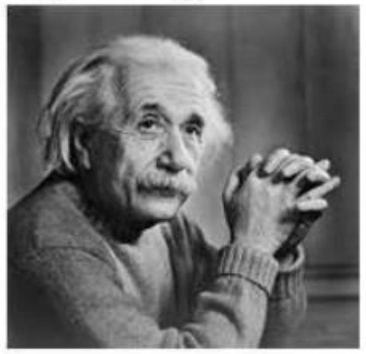
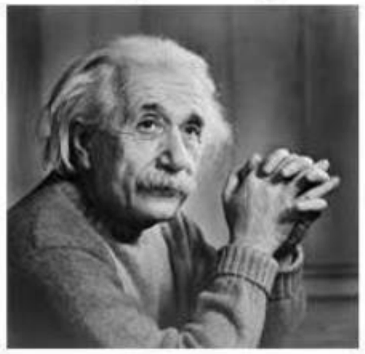
Intensity Transformations and Spatial Filtering

ICT4201: DIP


Input image


Output image

Introduction

- The principal objective of enhancement
- To process an image so that the result is **more suitable** than the original image for a **specific** application.
    - For example, A method that is quite useful for enhancing X-ray images may not necessarily be the best approach for enhancing pictures of Mars transmitted by a space probe.
- The term spatial domain : the image plane itself
    - Based on direct manipulation of pixels in an image.
    - Frequency domain processing techniques : based on the Fourier transform of an image.
- There is no general theory of image enhancement.
    - When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works.
- When dealing with machine perception, enhancement is easier to quantify.
    - For example, in an automated character-recognition system, the most appropriate enhancement method is the one that results in the best recognition rate, leaving aside other considerations such as computational requirements of one method versus another.

Image Transformation – Intensity Transformation Functions

- Consider this equation
  - $G(x,y) = T\{ f(x,y) \}$
- In this equation,
  - F(x,y) = input image on which transformation function has to be applied.
  - G(x,y) = the output image or processed image.
  - T is the transformation function.
- This relation between input image and the processed output image can also be represented as.
  - s = T (r)
  - where r is actually the pixel value or gray level intensity of f(x,y) at any point. And s is the pixel value or gray level intensity of g(x,y) at any point.
- Look-up table technique
  - – L LUT (L : # of gray levels, 2^N)
  - – s=LUT[r]
  - – Filling LUT with T

Basic Intensity Transformation Functions

- Three basic types of functions used frequently in image processing shown in the figure3.3:
  - linear (negative and identity transformations),
  - logarithmic (log and inverse-log transformations),
  - power-law (*n*th power and *n*th root transformations).
- The identity function is the trivial case in which the input and output intensities are identical.

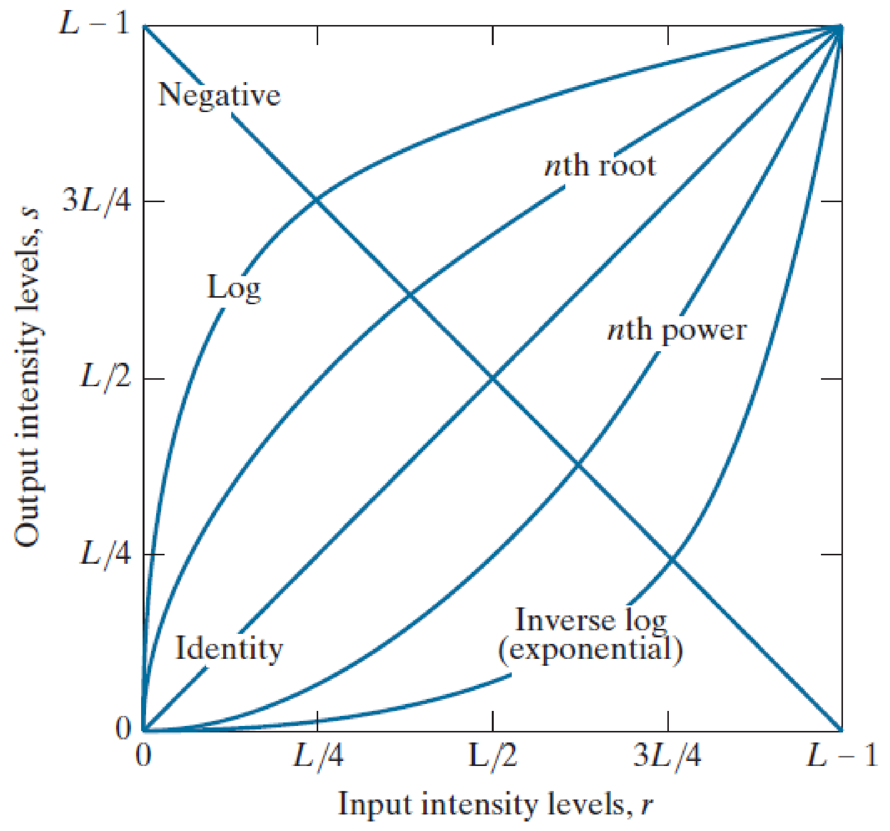
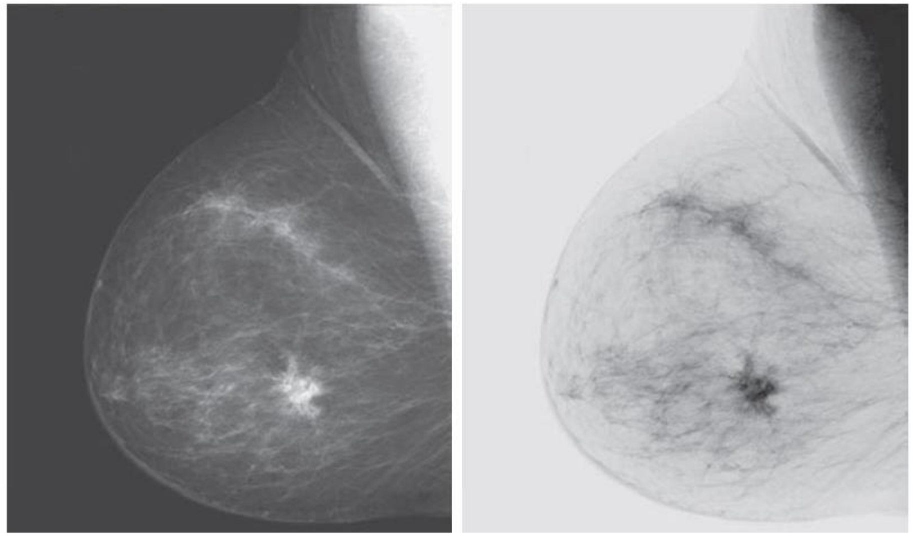


Image Negatives

- The negative of an image with intensity levels in the range [0, *L* - 1] is obtained by using the negative transformation function shown in Fig. 3.3, which has the form:
  - $s = L - 1 - r$
  - (0,1,2,…254,255)→(255,254,…,1,0)
- Reversing the intensity levels of a digital image in this manner produces the equivalent of a photographic negative. This type of processing is used, for example,
  - in enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size



Log Transformations

- The general form of the log transformation in Fig. 3.3 is
  - $s = c \log(1 + r)$
  - where c is a constant and it is assumed that $r \geq 0$
- Any curve having the general shape of the log function shown in Fig. 3.3 would accomplish spreading/compressing of intensity levels in an image.
- to expand the values of dark pixels in an image while compressing the higher-level values. (and vice versa)

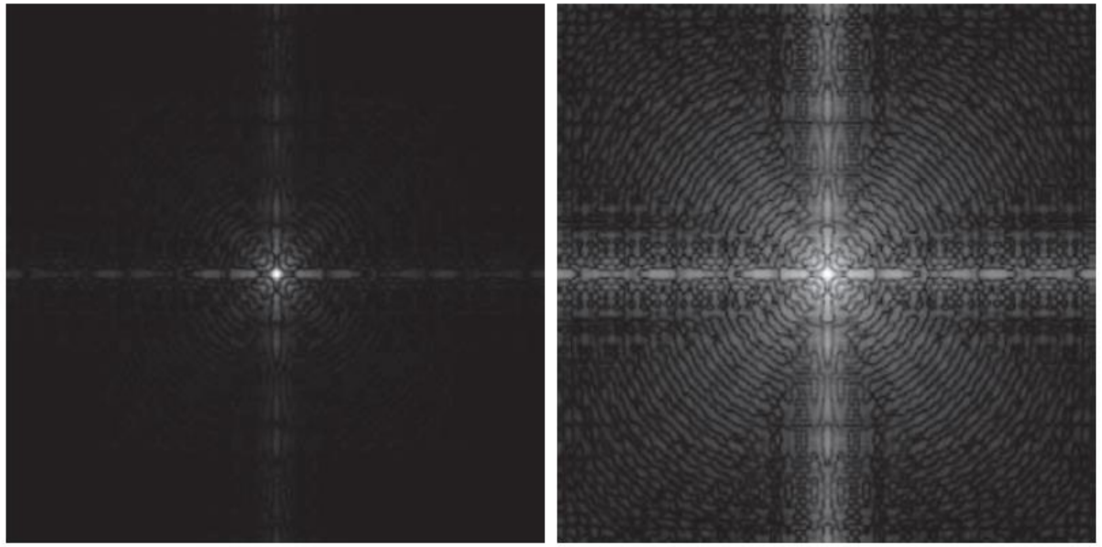
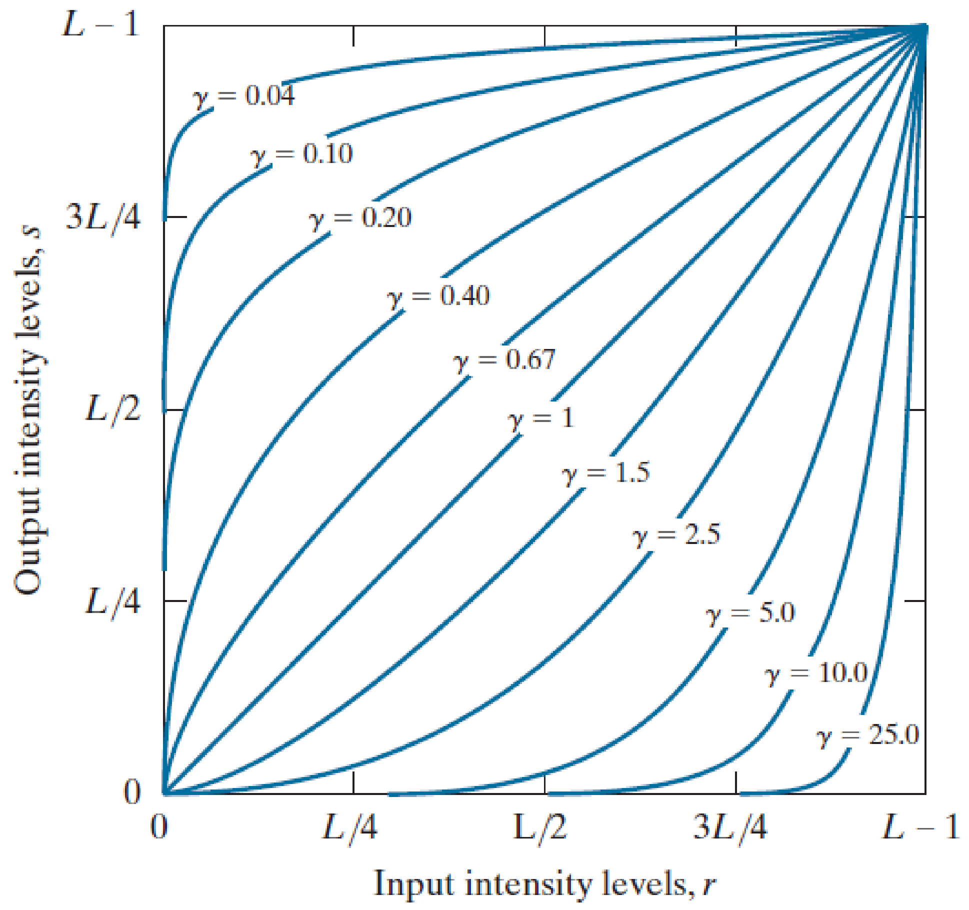

a b

FIGURE 3.5 (a) Fourier spectrum displayed as a grayscale image. (b) Result of applying the log transformation in Eq. (3-4) with $c = 1$. Both images are scaled to the range [0, 255].

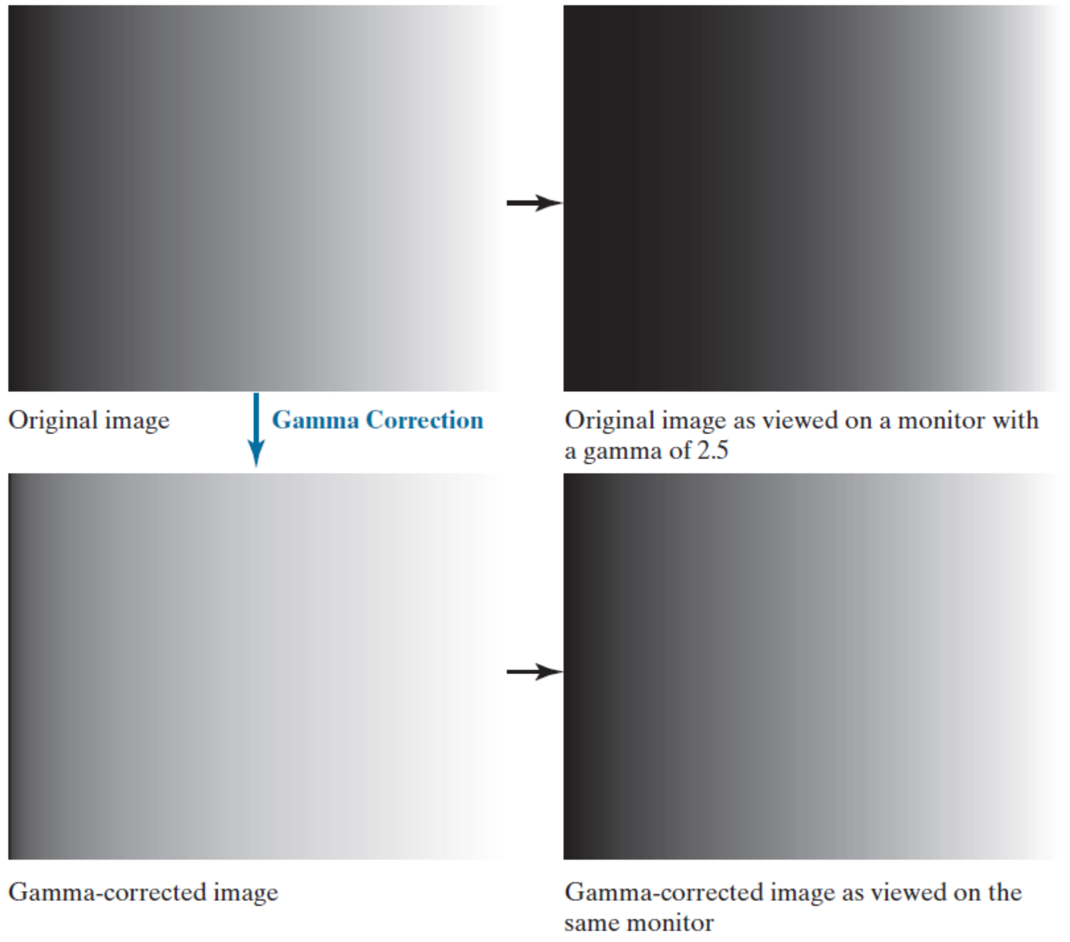
Power-Law (Gamma) Transformation

- Power-law transformations have the form
  - $s = cr^{ɣ}$
  - where *c* and ɣ are positive constants. Sometimes the equ. is written as $s = c(r + e)^{ɣ}$ to account for offsets (that is, a measurable output when the input is zero).
- Compresses the dynamic range of the image
  - the exponent is called gamma (g)
  - gamma correction
  - i.e. CRT monitors have an intensity to voltage response which is a power-law with gamma 1.8-2.5
- As in the case of the log transformation, power-law curves with fractional values of y map a narrow range of dark input values into a wider range of output values, with the opposite



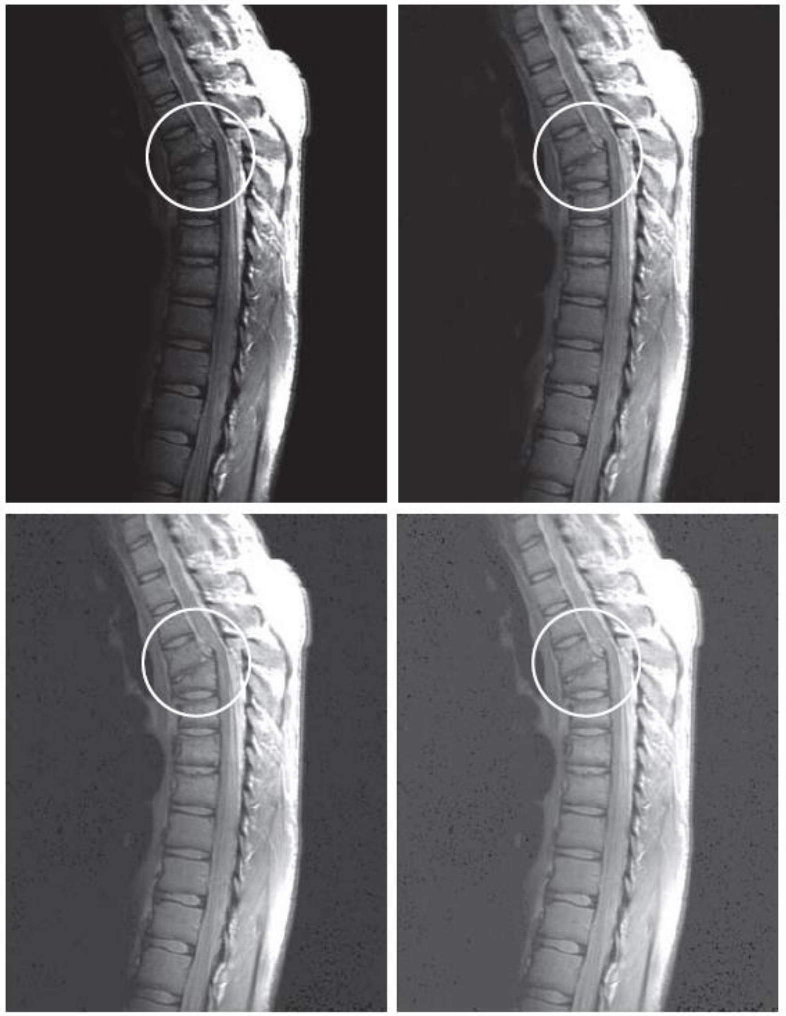
Gamma Correction

- The process used to correct these power-law response phenomena is called *gamma correction* or *gamma encoding*.
  - For example, cathode ray tube (CRT) devices have an intensity-to-voltage response that is a power function, with exponents varying from approximately 1.8 to 2.5.


Original image
Gamma Correction
Original image as viewed on a monitor with a gamma of 2.5
Gamma-corrected image
Gamma-corrected image as viewed on the same monitor

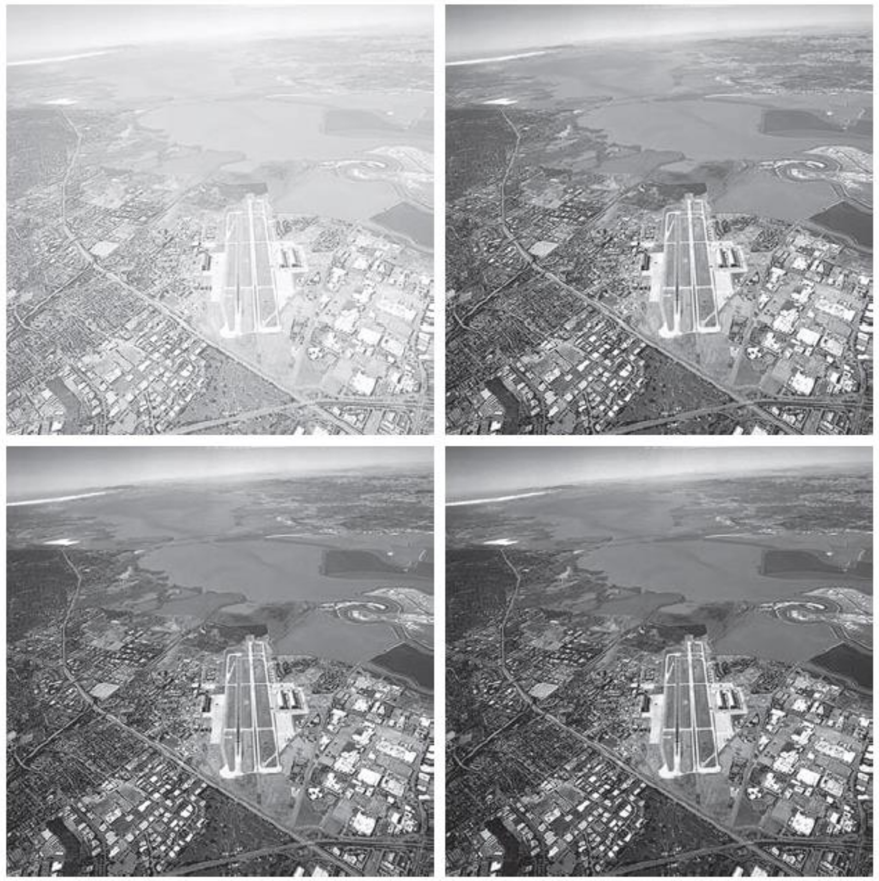
a b
c d

FIGURE 3.7 (a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).



a b
c d

FIGURE 3.8 (a) Magnetic resonance image (MRI) of a fractured human spine (the region of the fracture is enclosed by the circle). (b)–(d) Results of applying the transformation in Eq. (3-5) with $c = 1$ and $\gamma = 0.6, 0.4,$ and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)



a b
c d

FIGURE 3.9 (a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3-5) with $\gamma = 3.0, 4.0,$ and 5.0, respectively. ($c = 1$ in all cases.) (Original image courtesy of NASA.)

Piecewise-Linear Transformations - Contrast Stretching

- Low-contrast images can result from poor illumination, lack of dynamic range in the imaging sensor, or even the wrong setting of a lens aperture during image acquisition.
- *Contrast stretching* expands the range of intensity levels in an image so that it spans the ideal full intensity range of the recording medium or display device.
- If $r_1 = s_1$ and $r_2 = s_2$ the transformation is a linear function that produces no changes in intensity.
- If $r_1 = r_2 =$, $s_1 = 0$, and $s_2 = L - 1$ the transformation becomes a *thresholding function* that creates a binary image

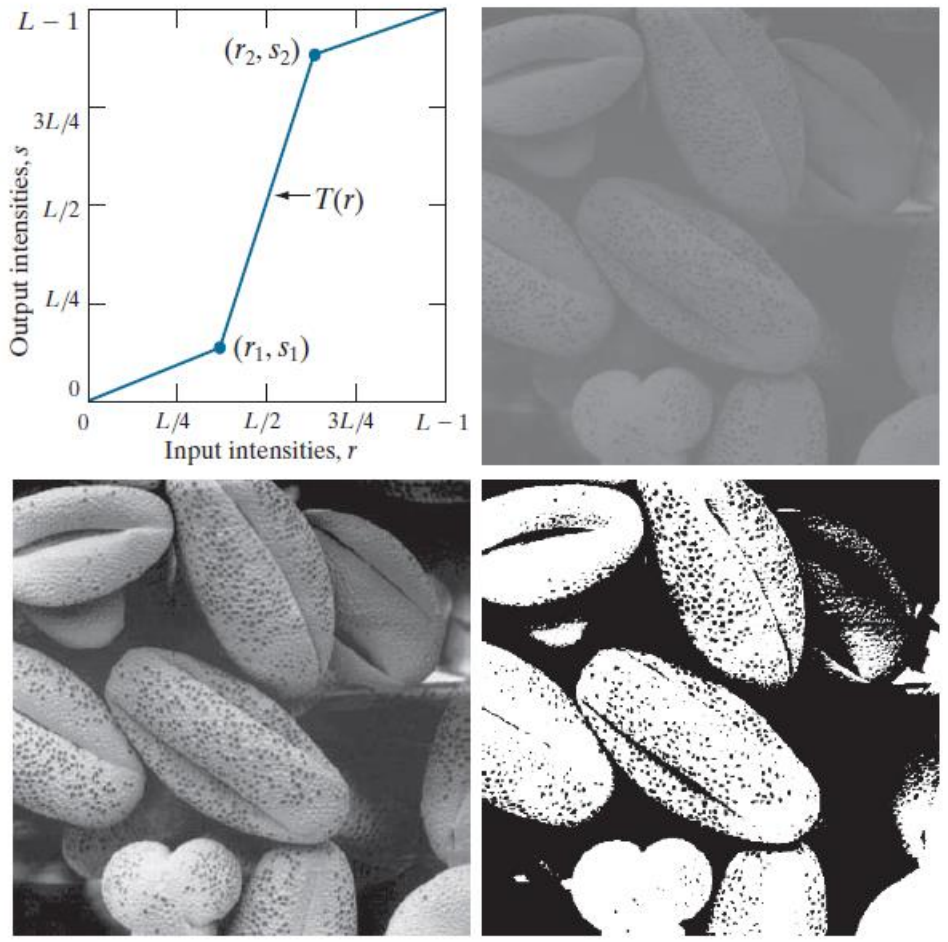
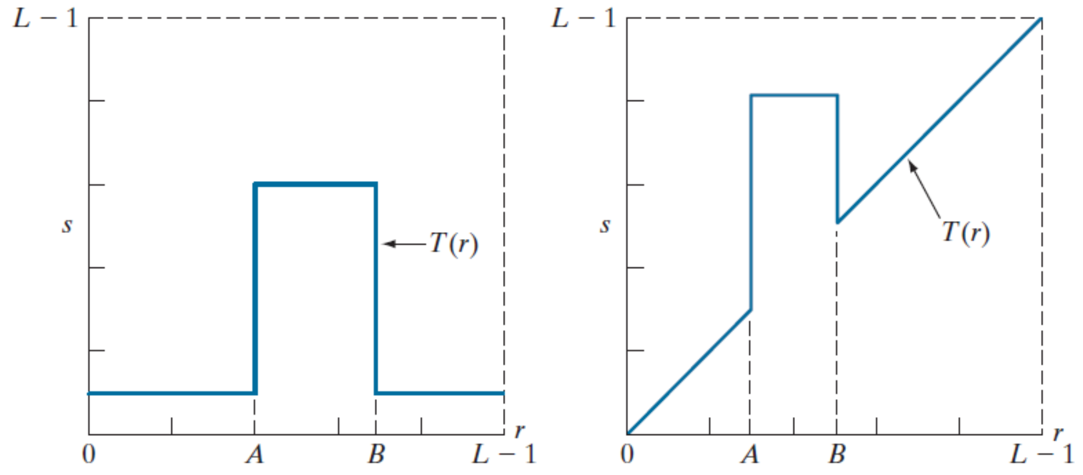


FIGURE 3.10 Contrast stretching. (a) Piecewise linear transformation function. (b) A low-contrast electron microscope image of pollen, magnified 700 times. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Intensity-Level Slicing

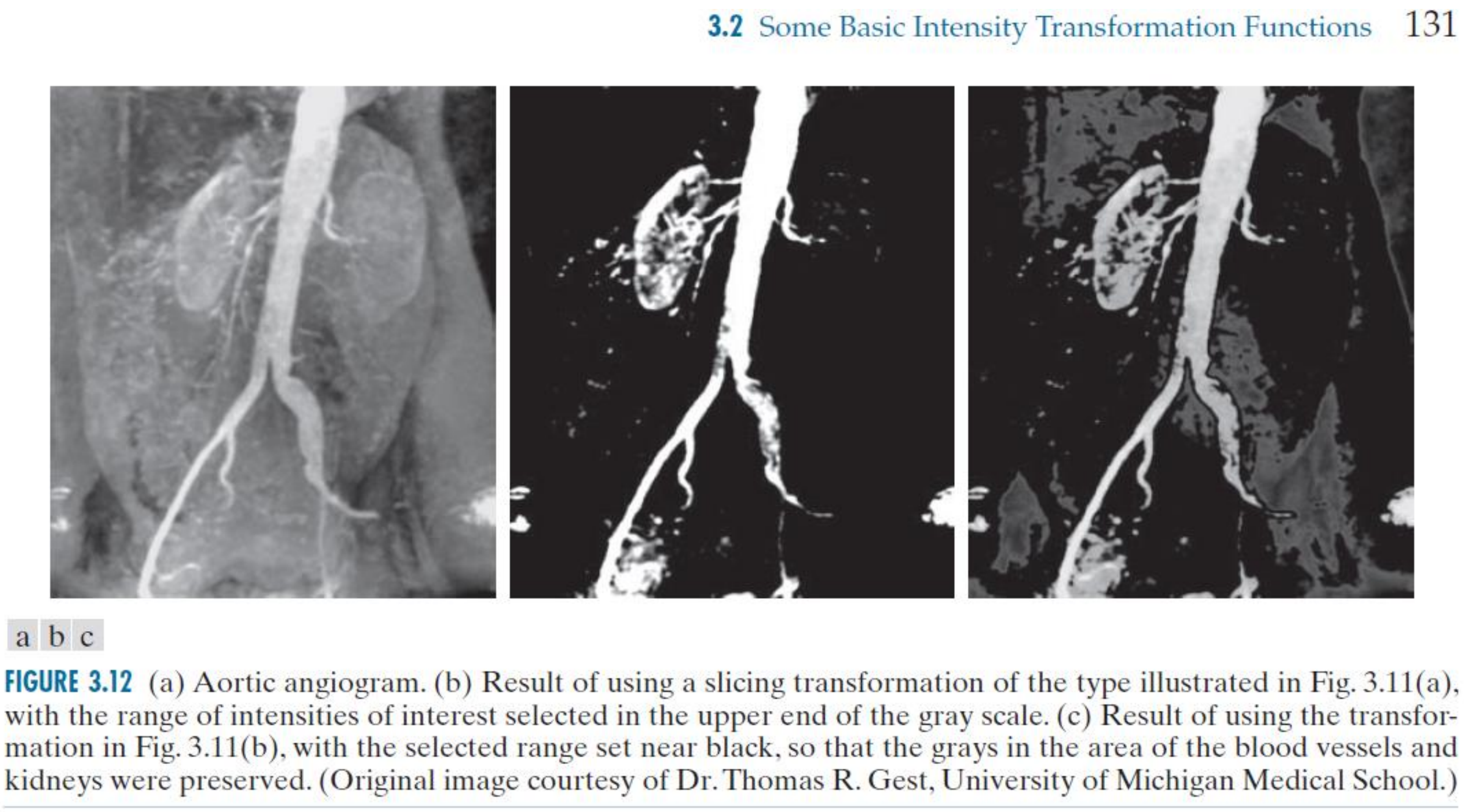
- There are applications in which it is of interest to highlight a specific range of intensities in an image. Some of these applications include enhancing features in satellite imagery, such as masses of water, and enhancing flaws in X-ray images.
- The method, called *intensity-level slicing*, can be implemented in several ways, but most are variations of two basic themes.
  - One approach is to display in one value (say, white) all the values in the range of interest and in another (say, black) all other intensities. This transformation, shown in Fig. 3.11(a), produces a binary image.
  - The second approach, based on the transformation in Fig. 3.11(b), brightens (or darkens) the desired range of intensities, but leaves all other intensity levels in the image unchanged.



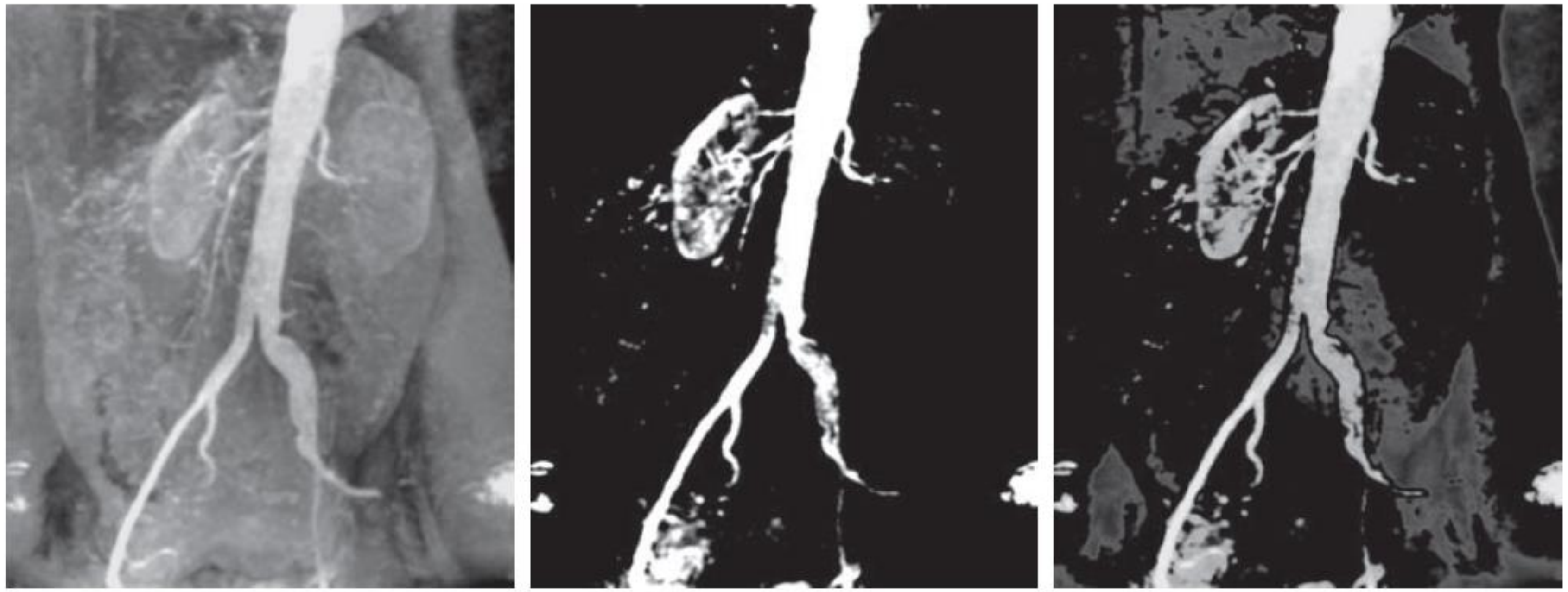
a b

FIGURE 3.11 (a) This transformation function highlights range $[A, B]$ and reduces all other intensities to a lower level. (b) This function highlights range $[A, B]$ and leaves other intensities unchanged.

Intensity-Level Slicing-- Example



3.2 Some Basic Intensity Transformation Functions 131

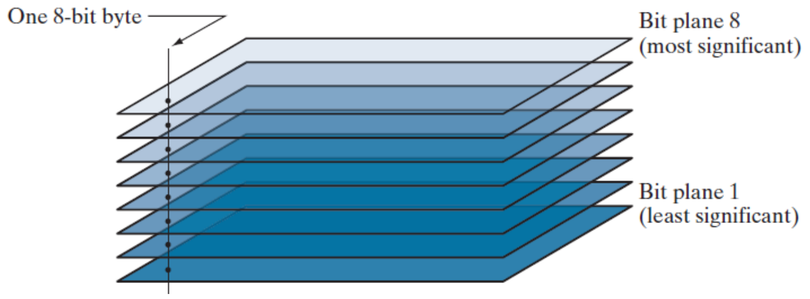


a b c

FIGURE 3.12 (a) Aortic angiogram. (b) Result of using a slicing transformation of the type illustrated in Fig. 3.11(a), with the range of intensities of interest selected in the upper end of the gray scale. (c) Result of using the transformation in Fig. 3.11(b), with the selected range set near black, so that the grays in the area of the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

Bit-Plane Slicing

FIGURE 3.13 Bit-planes of an 8-bit image.



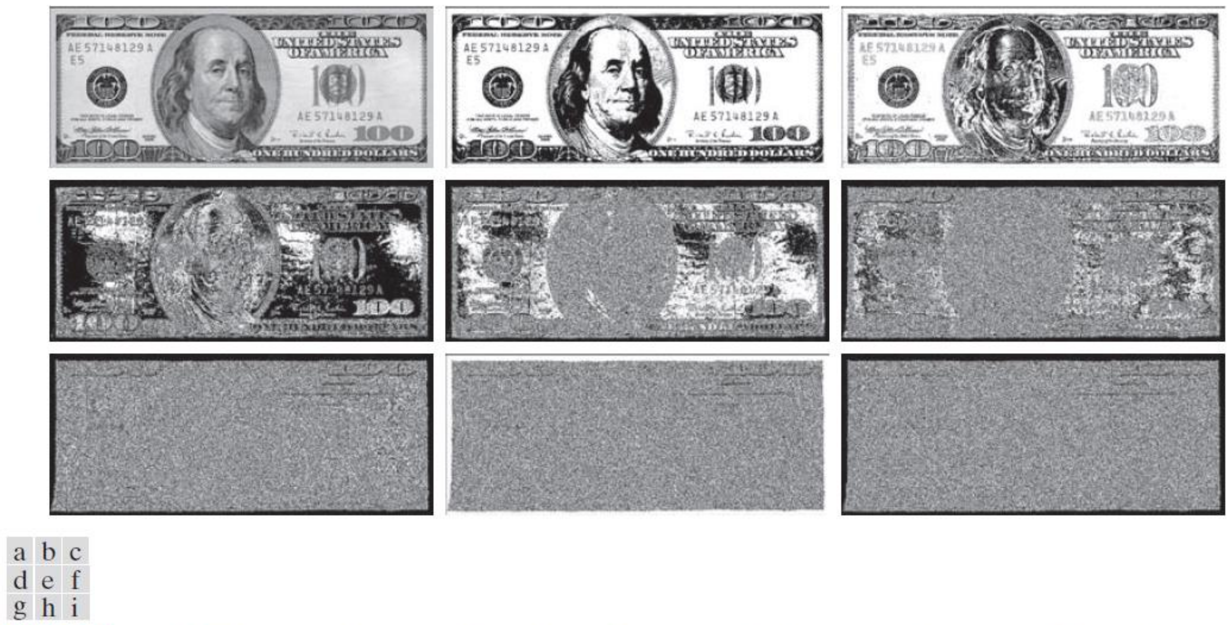


FIGURE 3.14 (a) An 8-bit gray-scale image of size 550×1192 pixels. (b) through (i) Bit planes 8 through 1, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image..

Histogram Processing

- The number of pixels having the same gray level
  - intensity distribution
- Useful for
  - – gathering image statistics
  - – spatial domain (real-time) processing
- • How to get the histogram
  - for (i=0; i<L; i++) hist[i]=0;
  - for (i=0; i<N; i++) hist[r]++;
- Histogram manipulation is a fundamental tool in image processing.
- Histograms are simple to compute and are also suitable for fast hardware implementations, thus making histogram-based techniques a popular tool for real-time image processing.

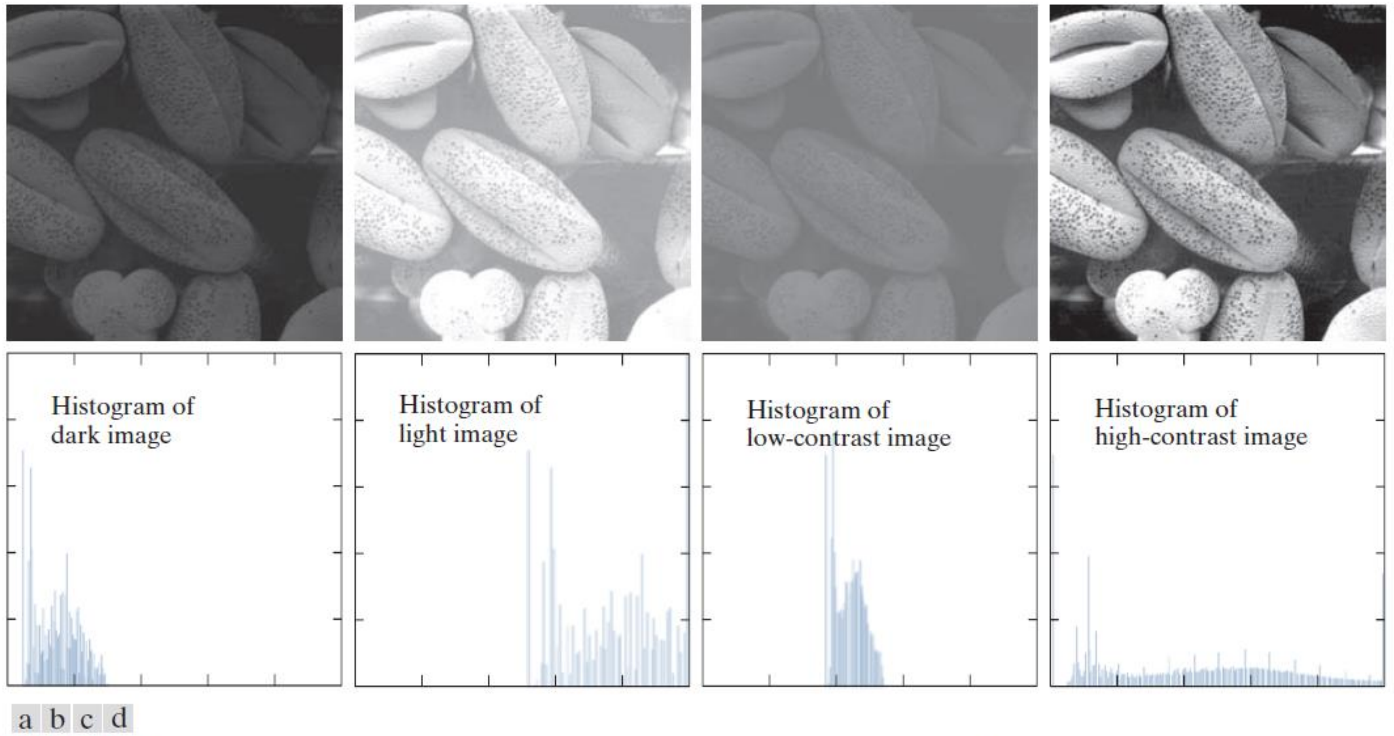


FIGURE 3.16 Four image types and their corresponding histograms. (a) dark; (b) light; (c) low contrast; (d) high contrast. The horizontal axis of the histograms are values of r_k and the vertical axis are values of $p(r_k)$.

Histogram Equalization

- $s = T(r)$
- PDF : $p_r(r), p_s(s)$

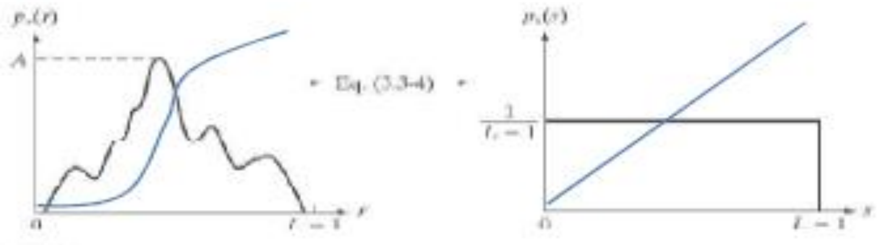
- Cumulative distribution function

$$s = T(r) = (L-1)\int_0^r p_r(w)dw$$

$$\frac{ds}{dr} = \frac{dT(s)}{dr} = (L-1)\frac{d}{dr}[\int_0^r p_r(w)dw] = (L-1)p_r(r)$$

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| = \frac{1}{L-1}$$

$p_s(s)$ is always a uniform PDF.



a b

FIGURE 3.18 (a) An arbitrary PDF. (b) Result of applying the transformation in Eq. (3.3-4) to all intensity levels r. The resulting intensities s have a uniform PDF independently of the form of the PDF of the r's.

- For discrete values, PDF → normalized histogram (divided by the total number of pixels)

$$s = T(r) = (L-1)\sum_{j=0}^{r} p_r(j)$$ → **Histogram Equalization**

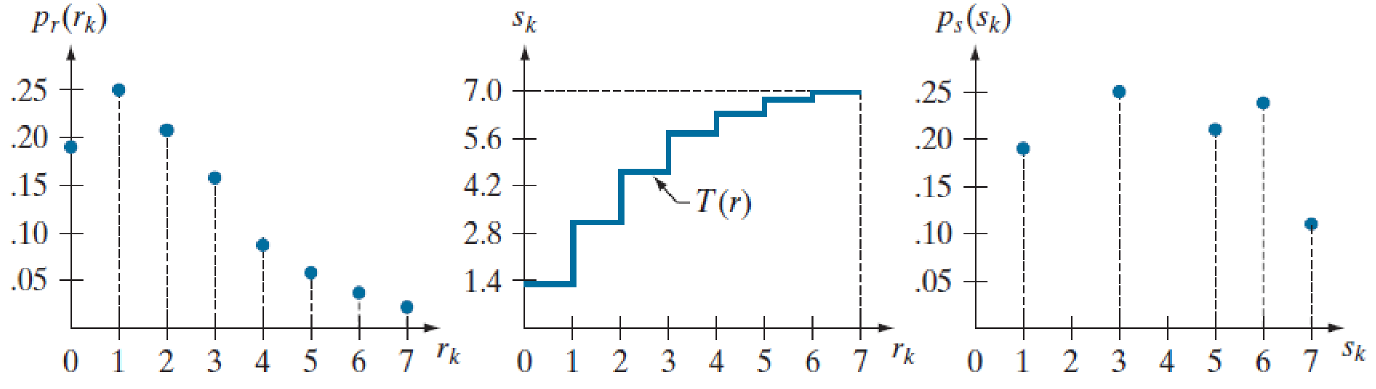
Example

TABLE 3.1
Intensity distribution and histogram values for a 3-bit, 64 × 64 digital image.

r_k	n_k	$p_r(r_k) = n_k/MN$
$r_0 = 0$	790	0.19
$r_1 = 1$	1023	0.25
$r_2 = 2$	850	0.21
$r_3 = 3$	656	0.16
$r_4 = 4$	329	0.08
$r_5 = 5$	245	0.06
$r_6 = 6$	122	0.03
$r_7 = 7$	81	0.02

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) \qquad k = 0,1,2,\ldots,L-1$$

$$s_0 = 1.33 \rightarrow 1 \quad s_2 = 4.55 \rightarrow 5 \quad s_4 = 6.23 \rightarrow 6 \quad s_6 = 6.86 \rightarrow 7$$
$$s_1 = 3.08 \rightarrow 3 \quad s_3 = 5.67 \rightarrow 6 \quad s_5 = 6.65 \rightarrow 7 \quad s_7 = 7.00 \rightarrow 7$$



a b c
FIGURE 3.19 Histogram equalization. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

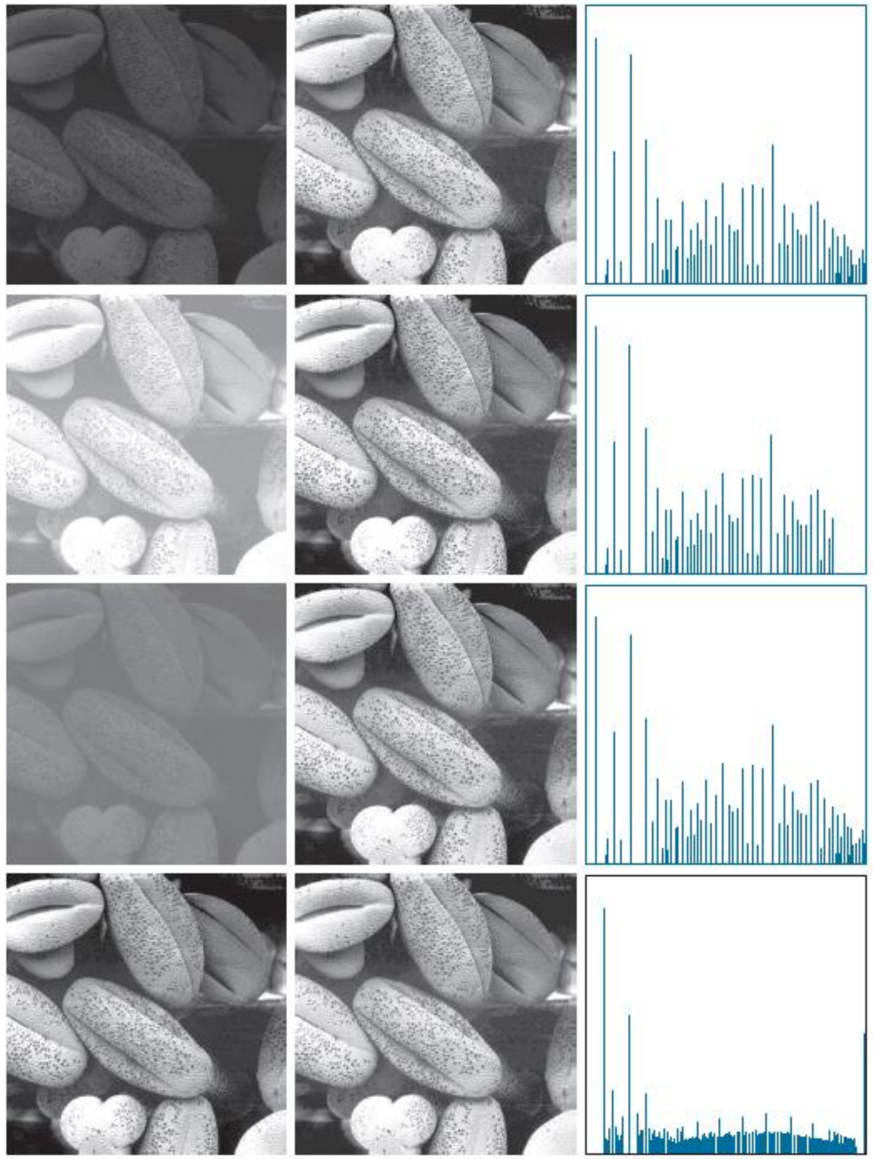
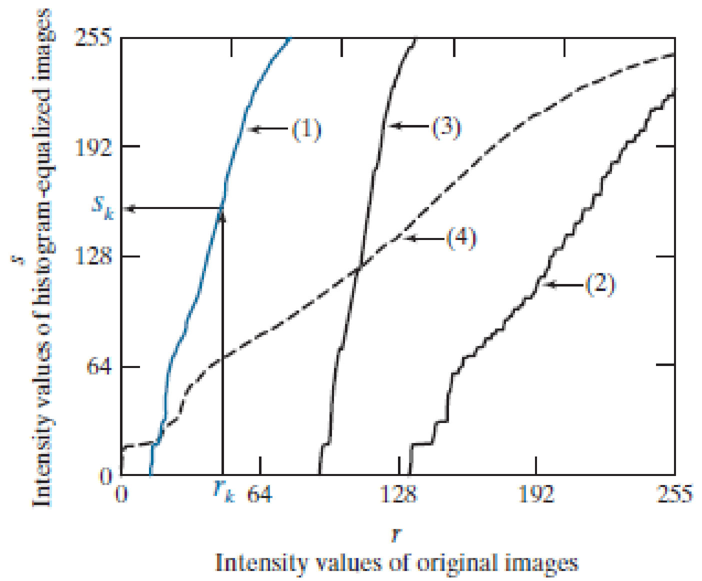


FIGURE 3.20 Left column: Images from Fig. 3.16. Center column: Corresponding histogram-equalized images. Right column: histograms of the images in the center column (compare with the histograms in Fig. 3.16).

FIGURE 3.21 Transformation functions for histogram equalization. Transformations (1) through (4) were obtained using Eq. (3-15) and the histograms of the images on the left column of Fig. 3.20. Mapping of one intensity value r_k in image 1 to its corresponding value s_k is shown.



Histogram Matching (Specification)

- The method used to generate images that have a specified histogram is called histogram matching or *histogram specification.*
  - Given histogram $p_z(z)$
  - $s = T(r) = (L-1)\int_0^r p_r(w)dw$: histogram-equalized

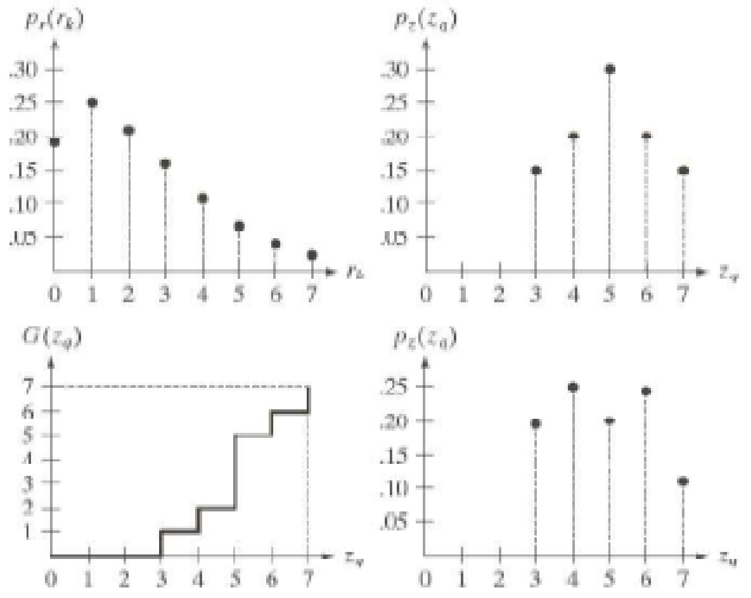
    $s = G(z) = (L-1)\int_0^z p_z(w)dw$

    $z = G^{-1}[T(r)] = G^{-1}(s)$

r	n	$p_r(r)$	s
0	790	0.19	1
1	1023	0.25	3
2	850	0.21	5
3	656	0.16	6
4	329	0.08	6
5	245	0.06	7
6	122	0.03	7
7	81	0.02	7

z	$p_z(z)$	$s=G(z)$
0	0.00	0
1	0.00	0
2	0.00	0
3	0.15	1
4	0.20	2
5	0.30	5
6	0.20	6
7	0.15	7

r	s	z
0	1	3
1	3	4
2	5	5
3	6	6
4	6	6
5	7	7
6	7	7
7	7	7



Using Histogram Statistics For Image Enhancement

- Statistics obtained directly from an image histogram can be used for image enhancement.
- Let *r* denote a discrete random variable representing intensity values in the range [0,*L* − 1], and let *p*(r_i) denote the normalized histogram component corresponding to intensity value r_i .
  - Where *p*(r_i) as an estimate of the probability that intensity r_i occurs in the image from which the histogram was obtained.

For an image with intensity levels in the range $[0, L-1]$, the nth moment of r about its mean, m, is defined as

$$\mu_n = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i) \qquad (3\text{-}24)$$

where m is given by

$$m = \sum_{i=0}^{L-1} r_i p(r_i) \qquad \textit{global} \text{ mean} \qquad (3\text{-}25)$$

The mean is a measure of average intensity and the variance (or standard deviation, σ), given by

$$\sigma^2 = \mu_2 = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i) \qquad (3\text{-}26)$$

is a measure of image contrast -- variance

Using Histogram Statistics For Image Enhancement

- We consider two uses of the mean and variance for enhancement purposes. The *global* mean and variance [Eqs. (3-25) and (3-26)] are computed over an entire image and are useful for gross adjustments in overall intensity and contrast.
- A more powerful use of these parameters is in *local enhancement*, where the *local* mean and variance are used as the basis for making changes that depend on image characteristics in a neighborhood about each pixel in an image.
- Let (*x*, *y*) denote the coordinates of any pixel in a given image, and let S_{xy} denote a neighborhood of specified size, centered on (*x*, *y*). The mean value of the pixels in this neighborhood is given by the expression, where $p_{S_{xy}}$ is the histogram of the pixels in region S_{xy}

$$m_{S_{xy}} = \sum_{i=0}^{L-1} r_i p_{S_{xy}}(r_i)$$

- The variance of the pixels in the neighborhood is similarly given by $\sigma_{S_{xy}}^2 = \sum_{i=0}^{L-1} (r_i - m_{S_{xy}})^2 p_{S_{xy}}(r_i)$
- Let *f* (*x*, *y*) denote the value of an image at any image coordinates (*x*, *y*), and let *g*(*x*, *y*) be the corresponding value in the enhanced image at those coordinates. Then,

$$g(x,y) = \begin{cases} C f(x,y) & \text{if } k_0 m_G \le m_{S_{xy}} \le k_1 m_G \text{ AND } k_2 \sigma_G \le \sigma_{S_{xy}} \le k_3 \sigma_G \\ f(x,y) & \text{otherwise} \end{cases} \qquad (3\text{-}29)$$

Examples

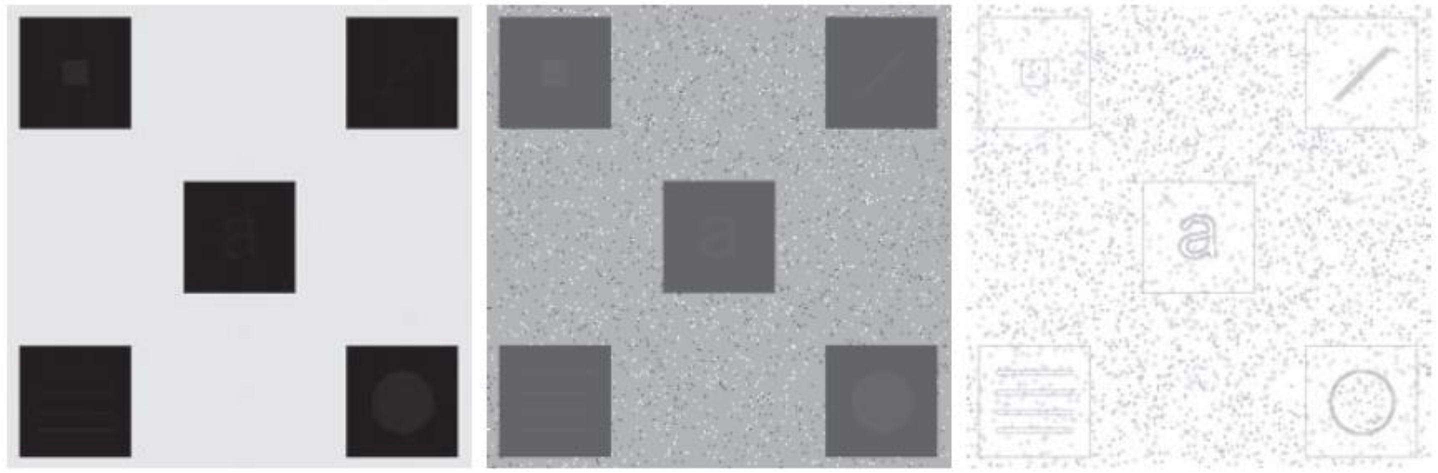

a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization.

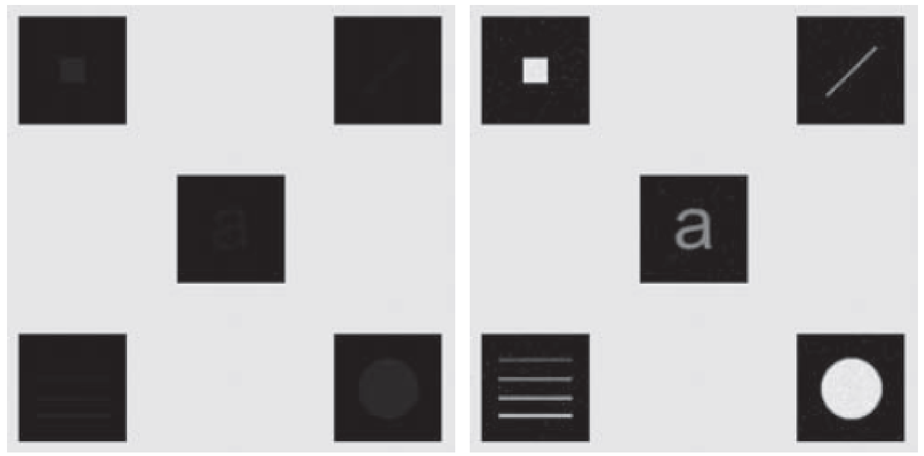

a b

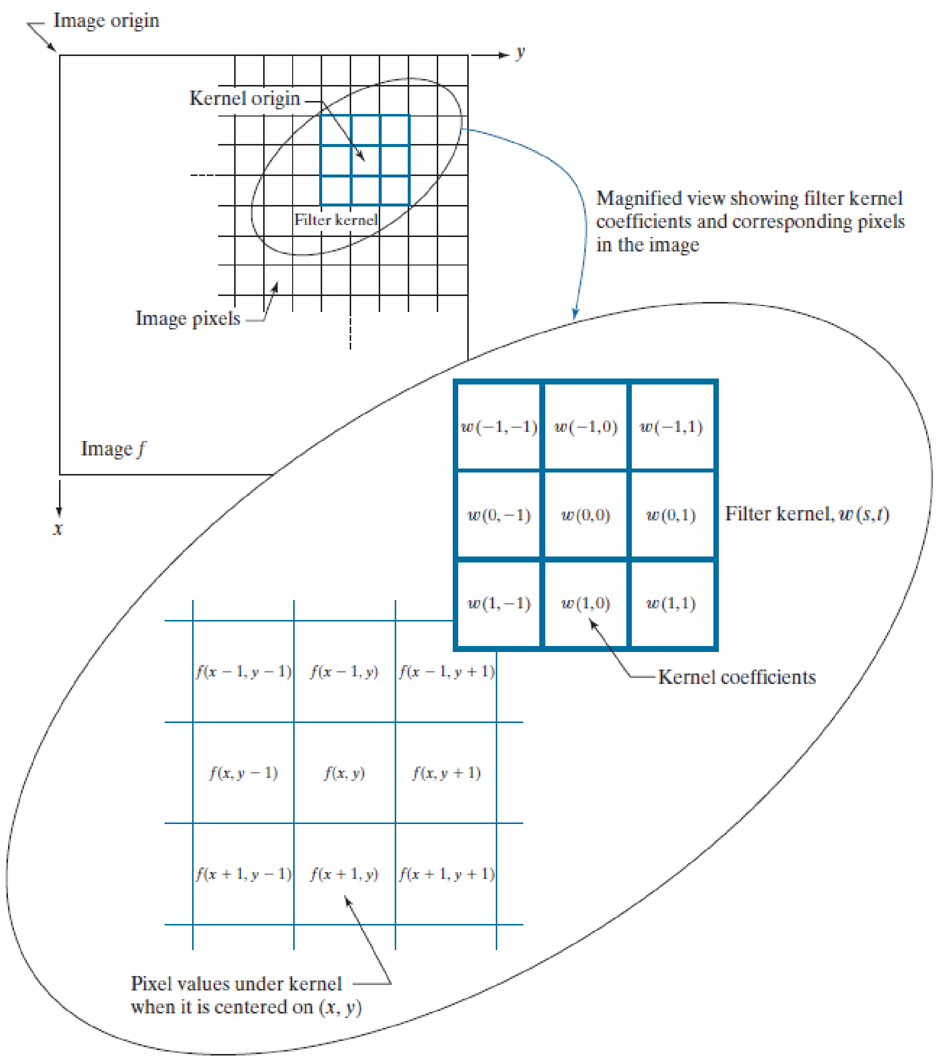
FIGURE 3.27 (a) Original image. (b) Result of local enhancement based on local histogram statistics. Compare (b) with Fig. 3.26(c).

Fundamentals of Spatial Filtering

- The name *filter* is borrowed from frequency domain processing where “filtering” refers to passing, modifying, or rejecting specified frequency components of an image.
  - For example, a filter that passes low frequencies is called a *lowpass filter*.
  - The net effect produced by a lowpass filter is to smooth an image by blurring it.
  - We can accomplish similar smoothing directly on the image itself by using *spatial filters*.
- Spatial filtering modifies an image by replacing the value of each pixel by a function of the values of the pixel and its neighbors.
  - If the operation performed on the image pixels is linear, then the filter is called a *linear spatial filter*. Otherwise, the filter is a *nonlinear spatial filter.*

Linear Spatial Filter

- A linear spatial filter performs a sum-of-products operation between an image *f* and a *filter kernel*, *w*.
- The *kernel* is an array whose size defines the neighborhood of operation, and whose coefficients determine the nature of the filter.
- Other terms used to refer to a spatial filter kernel are *mask*, *template*, and *window*. We use the term *filter kernel* or simply *kernel*.
- The fig. shows linear spatial filtering using a 3 × 3 kernel.
- At any point (*x*, *y*) in the image, the response, *g*(*x*, *y*), of the filter is the sum of products of the kernel coefficients and the image pixels encompassed by the kernel:



$$g(x,y) = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \dots + w(0,0)f(x,y) + \dots + w(1,1)f(x+1,y+1)$$

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \longrightarrow$$

linear spatial filtering of an image of size $M \times N$ with a kernel of size $m \times n$

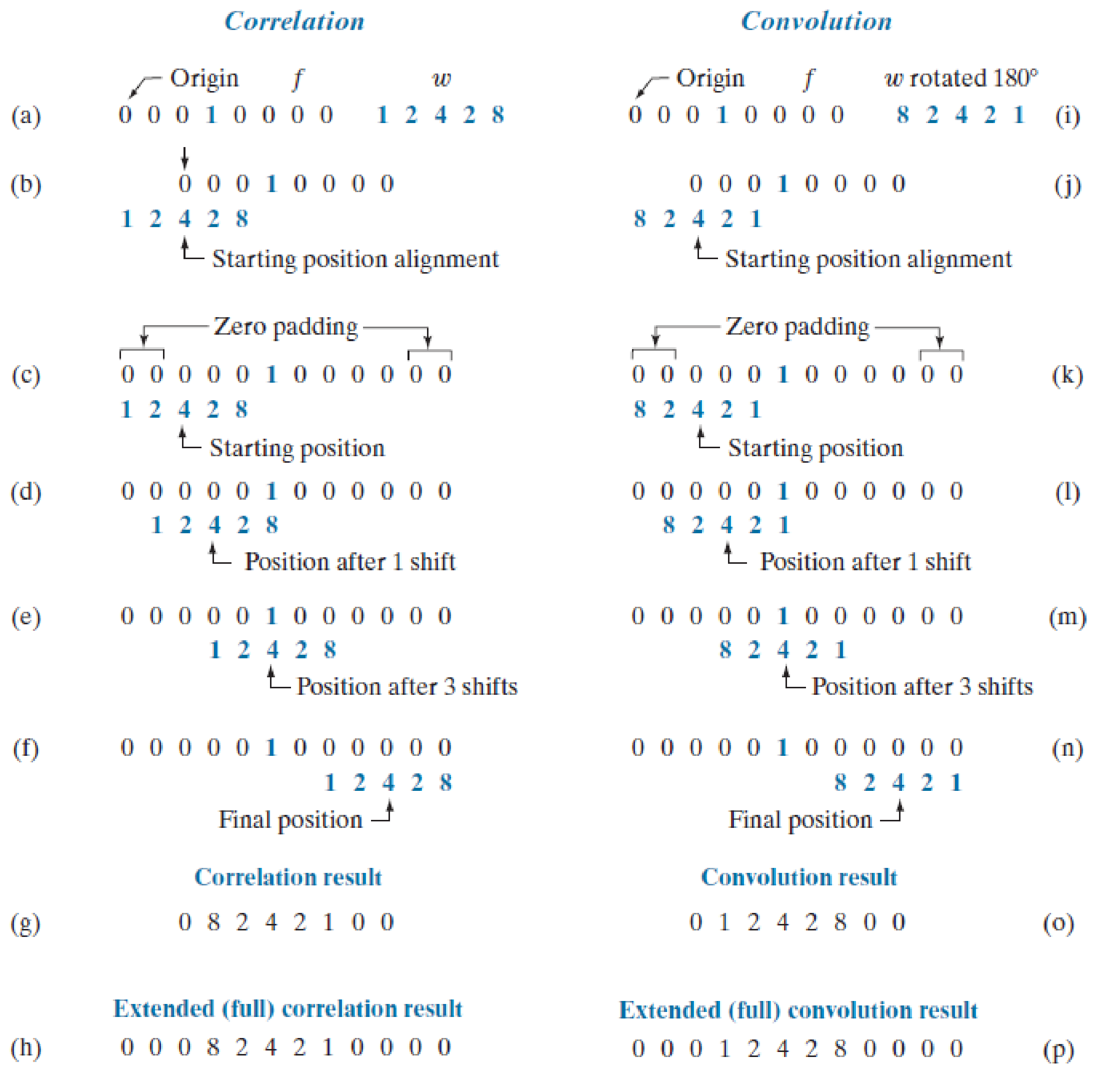
Spatial Correlation and Convolution

- Correlation consists of moving the center of a kernel over an image, and computing the sum of products at each location.
- The mechanics of *spatial convolution* are the same, except that the correlation kernel is rotated by 180°.
- For 1-D correlation and convolution,

$$g(x) = \sum_{s=-a}^{a} w(s) f(x+s)$$

- The kernel is of size 1 × 5, so *a* = 2 and *b* = 0 in this case.
- The first position

$$g(0) = \sum_{s=-2}^{2} w(s) f(s+0) = 0$$



Correlation
Convolution
Origin
f
w
w rotated 180°
(a) 0 0 0 1 0 0 0 0   1 2 4 2 8
(i) 0 0 0 1 0 0 0 0   8 2 4 2 1
Starting position alignment
Zero padding
Starting position
Position after 1 shift
Position after 3 shifts
Final position
Correlation result
(g) 0 8 2 4 2 1 0 0
Convolution result
(o) 0 1 2 4 2 8 0 0
Extended (full) correlation result
(h) 0 0 0 8 2 4 2 1 0 0 0 0
Extended (full) convolution result
(p) 0 0 0 1 2 4 2 8 0 0 0 0

Example

Origin f

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

w

$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}$$

(a)

Padded f

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

(b)

Initial position for w

$$\begin{array}{|ccc|cccc} \hline 1 & 2 & 3 & 0 & 0 & 0 & 0 \\ 4 & 5 & 6 & 0 & 0 & 0 & 0 \\ 7 & 8 & 9 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

(c)

Correlation result

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 8 & 7 & 0 \\ 0 & 6 & 5 & 4 & 0 \\ 0 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

(d)

Full correlation result

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 9 & 8 & 7 & 0 & 0 \\ 0 & 0 & 6 & 5 & 4 & 0 & 0 \\ 0 & 0 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

(e)

Rotated w

$$\begin{array}{|ccc|cccc} \hline 9 & 8 & 7 & 0 & 0 & 0 & 0 \\ 6 & 5 & 4 & 0 & 0 & 0 & 0 \\ 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

(f)

Convolution result

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 0 \\ 0 & 4 & 5 & 6 & 0 \\ 0 & 7 & 8 & 9 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix}$$

(g)

Full convolution result

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 0 & 0 \\ 0 & 0 & 4 & 5 & 6 & 0 & 0 \\ 0 & 0 & 7 & 8 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

(h)

How to perform convolution?

- In order to perform convolution on an image, following steps should be taken.
  - Flip the mask (horizontally and vertically) only once
  - Slide the mask onto the image.
  - Multiply the corresponding elements and then add them
  - Repeat this procedure until all values of the image has been calculated
- Why Convolution
  - Convolution can achieve blurring, sharpening, edge detection, noise reduction e.t.c.

Example of Convolution

Mask

1	2	3
4	5	6
7	8	9

Flipping the mask horizontally

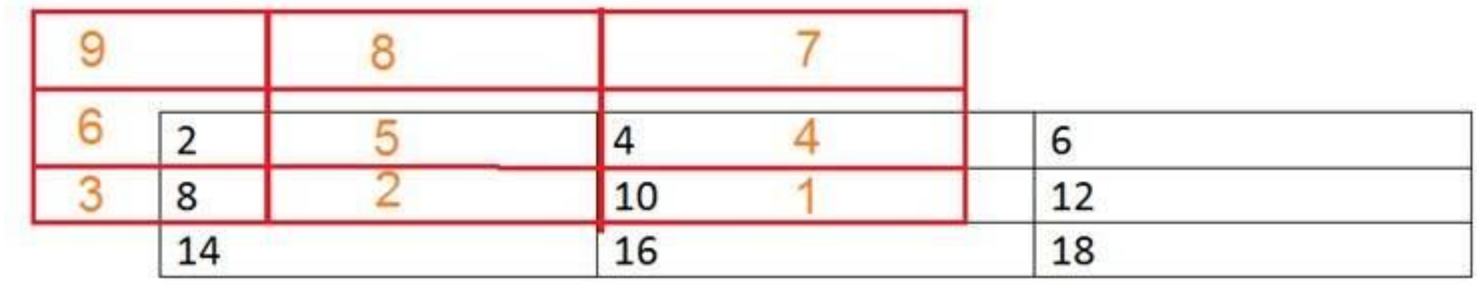
3	2	1
6	5	4
9	8	7

Flipping the mask vertically

9	*8*	*7*
6	*5*	*4*
3	*2*	*1*

Imag
e

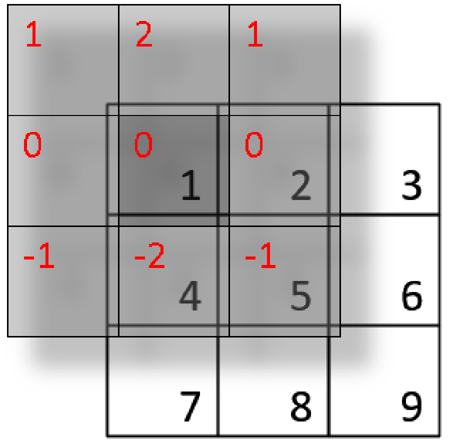
2	4	6
8	10	12
14	16	18



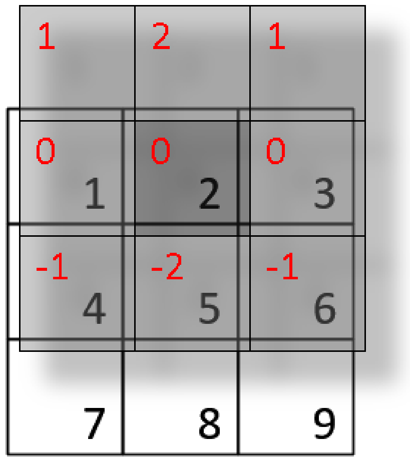
First pixel = (5*2) + (4*4) + (2*8) + (1*10) = 10 + 16 + 16 + 10 = 52

Now place 52 in the original image at the first index and repeat this procedure for each pixel of the image.

Another Example

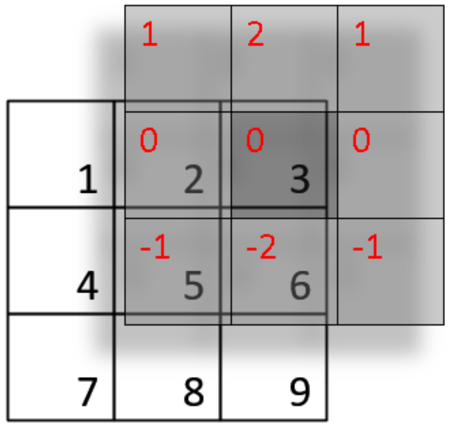


$$
\begin{aligned}
y[0,0] &= \sum_j \sum_i x[i,j] \cdot h[0-i, 0-j] \\
&= \quad x[-1,-1] \cdot h[1,1] + x[0,-1] \cdot h[0,1] + x[1,-1] \cdot h[-1,1] \\
&\quad + x[-1,0] \cdot h[1,0] \quad + x[0,0] \cdot h[0,0] \quad + x[1,0] \cdot h[-1,0] \\
&\quad + x[-1,1] \cdot h[1,-1] + x[0,1] \cdot h[0,-1] + x[1,1] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1 \\
&\quad + 0 \cdot 0 + 1 \cdot 0 + 2 \cdot 0 \\
&\quad + 0 \cdot (-1) + 4 \cdot (-2) + 5 \cdot (-1) \\
&= -13
\end{aligned}
$$



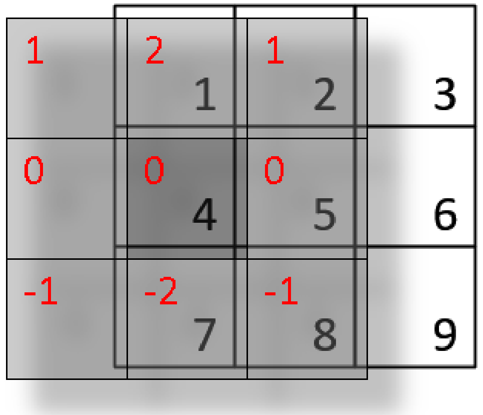
$$
\begin{aligned}
y[1,0] &= \sum_j \sum_i x[i,j] \cdot h[1-i, 0-j] \\
&= \quad x[0,-1] \cdot h[1,1] + x[1,-1] \cdot h[0,1] + x[2,-1] \cdot h[-1,1] \\
&\quad + x[0,0] \cdot h[1,0] \quad + x[1,0] \cdot h[0,0] \quad + x[2,0] \cdot h[-1,0] \\
&\quad + x[0,1] \cdot h[1,-1] + x[1,1] \cdot h[0,-1] + x[2,1] \cdot h[-1,-1] \\
&= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1 \\
&\quad + 1 \cdot 0 + 2 \cdot 0 + 3 \cdot 0 \\
&\quad + 4 \cdot (-1) + 5 \cdot (-2) + 6 \cdot (-1) \\
&= -20
\end{aligned}
$$

ANOTHER EXAMPLE



$$y[2,0] = \sum_j \sum_i x[i,j] \cdot h[2-i, 0-j]$$

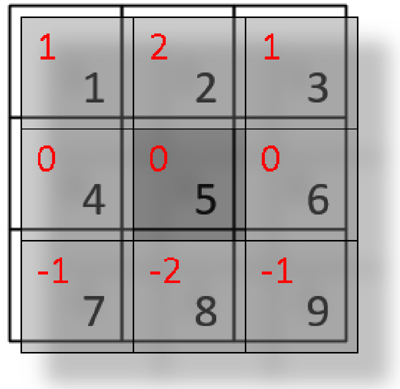
$$= \quad x[1,-1] \cdot h[1,1] + x[2,-1] \cdot h[0,1] + x[3,-1] \cdot h[-1,1]$$
$$+ x[1,0] \cdot h[1,0] \quad + x[2,0] \cdot h[0,0] \quad + x[3,0] \cdot h[-1,0]$$
$$+ x[1,1] \cdot h[1,-1] + x[2,1] \cdot h[0,-1] + x[3,1] \cdot h[-1,-1]$$
$$= \quad 0 \cdot 1 + 0 \cdot 2 + 0 \cdot 1$$
$$+ 2 \cdot 0 + 3 \cdot 0 + 0 \cdot 0$$
$$+ 5 \cdot (-1) + 6 \cdot (-2) + 0 \cdot (-1)$$
$$= -17$$



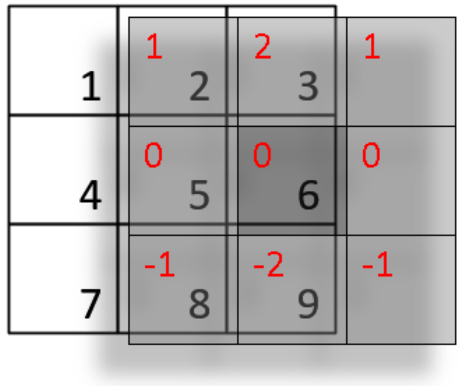
$$y[0,1] = \sum_j \sum_i x[i,j] \cdot h[0-i, 1-j]$$

$$= \quad x[-1,0] \cdot h[1,1] \quad + x[0,0] \cdot h[0,1] \quad + x[1,0] \cdot h[-1,1]$$
$$+ x[-1,1] \cdot h[1,0] \quad + x[0,1] \cdot h[0,0] \quad + x[1,1] \cdot h[-1,0]$$
$$+ x[-1,2] \cdot h[1,-1] + x[0,2] \cdot h[0,-1] + x[1,2] \cdot h[-1,-1]$$
$$= \quad 0 \cdot 1 + 1 \cdot 2 + 2 \cdot 1$$
$$+ 0 \cdot 0 + 4 \cdot 0 + 5 \cdot 0$$
$$+ 0 \cdot (-1) + 7 \cdot (-2) + 8 \cdot (-1)$$
$$= -18$$

ANOTHER EXAMPLE



$$
\begin{aligned}
y[1,1] &= \sum_j \sum_i x[i,j] \cdot h[1-i, 1-j] \\
&= \quad x[0,0]\cdot h[1,1] \quad + x[1,0]\cdot h[0,1] \quad + x[2,0]\cdot h[-1,1] \\
&\quad + x[0,1]\cdot h[1,0] \quad + x[1,1]\cdot h[0,0] \quad + x[2,1]\cdot h[-1,0] \\
&\quad + x[0,2]\cdot h[1,-1] + x[1,2]\cdot h[0,-1] + x[2,2]\cdot h[-1,-1] \\
&= \quad 1\cdot 1 + 2\cdot 2 + 3\cdot 1 \\
&\quad + 4\cdot 0 + 5\cdot 0 + 6\cdot 0 \\
&\quad + 7\cdot(-1) + 8\cdot(-2) + 9\cdot(-1) \\
&= -24
\end{aligned}
$$



$$
\begin{aligned}
y[2,1] &= \sum_j \sum_i x[i,j] \cdot h[2-i, 1-j] \\
&= \quad x[1,0]\cdot h[1,1] \quad + x[2,0]\cdot h[0,1] \quad + x[3,0]\cdot h[-1,1] \\
&\quad + x[1,1]\cdot h[1,0] \quad + x[2,1]\cdot h[0,0] \quad + x[3,1]\cdot h[-1,0] \\
&\quad + x[1,2]\cdot h[1,-1] + x[2,2]\cdot h[0,-1] + x[3,2]\cdot h[-1,-1] \\
&= \quad 2\cdot 1 + 3\cdot 2 + 0\cdot 1 \\
&\quad + 5\cdot 0 + 6\cdot 0 + 0\cdot 0 \\
&\quad + 8\cdot(-1) + 9\cdot(-2) + 0\cdot(-1) \\
&= -18
\end{aligned}
$$

ANOTHER EXAMPLE

1	2	3
4	5	6
7	8	9

$$\begin{array}{c|ccc} n \backslash m & -1 & 0 & 1 \\ \hline -1 & -1 & -2 & -1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 1 \end{array}$$

-13	-20	-17
-18	-24	-18
13	20	17

What is a mask?

- A mask is a filter.
- Concept of masking is also known as spatial filtering.
- Masking just deal with the filtering operation that is performed directly on the image.
- Mask can be a matrix which also called kernel.

What is filtering?

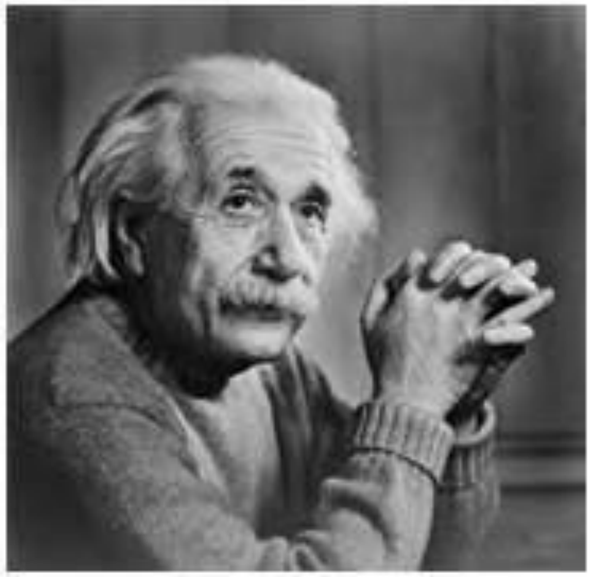
- The process of filtering is also known as convolving a mask with an image. As this process is same of convolution so filter masks are also known as convolution masks.
- Filtering process
  - The general process of filtering and applying masks is consists of moving the filter mask from point to point in an image. At each point (x,y) of the original image, the response of a filter is calculated by a pre defined relationship. All the filters values are pre defined and are a standard.
- Types of filters
  - Generally there are two types of filters. One is called as linear filters or smoothing filters and others are called as frequency domain filters.
- Why filters are used?
  - Filters are applied on image for multiple purposes. The two most common uses are as following:
    - Filters are used for Blurring and noise reduction
    - Filters are used or edge detection and sharpness

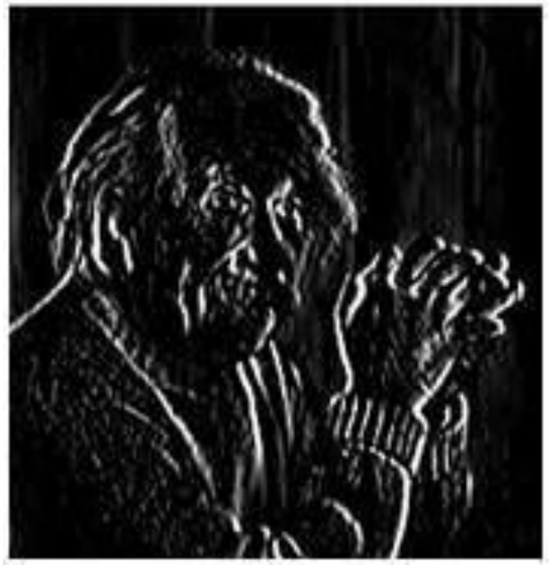
Uses of Filters

- Blurring and noise reduction
  - Filters are most commonly used for blurring and for noise reduction. Blurring is used in pre processing steps, such as removal of small details from an image prior to large object extraction.
- The common masks for blurring are.
  - Box filter
  - Weighted average filter
- In the process of blurring we reduce the edge content in an image and try to make the transitions between different pixel intensities as smooth as possible.
- Noise reduction is also possible with the help of blurring.
- Edge Detection and sharpness
  - Masks or filters can also be used for edge detection in an image and to increase sharpness of an image.

What are edges?

- We can also say that sudden changes of discontinuities in an image are called as edges. Significant transitions in an image are called as edges. A picture with edges is shown below.





Blurring

- In blurring, we simple blur an image.
- An image looks more sharp or more detailed if we are able to perceive all the objects and their shapes correctly in it.
  - For example. An image with a face, looks clear when we are able to identify eyes, ears, nose, lips, forehead e.t.c very clear.
- This shape of an object is due to its edges. So in blurring, we simple reduce the edge content and makes the transition form one color to the other very smooth.
- Blurring vs zooming
  - You might have seen a blurred image when you zoom an image. When you zoom an image using pixel replication, and zooming factor is increased, you saw a blurred image. This image also has less details, but it is not true blurring.
  - Because in zooming, you add new pixels to an image, that increase the overall number of pixels in an image, whereas in blurring, the number of pixels of a normal image and a blurred image remains the same.

Example

Filters for Blurring

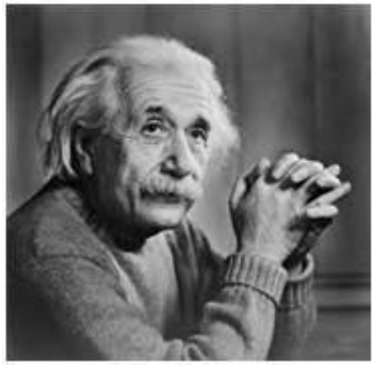
- Blurring can be achieved by many ways. The common type of filters that are used to perform blurring are.
  - Mean filter
  - Weighted average filter
  - Gaussian filter
- Mean filter
  - Mean filter is also known as Box filter and average filter. A mean filter has the following properties.
    - It must be odd ordered
    - The sum of all the elements should be 1
    - All the elements should be same
- Weighted average filter
  - In weighted average filter, we gave more weight to the center value. Due to which the contribution of center becomes more then the rest of the values. Due to weighted average filtering, we can actually control the blurring.
  - Properties of the weighted average filter are.
    - It must be odd ordered
    - The sum of all the elements should be 1
    - The weight of center element should be more then all of the other elements

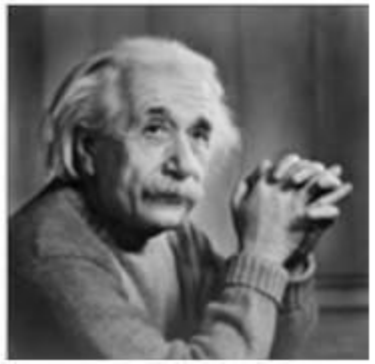
Mean filter Example

- Consider an 3×3 mean filter given below:
  - It has odd number i.e. 9 cells
  - Sum of all elements

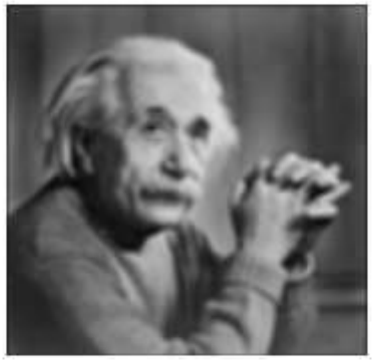
  1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 + 1/9 = 9/9 = 1
  - All elements are same.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

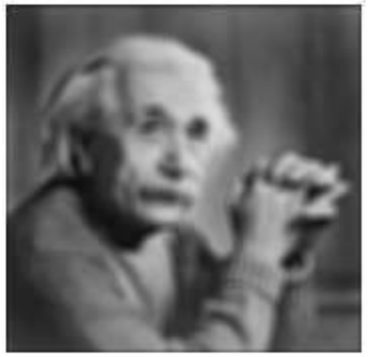




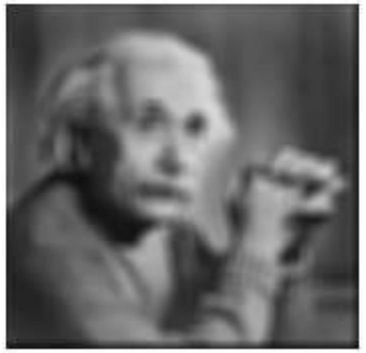
Using 5×5 filter



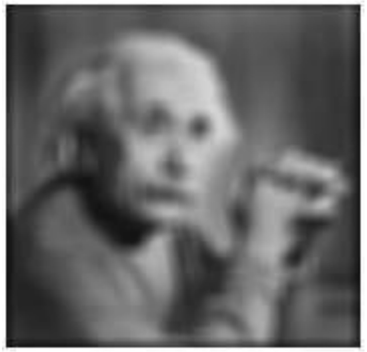
Using 7×7 filter



Using 9×9 filter



Using 11×11 filter



Weighted average filter

- In weighted average filter, we gave more weight to the center value. Due to which the contribution of center becomes more then the rest of the values. Due to weighted average filtering, we can actually control the blurring.
- Properties of the weighted average filter are.
  - It must be odd ordered
  - The sum of all the elements should be 1
  - The weight of center element should be more then all of the other elements

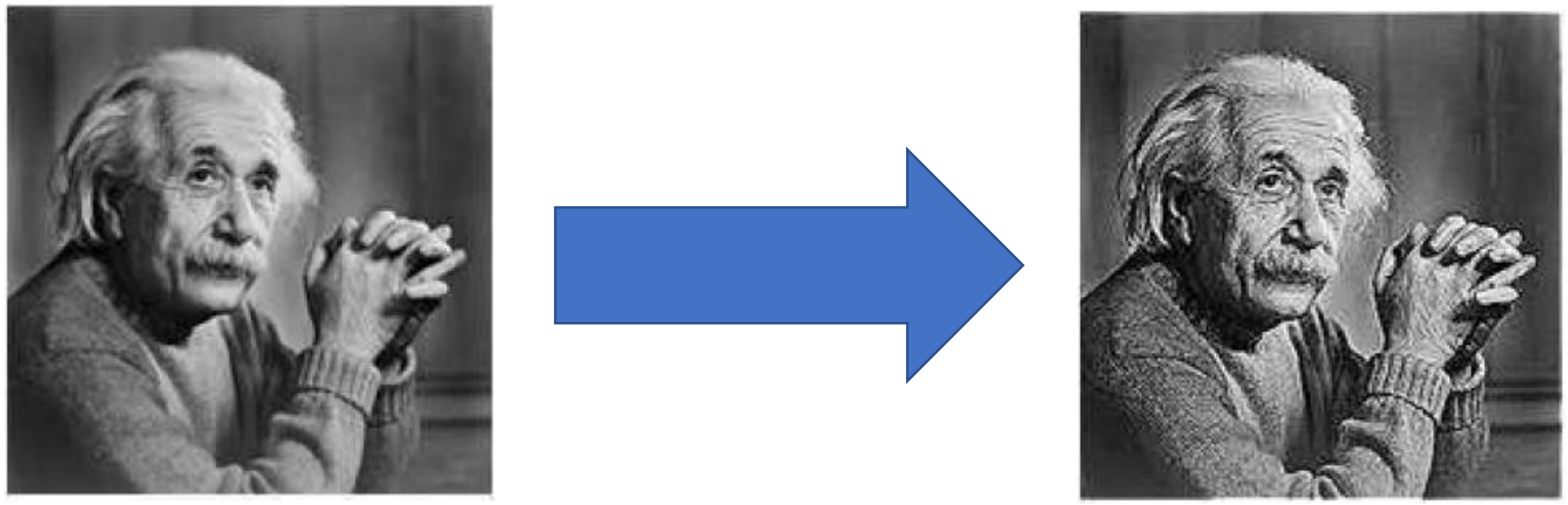
$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Concept of Edges

- We can also say that sudden changes of discontinuities in an image are called as edges.
- Significant transitions in an image are called as edges.
- **Types of edges**
  - Horizontal edges
  - Vertical Edges
  - Diagonal Edges
- **Why edges are detected?**
- Most of the shape information of an image is enclosed in edges. So first we detect these edges in an image and by using these filters and then by enhancing those areas of image which contains edges, sharpness of the image will increase and image will become clearer.
- Here are some of the masks for edge detection:
  - Prewitt Operator
  - Sobel Operator
  - Robinson Compass Masks
  - Krisch Compass Masks
  - Laplacian Operator.

Image **Sharpening**

- Sharpening is opposite to the blurring. In blurring, we reduce the edge content and in Sharpening, we increase the edge content. So in order to increase the edge content in an image, we have to find edges first.
- Edges can be find by one of the any method described above by using any operator.
- After finding edges, we will add those edges on an image and thus the image would have more edges, and it would look sharpen.
- This is one way of sharpening an image.

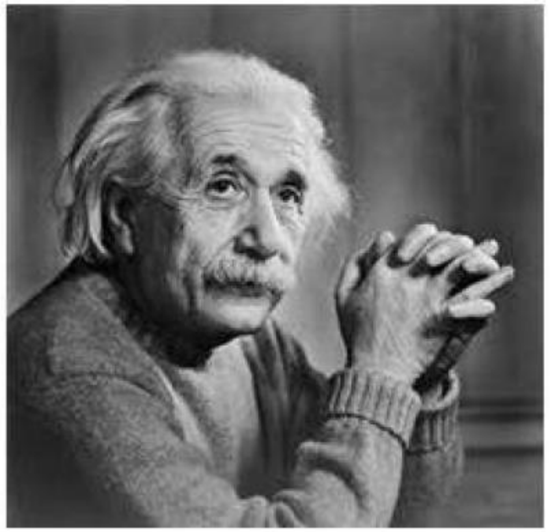


Prewitt operator

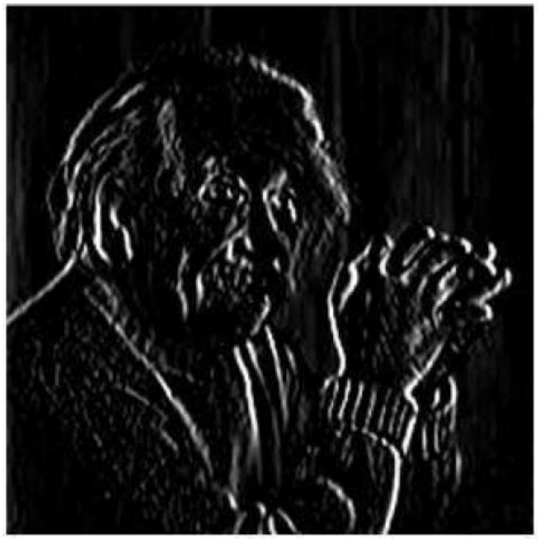
- Prewitt operator is used for edge detection in an image. It detects two types of edges
  - Horizontal edges
  - Vertical Edges
- Edges are calculated by using difference between corresponding pixel intensities of an image. All the masks that are used for edge detection are also known as derivative masks. Because image is also a signal so changes in a signal can only be calculated using differentiation. So that’s why these operators are also called as derivative operators or derivative masks.
- All the derivative masks should have the following properties:
  - Opposite sign should be present in the mask.
  - Sum of mask should be equal to zero.
  - More weight means more edge detection.
- Prewitt operator provides us two masks one for detecting edges in horizontal direction and another for detecting edges in an vertical direction.

Vertical direction USING Prewitt operator

- When we apply this mask on the image it prominent vertical edges.
- It simply works like as first order derivate and calculates the difference of pixel intensities in a edge region. As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge.
- This increase the edge intensity and it become enhanced comparatively to the original image.

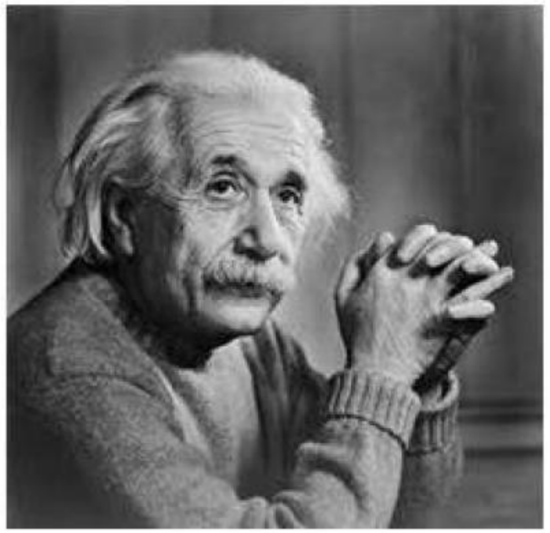


- 1	0	1
- 1	0	1
- 1	0	1

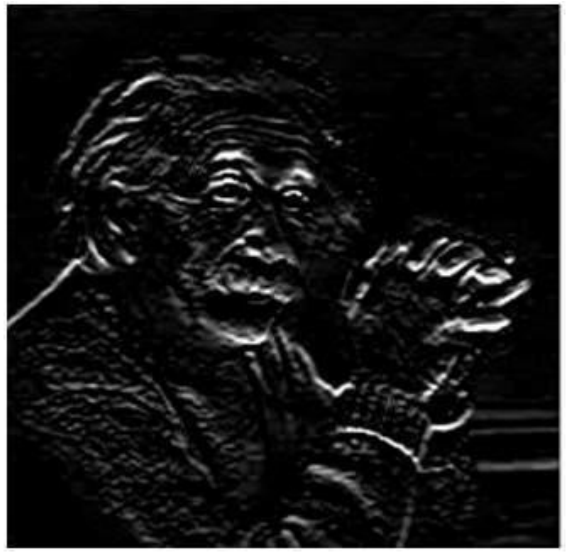


Horizontal direction USING Prewitt

- This mask will prominent the horizontal edges in an image. It also works on the principle of given mask and calculates difference among the pixel intensities of a particular edge.
- As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible.



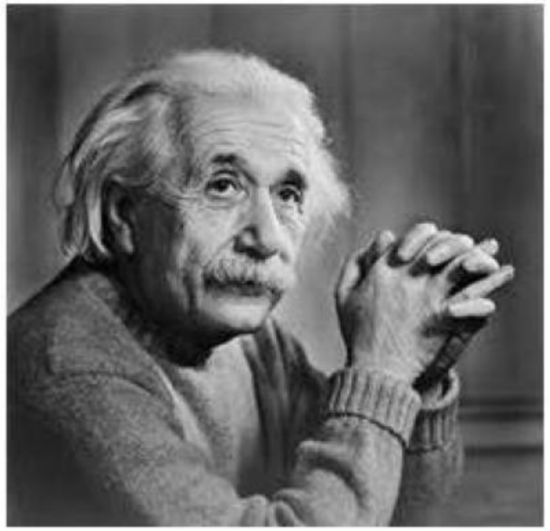
- 1	- 1	- 1
0	0	0
1	1	1



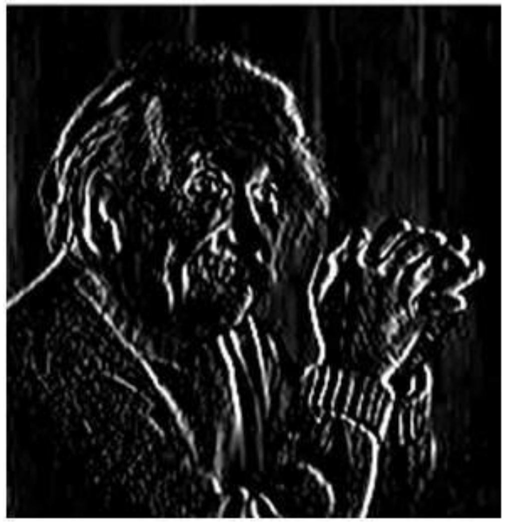
Sobel Operator

- Difference with Prewitt Operator
  - The major difference is that in Sobel operator the coefficients of masks are not fixed and they can be adjusted according to our requirement unless they do not violate any property of derivative masks.
- How vertical mask works?
  - When we apply this mask on the image it prominent vertical edges. It simply works like as first order derivate and calculates the difference of pixel intensities in a edge region.
  - As the center column is of zero so it does not include the original values of an image but rather it calculates the difference of right and left pixel values around that edge. Also the center values of both the first and third column is 2 and -2 respectively.
  - This give more weight age to the pixel values around the edge region. This increase the edge intensity and it become enhanced comparatively to the original image.
- How horizontal mask works?
  - This mask will prominent the horizontal edges in an image. It also works on the principle of above mask and calculates difference among the pixel intensities of a particular edge.
  - As the center row of mask is consist of zeros so it does not include the original values of edge in the image but rather it calculate the difference of above and below pixel intensities of the particular edge. Thus increasing the sudden change of intensities and making the edge more visible.

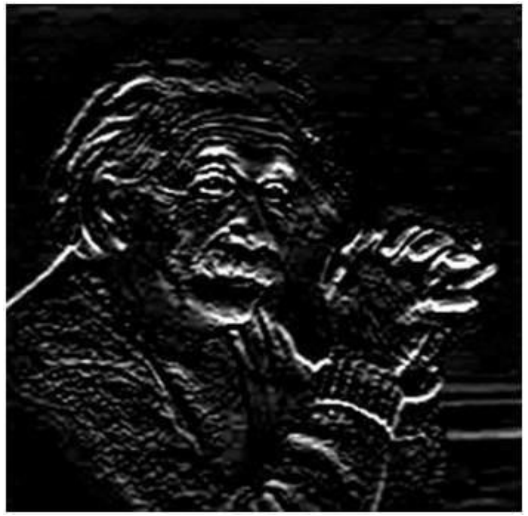
Sobel Operator example



- 1	0	1
-2	0	2
-1	0	1



-1	-2	-1
0	0	0
1	2	1



We can add more weights such as 5 instead of 2

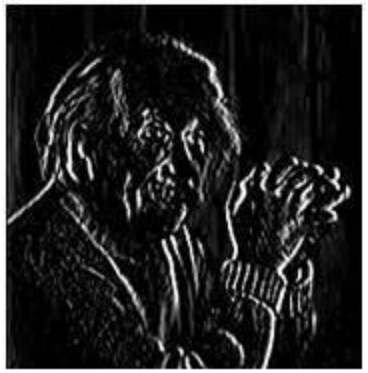
Robinson Compass Mask

- Robinson compass masks are another type of derivate mask which is used for edge detection. This operator is also known as direction mask. In this operator we take one mask and rotate it in all the 8 compass major directions that are following:
  - North
  - North West
  - West
  - South West
  - South
  - South East
  - East
  - North East
- There is no fixed mask. You can take any mask and you have to rotate it to find edges in all the above mentioned directions. All the masks are rotated on the bases of direction of zero columns.

Robinson Compass Mask example

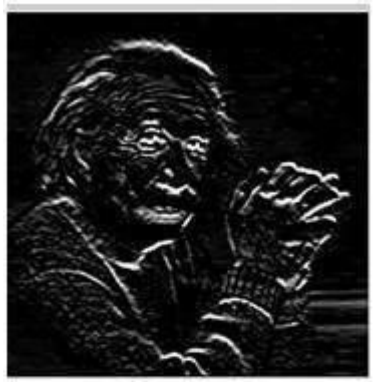
North Direction Mask

-1	0	1
-2	0	2
-1	0	1



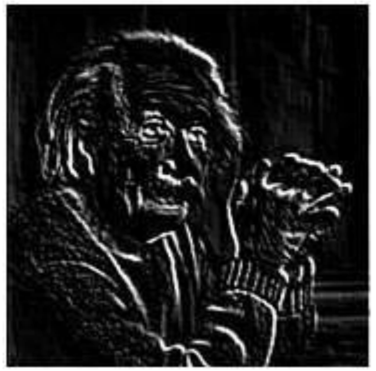
West Direction Mask

1	2	1
0	0	0
-1	-2	-1



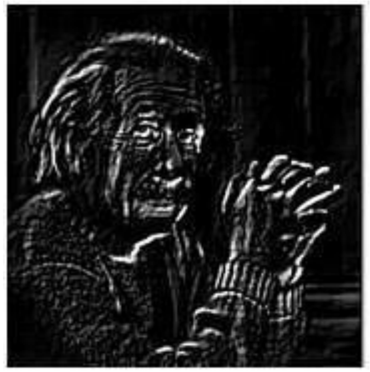
North West Direction Mask

0	1	2
-1	0	1
-2	-1	0



South West Direction Mask

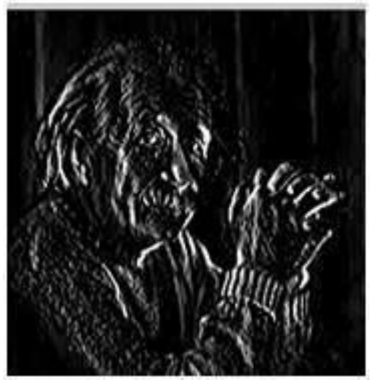
2	1	0
1	0	-1
0	-1	-2



Robinson Compass Mask example

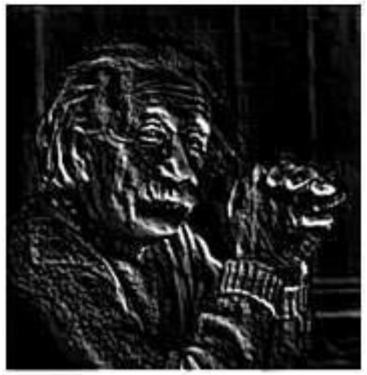
South Direction Mask

1	0	-1
2	0	-2
1	0	-1



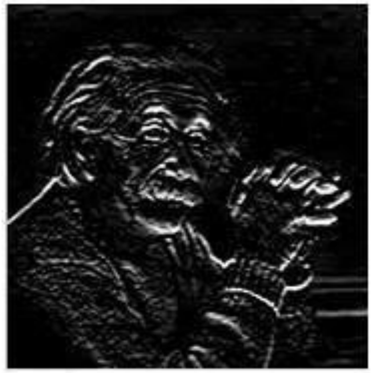
South East Direction Mask

0	-1	-2
1	0	-1
2	1	0



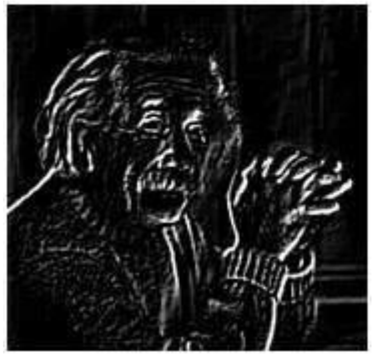
East Direction Mask

-1	-2	- 1
0	0	0
1	2	1



North East Direction Mask

-2	-1	0
-1	0	1
0	1	2

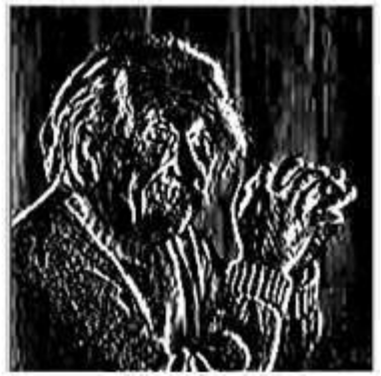


Kirsch Compass Mask

- The only difference between Robinson and kirsch compass masks is that in Robinson we have a standard mask but in Kirsch we change the mask according to our own requirements.
- By applying the masks of all direction we will get edges in all the direction.
  - Result is depends on the image. Suppose there is an image, which do not have any North East direction edges so then that mask will be ineffective.

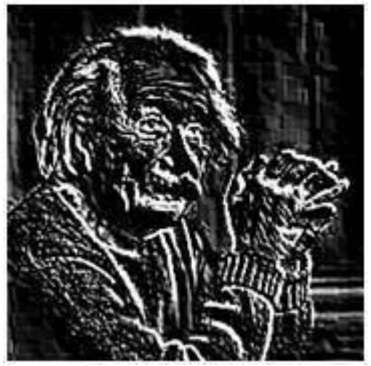
North Direction Mask

-3	-3	5
-3	0	5
-3	-3	5



North West Direction Mask

-3	5	5
-3	0	5
-3	-3	-3



Laplacian Operator

- Laplacian Operator is also a derivative operator which is used to find edges in an image. The major difference between Laplacian and other operators like Prewitt, Sobel, Robinson and Kirsch is that these all are first order derivative masks but Laplacian is a second order derivative mask.
- In this mask we have two further classifications:
  - Positive Laplacian Operator
  - Negative Laplacian Operator.
- Another difference between Laplacian and other operators is that unlike other operators Laplacian didn’t take out edges in any particular direction but it take out edges in following classification.
  - Inward Edges
  - Outward Edges

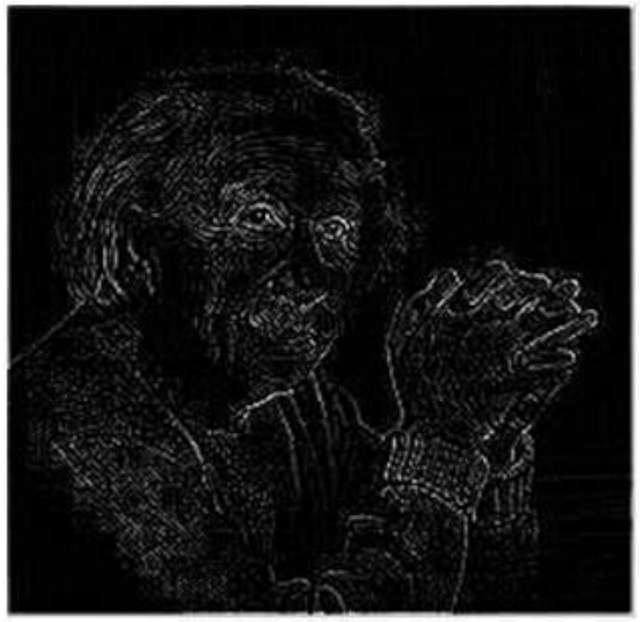
How Laplacian Operator works?

- Laplacian is a derivative operator; its uses highlight gray level discontinuities in an image and try to deemphasize regions with slowly varying gray levels. This operation in result produces such images which have grayish edge lines and other discontinuities on a dark background. This produces inward and outward edges in an image
- The important thing is how to apply these filters onto image. Remember we can’t apply both the positive and negative Laplacian operator on the same image. we have to apply just one but the thing to remember is that
  - if we apply positive Laplacian operator on the image then we subtract the resultant image from the original image to get the sharpened image.
  - Similarly if we apply negative Laplacian operator then we have to add the resultant image onto original image to get the sharpened image.

Laplacian Operator Example

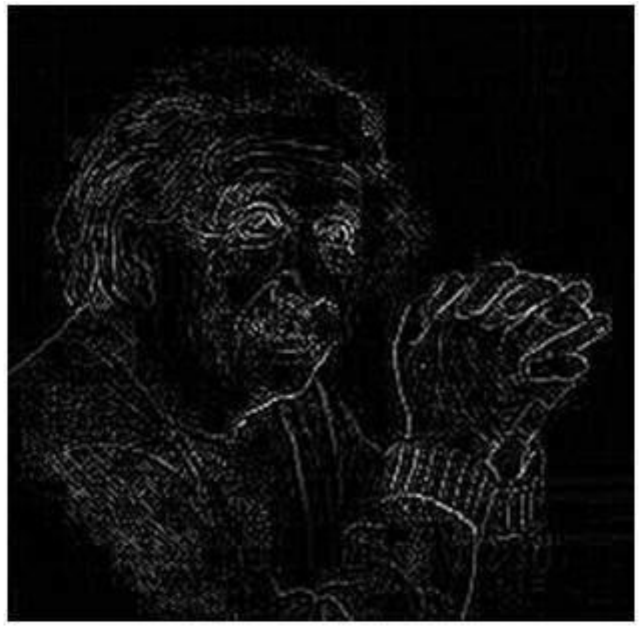
Positive Laplacian Operator

0	1	0
1	- 4	1
0	1	0



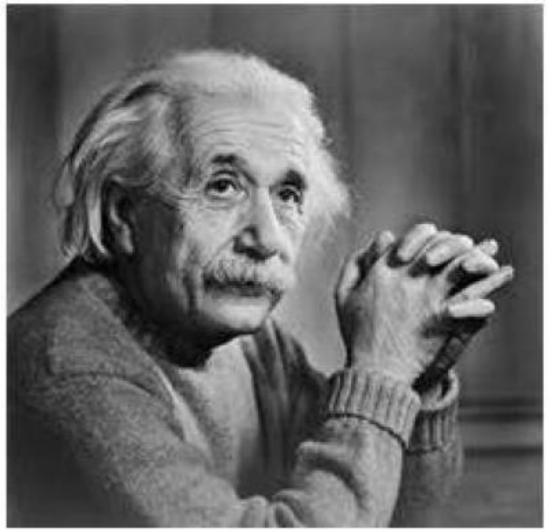
Negative Laplacian Operator

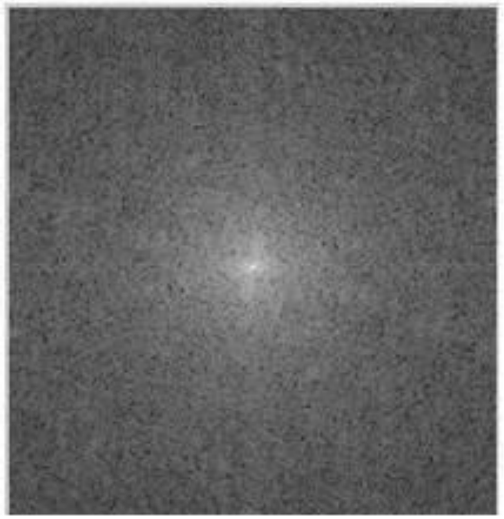
0	1	0
-1	4	-1
0	1	0



Convolution Theorem

- We consider images in spatial domain.
- Now consider the images in frequency domain. To get the frequency domain at first we have to understand the relation between frequency and spatial domain. This relationship can be explained by a theorem which is called as Convolution theorem.
  - $f(x, y) * h(x, y) \leftrightarrow F(u,v).H(u,v)$
  - $f(x, y).h(x, y) \leftrightarrow F(u,v) * H(u,v)$
  - $h(x, y) \leftrightarrow H(u,v)$

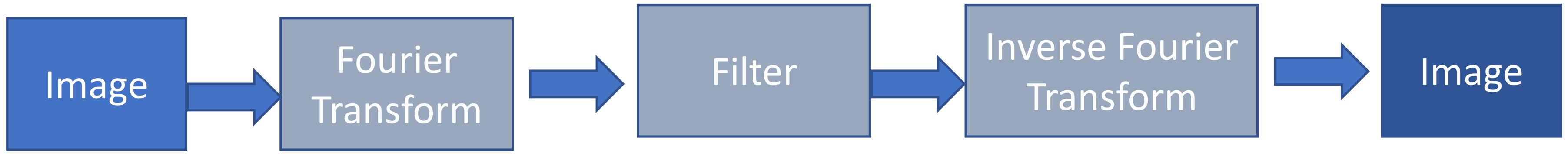




Convolution Theorem

- It can be stated as the convolution in spatial domain is equal to filtering in frequency domain and vice versa.
- The steps in filtering are given below.
  - At first step we have to do some pre – processing an image in spatial domain, means increase its contrast or brightness
  - Then we will take discrete Fourier transform of the image
  - Then we will center the discrete Fourier transform, as we will bring the discrete Fourier transform in center from corners
  - Then we will apply filtering, means we will multiply the Fourier transform by a filter function
  - Then we will again shift the DFT from center to the corners
  - Last step would be take to inverse discrete Fourier transform, to bring the result back from frequency domain to spatial domain
  - And this step of post processing is optional, just like pre processing , in which we just increase the appearance of image.

Image Filtering



- Filters
  - The concept of filter in frequency domain is same as the concept of a mask in convolution.
  - After converting an image to frequency domain, some filters are applied in filtering process to perform different kind of processing on an image. The processing include blurring an image, sharpening an image e.t.c.
- The common type of filters for these purposes are:
  - Ideal high pass filter
  - Ideal low pass filter
  - Gaussian high pass filter
  - Gaussian low pass filter

Blurring masks vs derivative masks

- **Blurring masks**
  - A blurring mask has the following properties.
  - All the values in blurring masks are positive
  - The sum of all the values is equal to 1
  - The edge content is reduced by using a blurring mask
  - As the size of the mask grow, more smoothing effect will take place
- **Derivative masks**
  - A derivative mask has the following properties.
  - A derivative mask have positive and as well as negative values
  - The sum of all the values in a derivative mask is equal to zero
  - The edge content is increased by a derivative mask
  - As the size of the mask grows , more edge content is increased

Mask VS Filtering

- Relationship between blurring mask and derivative mask with high pass filters and low pass filters.
  - Blurring masks are also called as low pass filter
  - Derivative masks are also called as high pass filter
- High pass frequency components and Low pass frequency components
  - The high pass frequency components denotes edges whereas the low pass frequency components denotes smooth regions.

Thank You