Digital Image Processing

Image Enhancement in the Spatial Domain

Low and High Pass Filtering

Topics

- Low Pass Filtering
  - Averaging
  - Median Filter
- High Pass Filtering
  - Edge Detection
- Line Detection

Low Pass Filtering

- Low pass filters block high frequency content of the image
- High frequency content correspond to boundaries of the objects

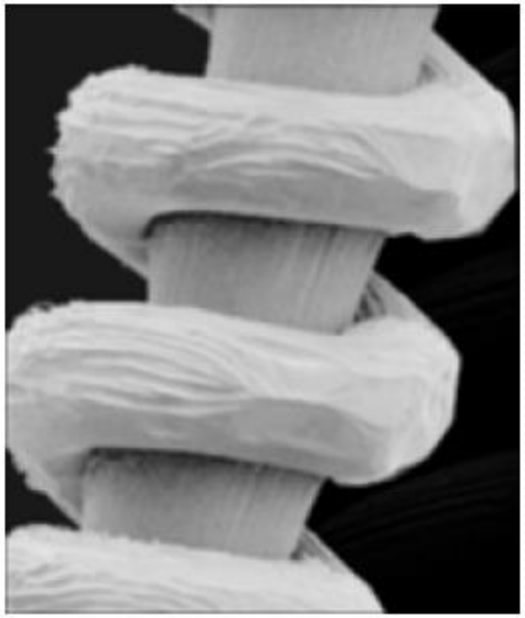


Image Averaging

- Image averaging is obtained by finding the average of K images. It is applied in de-noising the images.

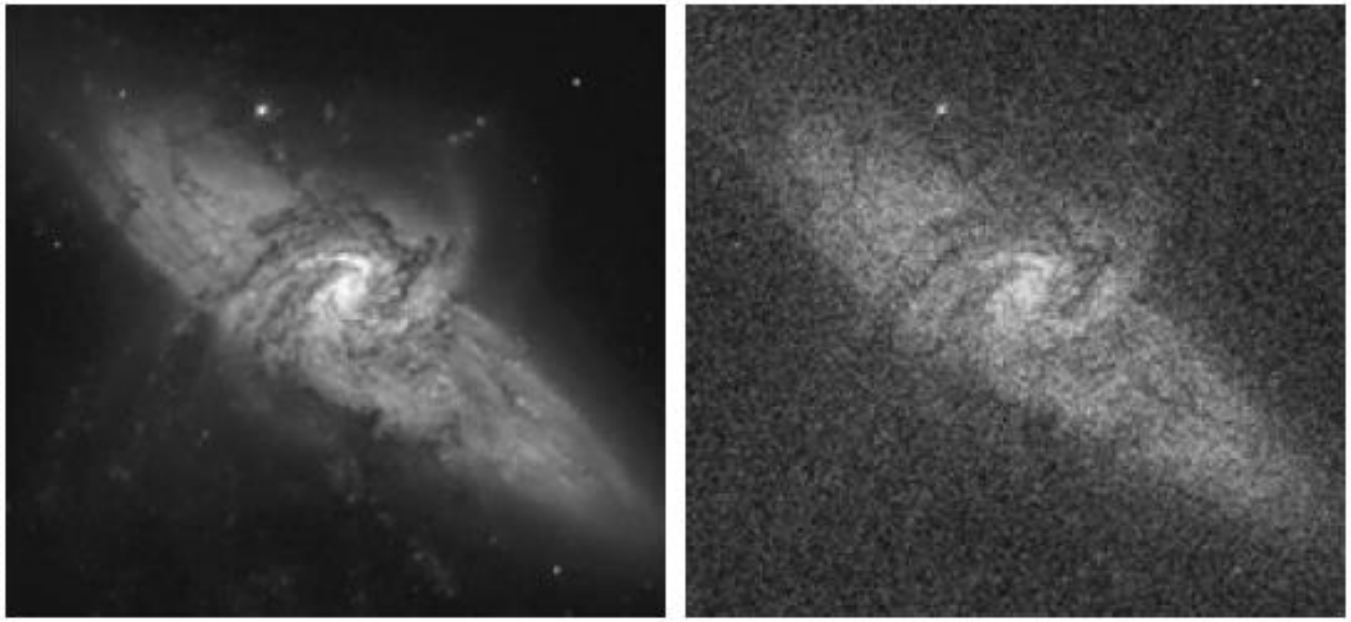
$$\bar{g}(x, y) = \frac{1}{K} \sum_{i=1}^{K} g_i(x, y)$$

- A noisy image is defined by:

$$g(x, y) = f(x, y) + \eta(x, y)$$

Assuming that the noise is uncorrelated with zero mean

Image Averaging



Average Images (8, 16, 64, and 128)

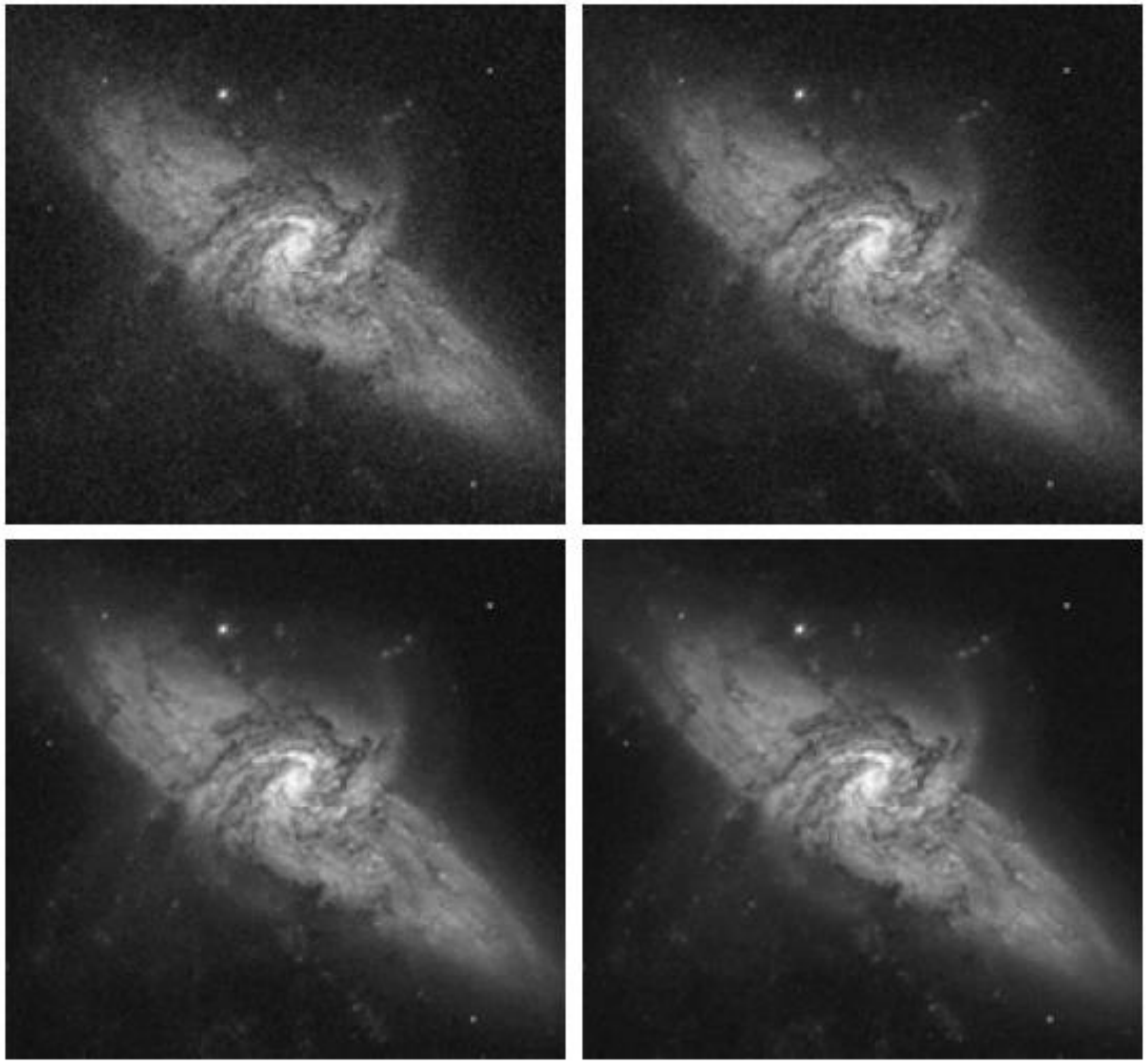
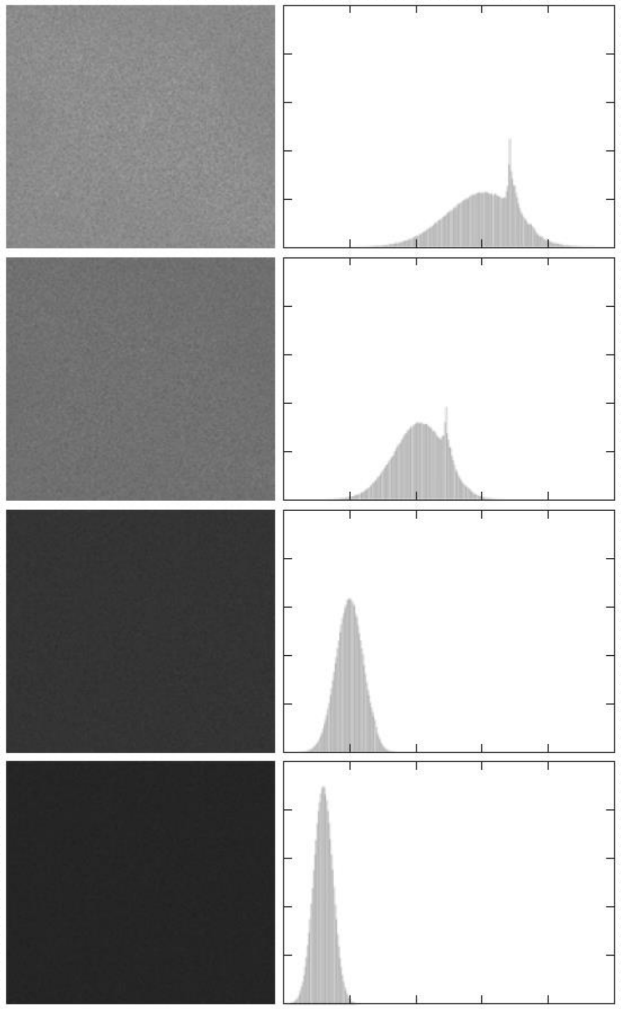


Image averaging in MATLAB

- Open the first image file   im1 = imread('test1.bmp');
- Open the second image file   im2 = imread('test2.bmp');
- Open the third image file   im3 = imread('test3.bmp');
- Find the average imAvg = (im1 + im2 + im3)/3;
- Display the image imshow(imAvg)

De-noising Effect on Histogram

Difference of Original Image and Average Images

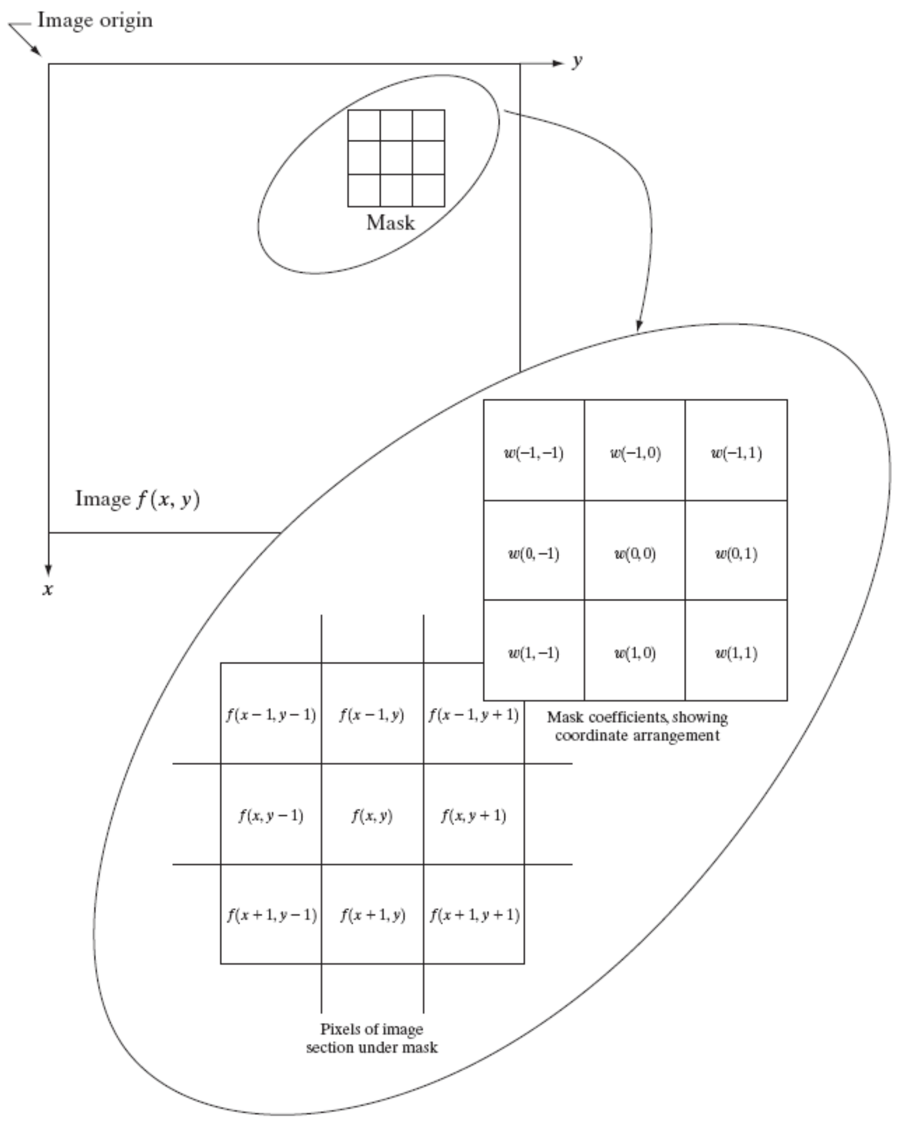


Corresponding Histograms

Spatial Filters

- To work on pixels in the neighborhood of a pixel, a sub-image is defined.
- The operation on the sub-image pixels is defined using a ***mask*** or ***filter*** with the same dimensions.
- Applying the operation to the image is referred to as convolution

Spatial Masks



Convolving Mask with Image

- Convolving mask with image is carried out by sliding the mask over the image, multiplying mask values with the pixel values falling beneath them and obtaining the sum.

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

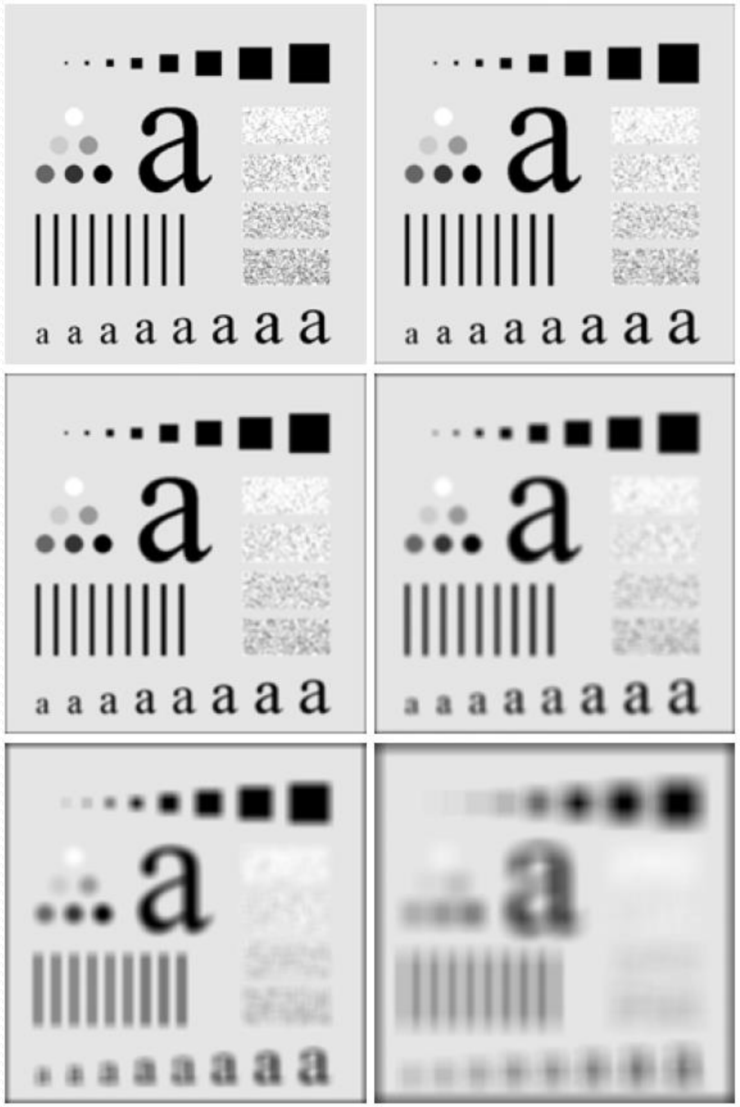
- The sum is used as the value for the position of the center of the mask over the image

Mean or Average Filter

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \qquad \frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

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

The Effect of Average Filter



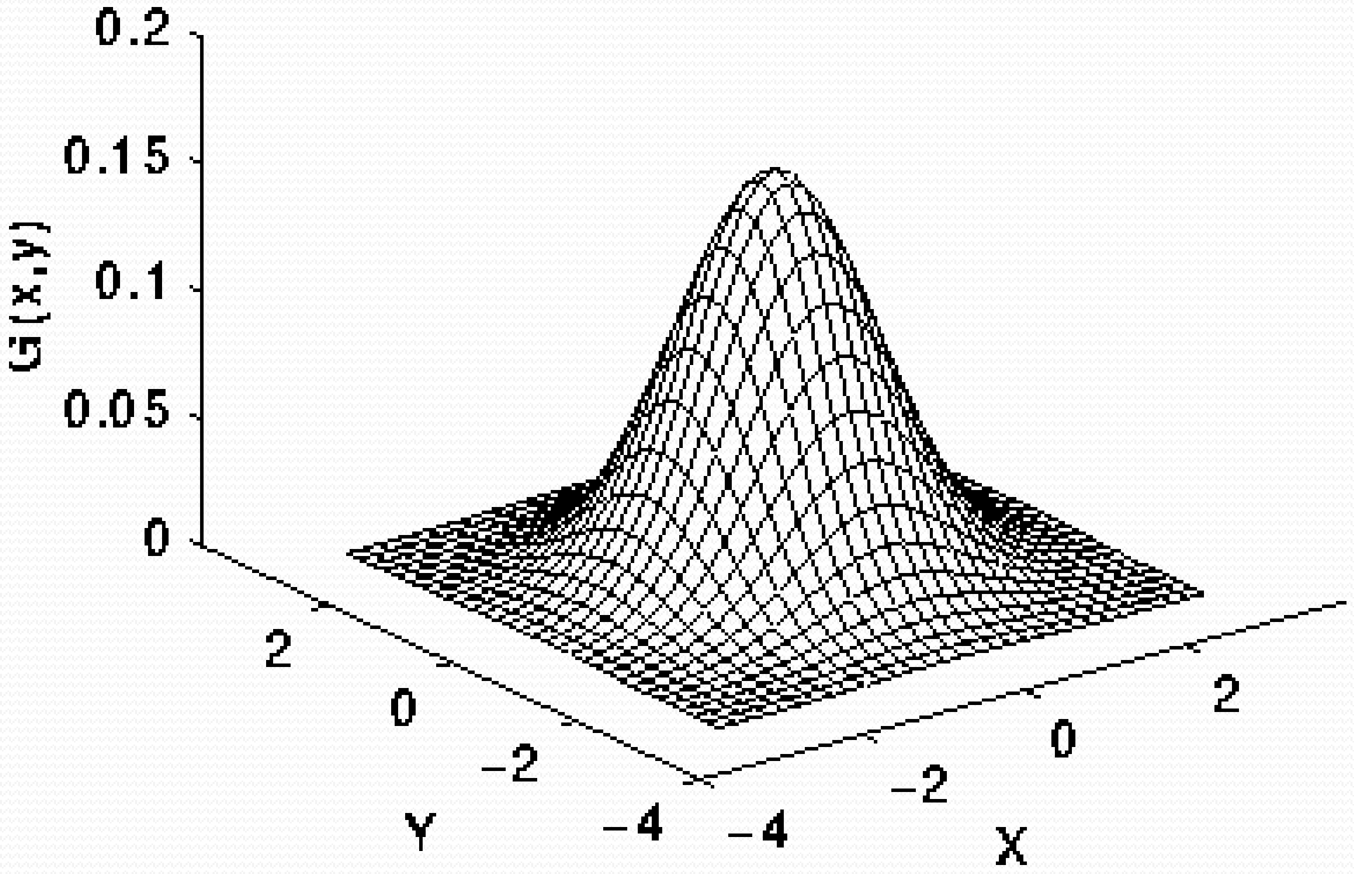
n=3, 5, 9, 15, and 35

Gaussian Filter

- The Gaussian smoothing operator is a 2-D convolution operator that is used to `blur' images and remove detail and noise.

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

Gaussian Filter



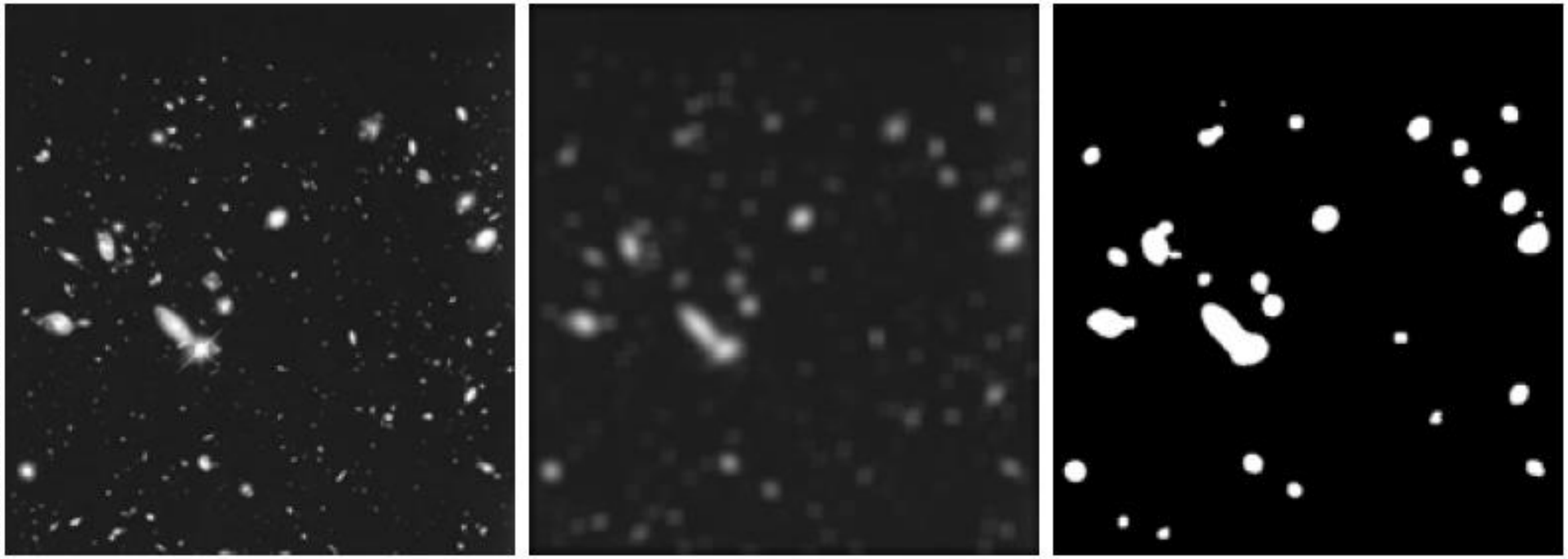
Gaussian Filter

$$\frac{1}{273}\begin{array}{|c|c|c|c|c|}\hline 1 & 4 & 7 & 4 & 1 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 7 & 26 & 41 & 26 & 7 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 1 & 4 & 7 & 4 & 1 \\ \hline \end{array}$$

Mean Filter in MATLAB

- I = imread('test.tif');
- H = fspecial('average', 3);
- I2 = imfilter(I, H);
- imshow(I), title('Original Image')
- figure, imshow(I2), title('Filtered image')

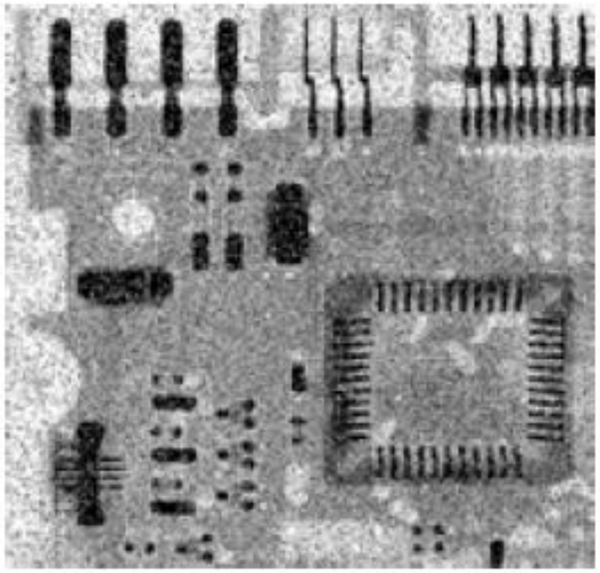
Mean Filter with Thresholding

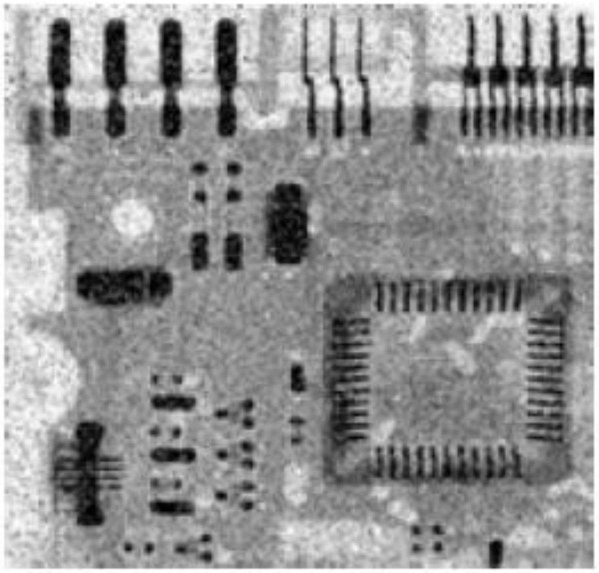


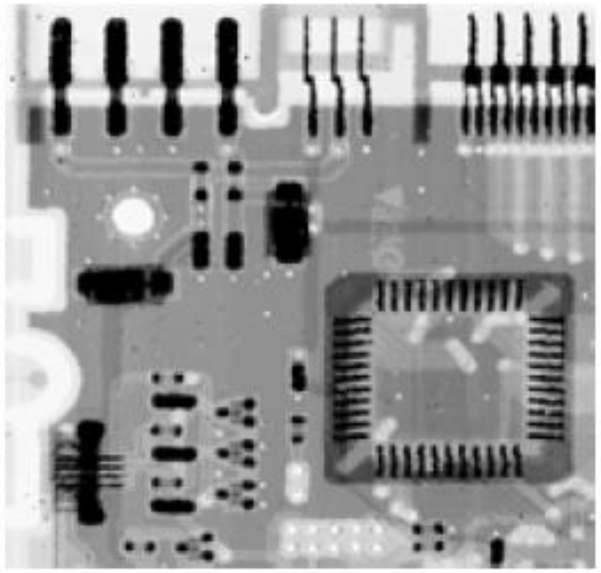
Median Filter

- Median filter replaces the pixel at the center of the filter with the median value of the pixels falling beneath the mask.
- Median filter does not blur the image but it rounds the corners.

Median Filter Effect


Original Image


Mean Filtered


Median Filtered

Median Filter in MATLAB

- img = imread(‘test.tif’);
- imSmoothed = medfilt2(img, [3 3]);

- [3 3] defines the dimension of the filter

High-pass or Sharpening Filters

- High pass filters let the high frequency content of the image pass through the filter and block the low frequency content.
- High pass filters can be modeled by first order derivative as :

$$\frac{\partial f}{\partial x} = f(x+1) - f(x).$$

- A second order derivative can also be used for extracting high frequency data

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

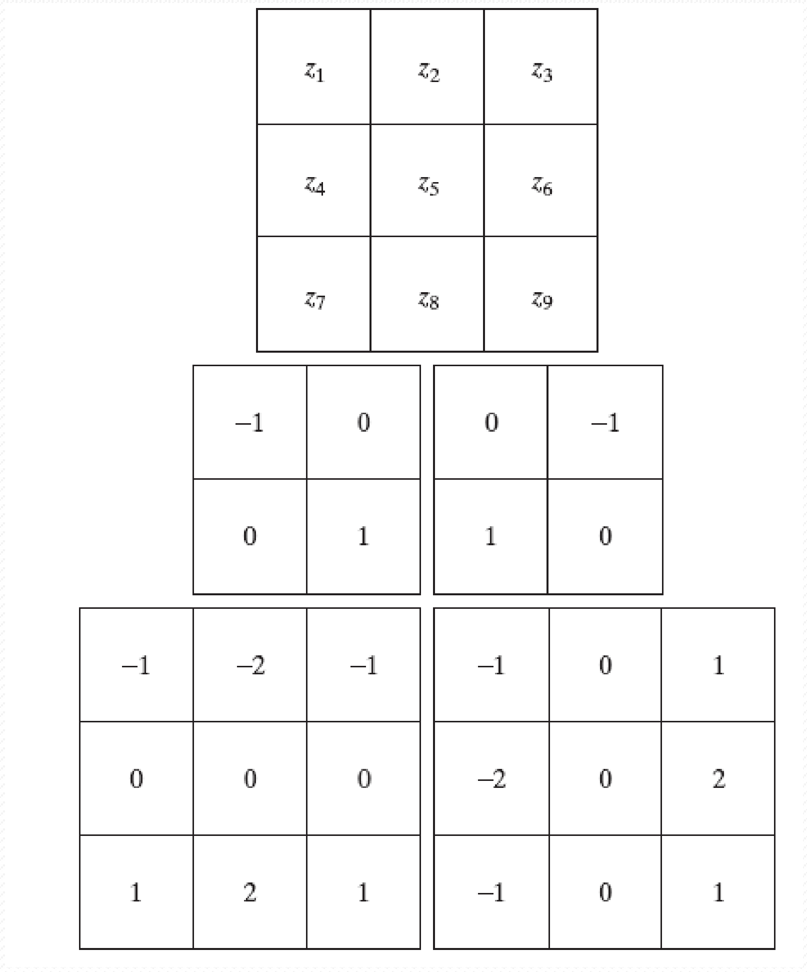
Use of First Derivative - Gradient

- Gradient is a vector which points in the direction of the greatest rate of increase of the scalar field, and whose magnitude is the greatest rate of change.

$$\nabla \mathbf{f} = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix}$$

$$\begin{aligned} \nabla f &= \operatorname{mag}(\nabla \mathbf{f}) \\ &= \left[G_x^2 + G_y^2\right]^{1/2} \\ &= \left[\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2\right]^{1/2} \end{aligned}$$

Edge Detection Filters



z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

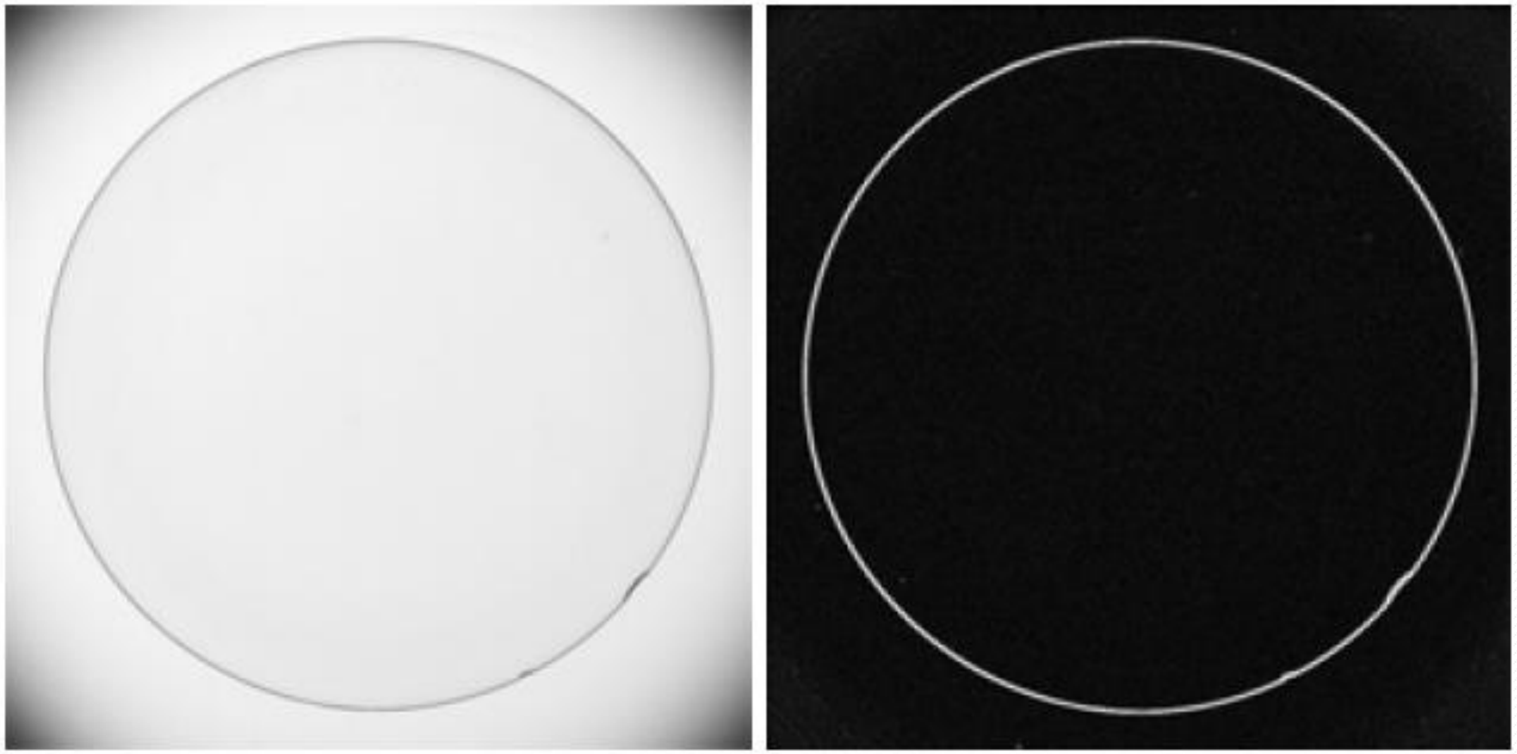
–1	0
0	1

0	–1
1	0

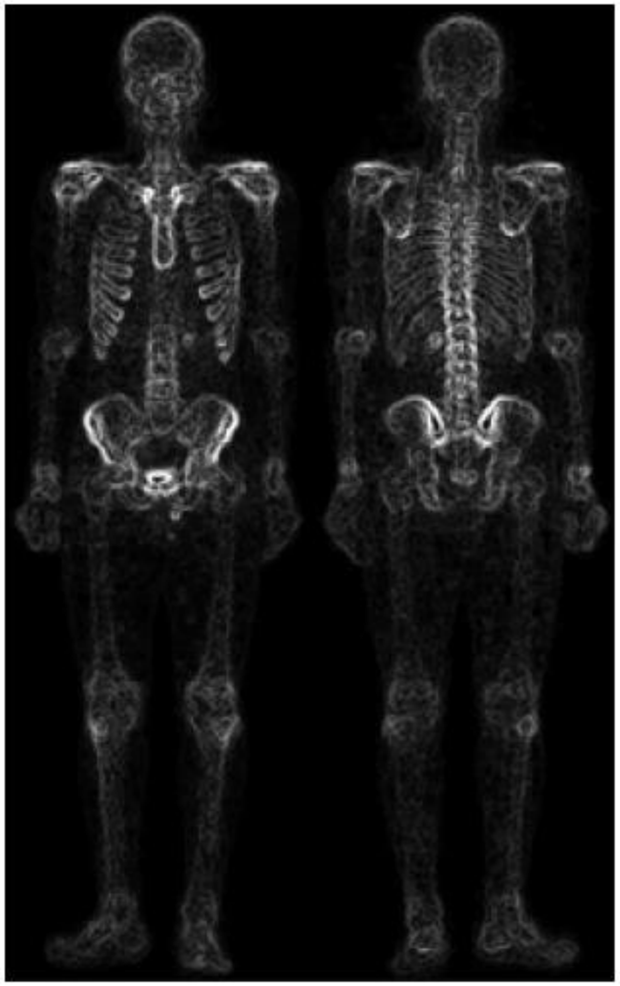
–1	–2	–1
0	0	0
1	2	1

–1	0	1
–2	0	2
–1	0	1

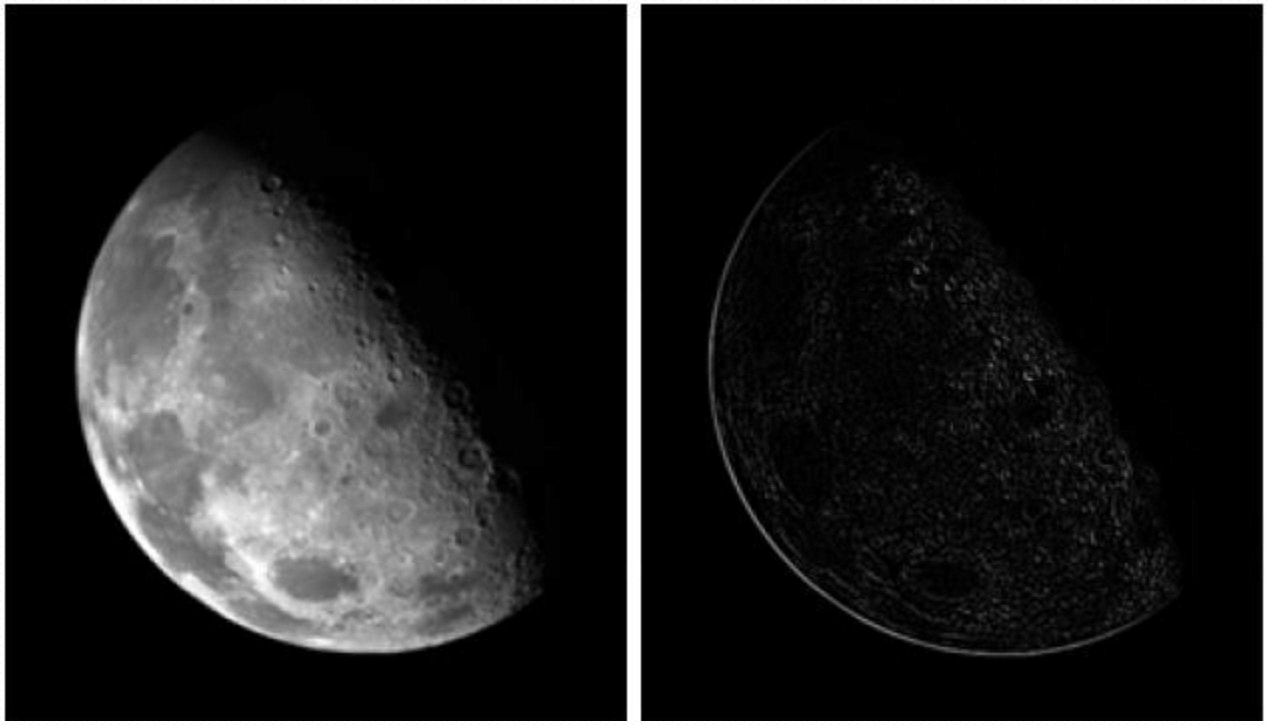
Edge Detection Example



Edge Image Example



Edge Image Example



Edge Detection in MATLAB

- Img = imread(‘test.tif’);
- edgeImg = edge(Img, ‘sobel’);
- imshow(edgeImg)

Canny Edge Detection

- The Process of Canny edge detection algorithm can be broken down to 5 different steps:
  - Apply Gaussian filter to smooth the image in order to remove the noise
  - Find the intensity gradients of the image
  - Apply non-maximum suppression to get rid of spurious response to edge detection
  - Apply double threshold to determine potential edges
  - Track edge by hysteresis

Non-Maximum Suppression

- Compare the edge strength of the current pixel with the edge strength of the pixel in the positive and negative gradient directions.
- If the edge strength of the current pixel is the largest compared to the other pixels in the mask with the same direction the value will be preserved. Otherwise, the value will be suppressed.

Double Thresholding

- Two threshold values are used with edge pixels: A larger threshold and a smaller threshold.

- The edge values larger than the first threshold are reliable edge pixels.

- The edge points with a value larger than the second threshold but smaller than the first threshold are unreliable edge pixels

Edge Tracking by Hysteresis

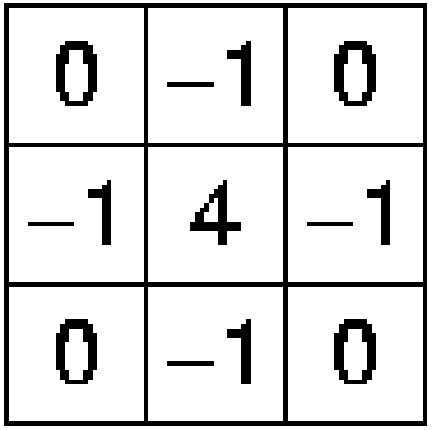
- Some weak edge pixels are coming from noise. To distinguish noise pixels from edge pixels, the neighboring pixels are examined
- If there is one strong edge pixel in the neighborhood of a weak edge pixel, that weak edge point should be preserved.

Using the Second Derivative for Edge Detection-Laplacian

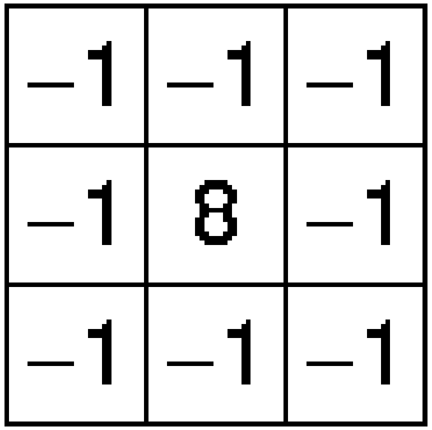
- The Laplace operator is a second order differential operator in the n-dimensional Euclidean space, defined as the divergence (∇) of the gradient (∇f).
- The Laplacian L(x,y) of an image with pixel intensity values I(x,y) is given by:

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$$

Laplacian Filters



0	−1	0
−1	4	−1
0	−1	0



−1	−1	−1
−1	8	−1
−1	−1	−1

Laplacian of Gaussian (LoG)

- Because Laplacian kernels are approximating a second derivative measurement on the image, they are very sensitive to noise.
- To counter this, the image is often Gaussian smoothed before applying the Laplacian filter.
- This pre-processing step reduces the high frequency noise components prior to the differentiation step.

Laplacian of Gaussian (LoG)

- The LoG (`Laplacian of Gaussian') kernel can be pre-calculated in advance so only one convolution needs to be performed at run-time on the image.

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$$

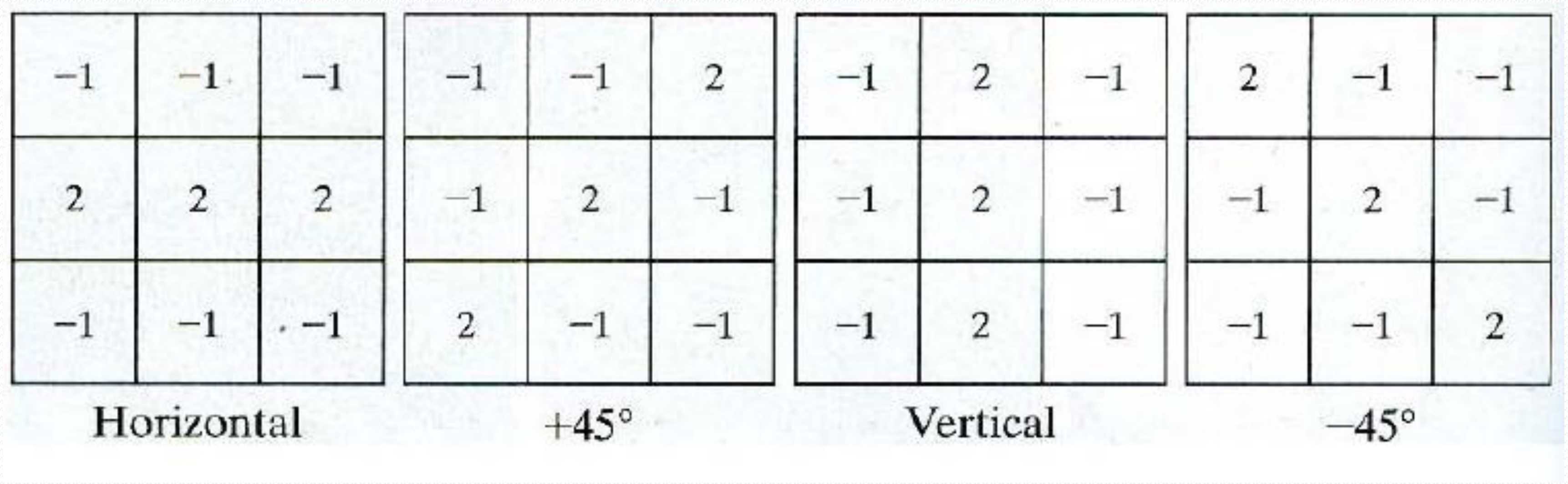
Corner Detection

- A corner can be defined as the intersection of two edges.
- A corner can also be defined as a point for which there are two dominant and different edge directions in a local neighborhood of the point.
- Therefore, a shift in any direction will give a large change in the pixel values

Assignment

- One of the important methods for corner detection is Harris Corner Detector.
- Write a report describing the details of the method step by step
- Implement the method using MATLAB. Explain the result of each step. Justify your results if they are not the expected values.

Line Detection Filters



−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

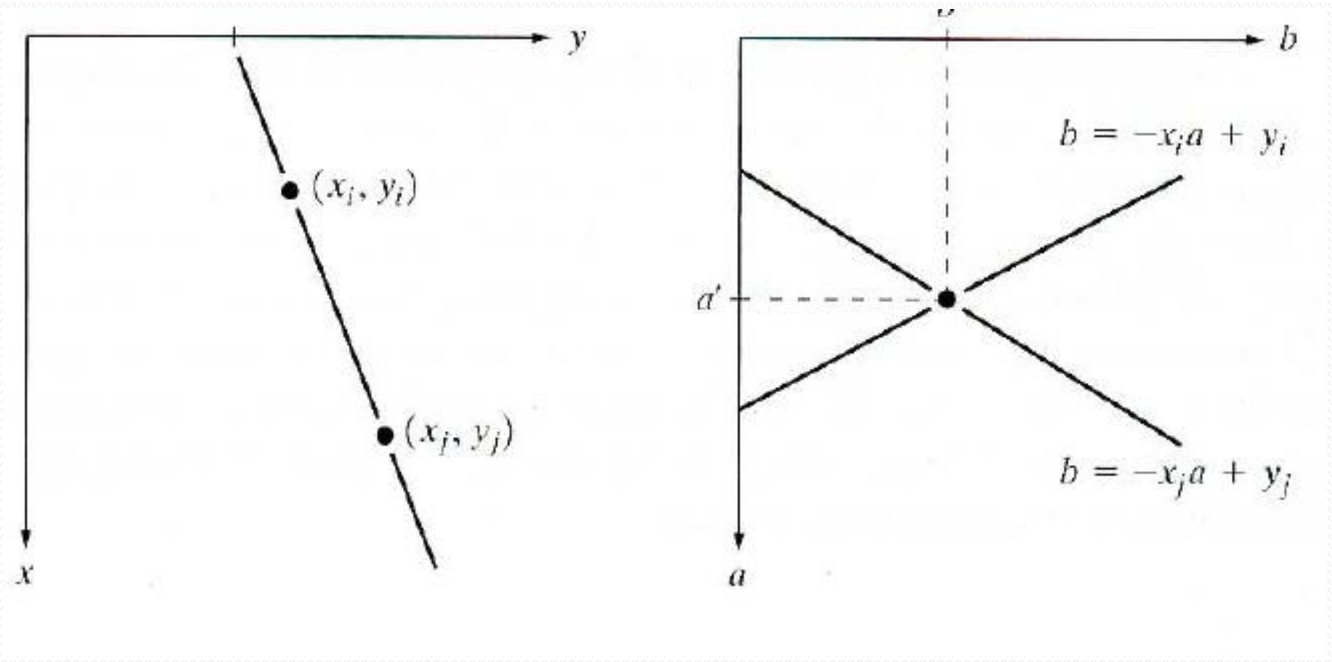
2	−1	−1
−1	2	−1
−1	−1	2

−45°

Is local Processing Adequate?

- Local processing gives information about the local neighborhood of pixels.
- The global characteristics of the pixels cannot be determined from their local characteristics.
- E.g. Line detection using gradient operator can provide the probability of a pixel being on a line but it cannot group the pixels into line segments.

Global Processing Example: Hough Transform



Line Detection using Hough Transform

- Define the line using its parametric equation y=mx+c
    - m is the slope of the line, c in the intersection with y axis
- Define a buffer (M, C) and initialize it to zero
- For all edge pixels Do
    - For all combination of m and c values Do
        - If the pixel satisfies $y=m_ix+c_i$ the increment the value of the corresponding buffer location (m_i , c_i)

Questions?