Chapter 3

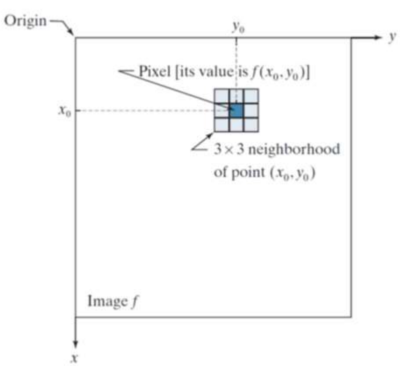
Intensity Transformations and Spatial Filtering

Intensity transformations

- Let $f(x, y)$ be an input image.
- Let $g(x, y)$ be the output image
- Let T be an operator on f defined over a neighborhood of point (x, y).
- The spatial domain processes are expressed as

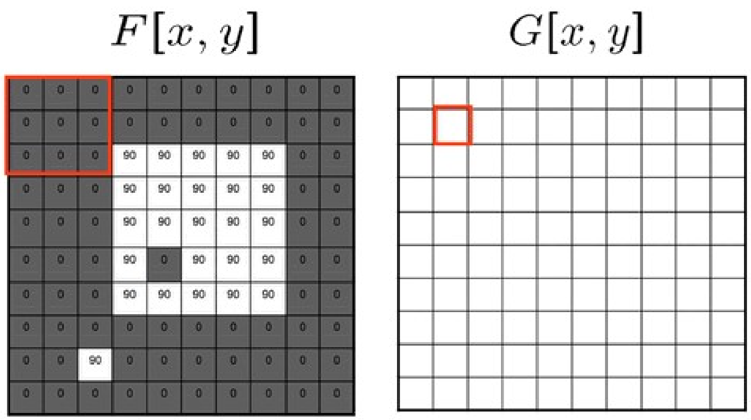
$$g(x, y) = T[f(x, y)]$$

- This operator is applied to the pixels of a single image.
- Basic implementation of on a single image is shown in figure.



- The point (x_0, y_0) is an arbitrary location in the image.
- The small region around (x_0, y_0) is a neighborhood of (x_0, y_0).
- Usually, the neighborhood is rectangular, centered on (x_0, y_0), and much smaller in size than the image.
- The center of the neighborhood is moved from pixel to pixel.
- Then the operator T is applied to the pixels in the neighborhood to obtain an output value at that location.

- The value of the output image g at (x_0, y_0) is equal to the result of applying T to the neighborhood with origin at (x_0, y_0) in f.
- For example, Let $(x_0, y_0) = (100, 150)$
- Suppose that the neighborhood is a square of size 3×3 and that operator T is "averaging" operator.
- Then the $g(100{,}150)$, is the sum of $f(100{,}150)$ and its 8-neighbors, divided by 9.
- The center of the neighborhood is then moved to the next adjacent location and the procedure is repeated to generate the next value of the output image g.
- The process starts at the top left pixel of the input image and proceeds pixel by pixel, one row at a time.

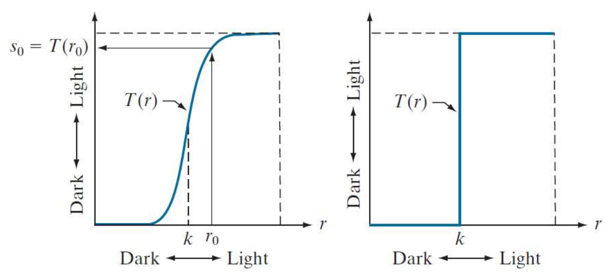


- The smallest possible neighborhood is of size 1×1.
- In this case, g depends only on the value of f at a single point (x, y) and T becomes an **intensity transformation** function of the form

$$s = T(r)$$

- s and r are intensities of g and f at any point (x, y), respectively.
- **Contrast stretching** and **thresholding** are examples of intensity transformation.
- **Contrast stretching** produce an image of higher contrast than the original, by darkening the intensity levels below k and brightening the levels above k.
- **Thresholding** produces a two level (binary) image.

Intensity transformation functions



(a) Contrast stretching function

(b) Thresholding function

Some basic intensity transformation functions

- Intensity transformations are the simplest image processing techniques.
- They are represented by $s = T(r)$ in general.
- r and s are intensities before and after applying transformation T.
- Three basic types of functions are used frequently in image processing:
  1. Linear
  2. Logarithmic
  3. Power-law

1. Image negatives using Linear intensity transformation

- The negative of an image with intensity levels in the range $[0, L-1]$ is obtained by using the negative transformation function.
- It is of the form

$$\boxed{s = L - 1 - r}$$

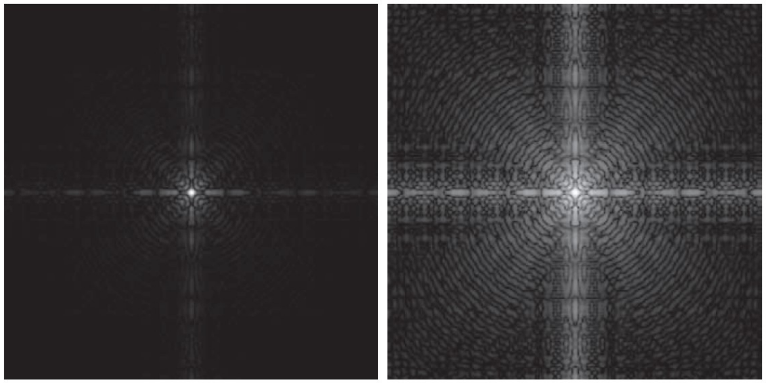
- Reversing the intensity levels of a digital image in this manner produces the equivalent of a photographic negative.
- It is used to enhance white or gray detail hidden in dark regions of an image.

2. Log transformations

- The general form of the log transformation is

$$\boxed{s = c\log(1 + r)}$$

- c is a constant and it is assumed that $r \geq 0$.
- This transformation maps a narrow range of low intensity values in the input into a wider range of output levels.
- Moreover, higher values of input levels are mapped to a narrower range in the output.
- It is used to expand the values of dark pixels in an image, while compressing the higher-level values.



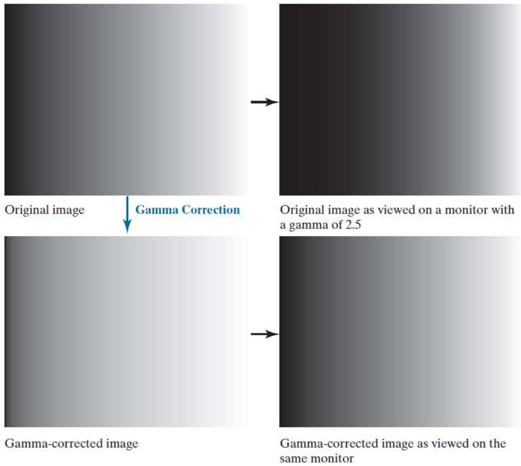
Result of applying the log transformation with $c = 1$.

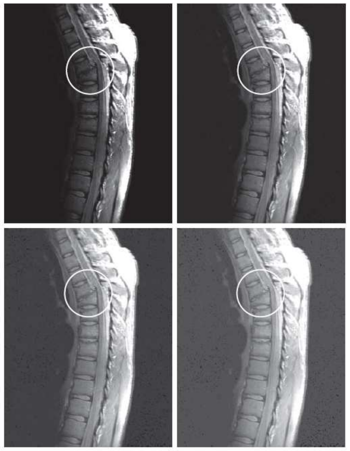
3. Power-law (gamma) transformations

- Power-law transformations have the form

$$\boxed{s = cr^{\gamma}}$$

where c and γ are positive constants.
- This maps a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels.
- The response of many devices used for image capture, printing, and display obey a power law.
- The process used to correct these power-law response phenomena is called gamma correction





- MRI of a fractured human spine.
- Results of applying Power-law (gamma) transformations with $c = 1$ and $\gamma = 0.6, 0.4$, and 0.3.

- Aerial image.
- Results of applying the Power-law (gamma) transformation with
- $\gamma = 3.0$, 4.0, and 5.0.
- $c = 1$ in all cases.

Histogram processing

- Let r_k, for $k = 0, 1, 2, \ldots, L-1$, denote the intensities of an $L-$level digital image, $f(x, y)$.
- The **unnormalized histogram** of f is defined as

$$\boxed{h(r_k) = n_k}$$

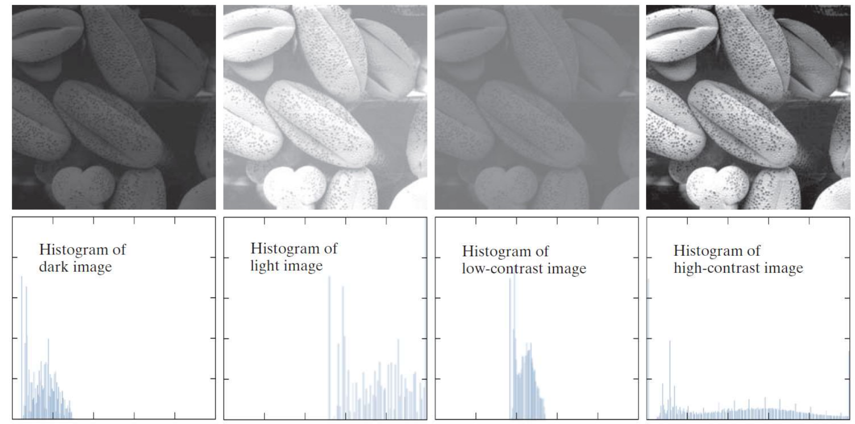
  where n_k is the number of pixels in f with intensity r_k.
- The subdivisions of the intensity scale are called **histogram bins**.
- The **normalized histogram** of f is defined as

$$\boxed{p(r_k) = \frac{h(r_k)}{MN} = \frac{n_k}{MN}}$$

- M and N are the number of image rows and columns.

- Mostly, we work with normalized histograms, which we refer to simply as histograms or image histograms.
- The sum of $p(r_k)$ for all values of k is always 1.
- The components of $p(r_k)$ are estimates of the probabilities of intensity levels occurring in an image.
- Histogram manipulation is a fundamental tool in image processing.
- Histograms are simple to compute and are also suitable for fast hardware implementations.
- Histogram-based techniques are popular tool for real-time image processing.
- Histogram shape is related to image appearance.

17



18

- In dark image, most populated histogram bins are concentrated on the lower (dark) end of the intensity scale.
- The most populated bins of the light image are biased toward the higher end of the scale.
- An image with low contrast has a narrow histogram located typically toward the middle of the intensity scale.
- The components of the histogram of the high-contrast image cover a wide range of the intensity scale.
- From this we can conclude that an image, which is tend to be distributed uniformly, will have an appearance of high contrast.

19

Fundamentals of spatial filtering

- Spatial filtering is used in a broad spectrum of image processing applications.
- "**Filtering**" refers to passing, modifying, or rejecting specified frequency components of an image.
- 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.

20

- 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**.

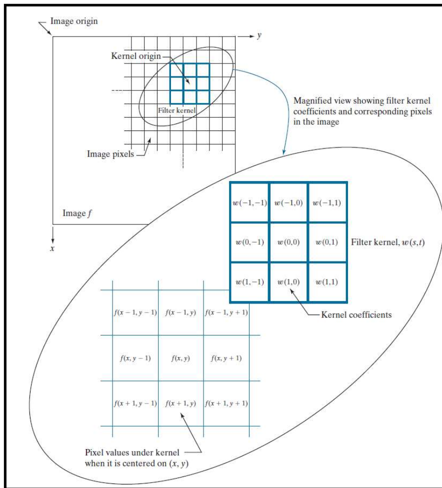
The mechanics of Linear Spatial Filtering

- A **linear spatial filter** performs a sum-of-products operation between an image f and a **filter kernel**, w.
- The kernel is an array…
  1. whose size defines the neighborhood of operation, and
  2. whose coefficients determine the nature of the filter.
- Other terms used to refer to a spatial filter kernel are **mask**, **template**, and **window**.

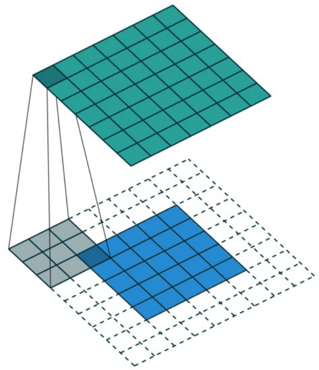
- At any point (x, y) in the image, the response, $g(x, y)$, of the filter is the sum of products of the kernel coefficients and the image pixels covered by the kernel.

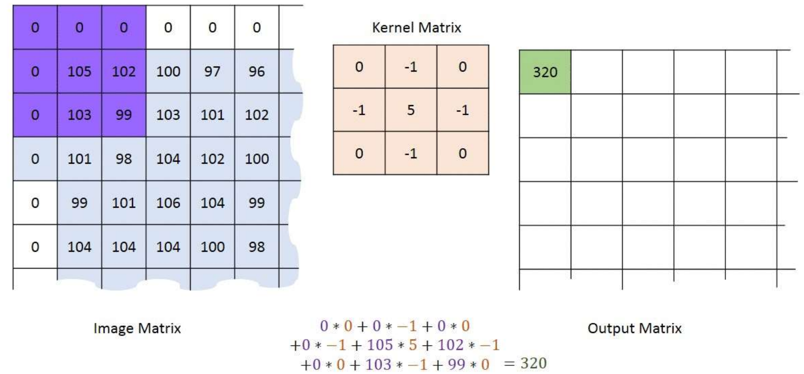
$$
\begin{aligned}
&g(x, y) \\
&= w(-1,-1)f(x-1, y-1) + w(-1,0)f(x-1, y) \\
&+ w(-1,1)f(x-1, y+1) + w(0,-1)f(x, y-1) + w(0,0)f(x, y) \\
&+ w(0,1)f(x, y+1) + w(1,-1)f(x+1, y-1) + w(1,0)f(x+1, y) \\
&+ w(1,1)f(x+1, y+1)
\end{aligned}
$$

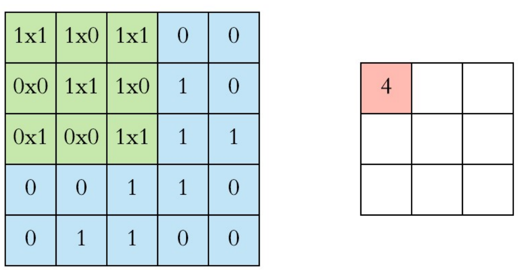
- As coordinates x and y are varied, the center of the kernel moves from pixel to pixel, generating the filtered image, g, in the process.
- Observe that the center coefficient of the kernel, $w(0,0)$, aligns with the pixel at location (x, y).
- Figure illustrates the mechanics of linear spatial filtering using a 3×3 kernel.



- The mechanics of linear spatial filtering using a 3 × 3 kernel.
- The pixels are shown as squares to simplify the graphics.
- Note that the origin of the image is at the top left, but the origin of the kernel is at its center.
- Placing the origin at the center of spatially symmetric kernels simplifies writing expressions for linear filtering.







- For a kernel of size $m \times n$, we assume that $m = 2a + 1$ and $n = 2b + 1$, where a and b are nonnegative integers.
- So there are kernels of odd size in both coordinate directions.
- In general, linear spatial filtering of an image of size $M \times N$ with a kernel of size $m \times n$ is given by the expression

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

- x and y are varied so that the center of the kernel visits every pixel in f once.
- For a fixed value of (x, y), this equation implements the sum of products, but for a kernel of arbitrary odd size.

Spatial correlation and Convolution

- **Spatial correlation** is described mathematically by

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \qquad (1)$$

- 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°.
- Thus, when the values of a kernel are symmetric about its center, correlation and convolution gives the same result.

Correlation in 1D

- In 1-D, the equation (1) becomes

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

- Following figure shows a 1-D function, f, and a kernel, w.

Origin f w

0 0 0 **1** 0 0 0 0 **1 2 4 2 8**

- The kernel is of size 1×5.
- Let so $a = 2$ and $b = 0$ in this case.

- Following figure shows the starting position used to perform correlation, in which w is positioned so that its center coefficient is coincident with the origin of f.

0 0 0 **1** 0 0 0 0

1 2 4 2 8

Starting position alignment

- The first thing we notice is that part of w lies outside f, so the summation is undefined in that area.
- A solution to this problem is to pad function f with enough 0's on either side.

Zero padding

0 0 0 0 0 **1** 0 0 0 0 0 0

1 2 4 2 8

Starting position

- In general, if the kernel is of size $1 \times m$, we need $\frac{m-1}{2}$ zeros on either side of f in order to handle the beginning and ending configurations of w with respect to f.
- The first correlation value is the sum of products in this initial position, computed using (2) with $x = 0$:

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

- This value is in the leftmost location of the correlation result.

- To obtain the second value of correlation, we shift the relative positions of w and f one pixel location to the right.

```
0 0 0 0 0 1 0 0 0 0 0 0
  1 2 4 2 8
    └ Position after 1 shift
```

- That means we consider $x = 1$ in equation (2) and compute the sum of products again.
- The result is $g(1) = 8$.
- When $x = 2$, we obtain $g(2) = 2$.
- When $x = 3$, we get $g(3) = 4$.

```
0 0 0 0 0 1 0 0 0 0 0 0
      1 2 4 2 8
        └ Position after 3 shifts
```

- Proceeding in this manner by varying x one shift at a time, we get

Correlation result

0 8 2 4 2 1 0 0

- Correlating a kernel w with a function that contains all 0's and a single 1 gives a copy of w, but rotated by 180°.
- Function that contains a single 1 with the rest being 0's is called a **discrete unit impulse**.
- Correlating a kernel with a discrete unit impulse yields a rotated version of the kernel at the location of the impulse.

Convolution in 1D

- Fig. shows the sequence of steps for performing convolution.
- The only difference here is that the kernel is pre-rotated by 180° prior to performing the shifting/sum of products operations.
- The result of pre-rotating the kernel is that now we have an exact copy of the kernel at the location of the unit impulse.

Convolution

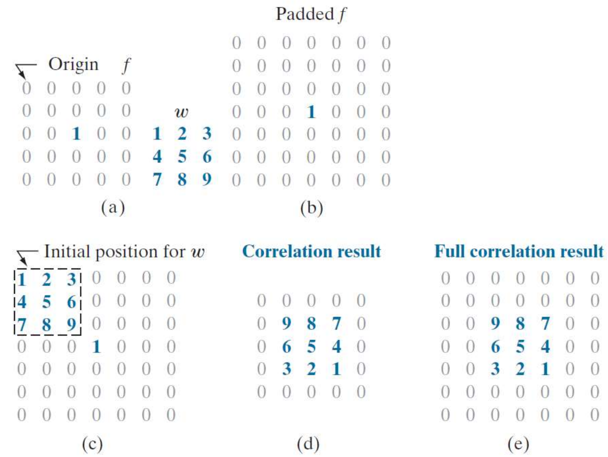
```
Origin  f           w rotated 180°
0 0 0 1 0 0 0 0     8 2 4 2 1

        0 0 0 1 0 0 0 0
8 2 4 2 1
        └ Starting position alignment

  Zero padding
0 0 0 0 0 1 0 0 0 0 0 0
8 2 4 2 1
        └ Starting position

0 0 0 0 0 1 0 0 0 0 0 0
  8 2 4 2 1
          └ Position after 1 shift

0 0 0 0 0 1 0 0 0 0 0 0
  8 2 4 2 1
          └ Position after 1 shift

0 0 0 0 0 1 0 0 0 0 0 0
      8 2 4 2 1
            └ Position after 3 shifts

0 0 0 0 0 1 0 0 0 0 0 0
              8 2 4 2 1
         Final position ┘
```

Convolution result

0 1 2 4 2 8 0 0

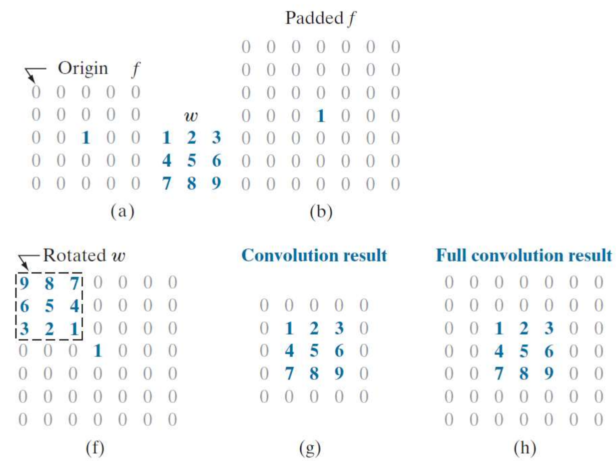
Correlation in 2D

- The 1-D concepts can be extended easily to images.
- For a kernel of size $m \times n$, we pad the image with a minimum of $\frac{m-1}{2}$ rows of 0's at the top and bottom and $\frac{n-1}{2}$ columns of 0's on the left and right.
- Suppose m and n are equal to 3.
- Then we pad f with one row of 0's above and below and one column of 0's to the left and right.
- The result is a copy of the kernel, rotated by 180°.



Convolution in 2D

- For convolution, we pre-rotate the kernel and repeat the sliding sum of products.
- Convolution of a function with an impulse copies the function to the location of the impulse.
- Correlation and convolution yield the same result if the kernel values are symmetric about the center.



Smoothing (lowpass) spatial filters

- Smoothing (also called averaging) spatial filters are used to reduce sharp transitions in intensity.
- One application of smoothing is noise reduction.
- Smoothing is used to reduce irrelevant detail in an image. “irrelevant” refers to pixel regions that are small with respect to the size of the filter kernel.
- Another application is for smoothing the false contours that result from using an insufficient number of intensity levels in an image.

41

Smoothing (lowpass) spatial filters

- Some smoothing (lowpass) spatial filters are
  1. Box filter kernels
  2. Lowpass gaussian filter kernels
  3. Order-statistic (nonlinear) filters

42

1. Box filter kernels

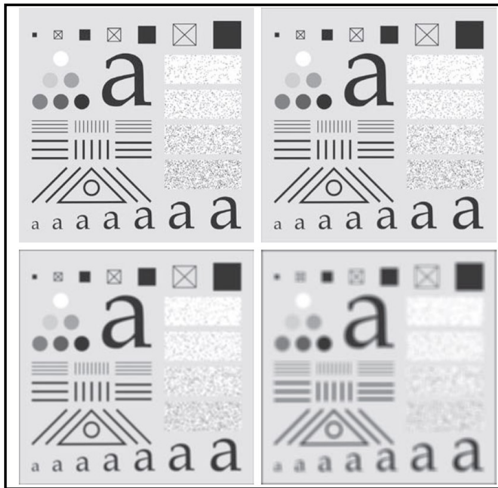
- Box kernel is a kernel whose coefficients have the same value (typically 1).
- The name “box kernel” comes from a constant kernel resembling a box when viewed in 3-D.
- A 3×3 box filter is of the form

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

43

- An $m \times n$ box filter is an $m \times n$ array of 1’s, with a normalizing constant in front.
- Value of a normalizing constant is 1 divided by the sum of the values of the coefficients i.e. $\frac{1}{mn}$ when all the coefficients are 1’s.
- Due to this, the average value of an area of constant intensity would equal that intensity in the filtered image.

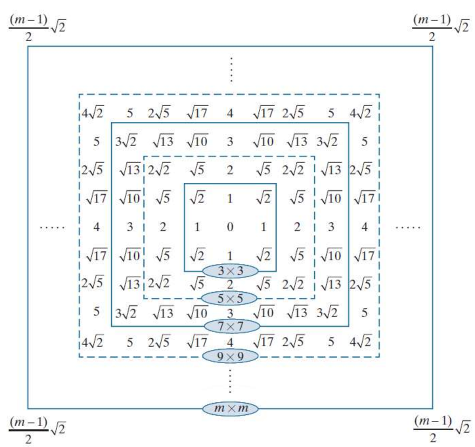
44



- First figure is a test pattern of size 1024 × 1024 pixels.
- Other 3 figures are results of lowpass filtering with box kernels of sizes, 3 × 3, 11 × 11, and 21 × 21.

2. Lowpass gaussian filter kernels

- Gaussian kernel is of the form
$$w(s,t) = Ke^{\frac{s^2+t^2}{2\sigma^2}}$$
- Variables s and t are real discrete numbers.
- Let $r = \sqrt{s^2+t^2}$ then,
$$w(s,t) = Ke^{\frac{r^2}{2\sigma^2}}$$
- Variable r is the distance from the center to any point on function w.
- Figure shows values of r for several kernel sizes using integer values for s and t.



- σ is standard deviation.
- The distance squared to the corner points for a kernel of size $m \times m$ is

$$r_{Max}^2 = \left[\frac{m-1}{2}\sqrt{2}\right]^2$$

- $\therefore r_{Max}^2 = \frac{m-1}{2}$

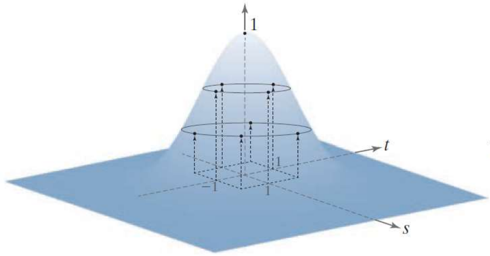
$$w(s,t) = Ke^{\frac{r^2}{2\sigma^2}}$$

- Considering $K = 1$ and $\sigma = 1$, we get,

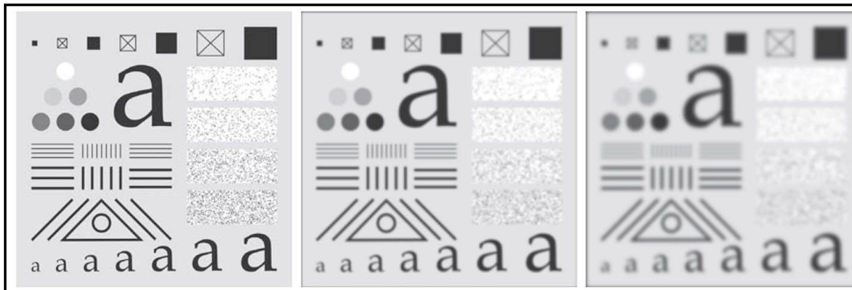
$w(s,t) = \frac{1}{4.8976} \times$

0.3679	0.6065	0.3679
0.6065	1.0000	0.6065
0.3679	0.6065	0.3679

- We can also plot the Gaussian kernel as follows:



- Gaussian kernel is also useful for smoothing.
- It produces better results as compared to the box filter.



a) A test pattern of size 1024×1024.
b) Result of a Gaussian kernel of size 21×21, with standard deviations $\sigma = 3.5$.
c) Result of using a kernel of size 43×43, with $\sigma = 7, K = 1$ in all cases.

Sharpening (highpass) spatial filters

- Sharpening highlights transitions in intensity.
- Uses of image sharpening range from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems.
- Sharpening is often referred to as highpass filtering.
- Here, high frequencies, which are responsible for fine details, are allowed to pass, while low frequencies are rejected.
- Sharpening is done using derivative operators like Laplacian operator and gradient operator.

1. Using Laplacian for image sharpening

- Let $f(x, y)$ be a function (image) of two variables.
- Then Laplacian of f is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad (1)$$

- Laplacian is a linear operator.
- Representing the derivatives of this equation in discrete form,

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \qquad (2)$$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y) \qquad (3)$$

- Using (2) and (3) in (1), we get,
  $\nabla^2 f$
  $= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$
- This equation can be implemented using convolution with the kernel,

0	1	0
1	−4	1
0	1	0

- The filtering mechanics for image sharpening are similar to lowpass filtering; we are simply using different coefficients here.

- Other Laplacian kernels are

1	1	1
1	−8	1
1	1	1

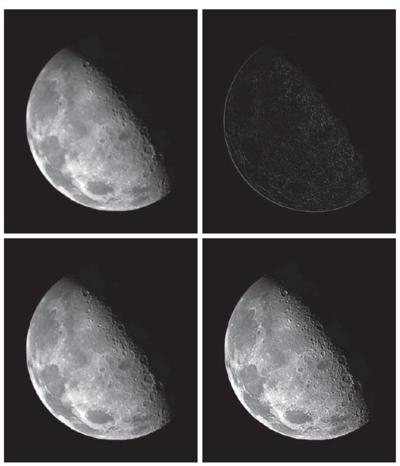
0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

- Finally sharpening is achieved using

$$g(x,y) = f(x,y) + c[\nabla^2 f(x,y)]$$

- Value of c is ± 1. For these values of c, we get different Laplacian kernels.



(a) Blurred image of the North Pole of the moon.

(b) Image obtained using the Laplacian kernel.

(c) Image sharpened with $c = -1$.

(d) Image sharpened using the same procedure, but with the different Laplacian kernel.

2. Using Gradient for image sharpening

- Let $f(x,y)$ be a function (image) of two variables.
- Then gradient of f is defined as a two-dimensional column vector

$$\nabla f = grad\,(f) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- This vector has the important geometrical property that it points in the direction of the greatest rate of change of f at location (x,y).
- This is a linear operator.

- The magnitude of gradient is given by

$$M(x,y) = \|\nabla f\| = magnitude\ (\nabla f) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

- It represents the value at (x, y) of the rate of change in the direction of the gradient vector.
- $M(x, y)$ is an image, called gradient image, and it is of the same size as the original.
- Gradient ∇f is linear but $\|\nabla f\|$ is not linear because of the squaring and square root operations.
- It is more suitable computationally to approximate the squares and square root operations by absolute values

$$M(x,y) \approx \left|\frac{\partial f}{\partial x}\right| + \left|\frac{\partial f}{\partial y}\right|$$

57

- Consider a 3×3 region of an image, where the z_i are intensity values. (Fig 1)
- The value of the center point, z_5, denotes the value of $f(x, y)$ at an arbitrary location, (x, y).
- z_1 denotes the value of $f(x-1, y-1)$; and so on.
- Then, using first order derivative approximation, we get **Roberts cross-gradient operator** (Fig 2)

$$M(x,y) \approx |z_9 - z_5| + |z_8 - z_6|$$

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

(Fig 1)

−1	0
0	1

0	−1
1	0

(Fig 2)

58

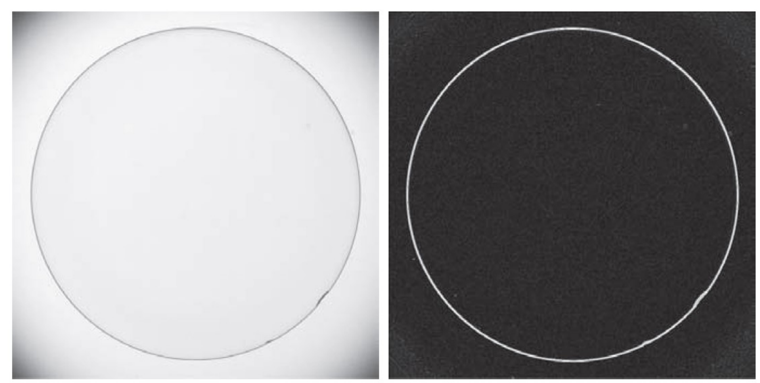
- Another gradient operator is **Sobel operator**.

$$M(x,y) = \sqrt{[(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)]^2 - [(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)]^2}$$

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

59



(a) Image of a contact lens (b) Sobel gradient

60