{cases} 0 & \text{if } x \leq T \\ 1, & \text{otherwise} \end{cases}$$

$$g_h(x) = \begin{cases} 0 & \text{if } |x| \leq T \\ x, & \text{if } |x| > T \end{cases}$$

$$g_s(x) = \begin{cases} 0, & \text{if } |x| \leq T \\ x - T, & \text{if } x > T \\ x + T, & \text{if } x < -T \end{cases}$$

Input image

117	170	130	54	84	209	164	148
135	151	137	96	56	157	225	189
136	152	174	146	64	84	146	90
123	139	182	133	51	71	56	74
119	137	172	146	119	67	65	70
90	123	166	184	203	101	49	64
85	102	162	194	164	80	38	56
73	84	155	185	147	163	87	57

Binary thresholding with $T = 120$

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

Hard thresholding with $T = 120$

0	170	130	0	0	209	164	148
135	151	137	0	0	157	225	189
136	152	174	146	0	0	146	0
123	139	182	133	0	0	0	0
0	137	172	146	0	0	0	0
0	123	166	184	203	0	0	0
0	0	162	194	164	0	0	0
0	0	155	185	147	163	0	0

Soft thresholding with $T = 120$

0	50	10	0	0	89	44	28
15	31	17	0	0	37	105	69
16	32	54	26	0	0	26	0
3	19	62	13	0	0	0	0
0	17	52	26	0	0	0	0
0	3	46	64	83	0	0	0
0	0	42	74	44	0	0	0
0	0	35	65	27	43	0	0

Neighborhood Operations

Each pixel value $x(m, n)$ is replaced by another, which is a linear or nonlinear function of the values of the pixels in its neighborhood.

8-connected neighborhood

$$\begin{bmatrix} x(m-1,n-1) & x(m-1,n) & x(m-1,n+1) \\ x(m,n-1) & x(m,n) & x(m,n+1) \\ x(m+1,n-1) & x(m+1,n) & x(m+1,n+1) \end{bmatrix}$$

4-connected neighborhood

$$\begin{bmatrix} & x(m-1,n) & \\ x(m,n-1) & x(m,n) & x(m,n+1) \\ & x(m+1,n) & \end{bmatrix}$$

Symmetric extension

44	32	32	44	44	23	23	44
51	23	23	51	23	32	32	23
51	23	23	51	23	32	32	23
44	32	32	44	44	23	23	44
23	23	23	23	44	32	32	44
44	44	44	44	23	23	23	23
44	44	44	44	23	23	23	23
23	23	23	23	44	32	32	44

Mirror image of itself at the borders.

Replication method of extension

23	23	23	51	23	32	32	32
23	23	23	51	23	32	32	32
23	23	23	51	23	32	32	32
32	32	32	44	44	23	23	23
23	23	23	23	44	32	32	32
44	44	44	44	23	23	23	23
44	44	44	44	23	23	23	23
44	44	44	44	23	23	23	23

Border values are repeated.

Periodic extension

44	32	23	23	44	32	23	23
23	23	44	44	23	23	44	44
23	32	23	51	23	32	23	51
44	23	32	44	44	23	32	44
44	32	23	23	44	32	23	23
23	23	44	44	23	23	44	44
23	32	23	51	23	32	23	51
44	23	32	44	44	23	32	44

Image is considered as one period of a 2-D periodic signal. The top and bottom edges are considered adjacent and so are the right and left edges.

1-D Linear Convolution

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) = \sum_{k=-\infty}^{\infty} h(k)x(n-k)$$

$\{x(0) = 4, x(1) = 3, x(2) = 1, x(3) = 2\}$ and
$\{h(0) = 1, h(1) = -2, h(2) = 1\}$

$y(0) = x(k)h(0-k) = (4)(1) = 4$

$y(1) = x(k)h(1-k) = (4)(-2) + (3)(1) = -5$

$y(2) = x(k)h(2-k) = (4)(1) + (3)(-2) + (1)(1) = -1$

$y(3) = x(k)h(3-k) = (3)(1) + (1)(-2) + (2)(1) = 3$

$y(4) = x(k)h(4-k) = (1)(1) + (2)(-2) = -3$

$y(5) = x(k)h(5-k) = (2)(1) = 2$

2-D Linear Convolution

$$y(m,n) = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x(k,l)h(m-k,n-l)$$

1. One of the images, say $h(k,l)$, is rotated in the (k,l) plane by 180 degrees about the origin to get $h(-k,-l)$.
2. Shifted by (m,n) to get $h(m-k,n-l)$.
3. The products $x(k,l)h(m-k,n-l)$ of all the overlapping samples are found.
4. The sum of all the products yields the convolution output $y(m,n)$ at (m,n).

Moving average filter

$$
\begin{aligned}
h(m,n) &= \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
&= \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} = h_c(m)h_r(n)
\end{aligned}
$$

$$x(m,n) = \begin{bmatrix} 1 & -1 & 3 & 2 \\ 2 & 1 & 2 & 4 \\ 1 & -1 & 2 & -2 \\ 3 & 1 & 2 & 2 \end{bmatrix}$$

Assuming zero-padding at the borders, the output of 1-D filtering of the rows of the input and the output of 1-D filtering of the columns of the partial output are, respectively,

$$\frac{1}{3}\begin{bmatrix} 0 & 3 & 4 & 5 \\ 3 & 5 & 7 & 6 \\ 0 & 2 & -1 & 0 \\ 4 & 6 & 5 & 4 \end{bmatrix} \qquad y(m,n) = \frac{1}{9}\begin{bmatrix} 3 & 8 & 11 & 11 \\ 3 & 10 & 10 & 11 \\ 7 & 13 & 11 & 10 \\ 4 & 8 & 4 & 4 \end{bmatrix}$$

Assuming replication at the borders, the extended input and the output are, respectively,

$$\begin{bmatrix} 1 & 1 & -1 & 3 & 2 & 2 \\ 1 & 1 & -1 & 3 & 2 & 2 \\ 2 & 2 & 1 & 2 & 4 & 4 \\ 1 & 1 & -1 & 2 & -2 & -2 \\ 3 & 3 & 1 & 2 & 2 & 2 \\ 3 & 3 & 1 & 2 & 2 & 2 \end{bmatrix} \qquad \frac{1}{9} \begin{bmatrix} 7 & 11 & 15 & 24 \\ 7 & 10 & 10 & 15 \\ 13 & 13 & 11 & 14 \\ 15 & 14 & 9 & 10 \end{bmatrix}$$

Only the output at the borders differ with different border extensions. The central part of the output is the same.

Gaussian lowpass filter

$$h(m,n) = \frac{e^{-\frac{(m^2+n^2)}{(2\sigma^2)}}}{K}, K = \sum_{m=-(N-1)/2}^{(N-1)/2} \sum_{n=-(N-1)/2}^{(N-1)/2} e^{-\frac{(m^2+n^2)}{(2\sigma^2)}}$$

assuming N is odd.

$$h(m,n) = \begin{bmatrix} 0.0113 & 0.0838 & 0.0113 \\ 0.0838 & 0.6193 & 0.0838 \\ 0.0113 & 0.0838 & 0.0113 \end{bmatrix}$$

$$= \begin{bmatrix} 0.1065 \\ 0.7870 \\ 0.1065 \end{bmatrix} \begin{bmatrix} 0.1065 & 0.7870 & 0.1065 \end{bmatrix}$$

Assuming zero-padding at the borders, the output of 1-D filtering of the rows of the input and the output of 1-D filtering of the columns of the partial output are, respectively,

$$\begin{bmatrix} 0.6805 & -0.3610 & 2.4675 & 1.8935 \\ 1.6805 & 1.2130 & 2.1065 & 3.3610 \\ 0.6805 & -0.4675 & 1.2545 & -1.3610 \\ 2.4675 & 1.3195 & 1.8935 & 1.7870 \end{bmatrix}$$

$$y(m,n) = \begin{bmatrix} 0.7145 & -0.1549 & 2.1663 & 1.8481 \\ 1.4675 & 0.8664 & 2.0542 & 2.7018 \\ 0.9773 & -0.0982 & 1.4133 & -0.5228 \\ 2.0144 & 0.9887 & 1.6238 & 1.2614 \end{bmatrix}$$

Assuming periodicity at the borders, the extended input and the output are, respectively,

$$xe(m,n) = \begin{bmatrix} 2 & 3 & 1 & 2 & 2 & 3 \\ 2 & 1 & -1 & 3 & 2 & 1 \\ 4 & 2 & 1 & 2 & 4 & 2 \\ -2 & 1 & -1 & 2 & -2 & 1 \\ 2 & 3 & 1 & 2 & 2 & 3 \\ 2 & 1 & -1 & 3 & 2 & 1 \end{bmatrix}$$

$$y(m,n) = \begin{bmatrix} 1.2130 & -0.0143 & 2.3679 & 2.1790 \\ 1.8027 & 0.8664 & 2.0542 & 2.8921 \\ 0.8777 & -0.0982 & 1.4133 & -0.3822 \\ 2.2545 & 0.9502 & 1.8866 & 1.7372 \end{bmatrix}$$

Highpass Filtering

Frequency, in image processing, is the rate of change of gray levels of an image with respect to distance. A high frequency component is characterized by large changes in gray levels over short distances and vice versa. High-pass filters pass high frequency components and suppress low frequency components. This type of filters are used for sharpening images and edge detection.

Laplacian operator of a function $f(x, y)$

$$\nabla^2 f(x, y) = \frac{\partial^2 f(x, y)}{\partial x^2} + \frac{\partial^2 f(x, y)}{\partial y^2}$$

For discrete signals, differencing approximates differentiation. At the point $x(m, n)$, the first differences along the horizontal and vertical directions, $\Delta_h(m, n)$ and $\Delta_v(m, n)$, are defined as

$$\Delta_h x(m, n) = x(m, n) - x(m, n - 1)$$
$$\Delta_v x(m, n) = x(m, n) - x(m - 1, n)$$

Using the first differences again,

$$\begin{aligned}
\Delta_v^2 x(m,n) &= \Delta_v x(m+1,n) - \Delta_v x(m,n) \\
&= (x(m+1,n) - x(m,n)) \\
&\quad - (x(m,n) - x(m-1,n)) \\
&= x(m+1,n) + x(m-1,n) - 2x(m,n) \\
\Delta_h^2 x(m,n) &= \Delta_h x(m,n+1) - \Delta_h x(m,n) \\
&= (x(m,n+1) - x(m,n)) \\
&\quad - (x(m,n) - x(m,n-1)) \\
&= x(m,n+1) + x(m,n-1) - 2x(m,n)
\end{aligned}$$

Summing the two second differences, we get the discrete approximation of the Laplacian as

$$\begin{aligned}\nabla^2 x(m,n) &= \Delta_h^2 x(m,n) + \Delta_v^2 x(m,n) \\ &= x(m,n+1) + x(m,n-1) + x(m+1,n) \\ &\quad + x(m-1,n) - 4x(m,n)\end{aligned}$$

Versions of Laplacian highpass filter

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Laplacian sharpening filter

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Let the input be the same used for lowpass filtering. With zero-padded and replicated inputs, the outputs of applying the Laplacian mask (Equation 2.2) are, respectively,

$$\begin{bmatrix} -3 & 9 & -9 & -1 \\ -5 & -2 & 2 & -14 \\ 0 & 9 & -7 & 16 \\ -10 & 0 & -3 & -8 \end{bmatrix} \quad \begin{bmatrix} -1 & 8 & -6 & 3 \\ -3 & -2 & 2 & -10 \\ 1 & 9 & -7 & 14 \\ -4 & 1 & -1 & -4 \end{bmatrix}$$

Using sharpening filter, with the same input used for lowpass filtering, the outputs with the input zero-padded and replicated are, respectively,

$$\begin{bmatrix} 4 & -10 & 12 & 3 \\ 7 & 3 & 0 & 18 \\ 1 & -10 & 9 & -18 \\ 13 & 1 & 5 & 10 \end{bmatrix} \quad \begin{bmatrix} 2 & -9 & 9 & -1 \\ 5 & 3 & 0 & 14 \\ 0 & -10 & 9 & -16 \\ 7 & 0 & 3 & 6 \end{bmatrix}$$

Median filter

The median of a list of N numbers

$$\{x(0), x(1), \ldots, x(N-1)\}$$

is defined as the middle number of the sorted list of $x(n)$, if N is odd. If N is even, the median is defined as the mean of the two middle numbers of the sorted list. For 2-D data, all the samples are listed as 1-D data for median computation.

The boundary replicated version of a 4×4 image and its median filtered version with a 3×3 window are

23	23	51	23	32	32

23	23	51	23	32	32
32	32	44	44	23	23
23	23	23	44	32	32
44	44	44	23	23	23
44	44	44	23	23	23

32	32	32	32
23	32	32	32
32	44	32	23
44	44	23	23

Chapter 3 Fourier Analysis

In Fourier analysis, a time-domain waveform is decomposed into its sinusoidal components of various frequencies.

1. It gives the strength of the various components, which is called the spectrum of the signal. The spectrum is the starting point in most of the analysis.

2. It is more efficient to find the system output using the sinusoidal components of the input signal.

Fourier analysis represents a signal as a linear combination of sinusoids or, equivalently, complex exponentials with pure imaginary exponents.

$$
\begin{aligned}
& \frac{1}{4}(4e^{j0\frac{2\pi}{4}n}+(2-j2\sqrt{3})e^{j\frac{2\pi}{4}n}+4e^{j2\frac{2\pi}{4}n} \\
+\ & (2+j2\sqrt{3})e^{j3\frac{2\pi}{4}n}) \\
=\ & \frac{1}{4}(4e^{j0\frac{2\pi}{4}n}+4e^{j(\frac{2\pi}{4}n-\frac{\pi}{3})}+4e^{j2\frac{2\pi}{4}n}+4e^{-j(\frac{2\pi}{4}n-\frac{\pi}{3})}) \\
=\ & 1+2\cos(\frac{2\pi}{4}n-\frac{\pi}{3})+\cos(2\frac{2\pi}{4}n)=x(n)
\end{aligned}
$$

DFT and IDFT

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk}, \;\; k = 0, 1, \ldots, N-1$$

where $W_N = e^{-j\frac{2\pi}{N}}$.

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) W_N^{-nk}, \;\; n = 0, 1, \ldots, N-1$$

The Fourier reconstruction of a waveform is with respect to the least squares error criterion.

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 3 \\ \sqrt{3} \\ 1 \\ -\sqrt{3} \end{bmatrix}$$

$$= \begin{bmatrix} 4 \\ 2 - j2\sqrt{3} \\ 4 \\ 2 + j2\sqrt{3}, \end{bmatrix}$$

$$\begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & j & -1 & -j \\ 1 & -1 & 1 & -1 \\ 1 & -j & -1 & j \end{bmatrix} \begin{bmatrix} 4 \\ 2 - j2\sqrt{3} \\ 4 \\ 2 + j2\sqrt{3} \end{bmatrix}$$

$$= \begin{bmatrix} 3 \\ \sqrt{3} \\ 1 \\ -\sqrt{3} \end{bmatrix}$$

Parseval's Theorem

$$\{4, 1, 2, 4\} \leftrightarrow \{11, 2 + j3, 1, 2 - j3\}$$

The sum of the squared magnitude of the data sequence is 37 and that of the DFT coefficients divided by 4 is also 37.

The 2-D DFT of a $N \times N$ image $x(m,n)$ is defined as

$$X(k,l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m,n) e^{-j\frac{2\pi}{N}(mk+nl)}$$

The 2-D IDFT is given by

$$x(m,n) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} X(k,l) e^{j\frac{2\pi}{N}(mk+nl)}$$

The impulse (on the left) and its 2-D DFT are

$$\begin{array}{|cccc|} \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \hline \end{array} \leftrightarrow \begin{array}{|cccc|} \hline 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \hline \end{array}$$

$$
\begin{aligned}
& \frac{1}{64}((-7.7071 - j50.163)e^{j\frac{2\pi}{8}m} \\
& +(-7.7071 + j50.163)e^{-j\frac{2\pi}{8}m}) \\
= \; & \frac{2}{64}|(-7.7071 - j50.163)| \\
& \cos(\frac{2\pi}{8}m + \angle(-7.7071 - j50.163)) \\
= \; & 1.586\cos(\frac{2\pi}{8}m - 98.7347^\circ) = x(m,n)
\end{aligned}
$$

The DFT of the $N \times N$ matrix $x(m,n)$ can be computed in two stages. For example, the 1-D DFT of each row of the input image results in

$$X(m,l) = \sum_{n=0}^{N-1} x(m,n) e^{-j\frac{2\pi}{N}nl}, \;\; m,l = 0,1,\ldots,N-1$$

Then, the 1-D DFT of each column of $X(m,l)$ yields the 2-D DFT.

$$X(k,l) = \sum_{m=0}^{N-1} X(m,l) e^{-j\frac{2\pi}{N}mk}, \;\; k,l = 0,1,\ldots,N-1$$

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 1 \\ -2 & 3 & 1 & 4 \\ 1 & 1 & 2 & 2 \\ 3 & 1 & 2 & 4 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$

The 2-D DFT of the image is

$$\begin{array}{c} \\ k \\ \downarrow \end{array} \begin{array}{c} l \rightarrow \\ \begin{bmatrix} 29 & -5+j4 & -7 & -5-j4 \\ 1+j4 & -3+j2 & 1+j8 & 1+j6 \\ -3 & -1-j4 & 9 & -1+j4 \\ 1-j4 & 1-j6 & 1-j8 & -3-j2 \end{bmatrix} \end{array} = X(k,l)$$

The 2-D DFT of a linear combination of a set of discrete images is equal to the same linear combination of their individual DFTs. Let $x_1(m,n) \leftrightarrow X_1(k,l)$ and $x_2(m,n) \leftrightarrow X_2(k,l)$. Then,

$$ax_1(m,n) + bx_2(m,n) \leftrightarrow aX_1(k,l) + bX_2(k,l)$$

where a and b are real or complex constants.

An image is periodic if it repeats its values over a period indefinitely, $x(m+M, n+N) = x(m, n)$ for all m, n. The smallest M, N satisfying the constraint are the periods in the two directions. Although a practical image is of finite extent, as the basis signals in Fourier analysis are the sinusoids (which are periodic), an image is assumed to be periodic in both the spatial and frequency domains.

$$x(m,n) = x(m + aM, n + bN), \quad \text{for all } m, n$$

$$X(k,l) = X(k + aM, l + bN), \quad \text{for all } k, l$$

where a and b are arbitrary integers. Useful information in a periodic signal is contained in any one period. The top and bottom edges are considered adjacent and so are the right and left edges.

A shift of a sinusoid results in changing its phase. Its magnitude is not affected. For a $N \times N$ image,

$$\begin{aligned} x(m,n) \leftrightarrow X(k,l) &\rightarrow x(m-m_0, n-n_0) \\ &\leftrightarrow X(k,l)e^{-j\frac{2\pi}{N}(km_0+ln_0)} \end{aligned}$$

$$x(m,n)e^{j\frac{2\pi}{N}(k_0m+l_0n)} \leftrightarrow X(k-k_0, l-l_0)$$

A specific use of this theorem is that the center-zero spectrum can be obtained with N even and $k_0 = l_0 = \frac{N}{2}$.

Circular convolution

Let $x(m,n) \leftrightarrow X(k,l)$ and $h(m,n) \leftrightarrow H(k,l),\ m,n,k,l$ $0,1,\ldots,N-1$. Then,

$$\sum_{p=0}^{N-1}\sum_{q=0}^{N-1} x(p,q)h(m-p,n-q) \leftrightarrow X(k,l)H(k,l)$$

$$x(m,n)h(m,n) \leftrightarrow \frac{1}{N^2}\sum_{p=0}^{N-1}\sum_{q=0}^{N-1} X(p,q)H(k-p,l-q)$$

The circular cross-correlation of $x(m,n)$ and $h(m,n)$ is given by

$$\begin{aligned} r_{xh}(m,n) &= \sum_{p=0}^{N-1}\sum_{q=0}^{N-1} x(p,q)h(p-m,q-n) \\ &\leftrightarrow H^*(k,l)X(k,l) \end{aligned}$$

$r_{hx}(m,n)$=$r_{xh}(N-m,N-n)$=IDFT$(X^*(k,l)H(k,l))$

Cross-correlation of an image $x(m,n)$ with itself is the autocorrelation operation.

$$r_{xx}(m,n) = \text{IDFT}(|X(k,l)|^2)$$

DFT values, of a real-valued $x(m,n)$, at diametrically opposite points form complex conjugate pairs.

$$X^*(N-k,N-l) = X(k,l)$$

An equivalent form of the symmetry is

$$X(\frac{N}{2} \pm k, \frac{N}{2} \pm l) = X^*(\frac{N}{2} \mp k, \frac{N}{2} \mp l)$$

$$\begin{bmatrix} \underline{29} & \underline{-5+j4} & \underline{7} & -5-j4 \\ \underline{1+j4} & \underline{-3+j2} & \underline{1+j8} & 1+j6 \\ \underline{-3} & \underline{-1-j4} & \underline{9} & -1+j4 \\ 1-j4 & \underline{1-j6} & 1-j8 & -3-j2 \end{bmatrix}$$

The 2-D DFT is a separable function in the variables m and n. Therefore, the DFT of a separable function $x(m,n) = x(m)x(n)$ is also separable. The product of the column vector with the row vector is equal to the 2-D function. That is,

$$x(m) \leftrightarrow X(k), x(n) \leftrightarrow X(l) \rightarrow X(k,l) = X(k)X(l)$$

Parseval's theorem

This theorem implies that the signal power can also be computed from the DFT representation of the image. Let $x(m,n) \leftrightarrow X(k,l)$ with the dimensions of the image $N \times N$.

$$\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}|x(m,n)|^2 = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}|X(k,l)|^2$$

The FT $X(j\omega)$ of $x(t)$ is defined as

$$X(j\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt$$

A sufficient condition for the existence of $X(j\omega)$ is that $x(t)$ is absolutely integrable. The IFT $x(t)$ of $X(j\omega)$ is defined as

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(j\omega)e^{j\omega t}d\omega$$

The amplitude of any component ($-\infty < \omega < \infty$) is $X(j\omega)\ d\omega/(2\pi)$, which is infinitesimal. The FT is a relative amplitude spectrum.

The FT $X(ju, jv)$ of $x(p, q)$ is defined as

$$X(ju, jv) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x(p, q) e^{-jup} e^{-jvq} dp\, dq$$

The IFT is given by

$$x(p, q) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X(ju, jv) e^{jup} e^{jvq} du\, dv$$

Enhancement in the Frequency Domain
Chapter 4

Processing of images in the frequency domain consists of:

1. Transformation of the input image and the system response from the spatial domain to the frequency domain.
2. Processing the image in the frequency domain.
3. Transformation of the processed image back to the spatial domain.

1. The DFT assumes periodicity of the finite input data. It has to be ensured that the output is represented with adequate accuracy in one period.
2. To meet this constraint, sufficient zero padding of the image and the filter is required. Further, the dimensions of one period have to be a power of 2 in order to use practically fast DFT algorithms.
3. It has to be ensured that both the image and the filter are in the same format with their origins aligned.

Convolve

$$x(m,n) = \begin{bmatrix} 1 & -1 & 3 & 2 \\ 2 & 1 & 2 & 4 \\ 1 & -1 & 2 & -2 \\ 3 & 1 & 2 & 2 \end{bmatrix}$$

and a 3×3 Gaussian lowpass filter with $\sigma = 0.5$. Assume periodicity at the borders.

This filter is also separable with the same coefficients in both the directions,

$$\{0.1065, 0.7870, 0.1065\}$$

Zero-padding and circularly shifting the column filter, we get

$$hz(m) = \{0.7870, 0.1065, 0, 0.1065\}$$

Only one zero is appended, since the convolution is periodic and the input is a 4×4 image. The 1-D DFT of this filter is

$$H(k) = \{1, 0.7870, 0.5740, 0.7870\}$$

The 2-D DFT, $X(k,l)$, of $x(m,n)$

22.00+j0.0	-2.00+j6.0	10.00+j0.0	-2.00-j6.0
5.00-j1.0	1.00+j5.0	-3.00+j3.0	-3.00-j3.0
-12.00+j0.0	-4.00-j2.0	8.00+j0.0	-4.00+j2.0
5.00+j1.0	-3.00+j3.0	-3.00-j3.0	1.00-j5.0

The partial convolution output

$$P(k,l) = X(k,l)H(k)$$

22.00+j0.0	-2.00+j6.0	10.00+j0.0	-2.00-j6.0
3.94-j0.7	0.79+j3.9	-2.36+j2.3	-2.36-j2.3
-6.89+j0.0	-2.30-j1.1	4.59+j0.0	-2.30+j1.1
3.94+j0.7	-2.36+j2.3	-2.36-j2.3	0.79-j3.9

The convolution output

$$Y(k,l) = P(k,l)H(l)$$

22.00+j0.0	-1.57+j4.7	5.74+j0.0	-1.57-j4.7
3.94-j0.7	0.62+j3.1	-1.36+j1.3	-1.86-j1.8
-6.89+j0.0	-1.81-j0.9	2.64+j0.0	-1.81+j0.9
3.94+j0.7	-1.86+j1.8	-1.36-j1.3	0.62-j3.1

The Laplacian filter is inseparable. The zero-padding and shifting in two directions results in

$$hz(m,n) = \begin{bmatrix} 5 & -1 & 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2-D DFT, $H(k,l)$, of zero-padded $hz(m,n)$

1.00	1.59	3.00	4.41	5.00	4.41	3.00	1.59
1.59	2.17	3.59	5.00	5.59	5.00	3.59	2.17
3.00	3.59	5.00	6.41	7.00	6.41	5.00	3.59
4.41	5.00	6.41	7.83	8.41	7.83	6.41	5.00
5.00	5.59	7.00	8.41	9.00	8.41	7.00	5.59
4.41	5.00	6.41	7.83	8.41	7.83	6.41	5.00
3.00	3.59	5.00	6.41	7.00	6.41	5.00	3.59
1.59	2.17	3.59	5.00	5.59	5.00	3.59	2.17

DFT is real-valued and even-symmetric, since the filter is also real-valued and even-symmetric.

Given the DFT coefficients of the waveform in the center-zero format,

$$\begin{aligned}\{ \quad & X(-2) = 4, X(-1) = 2 + j2\sqrt{3}, \\ & X(0) = 4, X(1) = 2 - j2\sqrt{3}\}\end{aligned}$$

we multiplied the set of coefficients, respectively, by the frequency responses

$$H_l(k) = \{0, 0, 1, 0\}, \quad H_h(k) = \{1, 0, 0, 0\},$$
$$H_{bp}(k) = \{0, 1, 0, 1\}, \quad H_{br}(k) = \{1, 0, 1, 0\}$$

to implement the different filters.

A lowpass filter in the frequency domain is given by

$$H(k,l) = \begin{cases} 1, & \text{for } D(k,l) \leq D_c \\ 0, & \text{for } D(k,l) > D_c \end{cases}$$

where $D(k,l) = \sqrt{k^2 + l^2}$ is the distance between the spectral point (k,l) and the center of the spectrum, and D_c is the cutoff radius.

A 4×4 distance matrix with the center at coordinates $(2, 2)$ is

$$D(k,l) = \begin{bmatrix} 2.8284 & 2.2361 & 2.0000 & 2.2361 \\ 2.2361 & 1.4142 & 1.0000 & 1.4142 \\ 2.0000 & 1.0000 & 0 & 1.0000 \\ 2.2361 & 1.4142 & 1.0000 & 1.4142 \end{bmatrix}$$

If we specify that the cutoff radius is 1.9, then the lowpass filter spectrum is

$$H(k,l) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

A highpass filter in the frequency domain is defined as

$$H(k,l) = \begin{cases} 0, & \text{for } D(k,l) \leq D_c \\ 1, & \text{for } D(k,l) > D_c \end{cases}$$

where $D(k,l) = \sqrt{k^2 + l^2}$ is the distance between the spectral point (k,l) and the center of the spectrum, and D_c is the cutoff radius. A highpass filter is also defined, in terms of the spectrum of the lowpass filter, as

$$H_h(k,l) = 1 - H_l(k,l)$$

where $H_h(k,l)$ and $H_l(k,l)$ are, respectively, the spectra of highpass and lowpass filters.

The spectrum of the lowpass Butterworth filter is defined as

$$H(k,l) = \frac{1}{1 + \left(\frac{D(k,l)}{D_c}\right)^{2n}}$$

where n is the order of the filter.

The spectrum of the Butterworth highpass filter is defined as

$$H(k,l) = \frac{1}{1 + \left(\frac{D_c}{D(k,l)}\right)^{2n}}$$

where D_c is the cutoff frequency and n is the order of the filter.

The spectrum of the lowpass Gaussian filter is defined as

$$H(k,l) = e^{-\frac{D^2(k,l)}{2D_c^2}}$$

The attenuation is $e^{-0.5} = 0.6065$ at $D(k,l) = D_c = \sigma$.

The Gaussian highpass filter is defined, in terms of the spectrum of that of its lowpass filter, as

$$H_h(k,l) = 1 - H_l(k,l) = 1 - e^{-\frac{D^2(k,l)}{2D_c^2}}$$

where $H_h(k,l)$ and $H_l(k,l)$ are, respectively, the spectra of highpass and lowpass filters.

Restoration Chapter 5

Inverse filtering

$$Y(k,l) = X(k,l)H_d(k,l)$$

where $Y(k,l)$, $X(k,l)$, and $H_d(k,l)$ are the corresponding DFTs of the degraded image, the input image and the impulse response of the degradation process. The image can be restored by the operation, called inverse filtering,

$$X(k,l) = \frac{Y(k,l)}{H_d(k,l)}$$

Wiener Filter

The problem is to find the estimate, $\widehat{x}(n)$, of $x(n)$ from $y(n)$ such that the least-squares error, E, is

$$E = \sum_{n=0}^{N-1} (x(n) - \widehat{x}(n))^2$$

minimized. Assuming that the estimated signal $\widehat{x}(n)$ is given by

$$\widehat{x}(n) = \sum_{k} y(n-k) h_r(k)$$

the task is to find the filter coefficients so that the least-squares error is minimized.

From Parseval's theorem,

$$E = \frac{1}{N} \sum_{k=0}^{N-1} |(X(k) - \hat{X}(k))|^2$$

$$\hat{X}(k) = H_r(k)Y(k) = H_r(k)H_d(k)X(k) + H_r(k)S(k)$$

$$X(k) - \hat{X}(k) = (1 - H_r(k)H_d(k))X(k) - H_r(k)S(k)$$

$$E = \frac{1}{N}\sum_{k=0}^{N-1} |(1 - H_r(k)H_d(k))X(k) - H_r(k,l)S(k)|^2$$
$$= \frac{1}{N}\sum_{k=0}^{N-1} |(1 - H_r(k)H_d(k))X(k)|^2 + |H_r(k)S(k)|^2$$
$$= \frac{1}{N}\sum_{k=0}^{N-1} |(1 - H_r(k)H_d(k))|^2|X(k)|^2 + |H_r(k)|^2|S(k)|^2$$

Setting the derivative of the last expression with respect to $H_r(k)$ equal to zero, we get

$$2(-(1-H_r^*(k)H_d^*(k))H_d(k)|X(k)|^2+H_r^*(k)|S(k)|^2)=0$$

$$H_r^*(k) = \frac{H_d(k)|X(k)|^2}{|H_d(k)|^2|X(k)|^2 + |S(k)|^2}$$

$$H_r(k) = \frac{H_d^*(k)}{|H_d(k)|^2 + |S(k)|^2/|X(k)|^2}$$

2-D Wiener filter

$$H_r(k,l) = \frac{H_d^*(k,l)}{|H_d(k,l)|^2 + |S(k,l)|^2/|X(k,l)|^2}$$

where $|H_d(k,l)|^2$ is the power spectrum of the degradation process, $H_r(k,l)$ is the DFT of the Wiener filter, $|S(k,l)|^2$ and $|X(k,l)|^2$ are the power spectral densities of the noise and the true image, respectively. The restored image $\widehat{x}(m,n)$ is obtained from the degraded image $y(m,n)$ as

$$\widehat{x}(m,n) = \text{IDFT}(H_r(k,l)Y(k,l))$$

1. Find the DFT $Y(k,l)$ of the degraded image $y(m,n)$.
2. Derive the Wiener filter $H_r(k,l)$.
3. Multiply pointwise $Y(k,l)$ with the Wiener filter $H_r(k,l)$.
4. Compute the IDFT $H_r(k,l)Y(k,l)$ to get the restored image.

Image Degradation Model

Let $x(m,n)$ be a $N \times N$ image and $h_d(m,n)$ be the $P \times Q$ impulse response of the process due to motion and $x(m,n) \leftrightarrow X(k,l)$. Then,

$$x(m-p, n-q) \leftrightarrow X(k,l) e^{-j\frac{2\pi}{N}(kp+lq)}$$

$$y(m,n) = \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} x(m-p, n-q)$$

$$H_d(k,l) = \sum_{p=0}^{P-1} \sum_{q=0}^{Q-1} e^{-j\frac{2\pi}{N}pk} e^{-j\frac{2\pi}{N}qk}$$

$$= e^{-j\frac{\pi}{N}(P-1)k} \left(\frac{\sin(\frac{\pi}{N}Pk)}{\sin(\frac{\pi}{N}k)} \right) e^{-j\frac{\pi}{N}(Q-1)l} \left(\frac{\sin(\frac{\pi}{N}Ql)}{\sin(\frac{\pi}{N}l)} \right)$$

Chapter 6 Geometric Transformations

In interpolation, the known values are appropriately weighted and summed to get the interpolated value at intermediate locations.

In the nearest-neighbor interpolation method, the interpolated pixel value is that of its nearest neighbor.

2	1
3	4

2	2	1	1
2	2	1	1
3	3	4	4
3	3	4	4

(0,0)	(0,1/3)	(0,2/3)	(0,1)
(1/3,0)	(1/3,1/3)	(1/3,2/3)	(1/3,1)
(2/3,0)	(2/3,1/3)	(2/3,2/3)	(2/3,1)
(1,0)	(1,1/3)	(1,2/3)	(1,1)

The formula for linear interpolation is

$$g(x) = f(x_0)\frac{(x_1 - x)}{(x_1 - x_0)} + f(x_1)\frac{(x - x_0)}{(x_1 - x_0)}$$

where $f(x)$ is the exact function for which the values are available only at a discrete set of data points, x_0 and x_1, x is the point at which interpolation is required and $g(x)$ is the interpolated approximation of $f(x)$.

2	1
3	4

2.0000	1.6667	1.3333	1.0000
2.3333	2.2222	2.1111	2.0000
2.6667	2.7778	2.8889	3.0000
3.0000	3.3333	3.6667	4.0000

In bilinear interpolation, linear interpolation is carried out in each of the two orthogonal coordinates of the image.

$$\begin{array}{lll} x(m,n) & & x(m,n+1) \\ & \bullet x(m',n') & \\ x(m+1,n) & & x(m+1,n+1) \end{array}$$

The bilinear interpolated value $x(m', n')$ of a pixel at the location (m', n') of an image is

$$x(m', n') =$$
$$(1 - a)[(1 - b)x(m, n) + bx(m, n + 1)]$$
$$+ a[(1 - b)x(m + 1, n) + bx(m + 1, n + 1)]$$

where $a = m' - m$ and $b = n' - n$ and the distance between pixel locations is one.

$$\mathbf{x} = \begin{bmatrix} 66 & 51 & 44 & 66 \\ 77 & 42 & 52 & 72 \\ 74 & 33 & 28 & 83 \\ 67 & 31 & 27 & 57 \end{bmatrix}$$

The bilinear 7×7 interpolated version of $\mathbf{x}$ is

$$\mathbf{xI} = \begin{bmatrix} 66 & 58.5 & 51 & 47.5 & 44 & 55 & 66 \\ 71.5 & 59 & 46.5 & 47.25 & 48 & 58.5 & 69 \\ 77 & 59.5 & 42 & 47 & 52 & 62 & 72 \\ 75.5 & 56.5 & 37.5 & 38.75 & 40 & 58.75 & 77.5 \\ 74 & 53.5 & 33 & 30.5 & 28 & 55.5 & 83 \\ 70.5 & 51.25 & 32 & 29.75 & 27.5 & 48.75 & 70 \\ 67 & 49 & 31 & 29 & 27 & 42 & 57 \end{bmatrix}$$

The affine transform, in homogenous form, is

$$\begin{bmatrix} m' \\ n' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m \\ n \\ 1 \end{bmatrix}$$

Appropriate values of the transformation matrix are to be used for each type of transformation.

Scaling
$a = 3/4,\ e = 1/2$ then $m' = (3/4)m,\ n' = (1/2)n$

The transformation matrix and its inverse are

$$\begin{bmatrix} 3/4 & 0 & 0 \\ 0 & 1/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 4/3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4×4 image and its 3×2 scaled version using the nearest-neighbor interpolation

$$\begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & 1 & 2 & 3 \\ 4 & 2 & 1 & 3 \\ 2 & 2 & 3 & 1 \end{bmatrix} \quad \begin{bmatrix} 2 & 3 \\ 1 & 2 \\ 2 & 3 \end{bmatrix}$$

Using the backward mapping (inverse of the transformation matrix), we get the middle matrix of the coordinates from that of the output. Using the nearest-neighbor interpolation, we round the coordinates of the middle matrix to get the right matrix. The values corresponding to these coordinates in the input matrix are the output values. For example, $(2, 0)$ in the output matrix corresponds to $(3, 0)$ in the input matrix and the output value is 2.

$$\begin{bmatrix} (0,0) & (0,1) \\ (1,0) & (1,1) \\ (2,0) & (2,1) \end{bmatrix} \begin{bmatrix} (0.0,0) & (0.0,2) \\ (1.3,0) & (1.3,2) \\ (2.7,0) & (2.7,2) \end{bmatrix} \begin{bmatrix} (0,0) & (0,2) \\ (1,0) & (1,2) \\ (3,0) & (3,2) \end{bmatrix}$$

Shear With $a = e = 1$, $c = d = f = 0$, and b taking any value, shear occurs along the m-axis. Let

$$b = 1, \qquad \text{then} \quad m' = m + n,\ n' = n$$

A 4×4 A 4×4 image **x** and its sheared version **y** using the nearest-neighbor interpolation are

$$\mathbf{x} = \begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & 1 & 2 & 3 \\ 4 & 2 & 1 & 3 \\ 2 & 2 & 3 & 1 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 4 & 1 & 3 & 0 \\ 2 & 2 & 2 & 4 \\ 0 & 2 & 1 & 3 \\ 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The maximum value index m' takes is $3+3=6$. Therefore, the size of the output image will be 7×4. The coordinates are as shown in the following matrix.

$$\begin{bmatrix} (0,0) & (0,1) & (0,2) & (0,3) \\ (1,0) & (1,1) & (1,2) & (1,3) \\ (2,0) & (2,1) & (2,2) & (2,3) \\ (3,0) & (3,1) & (3,2) & (3,3) \\ (4,0) & (4,1) & (4,2) & (4,3) \\ (5,0) & (5,1) & (5,2) & (5,3) \\ (6,0) & (6,1) & (6,2) & (6,3) \end{bmatrix}$$

Using the backward mapping (inverse of the transformation matrix), we get the coordinates of the corresponding input matrix as

$$\begin{bmatrix} (0,0) & (0,-1) & (0,-2) & (0,-3) \\ (1,0) & (0,1) & (-1,2) & (-2,3) \\ (2,0) & (1,1) & (0,2) & (-1,3) \\ (3,0) & (2,1) & (1,2) & (0,3) \\ (4,0) & (3,1) & (2,2) & (1,3) \\ (5,0) & (4,1) & (3,2) & (2,3) \\ (6,0) & (5,1) & (4,2) & (3,3) \end{bmatrix}$$

Let

$$d = 1, \qquad \text{then} \quad m' = m,\ n' = n + m$$

The transformation matrix and its inverse are

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

A 4×4 image and its sheared version using the nearest-neighbor interpolation are

$$\begin{bmatrix} 2 & 1 & 3 & 4 \\ 1 & 1 & 2 & 3 \\ 4 & 2 & 1 & 3 \\ 2 & 2 & 3 & 1 \end{bmatrix} \qquad \begin{bmatrix} 2 & 1 & 3 & 4 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 3 & 0 & 0 \\ 0 & 0 & 4 & 2 & 1 & 3 & 0 \\ 0 & 0 & 0 & 2 & 2 & 3 & 1 \end{bmatrix}$$

The maximum value index n' takes is 3+3 = 6. Therefore, the size of the output image will be 4×7. The coordinates are as shown in the matrix.

$$\begin{bmatrix} (0,0) & (0,1) & (0,2) & (0,3) & (0,4) & (0,5) & (0,6) \\ (1,0) & (1,1) & (1,2) & (1,3) & (1,4) & (1,5) & (1,6) \\ (2,0) & (2,1) & (2,2) & (2,3) & (2,4) & (2,5) & (2,6) \\ (3,0) & (3,1) & (3,2) & (3,3) & (3,4) & (3,5) & (3,6) \end{bmatrix}$$

Using the backward mapping we get the corresponding coordinates of the input matrix as

$$\begin{bmatrix} (0,0) & (0,1) & (0,2) & (0,3) & (0,4) & (0,5) & (0,6 \\ (1,-1) & (1,0) & (1,1) & (1,2) & (1,3) & (1,4) & (1,5 \\ (2,-2) & (2,-1) & (2,0) & (2,1) & (2,2) & (2,3) & (2,4 \\ (3,-3) & (3,-2) & (3,-1) & (3,0) & (3,1) & (3,2) & (3,3 \end{bmatrix}$$

Rotation

The transformation and its inverse are

$$\mathbf{T} = \begin{bmatrix} 0.7071 & -0.7071 & 0 \\ 0.7071 & 0.7071 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

$$\mathbf{T}^{-1} = \begin{bmatrix} 0.7071 & 0.7071 & 0 \\ -0.7071 & 0.7071 & 0 \\ 0 & 0 & 1.0000 \end{bmatrix}$$

A 3×3 input $\mathbf{x}$ and its 5×5 rotated version $\mathbf{y}$, (using the nearest-neighbor interpolation), are

$$\mathbf{x} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} & & 3 & & \\ & 2 & 3 & 6 & \\ 1 & 1 & \mathbf{5} & 9 & 9 \\ & 4 & 7 & 8 & \\ & & 7 & & \end{bmatrix}$$

A 2×2 input $\mathbf{x}$ and its 3×3 rotated version $\mathbf{y}$, (using the nearest-neighbor interpolation), are

$$\mathbf{x} = \begin{bmatrix} 1 & 2 \\ 3 & 3 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} & 2 & \\ 1 & \mathbf{4} & 4 \\ & 3 & \end{bmatrix}$$

The size of the rotated image is 3×3. The corresponding output image coordinates are

$$\begin{bmatrix} (-1,1) & (0,1) & (1,1) \\ (-1,0) & (0,0) & (1,0) \\ (-1,-1) & (0,-1) & (1,-1) \end{bmatrix}$$

The backward mapped m and n coordinates, rounded to 2 digits after the decimal point, are

$$\begin{bmatrix} (0.00, 1.41) & (0.71, 0.71) & (1.41, 0.00) \\ (-0.71, 0.71) & (0.00, 0.00) & (0.71, -0.71) \\ (-1.41, 0.00) & (-0.71, -0.71) & (0.00, -1.41) \end{bmatrix}$$

Correlation. 1-D

$$r_{xy}(m) = \sum_{n=-\infty}^{\infty} x(n)y(n-m), \quad m = 0, \pm 1, \pm 2, \ldots$$

2-D

$$r_{xy}(m,n) = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} x(k,l)y(k-m, l-n)$$

The normalized cross correlation $rn_{xy}(m,n)$ (correlation coefficient) of images $x(m,n)$ and $y(m,n)$ is defined as $a/\sqrt{bc}$,

$$a = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} (x(k,l) - \bar{x}_l)(y(k-m,l-n) - \bar{y})$$

$$b = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} (x(k,l) - \bar{x}_l)^2$$

$$c = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} (y(k-m,l-n) - \bar{y})^2$$

Radon Transform Chapter 7

In the Radon transform, an image is represented by its mappings with respect to a set of lines at various angles represented by polar coordinates. The values at various polar coordinates are the transform coefficients. When an image is represented in Radon transform form, we are able to form the image of the interior of an object with out intrusion. In its implementation, we use the 1-D DFT and interpolation operations.

slope-intercept form of a line

$$y = mx + c$$

normal form of a line

$$x\cos(\theta) + y\sin(\theta) = s$$

where s is always positive and $0 \leq \theta < 360°$. A line is expressed in terms of its perpendicular distance s from the line to the origin and the angle θ subtended between the perpendicular line and the x-axis.

Given a line

$$Ax + By + C = 0,$$

its normal form is

$$x\cos(\theta) + y\sin(\theta) = s,$$

where

$$\cos(\theta) = \pm\frac{A}{\sqrt{A^2 + B^2}}, \qquad \sin(\theta) = \pm\frac{B}{\sqrt{A^2 + B^2}},$$

with the condition that s should be always positive.

$$x + y - 2\sqrt{2} = 0,$$

let us put it in the normal form of a line. Shift the constant term to the other side and ensure that it is positive. We get

$$x + y = 2\sqrt{2}$$

Divide both sides by the square root of the sum of the squares of the constants associated with x and y. Since $\sqrt{1^2 + 1^2} = \sqrt{2}$, we get

$$\frac{1}{\sqrt{2}}x+\frac{1}{\sqrt{2}}y=2 \quad \text{or} \quad x\cos(45^\circ)+y\sin(45^\circ)=2$$

with $\theta = 45^\circ$ and $s = 2$. Parameters s and θ determine a specific line in the 2-D plane. The length is also given by $s = (\sqrt{\sqrt{2})^2 + (\sqrt{2})^2} = 2$. The x-axis and y-axis intercepts are the same, $2\sqrt{2}$. The coordinates of the point of intersection of the line and the normal is

$$x = s\cos(\theta) = \frac{2}{\sqrt{2}} = \sqrt{2},\ y = s\sin(\theta) = \frac{2}{\sqrt{2}} = \sqrt{2}$$

Radon transform $R(s,\theta)$ of $f(x,y)$

$$R(s,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos(\theta)+y\sin(\theta)-s)dx\,dy$$

$$-\infty < s < \infty,\ 0 \leq \theta < \pi$$

Using the coordinate transformation, the relations between the coordinate systems (x,y) and (s,n') (rotated) are given by

$$\begin{aligned} s &= x\cos(\theta) + y\sin(\theta) \\ n' &= -x\sin(\theta) + y\cos(\theta) \\ x &= s\cos(\theta) - n'\sin(\theta) \\ y &= s\sin(\theta) + n'\cos(\theta) \end{aligned}$$

In the rotated coordinate system (s, n'),

$$R(s,\theta)=\int_{-\infty}^{\infty} f(s\cos(\theta)-n'\sin(\theta), s\sin(\theta)+n'\cos(\theta))\, dn'$$

The back-projection of $R(s,\theta)$ is defined as

$$\widehat{f}(x,y) = \int_0^{\pi} R(x\cos(\theta)+y\sin(\theta),\theta)\, d\theta$$

where $\widehat{f}(x,y)$ is a blurred version of $f(x,y)$.

$$f(x,y) = \begin{cases} 1 & \text{for } x^2 + y^2 \leq r^2 \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} R(s,\theta) &= 2\int_0^{\sqrt{r^2-s^2}} f(s,n')\,dn' = 2\int_0^{\sqrt{r^2-s^2}} 1\,dn' \\ &= \begin{cases} 2\sqrt{r^2-s^2} & \text{for } |s| \leq r \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

Radon transform of a 2-D delayed impulse $f(x, y) = \delta(x - x_0, y - y_0)$

$$\begin{aligned} R(s,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x - x_0, y - y_0) \\ &\quad \delta(x\cos(\theta) + y\sin(\theta) - s)dx\,dy \\ &= \delta(x_0\cos(\theta) + y_0\sin(\theta) - s) \end{aligned}$$

As the strength of the impulse is concentrated only when its argument becomes zero, the Radon transform is given by
$x_0\cos(\theta)+y_0\sin(\theta)-s = 0$ or $s = x_0\cos(\theta)+y_0\sin(\theta)$

The point (x_0, y_0), where the impulse occurs in the spatial-domain, can be described in polar coordinates as

$$\begin{aligned} x_0 &= r\cos(\phi),\ y_0 = r\sin(\phi) \text{ and } r = \sqrt{x_0^2 + y_0^2}, \\ & \tan(\phi) = \frac{y_0}{x_0},\ x_0 \neq 0 \end{aligned}$$

The Radon transform, in terms of r and ϕ, is given by

$$\begin{aligned} s &= r\cos(\phi)\cos(\theta) + r\sin(\phi)\sin(\theta) \\ &= r\cos(\phi - \theta) = r\cos(\theta - \phi) \end{aligned}$$

As the impulse function is part of the Radon transform definition, the properties of the Radon transform are mostly due to the properties of the impulse function.

Property	Spatial domain $f(x, y)$
Linearity	$af_1(x, y) + bf_2(x, y)$
Symmetry	$f(x, y)$
Scaling	$f(ax, ay)$
Shift	$f(x - x_0, y - y_0)$
Periodicity	$f(x, y)$
Rotation	$f(r, \phi + \theta_0)$
Mass conservation	$M = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) dx dy$
Space limitedness	$f(x, y) = 0, \lvert x \rvert > 0.5D, \lvert y \rvert > 0.5$

Properties

$$R(s,\theta) = R(-s,\theta \pm 180^\circ)$$

$$\begin{aligned} f(x,y) &\leftrightarrow R(s,\theta) \rightarrow f(x-x_0, y-y_0) \\ &\leftrightarrow R(s - x_0\cos(\theta) - y_0\sin(\theta), \theta) \end{aligned}$$

$$f(x,y) \leftrightarrow R(s,\theta) \rightarrow f(kx,ky) \leftrightarrow \frac{1}{|k|}R(ks,\theta), \quad k \neq 0$$

The Radon transform is approximated for a $N \times N$ image $x(m,n)$ as

$$R(s,\theta) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m,n)\delta(m\cos(\theta)+n\sin(\theta)-s)$$

In the rotated coordinate system (s,n')

$$R(s,\theta) = \sum_{n'} x(s\cos(\theta)-n'\sin(\theta), s\sin(\theta)+n'\cos(\theta))$$

The back-projection is

$$\widehat{x}(m,n) = \sum_{\theta} R(m\cos(\theta)+n\sin(\theta),\theta)$$

$\widehat{x}(m,n)$ is a discrete and blurred version of $x(m,n)$.

The bottom left corner is the origin. Find the Radon transform of the 2×2 image

$$x(m,n) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The two points of the line are (0,1) and (1,0). The slope $m = \frac{1-0}{0-1} = -1$. Equation of the line

$$\frac{y-0}{x-1} = -1 \quad \text{or} \quad y = -x + 1 = mx + b$$

The slope corresponds to 135°. Now,

$$s = b \sin(135) = \frac{1}{\sqrt{2}}$$

In this example, the function $x(m,n)$ is a single line. The sum of the function along this line is 2. The parameters of the normal to the line are $s = \frac{1}{\sqrt{2}}$ and $\theta = 45°$. Therefore, the Radon transform of $x(m,n)$ is

$$R\left(\frac{1}{\sqrt{2}}, 45°\right) = 2$$

At all other distances, the Radon transform is 0.

Find the Radon transform of the 8×8 image

$$x(m,n) = \sin\left(\frac{2\pi}{8}n\right)$$

The 8-point 1-D DFT spectrum with $\theta = 90^\circ$ is

$$\{0, -j32, 0, 0, 0, 0, 0, j32\}$$

The IDFT of this spectrum

$$\frac{8}{\sqrt{2}}\{0, 1, \sqrt{2}, 1, 0, -1, -\sqrt{2}, -1\}$$

is the set of Radon transform coefficients

$$\{R(0, 0^\circ), R(1, 0^\circ), R(2, 0^\circ), R(3, 0^\circ),$$

$$R(-4, 0^\circ), R(-3, 0^\circ), R(-2, 0^\circ), R(-1, 0^\circ)\}$$

The 1-D FT of a projection $R(s,\theta)$ is

$$R(j\omega,\theta) = \int_{-\infty}^{\infty} R(s,\theta)e^{-j\omega s}ds$$

Substituting for $R(s,\theta)$, we get

$$\begin{aligned} R(j\omega,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(s\cos(\theta) - n'\sin(\theta), \\ &\quad s\sin(\theta) + n'\cos(\theta))e^{-j\omega s}\,dsdn' \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j\omega(x\cos(\theta)+y\sin(\theta))}dx\,dy \end{aligned}$$

Letting $\omega_1 = \omega\cos(\theta)$ and $\omega_2 = \omega\sin(\theta)$,

$$\begin{aligned} R(j\omega,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-j(\omega_1 x+\omega_2 y)}dx\,dy \\ &= F(j\omega_1, j\omega_2)\,|_{\omega_1=\omega\cos(\theta),\omega_2=\omega\sin(\theta)} \end{aligned}$$

The 2-D IFT of the FT, $F(j\omega_1, j\omega_2)$, of an image $f(x, y)$ is given by

$$f(x,y) = \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(j\omega_1, j\omega_2) e^{j(\omega_1 x + \omega_2 y)} d\omega_1 d\omega_2$$

Letting $\omega_1 = \omega\cos(\theta)$, $\omega_2 = \omega\sin(\theta)$, the differentials $d\omega_1\, d\omega_2$ become $\omega d\omega\, d\theta$. Then, using the polar coordinates and the Fourier-slice theorem, we get

$$f(x,y) = \frac{1}{4\pi^2}\int_{0}^{2\pi}\int_{0}^{\infty} F(j\omega\cos(\theta), j\omega\sin(\theta)) e^{j\omega(x\cos(\theta)+y\sin(\theta))} \omega d\omega\, d\theta$$

$$
\begin{aligned}
&= \frac{1}{4\pi^2}\int_0^{2\pi}\int_0^{\infty} R(j\omega,\theta)e^{j\omega(x\cos(\theta)+y\sin(\theta))}\omega d\omega\, d\theta \\
&= \frac{1}{4\pi^2}\int_0^{\pi}\int_{-\infty}^{\infty} R(j\omega,\theta)e^{j\omega(x\cos(\theta)+y\sin(\theta))}|\omega| d\omega\, d\theta \\
&= \frac{1}{4\pi^2}\int_0^{\pi}\left(\int_{-\infty}^{\infty}|\omega| R(j\omega,\theta)e^{j\omega s} ds\right)\Big|_{s=(x\cos(\theta)+y\sin(\theta))}
\end{aligned}
$$

A procedure for computing the Radon transform is as follows.

1. Compute the 2-D DFT of the image.
2. Interpolate the spectral values to get the spectrum on polar coordinates, for all angles of interest.
3. Compute the 1-D IDFT of the spectral values at all angles to get the Radon transform.

A procedure for computing the inverse RT

1. Compute the 1-D DFT of each of the projections of the image.
2. Multiply each DFT by the ramp filter.
3. Compute the 1-D IDFT of the spectral values at all angles to get the filtered Radon transform.
4. Obtain the filtered back-projected image using the back-projection definition, Equation (7.14), for each angle of projection.
5. Sum all the filtered backprojected images to reconstruct the image.

The steps of the Hough transform algorithm:

1. Select a set of points for the parameters (s, θ).
2. For each value of θ, compute the corresponding value of s using Equation (7.2) for all nonzero pixels.
3. Create an accumulator matrix which accumulates the number of occurrences of each pair of (s, θ), as all the pixels with value 1 in the input image are analyzed.
4. Find the accumulator values those are greater than a given threshold.

$$x(m,n) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$(s,\theta) = \begin{bmatrix} (0,0) & (0,45) & (0,90) & (0,135) \\ (1,0) & (1,45) & (1,90) & (1,135) \\ (2,0) & (2,45) & (2,90) & (2,135) \\ (3,0) & (3,45) & (3,90) & (3,135) \\ (4,0) & (4,45) & (4,90) & (4,135) \end{bmatrix}$$

$$acc(m,n) = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 4 & 2 & 1 & 2 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The process is similar to linear convolution and correlation, except that logical operations AND (denoted by &), OR (denoted by |) and NOT (denoted by ~) are used (a logical neighborhood operation) instead of arithmetic operations. Pixels are added to an object or deleted from it. Border extension has to be defined and windows (structuring elements) may have to be rotated by 180°.

The dilation of the binary image $x(m,n)$ and the window or mask $h(m,n)$ is defined as

$$\begin{aligned} y(m,n) &= |_k\,|_l(h(k,l)\ \&\ x(m-k,n-l)) \\ &= x(m,n) \oplus h(m,n),\ \forall(k,l)h(k,l) = 1 \end{aligned}$$

where m and n vary over the dimensions of the image and k and l vary over the dimensions of the structuring element.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$h(m,n) = \begin{bmatrix} 1 & 1 & 0 \\ 0 & \mathbf{1} & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad h(-m,-n) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & \mathbf{1} & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

The erosion of the binary image $x(m,n)$ and the structuring element $h(m,n)$ is defined as

$$\begin{aligned} y(m,n) &= \&_k \&_l (h(k,l) \,\&\, x(m+k, n+l)) \\ &= x(m,n) \ominus h(m,n),\ \forall (k,l) h(k,l) = 1 \end{aligned}$$

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & \mathbf{1} & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$
\begin{aligned}
x(m,n) \ominus h(m,n) &= \tilde{z}(m,n), \\
z(m,n) &= \tilde{x}(m,n) \oplus h(-m,-n) \\
x(m,n) \oplus h(m,n) &= \tilde{z}(m,n), \\
z(m,n) &= \tilde{x}(m,n) \ominus h(-m,-n)
\end{aligned}
$$

Dilation preceded by erosion is called the opening operation, defined as

$$\begin{aligned} y(m,n) &= x(m,n) \circ h(m,n) \\ &= (x(m,n) \ominus h(m,n)) \oplus h(m,n) \end{aligned}$$

After erosion and, then, dilation yields,

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 1 & \mathbf{1} & 0 \\ 0 & 0 & 0 & \mathbf{1} & 1 & 1 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \end{bmatrix}$$

- All the 1s in the object those are completely covered by the structuring element are preserved.
- All the 1s those can be reached by the structuring element, when it is placed at the 1s obtained in the erosion operation are also preserved.

Dilation followed by erosion is called the closing operation, defined as

$$\begin{aligned} y(m,n) &= x(m,n) \bullet h(m,n) \\ &= (x(m,n) \oplus h(m,n)) \ominus h(m,n) \end{aligned}$$

After dilation and, then, erosion yields,

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 1 & 1 & \mathbf{1} & 0 & 0 \\ 0 & \mathbf{1} & 1 & 1 & 1 & 1 & \mathbf{1} & 0 \\ 0 & 0 & \mathbf{1} & 1 & 1 & 1 & \mathbf{1} & 0 \\ 0 & 0 & 0 & \mathbf{1} & 1 & 1 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & \mathbf{1} & 0 & 0 \end{bmatrix}$$

hit-and-miss transformation is

$$\begin{aligned} x(m,n) \circledast h(m,n) &= (x(m,n) \ominus h_h(m,n)) \\ &\& \quad (\tilde{x}(m,n) \ominus h_{ms}(m,n)) \end{aligned}$$

$$h_h(m,n) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & \mathbf{0} & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad h_{ms}(m,n) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \mathbf{0} & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

input $x(m,n)$ and its complement $\tilde{x}(m,n)$ are

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

$$y(m,n) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Thinning

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Thickening of an image can be carried out by:

- Complement the input image $x(m, n)$ to get $\tilde{x}(m, n)$.
- Thin $\tilde{x}(m, n)$ to get $\tilde{y}(m, n)$.
- Complement $\tilde{y}(m, n)$ to get the thickened input image $y(m, n)$.

Additional processing may be required after each iteration, in case of extraneous pixels appearing in the output.

Skeletons. Let the input image be $x(m,n)$ and the distance of the pixels from the region marked with 1s in the complement of $x(m,n)$, $\tilde{x}(m,n)$, be $D(m,n)$.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \\ 0 & 5 & 10 & 10 & 10 & 10 & 5 & 0 \\ 0 & 5 & 10 & 15 & 15 & 10 & 5 & 0 \\ 0 & 5 & 10 & 15 & 15 & 10 & 5 & 0 \\ 0 & 5 & 10 & 10 & 10 & 10 & 5 & 0 \\ 0 & 5 & 5 & 5 & 5 & 5 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Distances in $D(m,n)$ are scaled by 5.

We start with a matrix $skel(m, n)$, of the same size as the input image, with all zero entries. Each pixel in this matrix is replaced by a 1, if the corresponding value in $D(m, n)$ is greater or equal to the largest value of its 4 nearest neighbors. For example, consider the neighborhood of $D(1, 1)$.

$$\begin{bmatrix} & 0 & \\ 0 & 5 & 5 \\ & 5 & \end{bmatrix}$$

As $D(1, 1) = 5$ is greater or equal to the largest value of its 4 nearest neighbors, $skel(1, 1) = 1$.

The skeleton of the input image is

$$skel(m,n) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

If we erode an image by a structuring element by one iteration, then the pixels in the border of the objects are set to zero, leaving the interior pixels unchanged. Now, if we subtract the output of erosion from the input, an image with object boundary is obtained.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The output of erosion and the extracted border are, respectively,

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Border pixels are assumed to be 0s.

Given a region defined by its boundary and the location of a pixel within it, the interior of the region is to be filled. Let the image be $x(m,n)$. The algorithm is defined by

$x_l(m,n) = (x_{l-1}(m,n) \oplus h(m,n)) \& \tilde{x}(m,n),\ l = 2,3,..$

where

$$h(m,n) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

and $x_1(m,n)$ is a matrix of the same size as $x(m,n)$ with all entries zero except a 1 at the given location inside the region.

Repeatedly, we keep dilating the current $x_l(m, n)$ with $h(m, n)$ and AND with the complement of the input image until there is no difference between two consecutive versions of $x_l(m, n)$. Without the AND operation, the dilation operation is uncontrolled and will fill up the entire image.

Let $x(m, n)$ be the given image and $(3, 3)$ is the given starting location. Then,

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The OR of $x_5(m, n)$ and $x(m, n)$ gives the filled region.

The dilation of the gray-level image $x(m,n)$ and the mask $h(m,n)$ is defined as

$$y(m,n) = \max_{k,l}\{x(m-k,n-l)\} = x(m,n) \oplus h(m,n)$$

Consider the 8×8 image and the 3×3 mask.

$$\begin{bmatrix} 88 & 100 & 104 & 101 & 114 & 110 & 110 & 107 \\ 92 & 102 & 104 & 107 & 112 & 110 & 104 & 92 \\ 103 & 105 & 111 & 112 & 114 & 108 & 86 & 39 \\ 106 & 107 & 112 & 113 & 107 & 73 & 26 & 25 \\ 111 & 114 & 14 & 104 & 64 & 23 & 25 & 28 \\ 117 & 115 & 97 & 45 & 15 & 13 & 23 & 29 \\ 119 & 93 & 32 & 0 & 15 & 11 & 19 & 23 \\ 88 & 15 & 0 & 2 & 10 & 13 & 11 & 15 \end{bmatrix}$$

Assuming that the border pixels are replicated, the output of the dilation operation is

$$\begin{bmatrix} 102 & 104 & 107 & 114 & 114 & 114 & 110 & 110 \\ 105 & 111 & 112 & 114 & 114 & 114 & 110 & 110 \\ 107 & 112 & 113 & 114 & 114 & 114 & 110 & 104 \\ 114 & 114 & 114 & 114 & 114 & 114 & 108 & 86 \\ 117 & 117 & 115 & 113 & 113 & 107 & 73 & 29 \\ 119 & 119 & 115 & 104 & 104 & 64 & 29 & 29 \\ 119 & 119 & 115 & 97 & 45 & 23 & 29 & 29 \\ 119 & 119 & 93 & 32 & 15 & 19 & 23 & 23 \end{bmatrix}$$

The erosion of the gray-level image $x(m,n)$ and the mask $h(m,n)$ is defined as

$$y(m,n) = \min_{k,l}\{x(m+k,n+l)\} = x(m,n) \ominus h(m,n)$$

The output of the erosion operation is

$$\begin{bmatrix} 88 & 88 & 100 & 101 & 101 & 104 & 92 & 92 \\ 88 & 88 & 100 & 101 & 101 & 86 & 39 & 39 \\ 92 & 92 & 102 & 104 & 73 & 26 & 25 & 25 \\ 103 & 14 & 14 & 14 & 23 & 23 & 23 & 25 \\ 106 & 14 & 14 & 14 & 13 & 13 & 13 & 23 \\ 93 & 14 & 0 & 0 & 0 & 11 & 11 & 19 \\ 15 & 0 & 0 & 0 & 0 & 10 & 11 & 11 \\ 15 & 0 & 0 & 0 & 0 & 10 & 11 & 11 \end{bmatrix}$$

$$x(m,n) \circ h(m,n) = (x(m,n) \ominus h(m,n)) \oplus h(m,n)$$
$$x(m,n) \bullet h(m,n) = (x(m,n) \oplus h(m,n)) \ominus h(m,n)$$

The output of the opening operation is

$$\begin{bmatrix} 88 & 100 & 101 & 101 & 104 & 104 & 104 & 92 \\ 92 & 102 & 104 & 104 & 104 & 104 & 104 & 92 \\ 103 & 103 & 104 & 104 & 104 & 101 & 86 & 39 \\ 106 & 106 & 104 & 104 & 104 & 73 & 26 & 25 \\ 106 & 106 & 14 & 23 & 23 & 23 & 25 & 25 \\ 106 & 106 & 14 & 14 & 14 & 13 & 23 & 23 \\ 93 & 93 & 14 & 0 & 11 & 11 & 19 & 19 \\ 15 & 15 & 0 & 0 & 10 & 11 & 11 & 11 \end{bmatrix}$$

The output of the closing operation is

$$\begin{bmatrix} 102 & 102 & 104 & 107 & 114 & 110 & 110 & 110 \\ 102 & 102 & 104 & 107 & 114 & 110 & 104 & 104 \\ 105 & 105 & 111 & 112 & 114 & 108 & 86 & 86 \\ 107 & 107 & 112 & 113 & 107 & 73 & 29 & 29 \\ 114 & 114 & 104 & 104 & 64 & 29 & 29 & 29 \\ 117 & 115 & 97 & 45 & 23 & 23 & 23 & 29 \\ 119 & 93 & 32 & 15 & 15 & 15 & 19 & 23 \\ 119 & 93 & 32 & 15 & 15 & 15 & 19 & 23 \end{bmatrix}$$

Edge detection Chapter 9

An edge is a line of interaction of two surfaces. Edge pixels are characterized by the abrupt change of intensity with the neighboring pixels. The boundaries of objects in an image are identified by edges. Edges are useful for tasks such as segmentation, registration and object identification. Edges provide a compact representation of objects than pixels. Edges are amplitude discontinuities between regions of an image. In the frequency-domain, an edge is characterized by the high frequency components of the spectrum of the image. Basically, edge detection constitutes highpass filtering.

Operator	Grad, n-direction	Grad, m-direction
Prewitt	$\begin{bmatrix} -1 & 0 & 1 \\ -1 & \mathbf{0} & 1 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ 0 & \mathbf{0} & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\begin{bmatrix} -1 & 0 & 1 \\ -2 & \mathbf{0} & 2 \\ -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & \mathbf{0} & 0 \\ 1 & 2 & 1 \end{bmatrix}$

The Prewitt operator averages intensity changes over six intervals. The Sobel operator gives twice the weight to the central pixels. The values of these masks are to be convolved by the corresponding pixel values of the image pointwise and divided by the sum of the magnitudes of the elements of the mask. These operators compute the differences (highpass filtering in one direction) of local sums (lowpass filtering in another direction), which has the effect of reducing the noise.

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$$

six sample neighborhoods of the image $x(m, n)$.

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

Using the Sobel operator,

	1	2	3	4	5	6
gm	4	0	3	4	-2	-3
gn	0	4	-3	-2	-4	-3
θ	0°	90°	−45°	−26°	63°	45°

Canny edge detection algorithm is based on three objectives.

1. The edges found should be true edges and all the edges should be found. The probability of finding a good edge should be maximized and that of a false edge should be minimized. Achieving this objective requires a high signal-to-noise ratio.
2. The location of the edge found must be as close as possible to the exact location.
3. There should be no multiple responses for a single edge.

Finding the edges in an image using the Canny edge detection algorithm consists of the following four basic steps.

1. The input image is smoothed by a Gaussian filter to improve the SNR.
2. The gradient magnitude and angle images are formed using a gradient filter.
3. The edge image is thinned by using non-maximum suppression of the gradient image.
4. By using two thresholds and connectivity constraint, the final edge image is formed.

The steps involved in implementing the LoG filter are:

1. Reduce the noise by filtering the image with the Gaussian lowpass filter.
2. Apply the Laplacian to the smoothed image.
3. Detect the zero-crossings to find the edge image.

$$g(x,y) = e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Taking the partial derivative with respect to x and y, we get

$$(-\frac{x}{\sigma^2})e^{-\frac{(x^2+y^2)}{2\sigma^2}} + (-\frac{y}{\sigma^2})e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Taking the partial derivative with respect to x and y again, we get $\nabla^2 g(x,y) =$

$$(\frac{x^2}{\sigma^2} - 1)(\frac{1}{\sigma^2})e^{-\frac{(x^2+y^2)}{2\sigma^2}} + (\frac{y^2}{\sigma^2} - 1)(\frac{1}{\sigma^2})e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

$$= \left(\frac{1}{\sigma^2}\right)\left(\frac{(x^2+y^2)}{\sigma^2} - 2\right)e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Segmentation Chapter 10

A region can be defined by its interior part or border (edge). The homogeneity P of a region R is defined as

$$P(R) = \begin{cases} 1, & \text{if } f(R) \in H \\ 0, & \text{otherwise} \end{cases}$$

where f is function defining the homogeneity and H is the predefined range of values of f. The function f can be defined in any suitable way. It could be the standard deviation of the region, the mean, the difference between the largest and smallest values of pixels, co-occurrence matrices or a gray level threshold.

Segmentation of an image is its partitioning into a set of N connected regions $R(n), n = 0, 1, \ldots, N-1$. Points to be noted are:

1. The sum of all the regions is exactly equal to the image.
2. A pixel of the image belongs to only one of the regions.
3. The predicate of a region holds for all its pixels.
4. The predicates of adjacent regions must be different.

Edge detection, thresholding, region growing and splitting and watershed segmentation are the basic approaches of segmentation.

Point Detection

The mask, which is one of the versions of the Laplacian masks presented in Chapter 2 multiplied by -1, for point detection is

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

The mask is applied to the image and the output is thresholded to detect the isolated points (segmentation of points). Note that, for edge detection, the Laplacian response is analyzed for zero crossings.

Line Detection

The masks for line detection at 4 directions (E-W, NW-SE, N-S, NE-SW) are, respectively,

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & \mathbf{2} & 2 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 2 & -1 & -1 \\ -1 & \mathbf{2} & -1 \\ -1 & -1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 2 & -1 \\ -1 & \mathbf{2} & -1 \\ -1 & 2 & -1 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & 2 \\ -1 & \mathbf{2} & -1 \\ 2 & -1 & -1 \end{bmatrix}$$

Threshold-based Segmentation

If we are able to determine the threshold between an object and the background, then the region corresponding to the object is labeled by selecting the pixels those are in the object range of gray levels. The thresholding process of an image $x(m,n)$, yielding its segmented version $R(m,n)$, is given by

$$R(m,n) = \begin{cases} o, & \text{for } x(m,n) \geq T \\ b, & \text{otherwise} \end{cases}$$

where T is the threshold and o and b represent the object and the background.

$$R(m,n) = \begin{cases} o1, & \text{for } x(m,n) > T1 \\ o2, & \text{for } T0 < x(m,n) \leq T1 \\ b, & \text{otherwise} \end{cases}$$

$$\begin{bmatrix} 185 & 182 & 45 & 2 \\ 188 & 140 & 10 & 5 \\ 189 & 74 & 2 & 7 \\ 164 & 21 & 5 & 6 \end{bmatrix}$$

The initial threshold is the average 76.5625.

$$\begin{bmatrix} 185 & 182 \\ 188 & 140 \\ 189 & \\ 164 & \end{bmatrix} \quad \begin{bmatrix} 45 & 2 \\ 10 & 5 \\ 74 & 2 & 7 \\ 21 & 5 & 6 \end{bmatrix}$$

The first group has values greater than 76.5625 with the average 174.6667. The second group has the remaining values with the average 17.7. The average of the values, 96.1833, is the new threshold value.

In the next iteration, we get the same value and the algorithm terminates. The segmented image is

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Thresholding by Otsu's Method

This method maximizes the between-class variance $\sigma_b^2(k)$ and it is based on the histogram of the image. Let the range of gray levels of the image be from 0 to $L-1$ and the normalized histogram values be $hn(k), k = 0, 1, \ldots L-1$. Let the threshold T be k, $k = 0, 1, \ldots, L-1$. Then, the pixels are placed in two groups $g1$ and $g2$ with the first group consists of pixels with gray levels in the range from 0 to k and the other group consists of pixels with gray levels in the range from $k+1$ to $L-1$.

$$\begin{aligned}\sigma_b^2(k) &= hc1(k)(ha1(k) - ha(L-1))^2 \\ &\quad + hc2(k)(ha2(k) - ha(L-1))^2\end{aligned}$$

where the normalized cumulative histograms and the average intensities are defined as

$$hc1(k) = \sum_{i=0}^{k} hn(i), hc2(k) = \sum_{i=k+1}^{L-1} hn(i) = 1 - hc1(k)$$

$$ha1(k) = \frac{1}{hc1(k)} \sum_{i=0}^{k} (i)\, hn(i), ha2(k) = \frac{1}{hc2(k)} \sum_{i=k+1}^{L-1} (i)\, hn(i)$$

$$ha(k) = \sum_{i=0}^{k} (i)\, hn(i), \quad ha(L-1) = \sum_{i=0}^{L-1} (i)\, hn(i)$$

$$hc2(k) = 1 - hc1(k), ha1(k) = \frac{ha(k)}{hc1(k)},$$

$ha2(k) = \frac{ha(L-1)-ha(k)}{1-hc1(k)}$, we get $\sigma_b^2(k)$ as

$$\sigma_b^2(k) = \frac{(ha(L-1)hc1(k) - ha(k))^2}{hc1(k)(1 - hc1(k))}$$

$$\begin{bmatrix} 2 & 7 & 6 & 6 \\ 5 & 6 & 5 & 5 \\ 6 & 5 & 5 & 6 \\ 7 & 6 & 4 & 5 \end{bmatrix}$$

k	0	1	2	3	4	5	6	7
$hn(k)$	0	0	0.06	0	0.06	0.37	0.37	0.12
$hc1(k)$	0	0	0.06	0.06	0.12	0.5	0.87	1
$ha(k)$	0	0	0.12	0.12	0.37	2.25	4.5	5.37
$\sigma_b^2(k)$	0	0	0.75	0.75	0.80	0.76	0.37	0

Since it is a 3-bit image, the gray level range, shown in the first row, varies from 0 to 7. For example, $\sigma_b^2(2)$

$$\frac{((5.375)(0.0625) - 0.125)^2}{(0.0625(1 - 0.0625))} = 0.7594$$

The index of the maximum variance 0.8058 is 4 and it is the optimum threshold value T. If the maximum variance occurs more than once, then the average value of the indices is taken as the threshold. The measure of the separability is given by

$$\frac{0.8058}{\sum_{k=0}^{7}(k-5.375)^2 hn(k)} = 0.5928$$

This measure varies from 0 (for an image with a single gray level) to 1 (for a 2-valued image with gray levels 0 and $L-1$ only).

Region-based Segmentation

For a pixel to be a part of a region, it should have the attributes characterizing the region and also should meet some connectivity constraints. Given a set of pixels, if we can identify at the least one 4-connected path between any pair of pixels, then it is a 4-connected region.

Typical attributes characterizing a region are:

- The average of the gray values of the pixels in a region is significantly different from that of the image.
- The standard deviation of the gray level values of the pixels in the region is within a distinct range.
- Pixels in the region exhibit distinct textural properties.

Region Growing

In the region growing method of segmentation, we start with a pixel, called a seed pixel and start checking the similarity of the attributes of the pixels and the connectivity. All the pixels satisfying the criteria are collected and they form the region. The process is continued until all the pixels of the image are assigned to some region.

Consider the 8×8 image $x(m, n)$.

$$\begin{bmatrix} 179 & 179 & 183 & 180 & 185 & 183 & 182 & 175 \\ 175 & 179 & 185 & 179 & 181 & 179 & 177 & 173 \\ 180 & 183 & 181 & 170 & 181 & \mathbf{176} & 174 & 174 \\ 181 & 181 & 182 & 180 & 174 & 176 & 179 & 175 \\ 185 & 184 & 185 & 177 & 172 & 176 & 174 & 170 \\ 184 & 184 & 182 & 182 & 175 & 172 & 165 & 167 \\ 177 & 185 & 181 & 175 & 171 & 166 & 165 & 169 \\ 179 & 184 & 177 & 173 & 170 & 171 & 171 & 172 \end{bmatrix}$$

Let the seed pixel be $x(2, 5) = 176$, shown in boldface. Let us use the 4-connectivity and the region is to be made of pixels with gray levels less than or equal to 176.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

After two more iterations, the final region map is obtained.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

With 8-connectivity, the final region matrix is

$$R(m,n) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Region Splitting and Merging

While region growing is a bottom-up approach, region splitting and merging is a top-down approach. The criteria of segmentation does not hold for the whole image and we divide the image into subimages. This division is carried out recursively until the criteria is met and the merging of all these subimages as required in the region being formed. The data structure most suitable for this algorithm is quadtree. This is a tree in which each node, except the leaves, has four children. Each segmented region is represented by a leaf.

The minimum distance between each pixel to its nearest pixel in the connected region defined by the 1s is

$$\begin{bmatrix} 4.47 & 3.60 & 2.82 & 2.23 & 1.41 & 1.00 & 1.41 & 2.23 \\ 4.12 & 3.16 & 2.23 & 1.41 & 1.00 & 0 & 1.00 & 2.00 \\ 4.00 & 3.00 & 2.00 & 1.00 & 0 & 0 & 1.00 & 1.41 \\ 3.60 & 2.82 & 2.23 & 1.41 & 1.00 & 0 & 0 & 1.00 \\ 3.16 & 2.23 & 1.41 & 1.00 & 1.00 & 0 & 0 & 1.00 \\ 3.00 & 2.00 & 1.00 & 0 & 0 & 0 & 1.00 & 1.41 \\ 3.16 & 2.23 & 1.41 & 1.00 & 1.00 & 0 & 1.00 & 2.00 \\ 3.60 & 2.82 & 2.23 & 2.00 & 1.41 & 1.00 & 1.41 & 2.23 \end{bmatrix}$$

The algorithm is essentially passing 2 masks over the image. The forward and backward masks are

$$h_f(m,n) = \begin{bmatrix} \infty & 1 & \infty \\ 1 & 0 & \infty \\ \infty & \infty & \infty \end{bmatrix}$$

$$h_b(m,n) = \begin{bmatrix} \infty & \infty & \infty \\ \infty & 0 & 1 \\ \infty & 1 & \infty \end{bmatrix}$$

Given image is sufficiently zero-padded and all the zero-valued pixels are assigned a value of ∞ and the pixels with value 1 are assigned the value 0, giving a initial distance matrix DS as

$$DS(m,n)=\begin{bmatrix} \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 & 0 & \infty & \infty \\ \infty & \infty & \infty & \infty & 0 & 0 & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

Now, the forward mask is passed over the matrix, starting from top left corner of the image. The mask is moved from left to right and top to bottom. At each pixel, the pixels of the mask are added with the corresponding pixels of the image. The minimum value of this set replaces current value in the DS matrix. At each pixel location, the updated values of the DS matrix are to be used in the computation, not those of the initial DS matrix.

For example, the pixels in the third row are updated as

$$\begin{bmatrix} \infty & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} \infty & \infty \\ \infty & 0 \end{bmatrix} = \begin{bmatrix} \infty & \infty \\ \infty & 0 \end{bmatrix}$$

$$\begin{bmatrix} \infty & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} \infty & \infty \\ 0 & \infty \end{bmatrix} = \begin{bmatrix} \infty & \infty \\ 1 & \infty \end{bmatrix}$$

$$\begin{bmatrix} \infty & 1 \\ 1 & 0 \end{bmatrix} + \begin{bmatrix} \infty & \infty \\ 1 & \infty \end{bmatrix} = \begin{bmatrix} \infty & \infty \\ 2 & \infty \end{bmatrix}$$

The result of the first pass is

$$DS(m,n)=\begin{bmatrix} \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & 0 & 1 & 2 & \infty \\ \infty & \infty & \infty & \infty & \infty & 0 & 0 & 1 & 2 & \infty \\ \infty & \infty & \infty & \infty & \infty & 1 & 0 & 0 & 1 & \infty \\ \infty & \infty & \infty & \infty & \infty & 2 & 0 & 0 & 1 & \infty \\ \infty & \infty & \infty & \infty & 0 & 0 & 0 & 1 & 2 & \infty \\ \infty & \infty & \infty & \infty & 1 & 1 & 0 & 1 & 2 & \infty \\ \infty & \infty & \infty & \infty & 2 & 2 & 1 & 2 & 3 & \infty \\ \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \end{bmatrix}$$

Now, the backward mask is passed over this matrix, starting from bottom right corner of the image. The mask is moved from right to left and bottom to top.

At each pixel, the pixels of the mask are added with the corresponding pixels of the image. The minimum value of this set replaces current value in the DS matrix. The result of the second pass is

$$DS(m,n) = \begin{bmatrix} 6 & 5 & 4 & 3 & 2 & 1 & 2 & 3 \\ 5 & 4 & 3 & 2 & 1 & 0 & 1 & 2 \\ 4 & 3 & 2 & 1 & 0 & 0 & 1 & 2 \\ 5 & 4 & 3 & 2 & 1 & 0 & 0 & 1 \\ 4 & 3 & 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 2 & 1 & 0 & 0 & 0 & 1 & 2 \\ 4 & 3 & 2 & 1 & 1 & 0 & 1 & 2 \\ 5 & 4 & 3 & 2 & 2 & 1 & 2 & 3 \end{bmatrix}$$

These distances are approximate. A more accurate result can be obtained using integer valued masks and by increasing the mask size.

Chapter 11 Object Description

The descriptor is a set of numbers characterizing the salient properties of the object. The descriptor is compared with that of the reference object for object recognition. A descriptor should completely characterize an object and, at the same time, it should be concise. A descriptor should be unique. Similar objects should have similar descriptors. A descriptor should be invariant with respect to scaling, rotation and translation.

A region of an image is characterized by its internal or external features. Internal features are based on the pixels comprising the region. Typical features are area, perimeter and compactness. External features are related to the boundary of the region.

Boundary Descriptors

A region is characterized by its boundary and the form of the boundary is called the shape. A point is on the boundary if there is at the least one of its neighbors is outside the region and the point itself is in the region.

Chain Codes

The set of coordinates of all the boundary pixels of a region is its description. However, we are looking for efficient ways to represent the shape. One way is to use a code for each principal direction the trace of the boundary could move.

Signatures

A signature is a 1-D representation of a 2-D region by the radial distances of its boundary. The radial distances are computed from the centroid of the boundary. Then, the signature is plotted, distances versus angle. The signature of a closed boundary is a periodic function.

Consider the 4×4 binary object $x(m,n)$ and its boundary $b(m,n)$

$$x(m,n) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \quad b(m,n) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & & & 1 \\ 1 & & & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$d(m,n) = \begin{bmatrix} 2.1220 & 1.6062 & 1.6300 & 2.1758 \\ 1.5576 & & & 1.6300 \\ 1.5326 & & & 1.6062 \\ 2.0669 & 1.5326 & 1.5576 & 2.1220 \end{bmatrix}$$

$$\theta(m,n) = \begin{bmatrix} -136.4688 & -163.3008 & 160.71 & 1 \\ -110.2249 & & & 109. \\ -72.4754 & & & 73.30 \\ -45.0000 & -17.5256 & 20.2249 & 46.46 \end{bmatrix}$$

in degrees.

Fourier descriptors

Closed boundaries of an object in an image can be compactly represented using Fourier coefficients. The two coordinates of all the boundary pixels are represented in the transform domain. Starting from a point in the boundary with coordinates (m_0, n_0), followed by

$$(m_l, n_l), \quad l = 1, 2, \ldots N-1$$

can be considered as a 1-D periodic complex data

$$d(l) = (m_l + jn_l), \quad l = 0, 1, \ldots N-1$$

of period N. The first and second coordinates represent, respectively, the real and imaginary parts of the complex data. Then, the DFT of $d(l)$, the set of N 1-D DFT coefficients, is the Fourier descriptor of the boundary with significant advantages.

$$x(m,n)=\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} x(m+1,n+1)=\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The complex data $b(l)$ formed from the boundary coordinates of $x(m,n)$ is

$$\{1+j1, 2+j1, 3+j1, 3+j2, 3+j3, 2+j3, 1+j3, 1+j2\}$$

The DFT of $b(l)$ is

$$\begin{aligned} B(k) \;=\; & \{16+j16, -6.8284-j6.8284, 0, 0, 0, \\ & -1.1716-j1.1716, 0, 0\} \end{aligned}$$

The complex data $b(l)$ formed from the boundary coordinates of $x(m+1, n+1)$ is

$$\{0+j0, 1+j0, 2+j0, 2+j1, 2+j2, 1+j2, 0+j2, 0+j1\}$$

The DFT of $b(l)$ is

$$\begin{aligned} B(k) \;=\; & \{8+j8, -6.8284-j6.8284, 0, 0, 0, \\ & -1.1716-j1.1716, 0, 0\} \end{aligned}$$

Geometrical Features

Area The area of a connected region $x(m,n)$ of a binary image, measured in pixels, is defined as

$$A = \sum_m \sum_n x(m,n)$$

It is the number of pixels with value 1 in the region.

Perimeter Let the coordinates of the perimeter of a region is given by $x(k)$ and $y(k)$. Then, the perimeter of the region is defined by

$$P = \sum_k \sqrt{(x(k) - x(k-1))^2 + (y(k) - y(k-1))^2}$$

It is the distance around the boundary of the region.

Compactness Compactness is defined, in terms of the perimeter and area, as

$$C = \frac{4\pi A}{P^2} = \frac{A}{P^2/(4\pi)}$$

For a circular region, which has the highest compactness, with radius r,

$$C = (4\pi(\pi r^2))/((2\pi r)(2\pi r)) = 1$$

For a square, $C = \pi/4$. It is a measure of the area enclosing ability of the shape of the region. It is the ratio of the area of the region and the area of the circle with the same perimeter as that of the region. Let the object be a unit square. Its area is 1 and perimeter is 4. The radius of a circle with the same perimeter is $(4/(2\pi))$ and its area is $4/\pi$.

Irregularity Irregularity of a region is defined by

$$I = \frac{\pi \max_k \left(x(k) - \bar{x}\right)^2 + \left(y(k) - \bar{y}\right)^2)}{A}$$

where $(\bar{x}, \bar{y})$ are the averages of the coordinates of the region. This is a measure of the density of the region. The numerator defines the area of the smallest circle enclosing the region. For circular shapes, I is unity. For a square, it is $I = 0.5\pi$.

Euler Number

A topological descriptor of an image $x(m, n)$ is that which remains the same for all its versions of continuous one-to-one transformations (rubber-sheet distortions). This descriptor is not affected by rotation or stretching. The Euler number E is defined as the difference between the number of connected components and the number of holes.

The $(p+q)$th order moment of a $N \times N$ region $x(k,l)$ is defined as

$$m_{pq} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} k^p l^q x(k,l), p = 0,1,2,\ldots, q = 0,1,2,..$$

$$m_{00} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} x(k,l),\ m_{10} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} kx(k,l),$$

$$m_{01} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} lx(k,l)$$

Zero-order moment is the area. First-order moments are this area multiplied by their distances of their center of gravity from the origin.

The coordinates of the centroid (center of mass) of the region is defined as

$$\bar{k} = \frac{m_{10}}{m_{00}} \quad \text{and} \quad \bar{l} = \frac{m_{01}}{m_{00}}$$

The central moments, which are translation-invariant since centroids are part of their definition, are defined as

$$\mu_{pq} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} (k-\bar{k})^p (l-\bar{l})^q x(k,l), p, q = 0, 1, 2, \ldots$$

The normalized central moments, which are invariant to translation, scaling and rotation, are defined as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \gamma = \frac{(p+q)}{2} + 1, \ (p+q) = 2, 3, \ldots$$

The first four normalized central moments are defined as

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2
\end{aligned}$$

$$x(k,l) = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The coordinates of the top left corner are $(0, 0)$.

$$\{m_{00} = 6,\ m_{10} = 6,\ m_{01} = 11,\ \bar{k} = 1,\ \bar{l} = 1.8333\}$$

$$\mu_{11} = (-1)(-0.8333)+(-1)(0.1667)+(1)(0.1667)$$

$$+(1)(1.1667) = 2,\ \mu_{20} = 4,\ \mu_{02} = 2.8333$$

$$\eta_{11} = 0.0556,\ \eta_{20} = 0.1111,\ \eta_{02} = 0.0787,$$

$$\phi_1 = 0.1898,\ \phi_2 = 0.0134$$

First two moments of the four objects in Figure 11.7(b)

ϕ_1	0.4519	0.2320	0.4554	0.4510
ϕ_2	0.1149	0.0269	0.1082	0.1116

Texture is a pattern resembling a mosaic, made by a physical composition of an object using constituents of various sizes and shapes. Statistical features, taken over the whole image or in its neighborhoods, are used to characterize a texture.

Histogram Based Features

0	19	20	22
53	4	23	25
116	16	17	24
110	90	4	23

The histogram of this image and its normalized version (with a precision of 2 digits) are

g_lev	0	4	16	17	19	20	22	23
his	1	2	1	1	1	1	1	2
$hisn$	0.06	0.13	0.06	0.06	0.06	0.06	0.06	0.13

g_lev	24	25	53	90	110	116
his	1	1	1	1	1	1
$hisn$	0.06	0.06	0.06	0.06	0.06	0.06

which also is the probability $p(u)$ of the occurrences of the gray levels.

Mean The mean m, which is the average intensity, is given by

$$m = \sum_{u=0}^{L-1} up(u)$$

For the example, the mean is 35.3750 with $L = 256$.

Standard deviation The standard deviation σ, which is the average contrast and the 2nd moment, is defined as

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

For the example, $\sigma = 35.8153$. The square of the standard deviation is the variance.

Smoothness A measure of the smoothness of the texture is defined as

$$S = 1 - \frac{1}{1 + \sigma_n^2}$$

where σ_n^2 is a normalized version of the variance and

$$\sigma_n^2 = \frac{\sigma^2}{(L-1)^2}$$

For the example image,

$$S = 1 - \frac{1}{1 + (35.8153^2/255^2)} = 1 - \frac{1}{1.0197} = 0.0193$$

For regions with constant intensity, $S = 0$ and it increases with increasing value of σ towards the limit 1.

Skew The skew, which indicates the asymmetry of the histogram about the mean and the third moment, is defined as

$$Sk = \sum_{u=0}^{L-1} (u - m)^3 p(u)$$

Sk is zero for a symmetric histogram. It is positive for a right skew (spreads to the right) and negative for a left skew (spreads to the left). Sk is also normalized in the same way and the normalized value is 0.9341 (positive skew) for the example.

Uniformity This measure, uniformity of energy, is given by

$$U = \sum_{u=0}^{L-1} p^2(u)$$

U is maximum when all the intensity levels are the same and it has a lower value otherwise. For the example image, $U = 0.0781$.

Entropy The entropy, which is measure of randomness, is given by

$$E = -\sum_{u=0}^{L-1} p(u) \log_2(p(u))$$

A lower value indicates a higher redundancy in the image data and should give a high compression ratio, when compressed. $E = 3.75$.

Co-occurrence Matrix Based Features

A pair of pixels, with gray levels a and b, occurring with the same spatial relationship is co-occurrence. Co-occurrence matrices carry information of the spatial relationships between pixels. Let the number of gray levels in $x(m,n)$ be L, $\{0,1,\ldots,L-1\}$. A co-occurrence matrix $g(m,n)$ is a $L \times L$ matrix in which each element $g(m,n)$ represents the number of occurrences of a pair of pixels with intensities I_m and I_n, in a given spatial relationship in the image $x(m,n)$. It is the joint probability distribution of pairs of pixels.

The probability $p(I_m, I_n)$ of the co-occurrences of the gray levels I_m and I_n is defined as

$$p(I_m, I_n) = \frac{n(I_m, I_n)}{M} = \frac{g(m, n)}{M}$$

where $n(I_m, I_n)$ is the number of occurrences with $x(k, l) = I_m$ and $x(p, q) = I_n$ and M is the total number of occurrences in $g(m, n)$, the co-occurrence matrix.

$$x(m,n) = \begin{bmatrix} 52 & 71 & 72 & 74 & 64 & 55 & 43 & 74 \\ 105 & 56 & 75 & 77 & 64 & 60 & 53 & 78 \\ 168 & 68 & 69 & 76 & 69 & 62 & 58 & 71 \\ 162 & 142 & 56 & 75 & 73 & 64 & 60 & 53 \\ 162 & 180 & 89 & 67 & 79 & 68 & 63 & 30 \\ 186 & 175 & 156 & 61 & 78 & 72 & 63 & 53 \\ 210 & 171 & 192 & 87 & 67 & 77 & 67 & 59 \\ 250 & 158 & 188 & 140 & 59 & 77 & 69 & 61 \end{bmatrix}$$

$$x_q(m,n) = \begin{bmatrix} 1 & 2 & 2 & 2 & 2 & 1 & 1 & 2 \\ 3 & 1 & 2 & 2 & 2 & 1 & 1 & 2 \\ 5 & 2 & 2 & 2 & 2 & 1 & 1 & 2 \\ 5 & 4 & 1 & 2 & 2 & 2 & 1 & 1 \\ 5 & 5 & 2 & 2 & 2 & 2 & 1 & 0 \\ 5 & 5 & 4 & 1 & 2 & 2 & 1 & 1 \\ 6 & 5 & 6 & 2 & 2 & 2 & 2 & 1 \\ 7 & 4 & 5 & 4 & 1 & 2 & 2 & 1 \end{bmatrix}$$

Let the spatial relationship be $x(m,n)$ and $x(m+1,n+1)$. That is, a pixel and its immediate bottom right (diagonal) neighbor form the pair. With this spatial relationship and $x_q(m,n)$, we get the co-occurrence matrix $g(m,n)$ as

$$g(m,n) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 7 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 8 & 16 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The joint probability matrix $p(m,n)$ is defined as

$$p(m,n) = \frac{g(m,n)}{M}, \quad M = \sum_m \sum_n g(m,n)$$

For the example, with $M = 49$, we get $p(m,n)$ as

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.02 & 0.14 & 0.10 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.16 & 0.32 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.02 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.04 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.04 & 0.08 & 0.02 & 0 \\ 0 & 0 & 0 & 0 & 0.04 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Maximum probability

It is in the range 0 to 1 and indicates the maximum value of $p(m,n)$. For the example, it is $p(2,2) = 0.3265$.

Entropy

$$E = -\sum_{m=0}^{N-1}\sum_{n=0}^{N-1} p(m,n)\log_2 p(m,n)$$

For a $p(m,n)$ with all zero entries, $E = 0$. For a $p(m,n)$ with all entries equal, $E = 2\log_2 N$. For the example, $E = 2.8951$ with $N = 8$.

Contrast

The contrast in intensity of a pixel and its neighbor over the image is given by this measure. The range of values for C is from 0 to $(N-1)^2$.

$$C = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} (m-n)^2 p(m,n)$$

For the example, $C = 0.6939$.

Energy (Uniformity)

This measure is an indicator of the energy. Its range is from 0 to 1.

$$U = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} p^2(m,n)$$

For the example, $U = 0.1770$.

Homogeneity

$$H = \sum_{m=0}^{N-1} \sum_{N=0}^{N-1} \frac{p(m,n)}{1 + |(m-n)|}$$

This measure indicates the closeness of the distribution of the values in the co-occurrence matrix to its diagonal and it is in the range 0 to 1. For the example, $H = 0.7619$.

Correlation

$$R = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} \frac{(m - \bar{m})(n - \bar{n})p(m,n)}{\sigma_m \sigma_n}, \sigma_m \neq 0, \sigma_n \neq 0$$

$$\bar{m} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} mp(m,n), \quad \bar{n} = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} np(m,n)$$

$$\sigma_m^2 = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} (m - \bar{m})^2 p(m,n),$$

$$\sigma_n^2 = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} (n - \bar{n})^2 p(m,n)$$

This measure, with range -1 to 1, indicates the similarity of a pixel to its neighbor over the entire image. For the example, $R = 0.8508$.

Texture measures based on co-occurrence matrix

Figure	max p	E	H	U
11.8	0.0008	12.8600	0.1687	0.0002
11.9	0.0072	13.7058	0.1594	0.0002
11.10	0.0006	13.7221	0.1262	0.0001
11.11	0.0018	14.5118	0.0965	5.7966e-05

C	R
380.6666	0.9188
756.0718	0.8703
838.6651	0.7726
2.2376e+03	0.6996

These measures can be used to differentiate the various types of textures.

Principal Component Analysis

Matrix representation of data is transformed to its diagonal form. The data gets uncorrelated and sufficient number of components can be used to approximate the data with a desired accuracy.

Let there be M vector variables, with each having N samples. M variables form a $N \times M$ matrix

$$\boldsymbol{X} = [\boldsymbol{x}_0, \boldsymbol{x}_1 \cdots \boldsymbol{x}_{M-1}]$$

We want to find a matrix

$$\boldsymbol{Y} = [\boldsymbol{y}_0, \boldsymbol{y}_1 \cdots \boldsymbol{y}_{M-1}]$$

such that

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{R}$$

and the columns of $\boldsymbol{Y}$ are mutually orthogonal.

$$\boldsymbol{Y}^T\boldsymbol{Y} = (\boldsymbol{X}\boldsymbol{R})^T(\boldsymbol{X}\boldsymbol{R}) = \boldsymbol{D}$$

where $\boldsymbol{D}$ is a diagonal matrix. Using the property

$$(\boldsymbol{A}\boldsymbol{B})^T = \boldsymbol{B}^T\boldsymbol{A}^T$$

and multiplying both sides by $1/(N-1)$, we get

$$\frac{1}{N-1}\boldsymbol{Y}^T\boldsymbol{Y} = \boldsymbol{R}^T\frac{(\boldsymbol{X}^T\boldsymbol{X})}{N-1}\boldsymbol{R} = \boldsymbol{D}$$

Assuming $N = 4$ and $M = 2$, let us find the corresponding PCA components and their covariance. Then, let us reconstruct the original data from the PCA components. Let

$$a(n,1) = \begin{bmatrix} 4 \\ 5 \\ 6 \\ 7 \end{bmatrix}, \qquad b(n,2) = \begin{bmatrix} 22 \\ 23 \\ 23 \\ 24 \end{bmatrix}$$

The mean of the columns are $an1 = 5.5$ and $bn2 = 23$, respectively. Subtracting the respective means from the column vectors and concatenating, we get the 4×2 matrix

$$x(n,m) = \begin{bmatrix} -1.5000 & -1.0000 \\ -0.5000 & 0 \\ 0.5000 & 0 \\ 1.5000 & 1.0000 \end{bmatrix}$$

The covariance of this matrix is the scaled product of its transpose with itself. The covariance of the matrix $x(n,m)$ is the scaled product of its transpose with itself.

$$C(m,m) = \frac{1}{3}\begin{bmatrix} -1.5000 & -0.5000 & 0.5000 & 1.5000 \\ -1.0000 & 0 & 0 & 1.0000 \end{bmatrix}$$

$$\begin{bmatrix} -1.5000 & -1.0000 \\ -0.5000 & 0 \\ 0.5000 & 0 \\ 1.5000 & 1.0000 \end{bmatrix} = \begin{bmatrix} 1.6667 & 1.0000 \\ 1.0000 & 0.6667 \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_{aa} & \sigma_{ab} \\ \sigma_{ba} & \sigma_{bb} \end{bmatrix}$$

The covariance matrix is square with the dimensions equal to the number of images. This matrix is always symmetric.

In order to find the eigenvectors of this matrix, we have to find its eigenvalues. They are found by equating the determinant of $\lambda \boldsymbol{I} - \boldsymbol{C}$ to zero (its characteristic equation), where $\boldsymbol{I}$ is the identity matrix.

$$\begin{vmatrix} \lambda - 1.6667 & -1 \\ -1 & \lambda - 0.6667 \end{vmatrix} = 0 \quad \text{or} \quad \lambda^2 - 2.3333\lambda + 0.$$

The two eigenvalues are $\{2.2847, 0.0486\}$.

The eigenvectors are

$$\boldsymbol{R} = \begin{bmatrix} -0.8507 & -0.5257 \\ -0.5257 & 0.8507 \end{bmatrix}$$

The principal components are found as

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{R} = \begin{bmatrix} -1.5000 & -1.0000 \\ -0.5000 & 0 \\ 0.5000 & 0 \\ 1.5000 & 1.0000 \end{bmatrix}$$

$$\begin{bmatrix} -0.8507 & -0.5257 \\ -0.5257 & 0.8507 \end{bmatrix} = \begin{bmatrix} 1.8017 & -0.0621 \\ 0.4253 & 0.2629 \\ -0.4253 & -0.2629 \\ -1.8017 & 0.0621 \end{bmatrix}$$

The input can be reconstructed by

$$\boldsymbol{Y}\boldsymbol{R}^T + \begin{bmatrix} an1 & bn2 \\ an1 & bn2 \\ an1 & bn2 \\ an1 & bn2 \end{bmatrix} = \begin{bmatrix} 1.8017 & -0.0621 \\ 0.4253 & 0.2629 \\ -0.4253 & -0.2629 \\ -1.8017 & 0.0621 \end{bmatrix} \begin{bmatrix} -0.8507 & -0.5257 \\ -0.5257 & 0.8507 \end{bmatrix} + \begin{bmatrix} 5.5 & 23 \\ 5.5 & 23 \\ 5.5 & 23 \\ 5.5 & 23 \end{bmatrix}$$

$$
\begin{aligned}
&= \begin{bmatrix} -1.5000 & -1.0000 \\ -0.5000 & 0 \\ 0.5000 & 0 \\ 1.5000 & 1.0000 \end{bmatrix} + \begin{bmatrix} 5.5 & 23 \\ 5.5 & 23 \\ 5.5 & 23 \\ 5.5 & 23 \end{bmatrix} \\
&= \begin{bmatrix} 4 & 22 \\ 5 & 23 \\ 6 & 23 \\ 7 & 24 \end{bmatrix}
\end{aligned}
$$

Recognition Chapter 12

There are two main types of classification: (i) supervised classification and (ii) unsupervised classification. In supervised classification, features are specified apriori and objects are classified using them. Typical methods used are minimum distance, k-nearest neighbors, decision trees, and statistical (based on probability distribution models). The decision is prior. In unsupervised classification, we classify the objects by the constraints imposed by the features. Partition the data into groups by clustering. Unknown, but distinct set of feature classes are generated. Decision is posterior.

The k-nearest Neighbors Classifier

In this method, the feature set of a test object is compared with the reference set and the test object is assigned to the class whose features differ, with respect to some measure, by the least from that of the test object. In terms of distance, computing the distance between the k closest points in the reference sets of feature vectors is the measure. The method is simple, capable of classifying overlapping classes and classes with complex structures. With $k = 1$, it becomes the minimum distance classifier.

The Minimum-Distance-to-Mean Classifier
In this approach, the classification of an object is based on discriminant functions. Let the feature vector $\boldsymbol{x}$ of three classes be $\boldsymbol{m}_1$, $\boldsymbol{m}_2$, $\boldsymbol{m}_3$. Three discriminant functions, $d_1(\boldsymbol{x})$, $d_2(\boldsymbol{x})$ and $d_3(\boldsymbol{x})$, have to be found such that the discriminant function corresponding to an unclassified feature vector will yield a value that is greater than those of the functions to which it does not belong. In the case of two or more functions evaluating to the same value, the decision is arbitrary or based on some additional factors.

Let the elements of a test vector be

$$\boldsymbol{x} = \{x_1, x_2, \ldots, x_M\}$$

Let the mean vector of the N classes be

$$\{\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_N\}$$

$$d_n(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{m}_n^T - 0.5\boldsymbol{m}_n\boldsymbol{m}_n^T, \quad n = 1, 2, \ldots, N$$

Let us get the discriminant functions for the last example. For the first feature vector $\{6400, 320\}$

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 6400 \\ 320 \end{bmatrix} - 0.5 \begin{bmatrix} 6400 & 320 \end{bmatrix} \begin{bmatrix} 6400 \\ 320 \end{bmatrix}$$

$$d_1(\boldsymbol{x}) = 6400x_1 + 320x_2 - 20531200$$

Similarly, for class2, we get

$$d_2(\boldsymbol{x}) = 2500x_1 + 5000x_2 - 15625000$$

For class3, we get

$$d_3(\boldsymbol{x}) = 500x_1 + 1000x_2 - 625000$$

For the first feature vector, these three functions yield

$$\{20531200, 1975000, 2895000\}$$

As expected, the first function has the greatest value. Similarly, for the second feature vector, these three functions yield

$$\{-2931200, 15625000, 5625000\}$$

For the third feature vector, these three functions yield

$$\{-17011200, -9375000, 625000\}$$

The difference between two discriminant functions is the boundary discriminant function for them. For example, the boundary discriminant function for class1 and class2 is

$$\begin{aligned} d_{12}(\boldsymbol{x}) &= d_1(\boldsymbol{x}) - d_2(\boldsymbol{x}) \\ &= (6400x_1 + 320x_2 - 20531200) \\ &\quad - (2500x_1 + 5000x_2 - 15625000) \\ &= 3900x_1 - 4680x_2 - 4906200 \end{aligned}$$

Decision Tree Classification

In this approach, the feature space is split into unique regions sequentially. A decision is arrived with out testing all classes and, therefore, it is advantages when the number of classes is large. Further, the convergence of this algorithm is guaranteed irrespective of the nature of the feature space.

Object	A	B	C	D	E
Holes	1	2	0	1	0
End points	2	0	2	0	3

The first step is to sort the entries in each row of the feature vectors in ascending order. The maximum difference of adjacent entries in the first row is 1. It is 2 in the second row. A threshold, that is the average of the two adjacent entries, $(0, 2)$, that produced the maximum difference, is set. Using this threshold $f2 = (0 + 2)/2 = 1$, the rows are partitioned

$$\left[\begin{array}{cc|ccc} 0 & 0 & 1 & 1 & 2 \\ 0 & 0 & 2 & 2 & 3 \end{array}\right]$$

This partitioning continues until each partition is just one column. Now, the order of the first row is restored and we get

$$\left[\begin{array}{cc|ccc} 2 & 1 & 1 & 0 & 0 \\ 0 & 0 & 2 & 2 & 3 \end{array}\right]$$

Now, the left side partition includes the letters B and D, which can be partitioned with a threshold $f1 = (1 + 2)/2 = 1.5$. The unsorted and sorted feature vectors for the other 3 characters are

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 2 & 2 & 3 \end{array}\right] \quad \left[\begin{array}{cc|c} 0 & 0 & 1 \\ 2 & 2 & 3 \end{array}\right]$$

The letter A can be isolated with a threshold $f1 = 0.5$. Letters C and E can be isolated with a threshold $f2 = 2.5$.

Bayesian approach to statistical methods of classification is based not only on the set of samples but also on the pertinent prior information. Bayesian approach provides discriminant or decision functions, which maximize the number of correct classifications and minimize the incorrect ones. Let there be N features $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}^T$ representing the M classes of objects, $\{\omega_1, \omega_2, \ldots, \omega_M\}$. Let the a priori probability of an arbitrary object belongs to class ω_i be $\{p(\omega_1), p(\omega_2), \ldots, p(\omega_M)\}$. Let the density distribution of all the objects be $p(\boldsymbol{x})$. Let the conditional density distribution of all the objects belonging to class ω_i be $p(\boldsymbol{x}/\omega_i)$.

Using the Bayes' theorem, the decision rule is

if $p(\boldsymbol{x}/\omega_i)p(\omega_i) > p(\boldsymbol{x}/\omega_j)p(\omega_j)$ for all $i \neq j$,

then assign $\boldsymbol{x}$ to ω_i

Since it is difficult to estimate the actual $p(\boldsymbol{x}/\omega_i)$, in practice, the Gaussian (normal) density function is often assumed.
For normal distribution,

$$p(\boldsymbol{x}/\omega_i) = \frac{1}{(2\pi)^{(N/2)}|\boldsymbol{C}_i|^{0.5}} e^{-0.5(\boldsymbol{x}-\boldsymbol{m}_i)^T \boldsymbol{C}_i^{-1}(\boldsymbol{x}-\boldsymbol{m}_i)}$$

where the mean $\boldsymbol{m}_i$ and the covariance matrix $\boldsymbol{C}_i$ are approximated as

$$\boldsymbol{m}_i = \frac{1}{N_i} \sum_{x \in \omega_i} \boldsymbol{x}$$

$$\boldsymbol{C}_i = \frac{1}{N_i - 1} (\boldsymbol{x_i} - \boldsymbol{m_i})^T (\boldsymbol{x_i} - \boldsymbol{m_i})$$

The determinant of $\boldsymbol{C}_i$ is $|\boldsymbol{C}_i|$. Since $p(\boldsymbol{x}/\omega_i)$ in exponential form, the decision rule

$$d_i(\boldsymbol{x}) = p(\boldsymbol{x}/\omega_i) p(\omega_i)$$

is changed to the form

$d_i(\boldsymbol{x})$=log$_e(p(\boldsymbol{x}/\omega_i)p(\omega_i))$=log$_e(p(\boldsymbol{x}/\omega_i))$+log$_e(p(\omega_i))$

for convenience of manipulation. This change in form does not alter the numerical order of the decision functions required for classification.

Substituting the exponential expression for $p(\boldsymbol{x}/\omega_i)$, we get the Bayes decision function

$$d_i(\boldsymbol{x}) = \log_e(p(\omega_i)) - 0.5\log_e(|\boldsymbol{C}_i|) - 0.5((\boldsymbol{x}-\boldsymbol{m}_i)^T\boldsymbol{C}_i^{-1}(\boldsymbol{x}-\boldsymbol{m}_i)), i = 1, 2, \ldots, M$$

As the term $-(N/2)\log_e(2\pi)$ does not affect the numerical order of the decision functions, it is dropped. If all the covariance matrices are the same, then

$$d_i(\boldsymbol{x}) = \log_e(p(\omega_i)) + \boldsymbol{x}^T\boldsymbol{C}^{-1}\boldsymbol{m}_i - 0.5\boldsymbol{m}_i^T\boldsymbol{C}^{-1}\boldsymbol{m}_i$$

Further, if $\boldsymbol{C}$ is the identity matrix and $p(\omega_i) = 1/M,\ i = 1, 2, \ldots, M$, then

$$d_i(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{m}_i - 0.5 \boldsymbol{m}_i^T \boldsymbol{m}_i, \quad i = 1, 2, \ldots, M$$

which is the same for the decision function of the minimum distance classifier.

Chapter 13 Image Compression

Image compression is finding the minimum amount of data required to represent a certain amount of information and it is a necessity for efficient storage and transmission of images. The basic property of images that enables compression is that the spectrum of practical images tends to fall off to insignificant levels at high frequencies.

Redundancies in an image

The redundancies in an image can be classified into three major types:
(i) coding redundancy
(ii) interpixel redundancy
(iii) irrelevant information.

DWT

The DWT is more often used for image compression. The major advantage is the structure of the DWT transformed image. Exploiting this structure, the use of the DWT gives better reconstructed images and compression ratios, particularly at lower bit rates.

Filters

The most suitable DWT filters are of biorthogonal type. The advantage is that longer symmetric filters are available. A symmetric filter handles the border problem more effectively. Further, symmetric filters provide linear phase response, which is essential in the analysis of images. The CDF 9/7 filter is often used for lossy image compression. For lossless image compression, the 5/3 spline filter is often used.

Example Image

13	14	2	14
8	2	2	8
15	15	2	15
15	8	13	2

Assignment of Huffman Codes

Char	Freq	Rel	Code A	Bits	Code B	Bits
2	5	$\frac{5}{16}$	1	5	11	10
8	3	$\frac{3}{16}$	001	9	01	6
13	2	$\frac{2}{16}$	0001	8	001	6
14	2	$\frac{2}{16}$	0000	8	000	6
15	4	$\frac{4}{16}$	01	8	10	8

Entropy

Let the number of distinct values in an image is N and the frequency of their occurrence be

$$s_1, s_2, \ldots, s_N$$

Entropy is defined as

$$E = \sum_{k=1}^{N} p(s_k) \log_2 \left(\frac{1}{p(s_k)} \right)$$

where $p(s_k)$ is the probability of occurrence of s_k.

The term $\log_2(1/p(s_k))$ gives the number of bits required to represent $1/p(s_k)$. By multiplying this factor with $p(s_k)$, we get the bpp required to code s_k. The sum of bpp for all the distinct values yields the bpp to code the image. This equation can be equivalently written as

$$E = -\sum_{k=1}^{N} p(s_k) \log_2(p(s_k))$$

Signal-to-noise ratio

signal-to-noise ratio, expressed in decibels, defined as

$$\text{SNR}=10\log_{10}\left(\frac{\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}\widehat{x}^2(n_1,n_2)}{\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1}(x(n_1,n_2)-\widehat{x}(n_1,n_2))^2}\right)$$

Lossless Predictive Coding

Consider the 4×4 image

51	50	52	47
53	44	39	48
40	63	131	212
114	128	154	155

Let us use a filter with one coefficient with value 1. Then,

$$\widehat{x}(m,n) = \text{round}\,(x(m,n-1))$$

$$e(m,n) = x(m,n) - \widehat{x}(m,n) = x(m,n) - x(m,n-1)$$

except for the first pixels of the rows.

Predictive coding representation of the image is

51	-1	2	-5
53	-9	-5	9
40	23	68	81
114	14	26	1

The first column values remain the same. For the first row, the second, third and fourth column values are $50-51=-1$, $52-50=2$ and $47-52=-5$, respectively. The values of the other three rows are found similarly. For decompressing of the first row, the second, third and fourth column values are $51-1=50$, $50+2=52$ and $52-5=47$. In the input image, there are 16 symbols each with probability 1/16. Therefore, the entropy is

$$-(16(1/16)\log 2(1/16))=4$$

In the coded image, there are 15 symbols (-5 repeats twice), 14 with probability 1/16 and one with 2/16. Therefore, the entropy is

$$-(14(1/16)\log 2(1/16) + (2/16)\log 2(2/16))$$

$$= 31/8 = 3.8750$$

To code the difference, we need one bit more and, usually, the independent symbols in the coded image is more than that of the input image. However, due to the density of the histogram at the center, the coded image has a lower entropy, as presented in the next example.

Example Image

172	188	189	186
178	187	189	192
188	190	196	197
191	193	197	199

Level-shifted Image

44	60	61	58
50	59	61	64
60	62	68	69
63	65	69	71

Compression Algorithm

- Level shift the image. Let the gray levels be represented using p bits. Then, each level-shifted pixel value of the image is obtained by subtracting $2^{(p-1)}$. This ensures that the DWT coefficients are more evenly distributed around zero and quantization will be more effective.

- Compute the 2-D DWT of the level-shifted image, resulting in $N \times N$ coefficients, usually with real values, over some range.

- Quantize the coefficients to q quantization levels, so that the fidelity of the reconstructed image is adequate. Each value in a range is mapped to an integer value.

- Threshold the coefficients, if necessary, so that coefficients with value less than a chosen threshold are replaced by zero.

- Code the resulting sequence of symbols using a suitable coder so that the redundancy is exploited to reduce the number of bits required to store the compressed image.

The row DWT on the top and the 2-D Haar DWT (bottom) of the level-shifted image

$\frac{1}{\sqrt{2}}$

104	119	-16	3
109	125	-9	-3
122	137	-2	-1
128	140	-2	-2

106.5	122.0	-12.5	0
125.0	138.5	-2.0	-1.5
-2.5	-3.0	-3.5	3.0
-3.0	-1.5	0	0.5

Quantized Image

48	55	-5	0
56	62	0	0
-1	-1	-1	1
-1	0	0	0

Huffman code of the image

$$\{48, 55, 56, 62, \quad -5, 0, 0, 0,$$

$$-1, -1, -1, 0, \quad -1, 1, 0, 0\}$$

The Huffman code of the image is

$$\{0010\ 00001\ 00000\ 00011$$

$$00010\ 1\ 1\ 1\ 01\ 01\ 01\ 1\ 01\ 0011\ 1\ 1\}$$

2-D Haar DWT of the reconstructed level-shifted image

105.5238	120.9127	-10.9921	0
123.1111	138.5000	0	0
-2.1984	-2.1984	-2.1984	2.1984
-2.1984	0	0	0

Level-shifted reconstructed image

45.0675	58.2579	60.4563	58.2579
49.4643	58.2579	60.4563	62.6548
60.4563	60.4563	69.2500	69.2500
62.6548	62.6548	69.2500	69.2500

Reconstructed image

173.0675	186.2579	188.4563	186.2579
177.4643	186.2579	188.4563	190.6548
188.4563	188.4563	197.2500	197.2500
190.6548	190.6548	197.2500	197.2500

Color Image Processing Chapter 14

Any color can be specified by a set of basis colors. Similar to the availability of various transforms suitable for various applications, various color models are available to suit various color image processing tasks. RGB model is mostly used for image acquisition and display. CMY and CYMK models are used in color printing. The HSI model is suitable for image processing operations since it decouples the color component from the intensity value of the image.

The red, green and blue component pixel values at coordinates $(73:76, 173:176)$, respectively, are

$$xr = \begin{bmatrix} 245 & 246 & 246 & 248 \\ 246 & 247 & 246 & 247 \\ 246 & 246 & 245 & 248 \\ 248 & 247 & 246 & 248 \end{bmatrix}$$

$$xg = \begin{bmatrix} 191 & 196 & 193 & 190 \\ 192 & 197 & 192 & 189 \\ 192 & 197 & 192 & 189 \\ 192 & 198 & 191 & 188 \end{bmatrix}$$

$$xb = \begin{bmatrix} 222 & 225 & 223 & 223 \\ 223 & 226 & 223 & 222 \\ 223 & 225 & 222 & 222 \\ 223 & 224 & 221 & 224 \end{bmatrix}$$

In this neighborhood (about the center of the top-right quadrant), the image is primarily white and, therefore, the intensities of all the three components are almost equal and high.

The CMY (cyan, magenta and yellow) model can be obtained from the RGB model using the relationship, assuming color values have been normalized to the range 0 to 1,

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Note that cyan subtracts (absorbs) red component and, therefore, when white light is reflected from an object with cyan color, the red component will be zero. Similarly, magenta and yellow surfaces do not reflect green and blue, respectively.

In the HSI (hue, saturation and intensity) model, the intensity component is decoupled from the color information, making it highly suitable for developing image processing algorithms. Humans also describe a color using these components rather than in terms of red, green, and blue components.

Hue The true color attribute identifies colors red, green, yellow, etc.

Saturation Indicates the amount of white color mixed (color purity). More white in the color will result in a low saturation value.

Intensity Is a measure of brightness. The intensity of a dark color is low.

This is a perception-based color model. The conversion of a RGB image to a HSI image is governed by the following equations.

$$H = \begin{cases} \theta, & \text{for } B \leq G \\ 360 - \theta, & \text{for } B > G \end{cases},$$

$$\theta = \cos^{-1}\left(\frac{0.5((R-G)+(R-B))}{\sqrt{(R-G)^2+(R-B)(G-B)}}\right)$$

$$S = 1 - \frac{3(\min(R,G,B))}{(R+G+B)},\ I = \frac{(R+G+B)}{3}$$

The primary colors are separated by 120°. This model is derived by making the RGB color cube stand on its black corner with intensity value zero. Then, the white corner, with intensity value one, is at the top. The two corners are joined by the vertical intensity line, which gives the intensity component of a pixel. The intensity value I of any pixel $x(H, S, I)$ is given by the intersection of this line with a plane containing the pixel and perpendicular to the intensity line. The intensity is the average of those of the 3 components.

The red color is set as the reference for measuring the hue H of a pixel. The reference line is from the center of the figure to the red color corner. The color of a pixel $x(H, S, I)$ H is the angle, measured in the anticlockwise direction, between this reference line and the line joining the pixel and the center of the figure. Therefore, $H = 0^\circ$ for red color and it is measured along the circumference of the circle.

The saturation component S of a pixel is the length of the line between the center of the figure and the pixel (radial distance). It indicates the purity of the color. If the color is achromatic, then $S = 0$. For a pure color, $S = 1$. This value is dependent on the number of colors contributing to the color perception. The higher the number, the lower is the value of S. The smallest value of the RGB components determines the amount of white color possible.

Color	RGB values	H	S	I
Red	[1 0 0]	0	1	0 to 1/3
Green	[0 1 0]	$120^\circ/360$	1	0 to 1/3
Blue	[0 0 1]	$240^\circ/360$	1	0 to 1/3
Black	[0 0 0]	0	0	0
White	[1 1 1]	0	0	1
Cyan	[0 1 1]	$180^\circ/360$	1	0 to 2/3
Magenta	[1 0 1]	$300^\circ/360$	1	0 to 2/3
Yellow	[1 1 0]	$60^\circ/360$	1	0 to 2/3

Table shows HSI model values for images with pure primary colors, black, white and pure secondary colors with intensity varying from 0 to 1. The values in the table can be verified using the defining equations.

The conversion of a HSI image to a RGB image is governed by the following equations.
RG sector ($0° \leq H < 120°$) :

$$B = I(1 - S)$$

$$R = I\left(1 + \frac{S\cos(H)}{\cos(60° - H)}\right)$$

$$G = 3I - (R + B)$$

GB sector ($120^\circ \leq H < 240^\circ$) :

$$H = H - 120^\circ$$

$$R = I(1 - S)$$

$$G = I\left(1 + \frac{S\cos(H)}{\cos(60^\circ - H)}\right)$$

$$B = 3I - (R + G)$$

BR sector ($240^\circ \leq H \leq 360^\circ$) :

$$H = H - 240^\circ$$

$$G = I(1 - S)$$

$$B = I\left(1 + \frac{S\cos(H)}{\cos(60^\circ - H)}\right)$$

$$R = 3I - (B + G)$$

The NTSC color model is used for television in some countries. The advantage is that it is suitable for both color and monochrome television. The conversion between the formats can be carried using a transformation and its inverse. The luminance (intensity) is represented by the Y component and I and Q carry color information jointly, hue and saturation.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

For a gray scale image with no color, as the RGB components are equal, the first row of the transformation matrix adds to 1 and the other two adds to zero. In finding the Y component, more weight is given to the green component in order to match the response of the human visual system. The inverse transformation is

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.0 & 0.956 & 0.621 \\ 1.0 & -0.272 & -0.647 \\ 1.0 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

The YCbCr model is mostly used in digital video. The YCbCr model is a format in which Y represents the intensity and Cb and Cr represent the chrominance. Cb component is the difference between blue component and a reference value. Cr component is the difference between red component and a reference value. The energy of an image is more evenly distributed among its three components in the RGB format. In the YCbCr format, the intensity carries most of the energy. Therefore, the chrominance component can be effectively compressed requiring reduced storage requirements.

The luminance is defined as a weighted average of that of the three components. Let the intensity values of an image from 0 to 255 be scaled to 0 to 1 obtained by dividing by 255.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

In this formula, let the RGB input values be 0 to 1. Then, output Y varies from 16 to 235. Outputs Cb and Cr vary from 16 to 240. Scaling the output by 255, we get the outputs in the range 0 to 1.

The inverse transformation is

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 0.0046 & 0.0000 & 0.0063 \\ 0.0046 & -0.0015 & -0.0032 \\ 0.0046 & 0.0079 & 0.0000 \end{bmatrix}$$

$$\left[\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} - \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}\right]$$

An image can also be represented by a combination of a look-up table (LUT) and a matrix of values pointing to it. This representation is called an indexed image . The LUT is also called a color map. For each pixel of the image, the image matrix contains a value that is an index into the color map. The colormap is a $m \times 3$ matrix containing floating-point values in the range [0 1]. Each row of colormap indicates the red, green and blue components of a single color.

Conversion of a RGB image into a gray scale image
The formula used is

$$x(m,n) = 0.2989r(m,n)+0.5870g(m,n)+0.1140b(m$$

where $x(m,n)$ is the value of the gray level image and $r(m,n)$, $g(m,n)$ and $b(m,n)$ are, respectively, the red, green and blue components of the color image at coordinates (m,n).

$$color_map1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} color_map2 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

0	1-64	65-128	129-192	255
Blue	Magenta	Yellow	Red	Green
Yellow	Green	Blue	White	Magenta

Let the partial derivatives of the RGB components along the two directions, at each pixel, be

$$\left\{\frac{\partial R}{\partial x}, \frac{\partial G}{\partial x}, \frac{\partial B}{\partial x}\right\} \quad \text{and} \quad \left\{\frac{\partial R}{\partial y}, \frac{\partial G}{\partial y}, \frac{\partial B}{\partial y}\right\}$$

The partial derivatives are approximated using gradient operators, such as Sobel. Then,

$$gxx = \left(\frac{\partial R}{\partial x}\right)^2 + \left(\frac{\partial G}{\partial x}\right)^2 + \left(\frac{\partial B}{\partial x}\right)^2,$$

$$gyy = \left(\frac{\partial R}{\partial y}\right)^2 + \left(\frac{\partial G}{\partial y}\right)^2 + \left(\frac{\partial B}{\partial y}\right)^2,$$

$$gxy = \frac{\partial R}{\partial x}\frac{\partial R}{\partial y} + \frac{\partial G}{\partial x}\frac{\partial G}{\partial y} + \frac{\partial B}{\partial x}\frac{\partial B}{\partial y}$$

The angle of the gradient is given by

$$\theta 1 = 0.5 \tan^{-1}\left(\frac{2gxy}{gxx - gyy}\right),\ \theta 2 = \theta 1 + \frac{\pi}{2}$$

The magnitude of the gradient in the direction of $\theta 1$ and $\theta 2$ is computed using the expression

$$\sqrt{0.5((gxx+gyy)+(gxx-gyy)\cos(2\theta)+2gxy\sin(2\theta))}$$

The maximum of the two values is taken as the magnitude of the gradient, which is then thresholded.

Segmentation

Let the average color of the region, to be segmented, of the RGB image $x(m,n)$ be $\{ar, ag, ab\}$. Then, the square root of the sum of the Euclidean distance between the reference and image pixel color components is computed and then it is subjected to a threshold. The distances are computed using the equation.

$$\sqrt{(xr(m,n)-ar)^2+(xg(m,n)-ag)^2+(xb(m,n)-ab)^2)}$$

The pixels with distances above the threshold are not in the segment and are assigned the value zero. The other pixels are assigned the value 1.