Image Segmentation

Reference:
Digital Image Processing
Gonzalez & Woods

Segmentation

- Segmentation subdivides an image into its constituent regions or objects. It attempts to:
  - Partition an image into *meaningful* regions with respect to a particular application.
- Segmentation is based on measurements taken from the image.
  - Might be graylevel, colour, texture, depth or motion.
  - It should stop when the objects of interest in an application have been isolated.

The Segmentation Problem

- Segmentation algorithms are generally based on two basic properties of intensity values:
  - Discontinuity, ex. used to detect edges
  - Similarity, ex. thresholding, region growing

Segmentation Applications

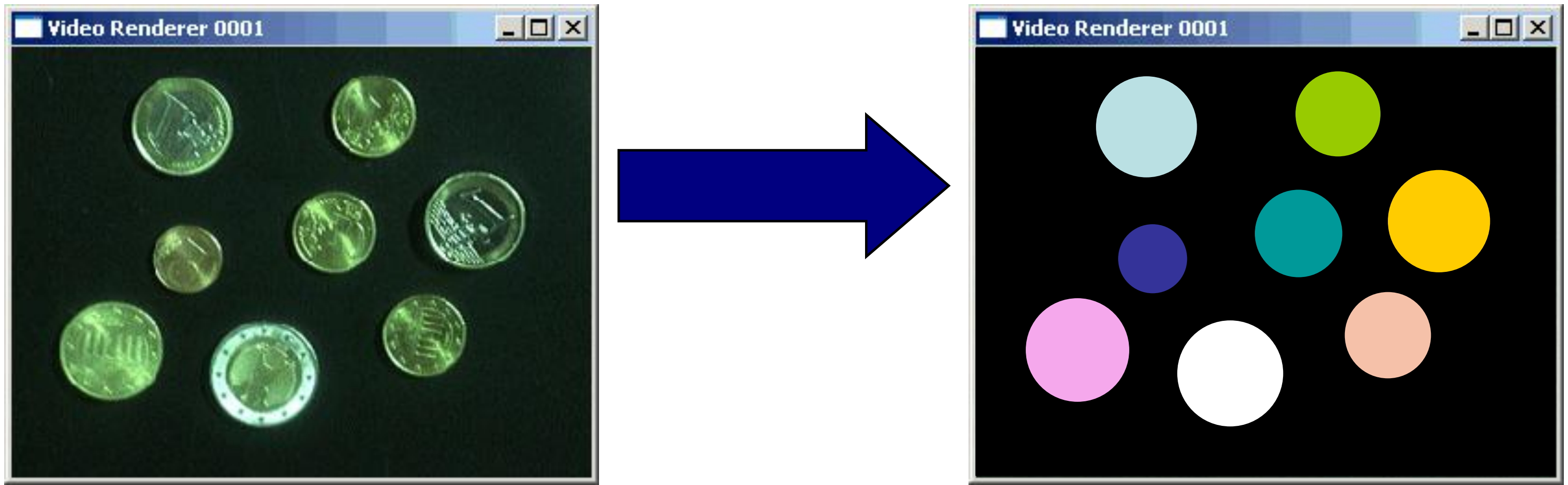
- Applications of image segmentation include:
  - Identifying objects in a scene for object-based measurements such as size and shape.
  - Identifying objects in a moving scene for object-based video compression (MPEG4).
  - Identifying objects which are at different distances from a sensor using depth measurements from a laser range finder enabling path planning for a mobile robot.

The Segmentation Problem

Segmentation attempts to:

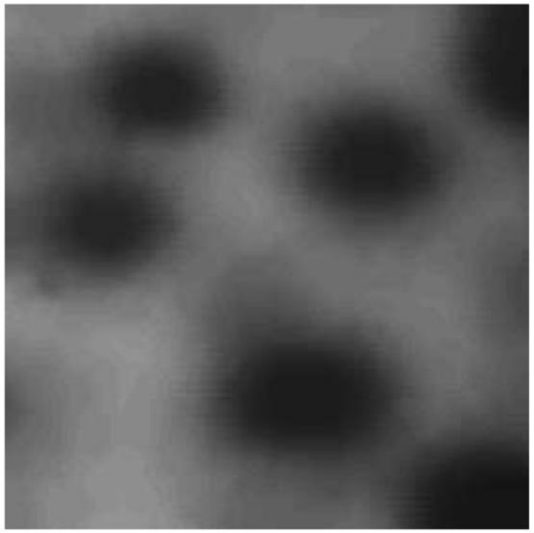
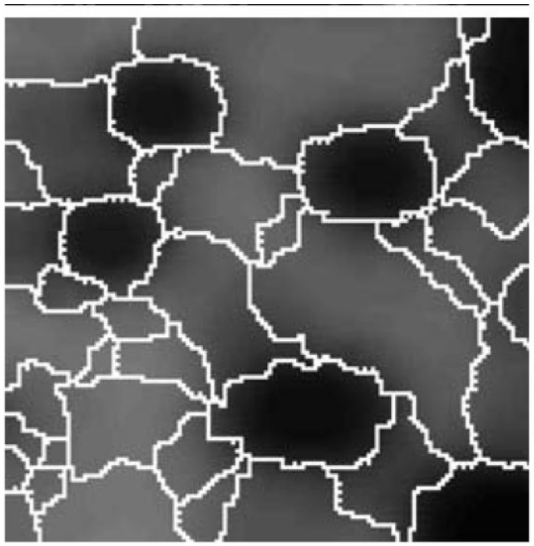
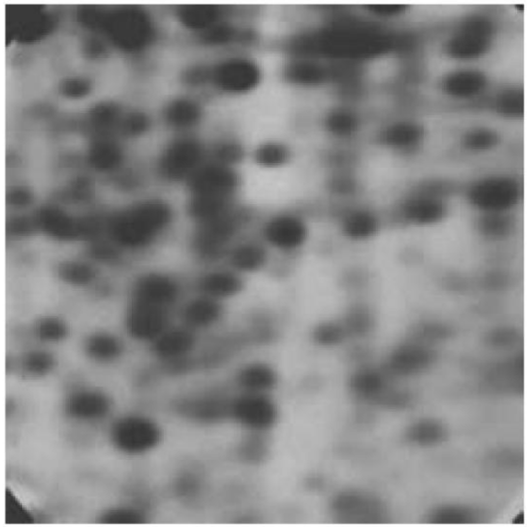
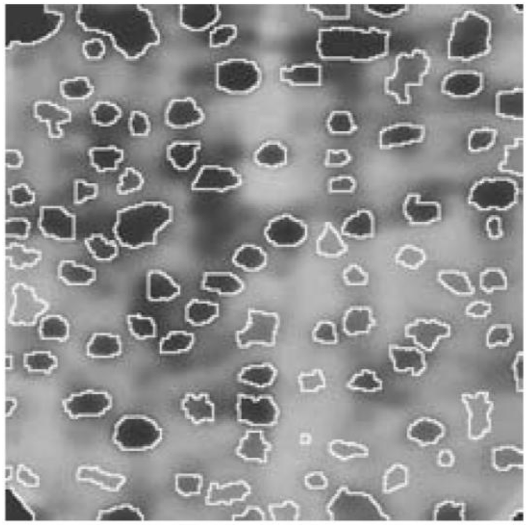
– partition the pixels of an image into groups that strongly correlate with the objects in an image.

–Typically the first step in any automated computer vision application.



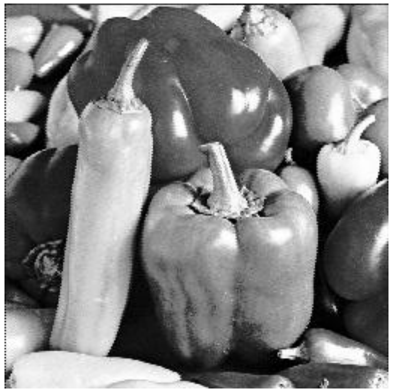
Segmentation Examples

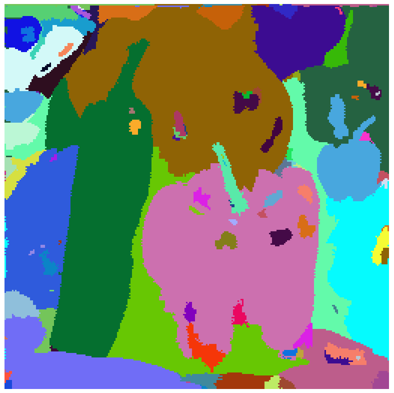
Images taken from Gonalez & Woods, Digital Image Processing (2002)

Segmentation Example

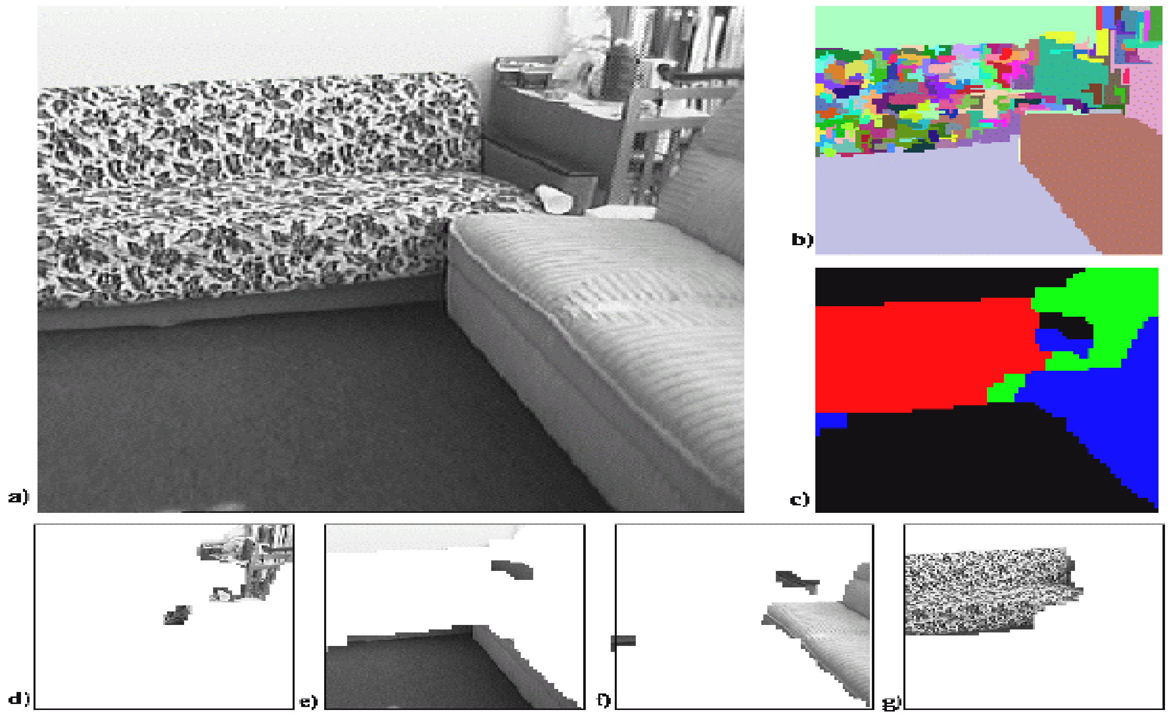
- Example 1
  - Segmentation based on greyscale.
  - Very simple ‘model’ of grayscale leads to inaccuracies in object labelling.





Segmentation Example

- Example 2
  - Segmentation based on texture.
  - Enables object surfaces with varying patterns of grey to be segmented.



Detection Of Discontinuities

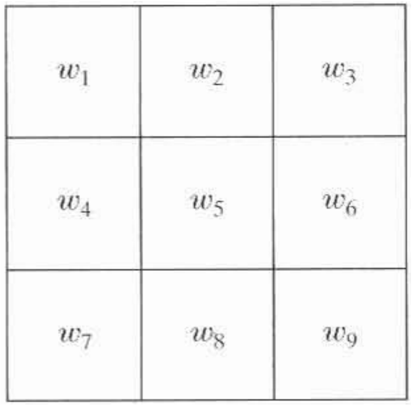
- There are three basic types of grey level discontinuities in digital images:
  - Points
  - Lines
  - Edges
- We typically find discontinuities using masks and correlation.

Detection Of Discontinuities

•We say that a point, line and edge has been detected at the location on which the mask is centered if $|R| \geq T$ where,

$$R = w_1 z_1 + w_2 z_2 + \dots\dots + w_9 z_9$$

and *T* is a non-negative threshold.

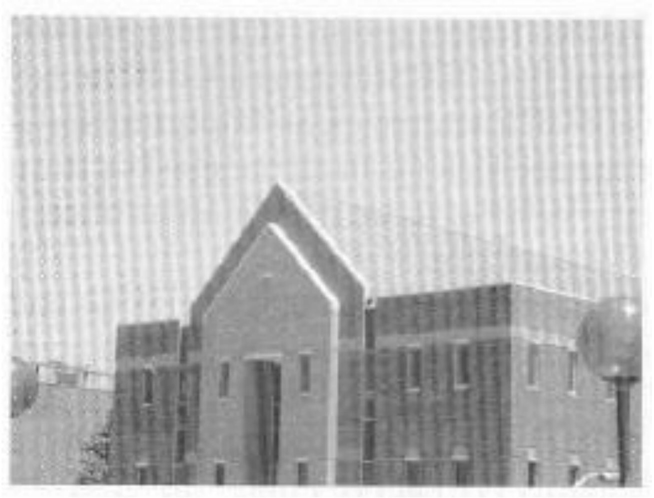


w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Detection Of Discontinuities

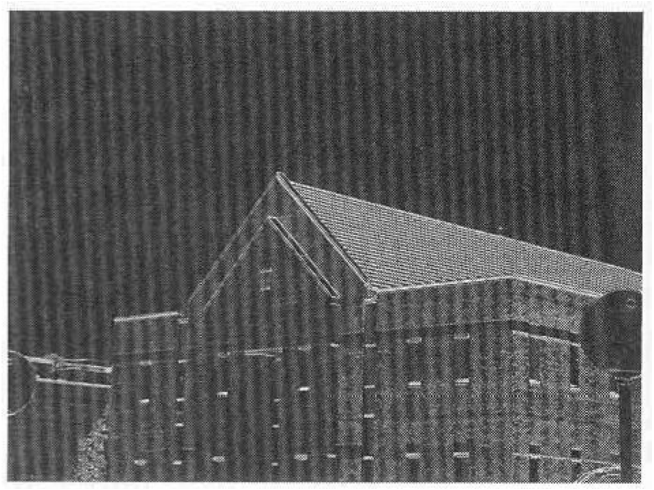
- A point detection mask:

-1	-1	-1
-1	8	-1
-1	-1	-1



- A line detection mask:

-1	-1	-1
2	2	2
-1	-1	-1



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Point Detection

•Point detection can be achieved simply by using the mask below:

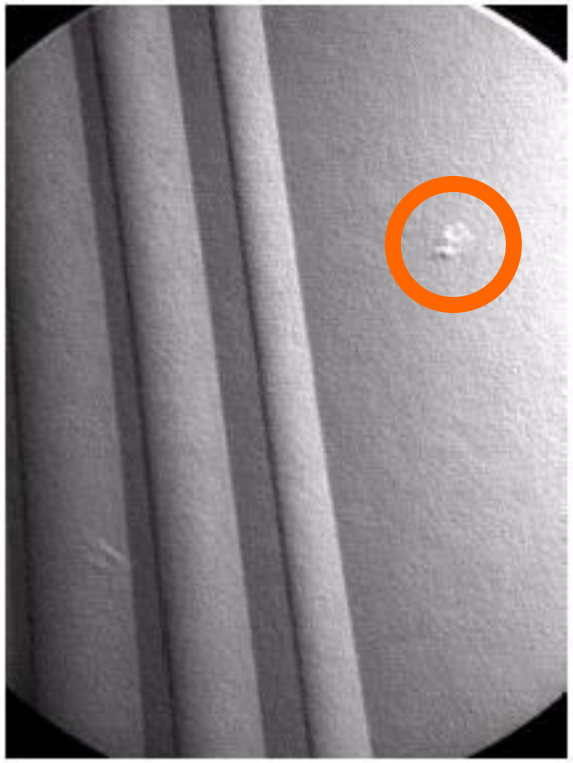
–1	–1	–1
–1	8	–1
–1	–1	–1

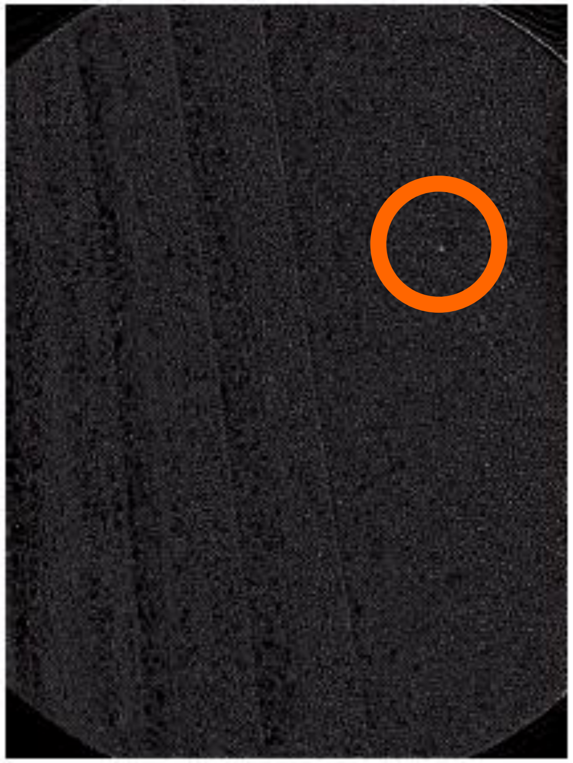
•Points are detected at those pixels in the subsequent filtered image that are above a set threshold.

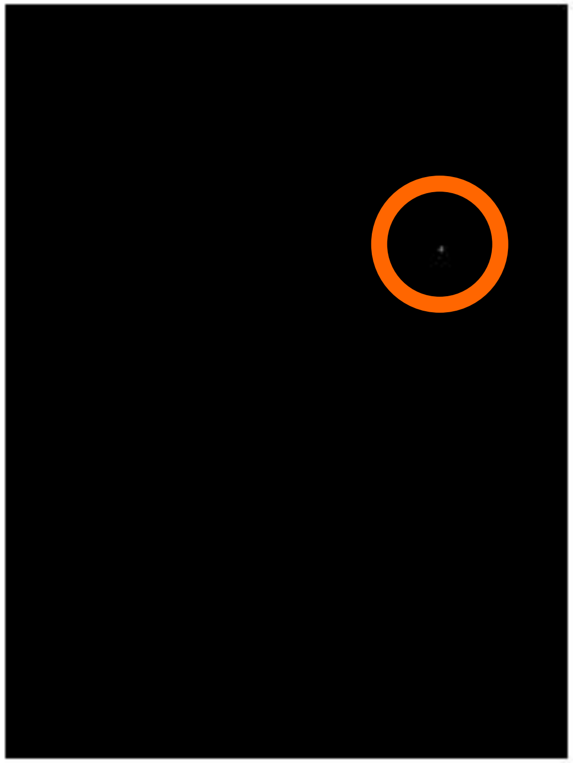
Point Detection

- The formulation measures weighted differences between center point and its neighbours.
  - An isolated point will be quite different from its background.
  - Mask coefficients sum to zero, indicating that mask response will be zero in areas of constant gray level.

Point Detection (cont…)


X-ray image of a turbine blade


Result of point detection


Result of thresholding

Line Detection

- Methods:
  - Use of Convolution Mask
  - Hough Transform
- The masks below will extract lines that are one pixel thick and running in a particular direction:

−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

$-45°$

Line Detection

- Let R_1, R_2, R_3, R_4 be the mask responses, when they are run on an image given by equation:

$$R = w_1 z_1 + w_2 z_2 + \ldots\ldots + w_9 z_9$$

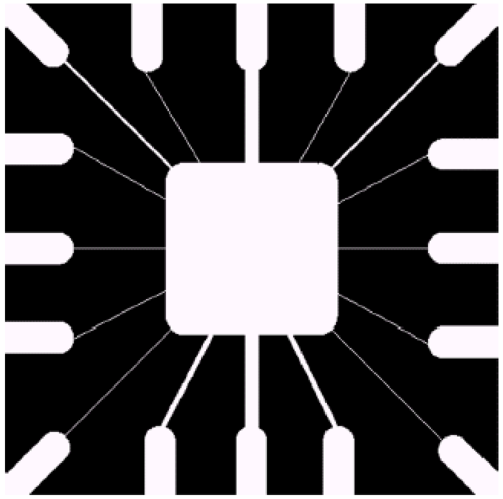
- If, at a certain point in the image, $|R_i| > |R_j|$ for all $j \neq i$, that point is more likely to be associated with a line in the direction of mask *i*.

Line Detection

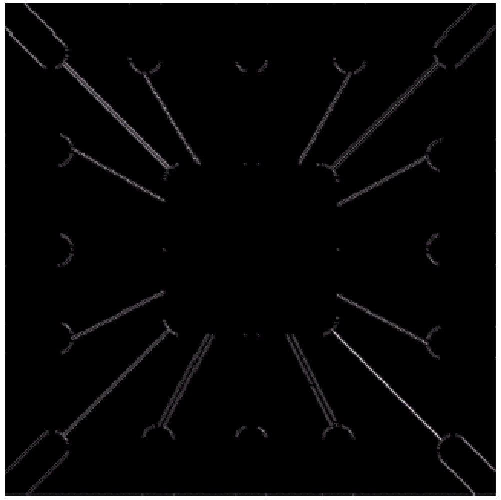
- If we are interested in detecting all lines in an image in the direction defined by a given mask, we run the mask through the image and threshold the absolute value of the result.
  - The points that are left are the strongest responses, one pixel thick, corresponding closest to the direction defined by the mask.

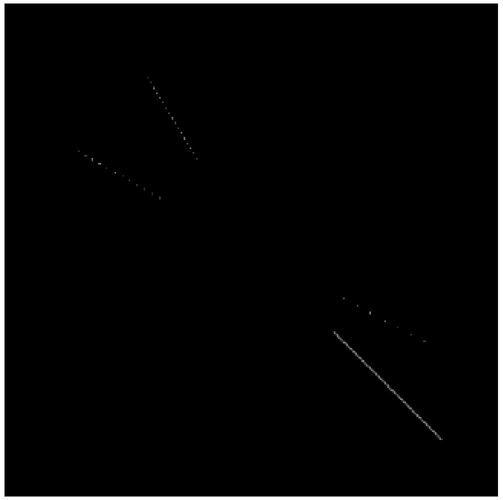
Line Detection

Binary image of a wire bond mask



After processing with -45° line detector

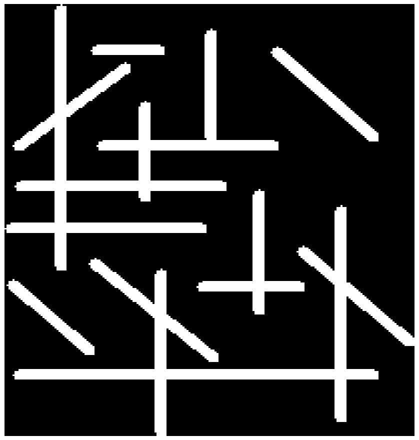


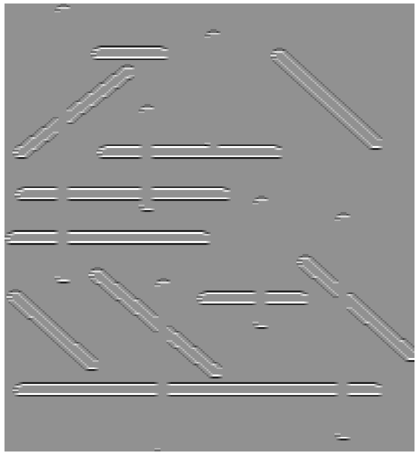


Result of thresholding filtering result

Line Detection

- The mask for horizontal orientation is applied.
- Lines are 5 pixels wide, mask is tuned for detecting 1 pixel wide line.
- Line detector acts as edge detector.



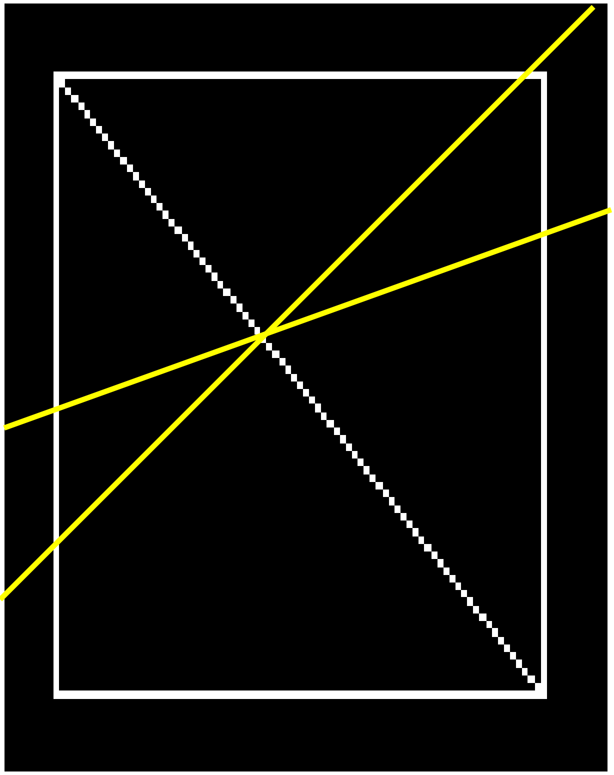


Line Detection: Hough Transform

- Method to isolate the shapes from an image.
- Performed after edge detection.
- Not affected by noise or gaps in the edges.
- Technique:
  - Thresholding is used to isolate pixels with strong edge gradient.
  - Parametric equation of straight line is used to map the edge points to the Hough parameter space.
  - Points of intersection in the Hough parameter space gives the equation of line on actual image.

Hough Transform

- Basic idea:
  - Each straight line in this image can be described by an equation.
  - Each white point if considered in isolation could lie on an infinite number of straight lines.
  - In the Hough transform each point votes for every line it could be on.
  - The lines with the most votes win.

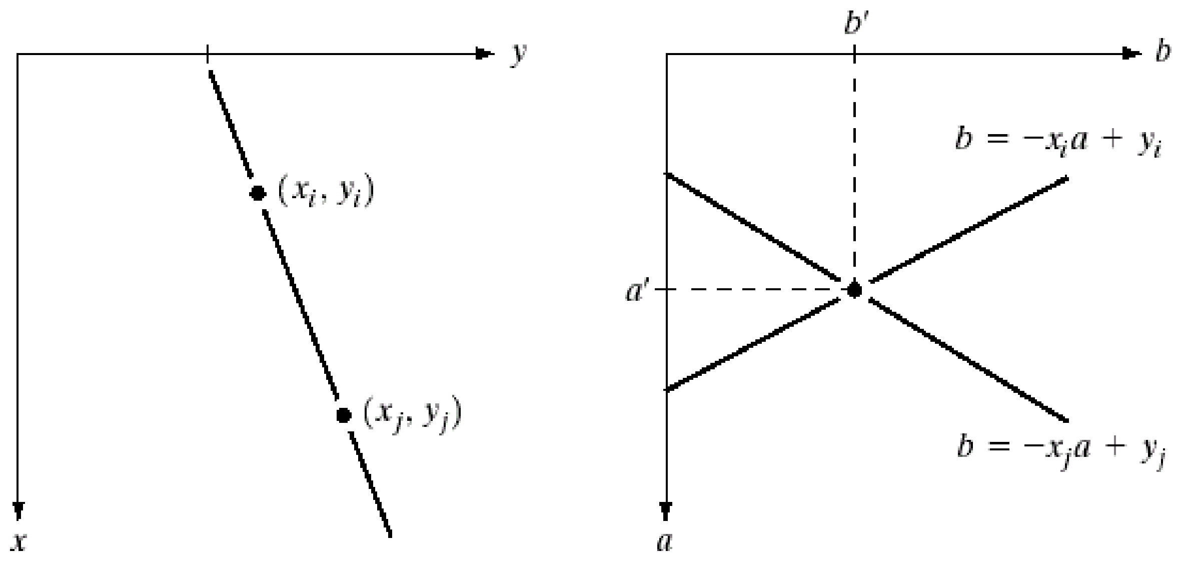


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Hough Transform

- Example: *xy*-plane v.s. *ab*-plane (parameter space)

$$y_i = ax_i + b$$



a b

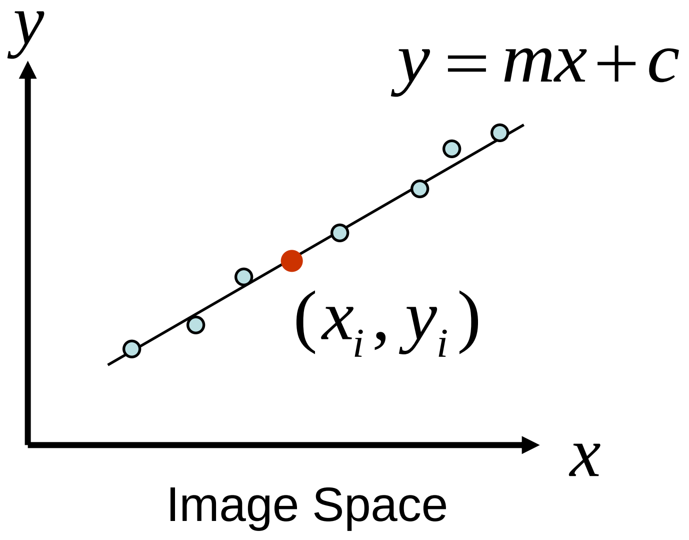
FIGURE 10.17 (a) *xy*-plane. (b) Parameter space.

Image and Parameter Space

Equation of Line: $y = mx + c$

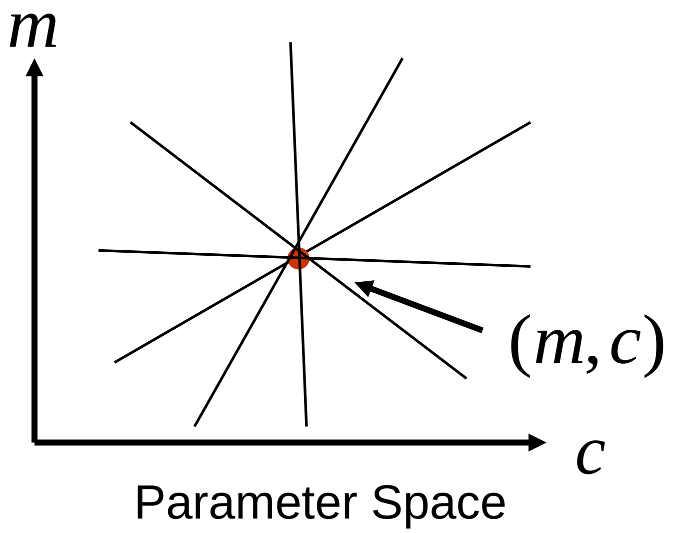
Find: (m, c)

Consider point: (x_i, y_i)



$$y_i = mx_i + c \quad or \quad c = -x_i m + y_i$$

Parameter space also called Hough Space



Line Detection by Hough Transform

Algorithm:

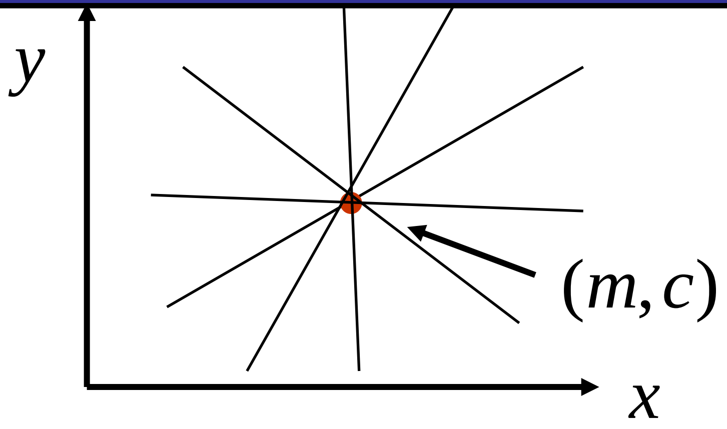
- Quantize Parameter Space (m,c)
- Create Accumulator Array $A(m,c)$
- Set $A(m,c)=0 \quad \forall m,c$
- For each image edge (x_i, y_i) increment:

$$A(m,c)=A(m,c)+1$$

if (m,c) lies on the line:

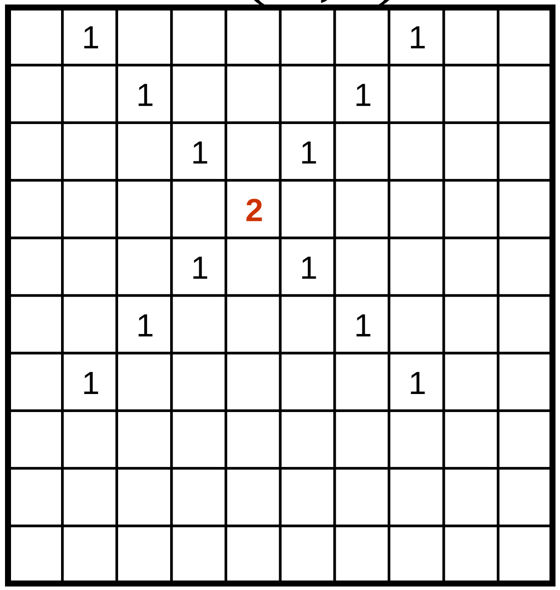
$$c=-x_i m+y_i$$

- Find local maxima in $A(m,c)$



Parameter Space

$A(m,c)$



	1						1		
		1				1			
			1		1				
				2					
			1		1				
		1				1			
	1						1		

Better Parameterization

NOTE: $-\infty \leq m \leq \infty$

Large Accumulator

More memory and computations

Improvement: (Finite Accumulator Array Size)

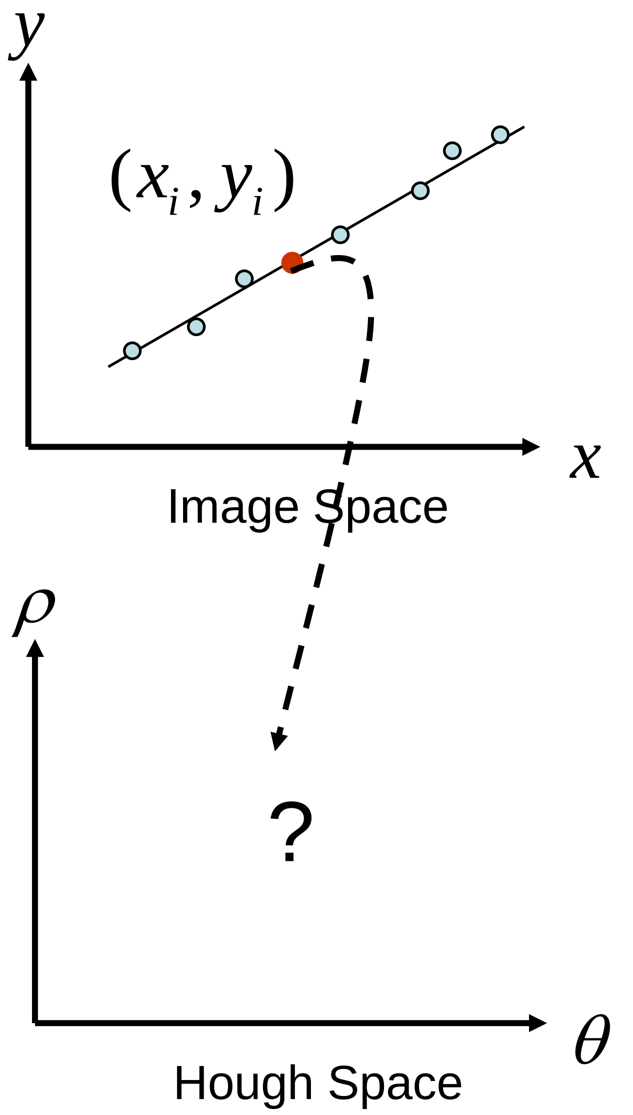
Line equation: $\rho = x\cos\theta + y\sin\theta$

Here: $0 \leq \theta \leq 2\pi$

$0 \leq \rho \leq \rho_{\max}$

Given points (x_i, y_i) find (ρ, θ)

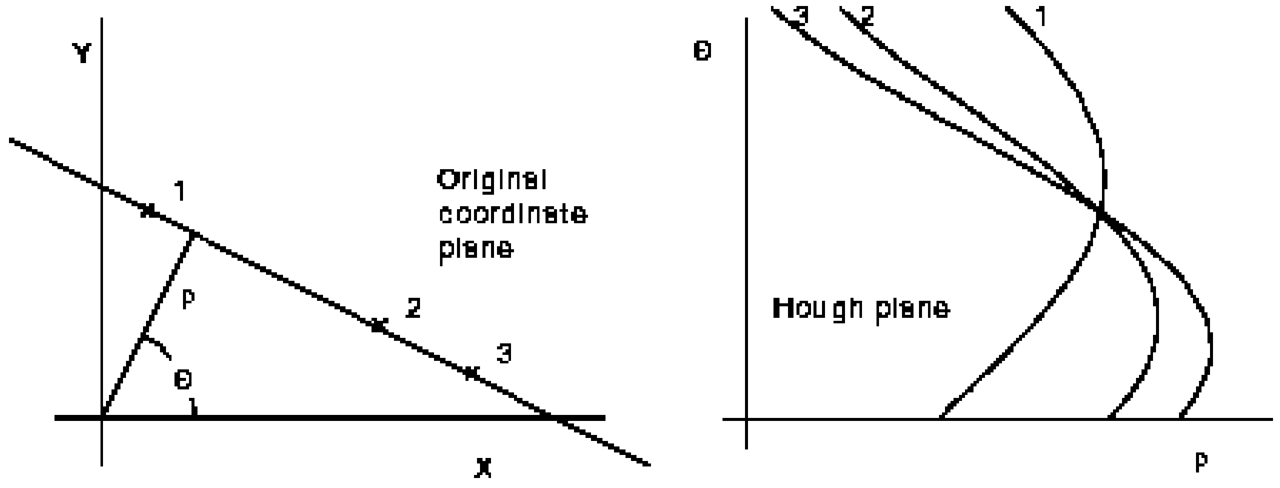
Hough Space Sinusoid



Hough Transform

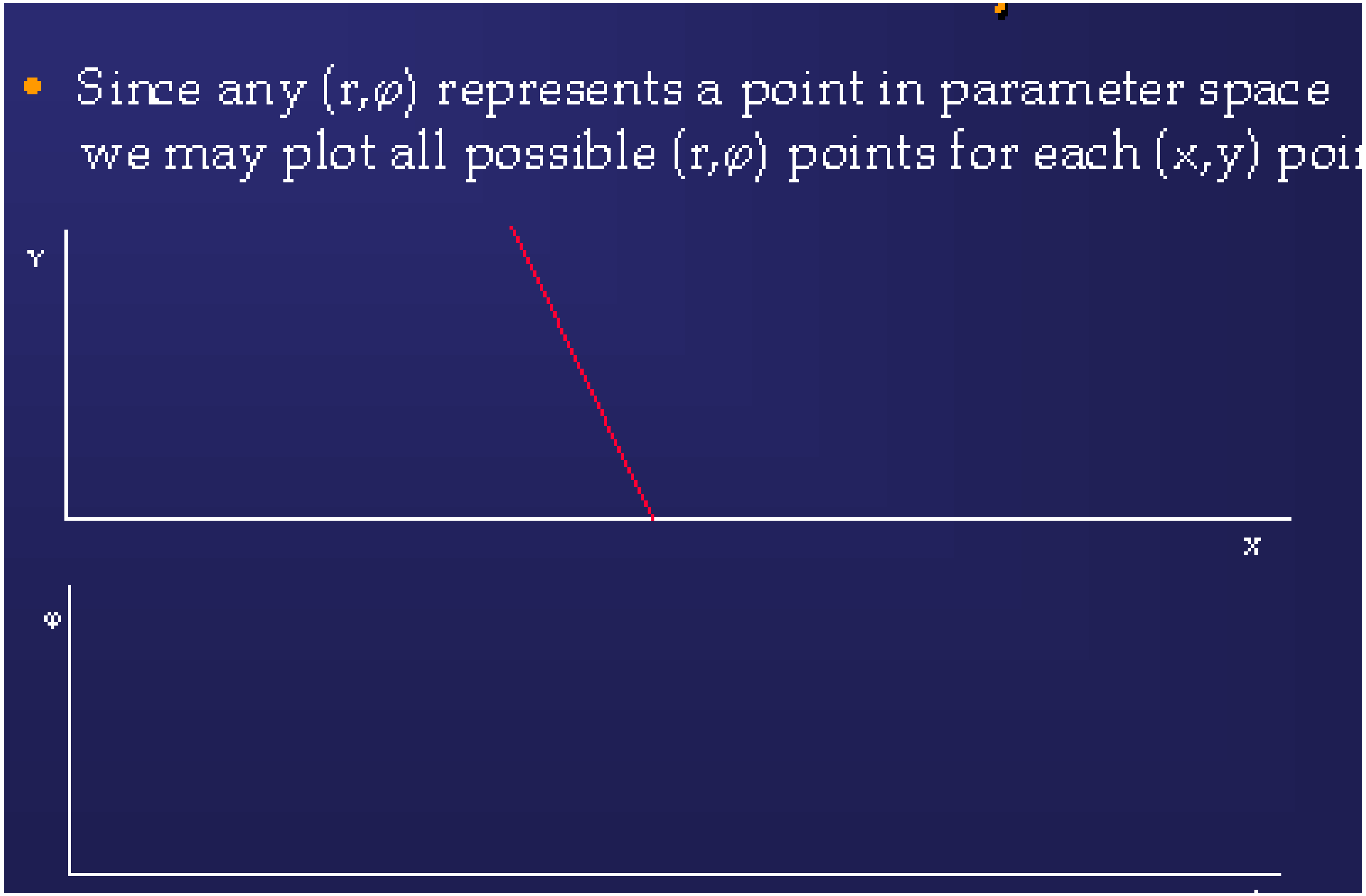
- Parametric equation of a straight Line:

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



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

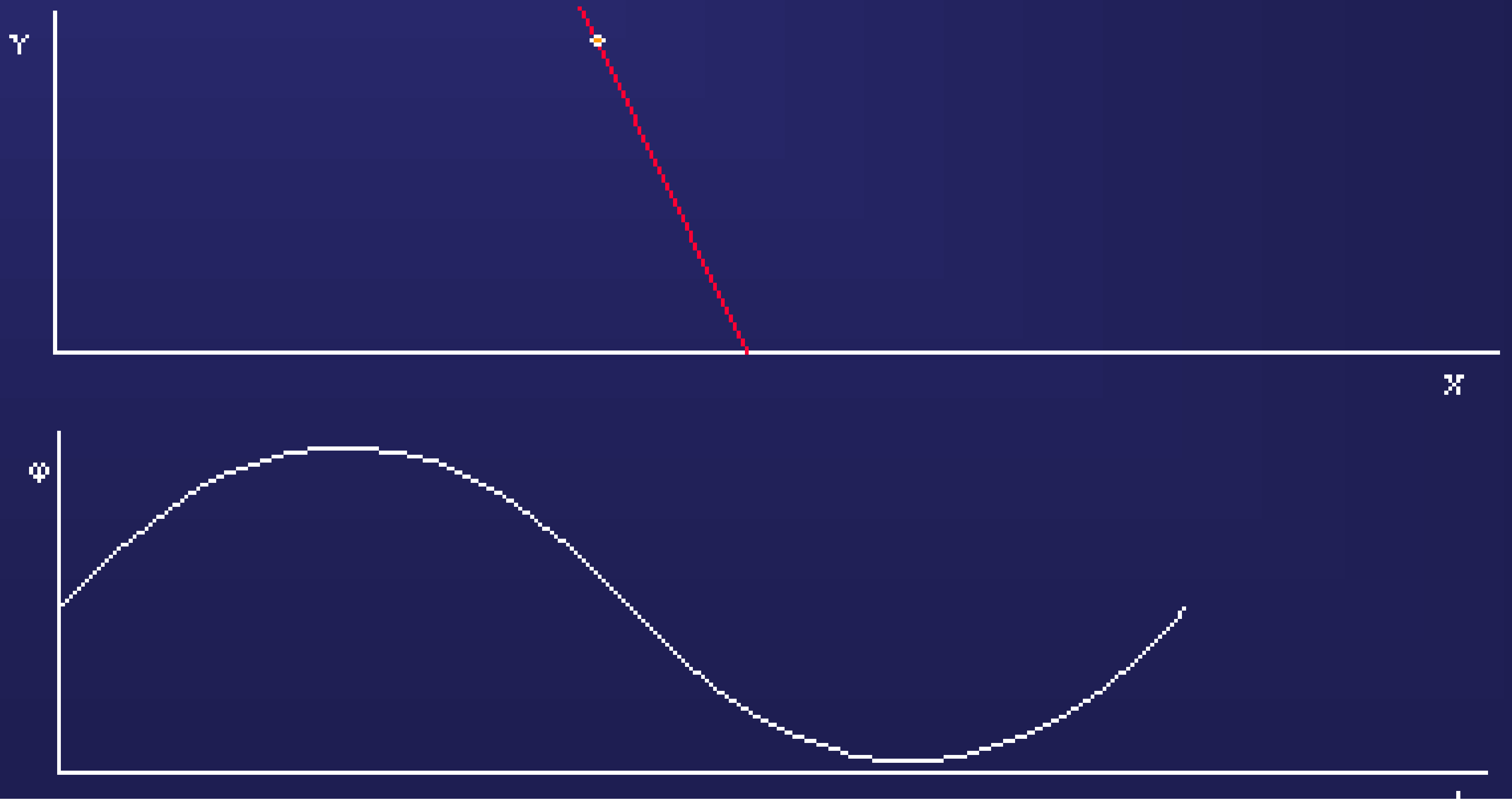
Hough Transform



- Since any (r,φ) represents a point in parameter space we may plot all possible (r,φ) points for each (x,y) poi

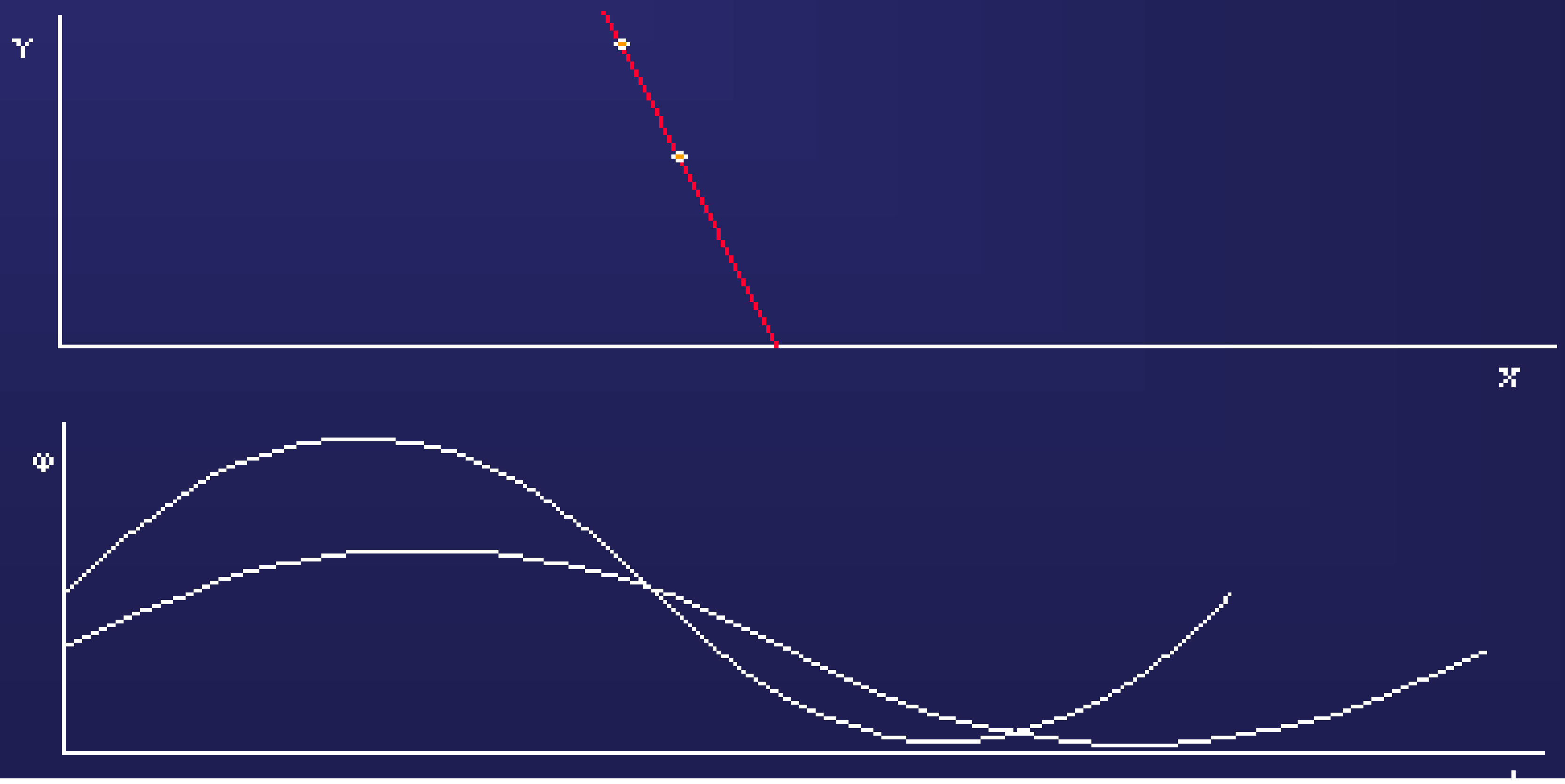
Hough Transform

- Taking any particular (x,y) point its set of (r,φ) points is a sinusoid through parameter space



Hough Transform

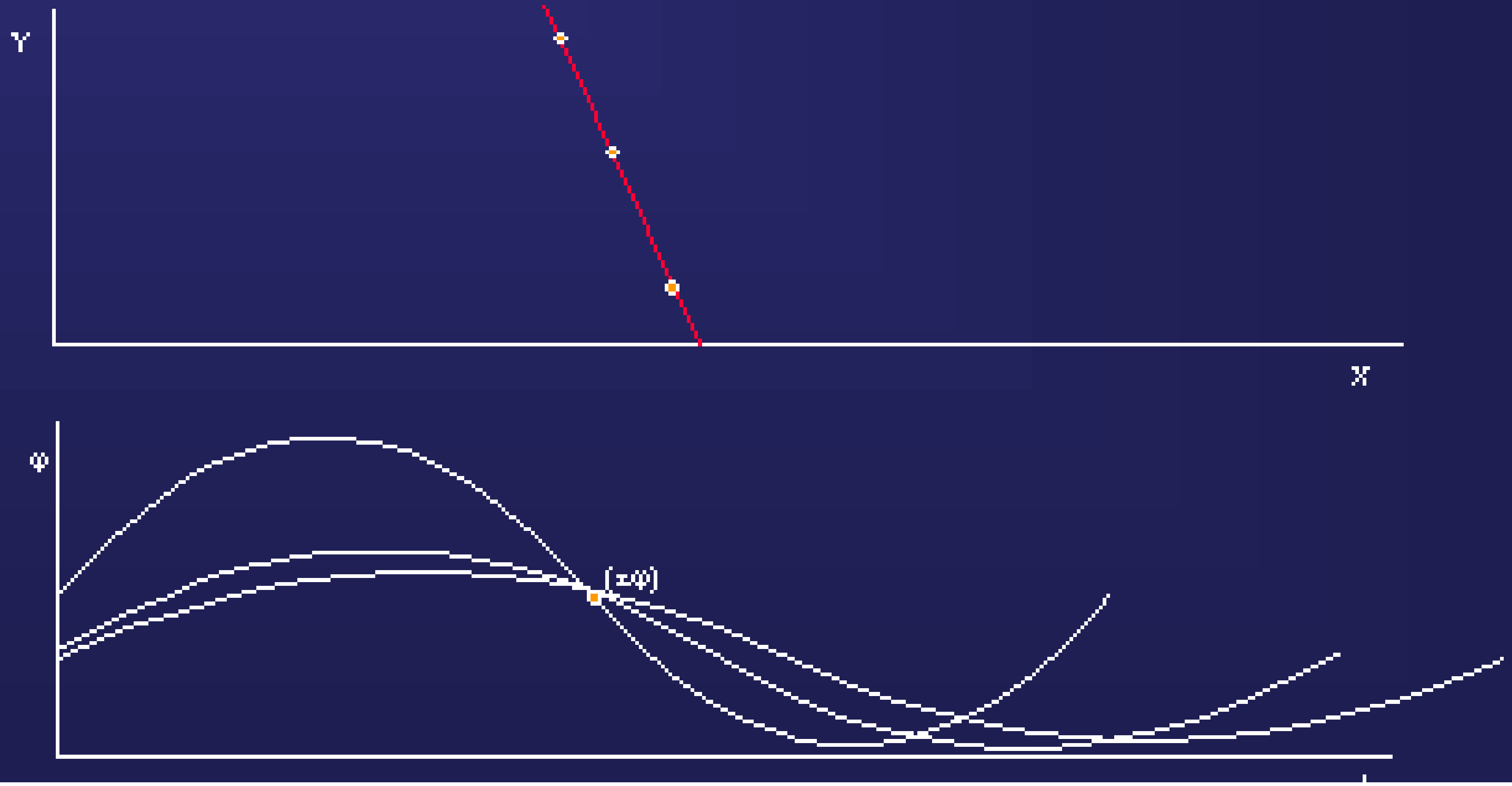
- Each sinusoid tells us every possible line that can pass through the corresponding (x,y) point



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

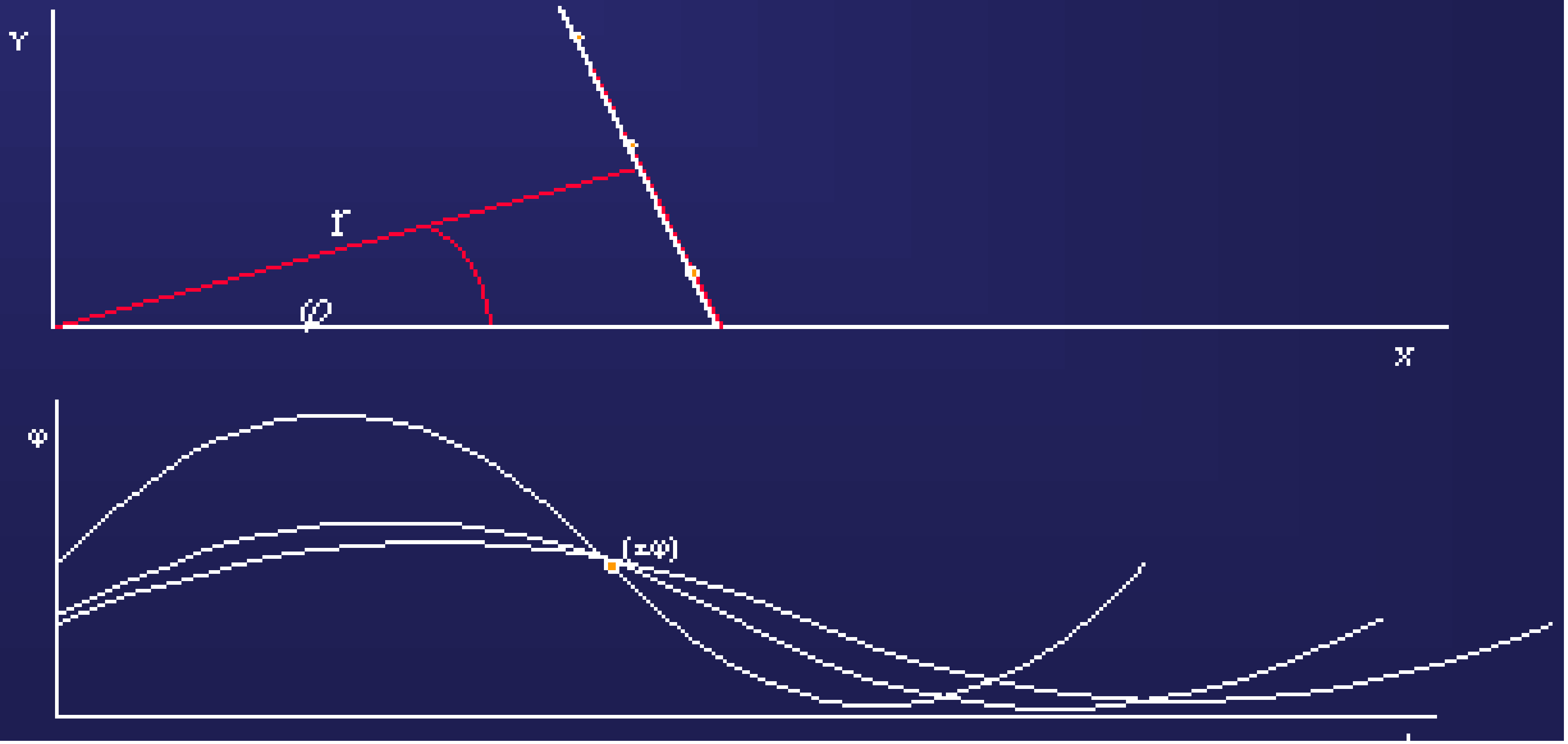
Hough Transform

- The line upon which the (x,y) points lie is then given by the (r,φ) pair on which all the sinusoids *agree* (i.e. intersect)



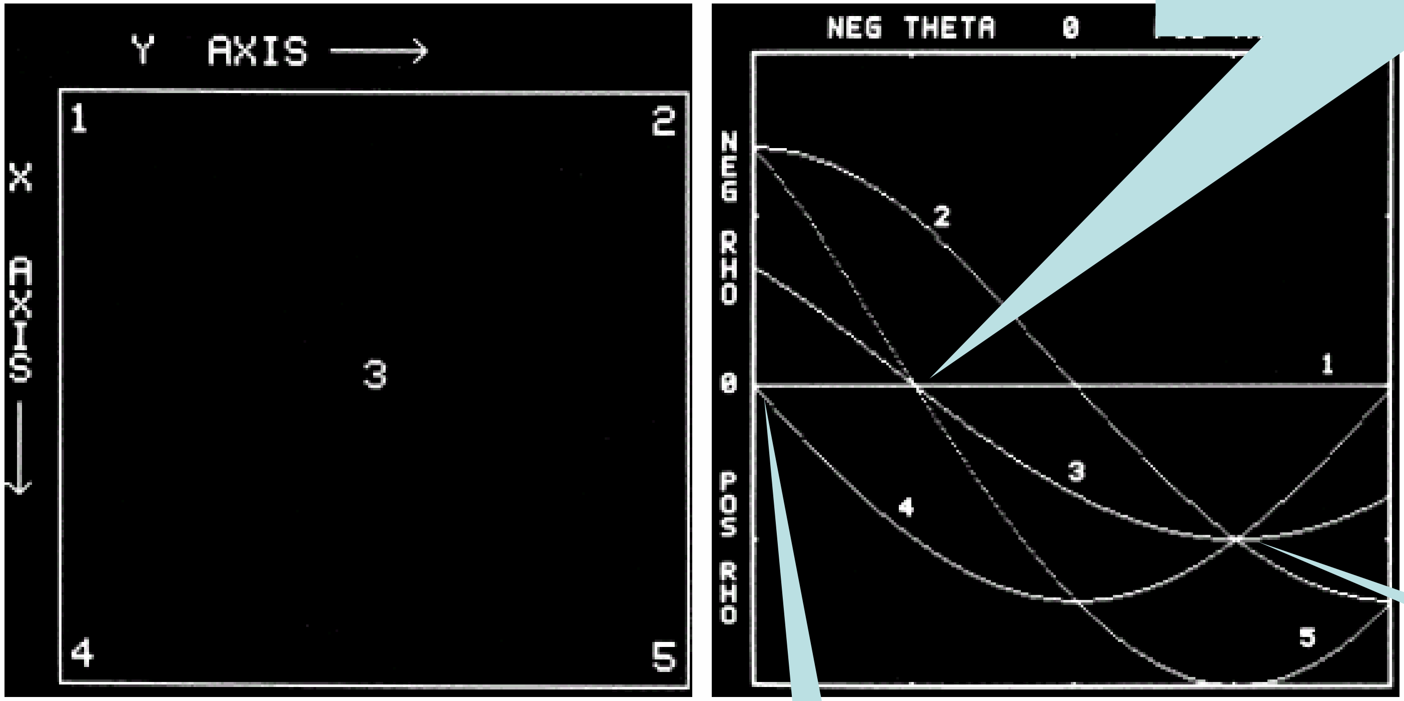
Hough Transform

- Finding this intersection we then have the required (r,φ) parameters and therefore the line in the image



Hough Transform

Images taken from Gonzalez & Woods, Digital Image Processing (2002)



The intersection of the curves corresponding to points 1,3,5

2,3,4

1,4

FIGURE 10.20 Illustration of the Hough transform. (Courtesy of Mr. D. R. Cate, Texas Instruments, Inc.)

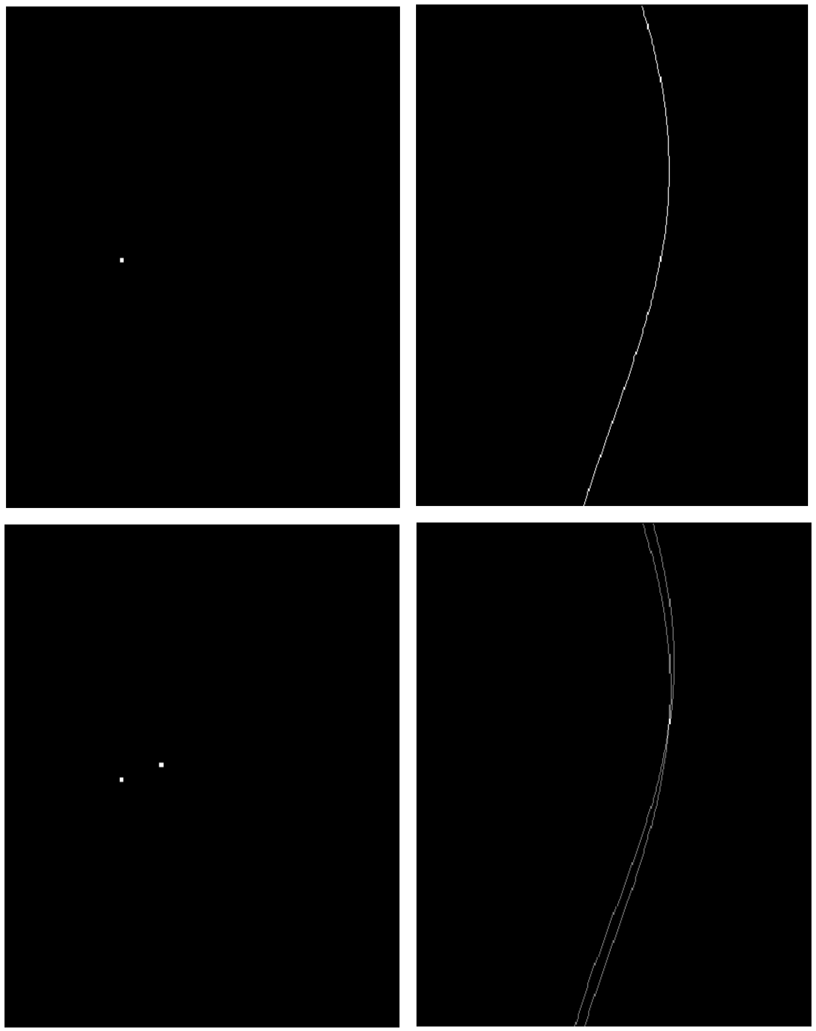
Hough Transform

One point in image space corresponds to a sinusoidal curve in image space

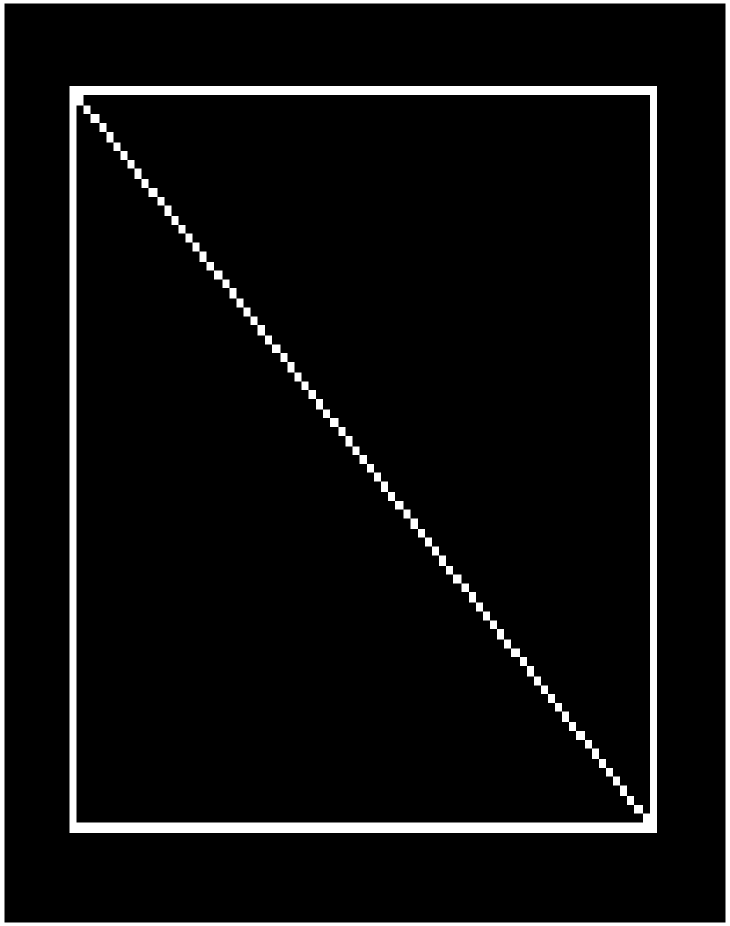
Two points correspond to two curves in Hough space

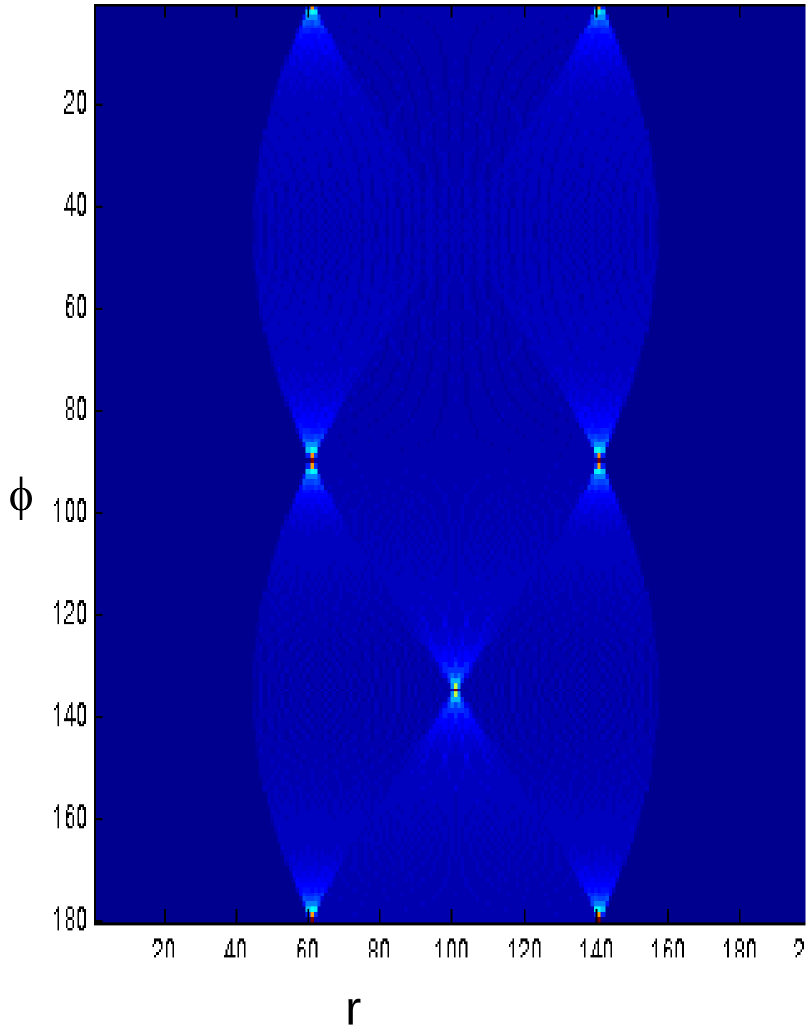
The intersection of those two curves has “two votes”.

This intersection represents the straight line in image space that passes through both points



Hough Transform





Hough Transform

Note that most points in the vote array are very dark, because they get only one vote.

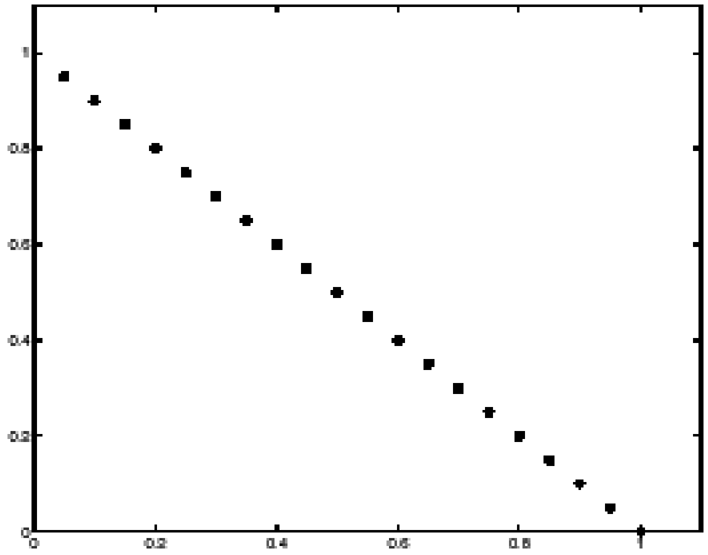
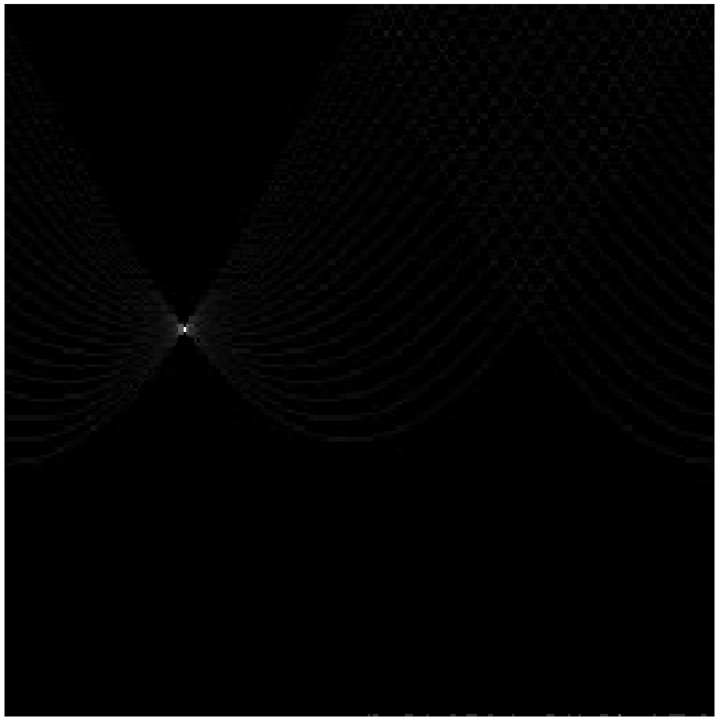


Image space



Votes

Horizontal axis is θ, vertical is rho.

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Hough Transform

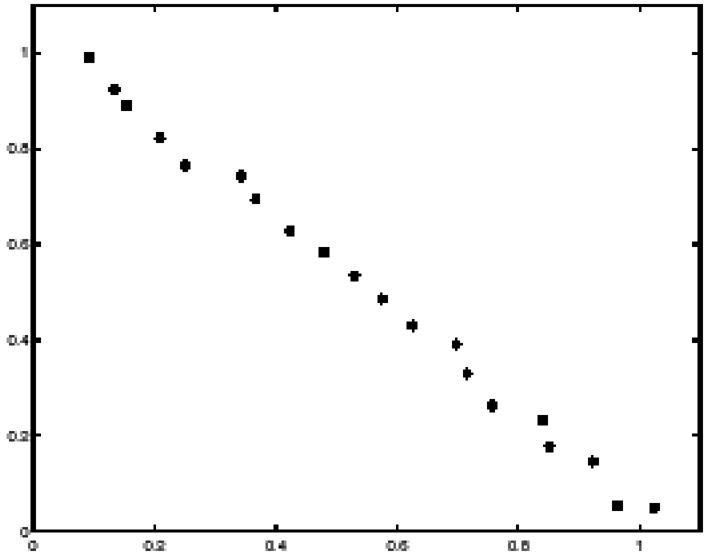
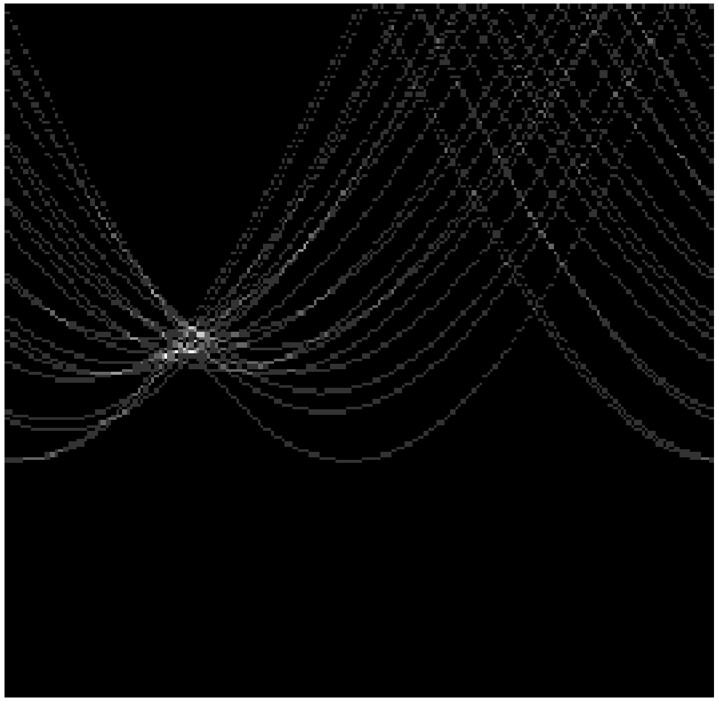


Image space



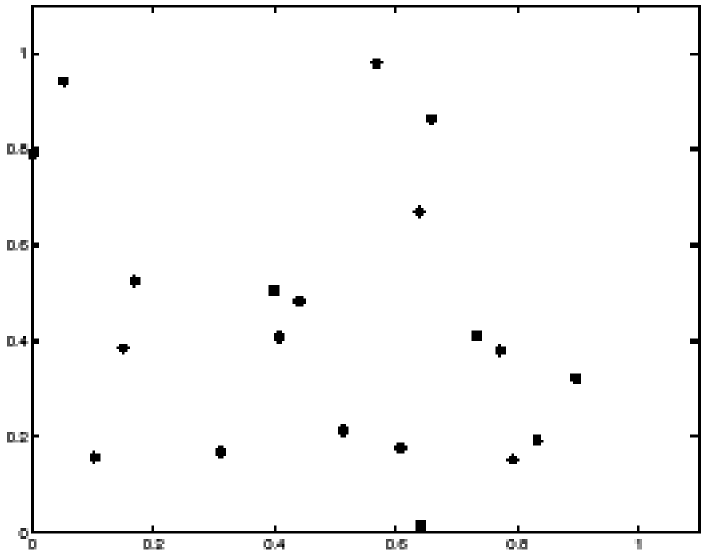
Votes

Horizontal axis is θ, vertical is rho.

Hough Transform

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Lots of noise can lead to large peaks in the array.



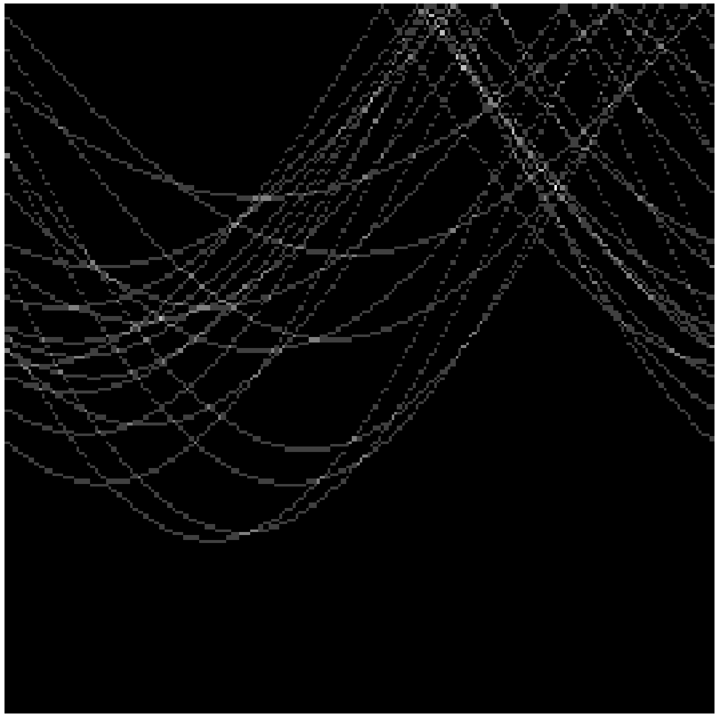
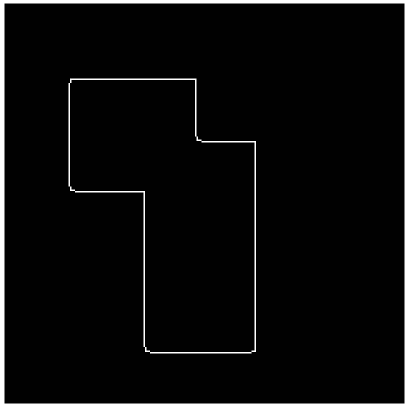


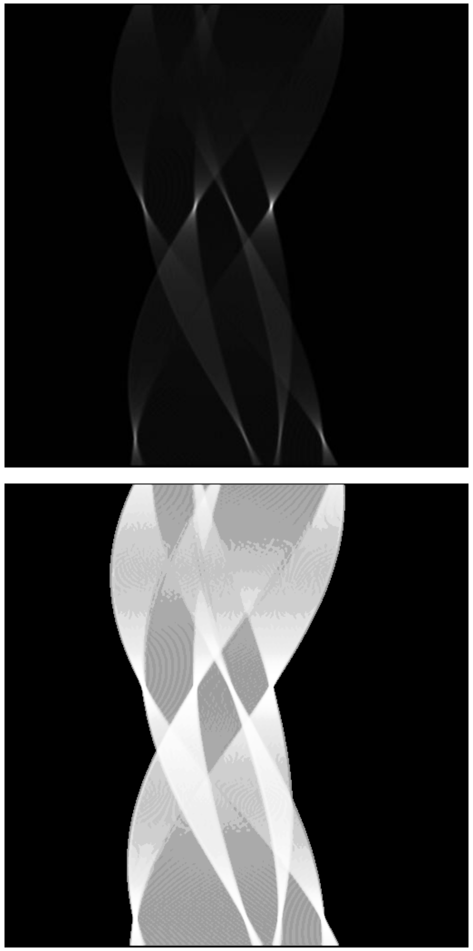
Image space

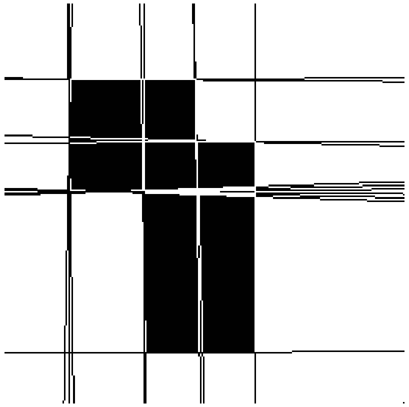
Votes

Horizontal axis is θ, vertical is rho.

Line Detection: Hough Transform

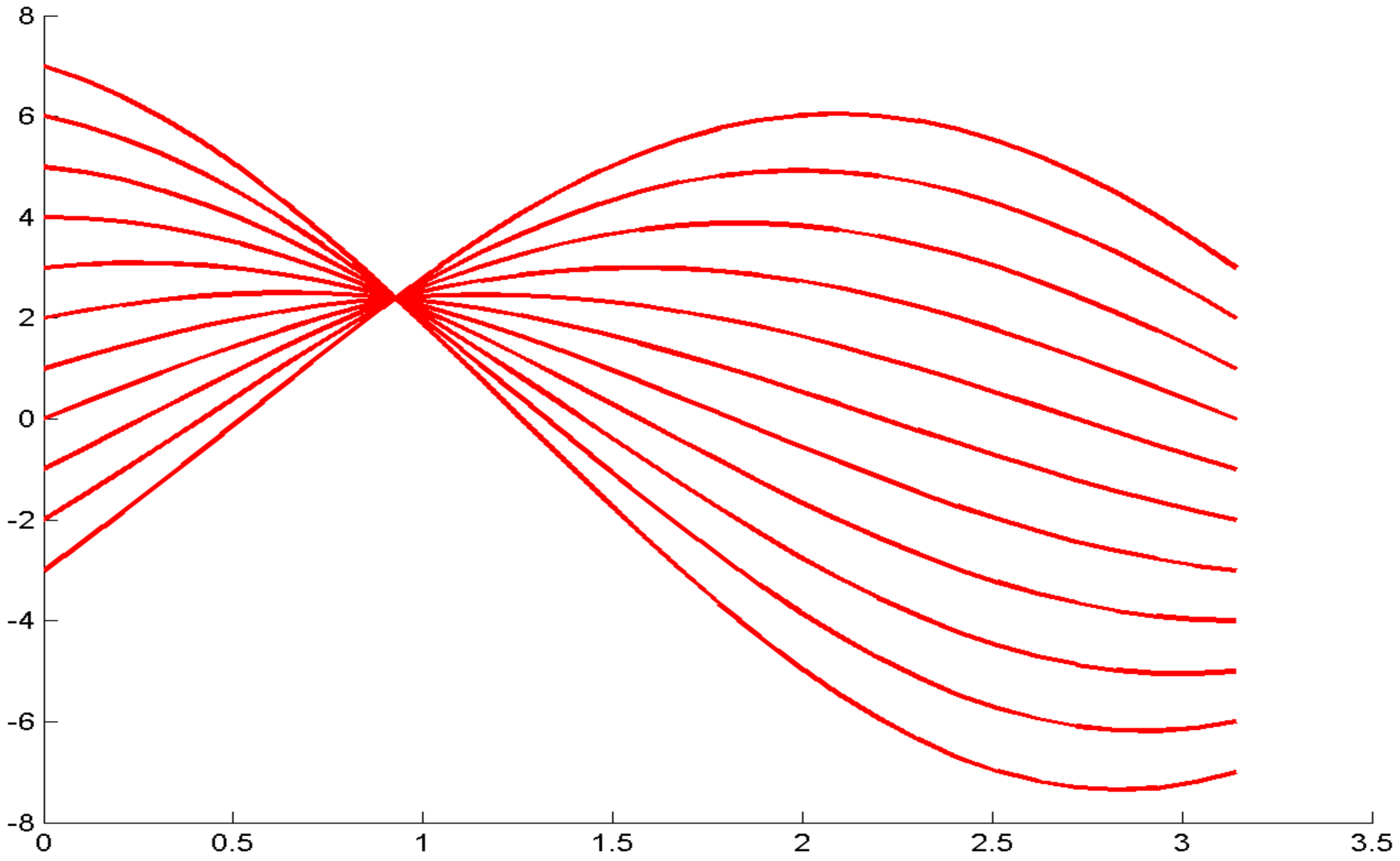

Edge Detected Image


Mapping of Edge Points in Hough Parameter Space



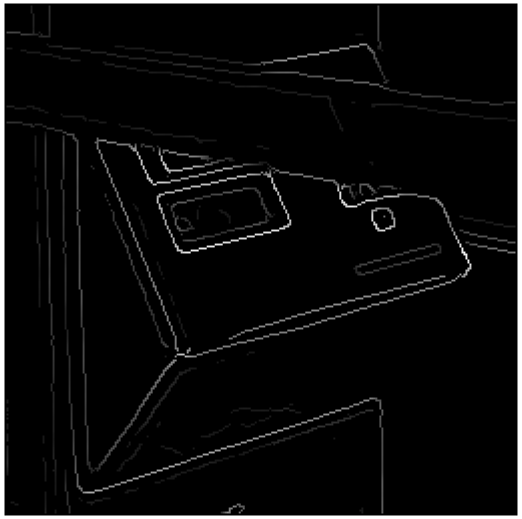
Hough Transform

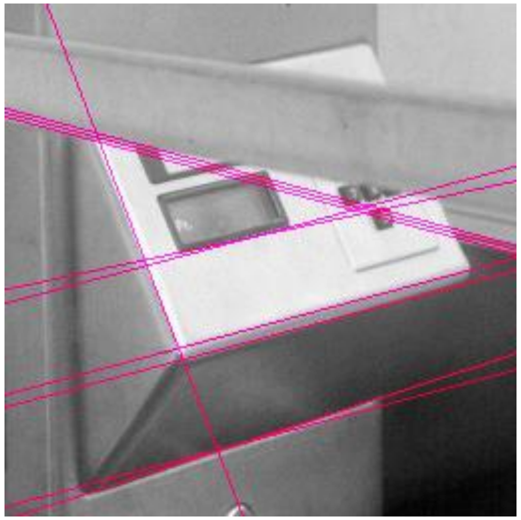
- Line: 0.6x + 0.4y = 2.4
  - Sinusoids intersect at: *θ* = 2.4, *r* = 0.9273

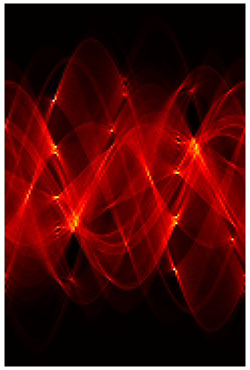


Hough Transform

Original


Edge detection


Found lines


Parameter space

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Hough Transform for Circles

- The parametric equation of the circle can be written as

$$(x-a)^2+(y-b)^2=r^2$$

- The equation has three parameters – *a*, *b*, *r*.
- The curve obtained in the Hough Transform space for each edge point will be a right circular cone.
- Point of intersection of the cones gives the parameters *a*, *b*, *r*.

Hough Transform for Circles

- Gradient at each edge point is known.
- We know the line on which the center will lie

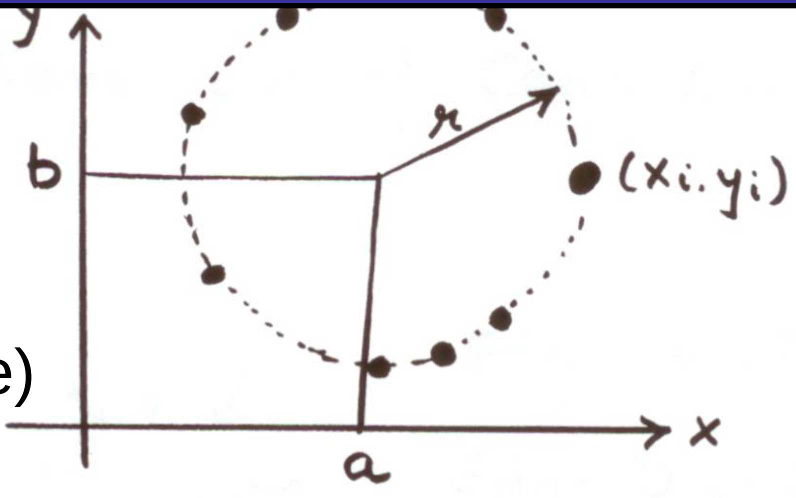
$$x_0 = x_i - R\cos\theta$$
$$y_0 = y_i - R\sin\theta$$

- If the radius is also known then center of the circle can be located.

Hough Transform for Circles

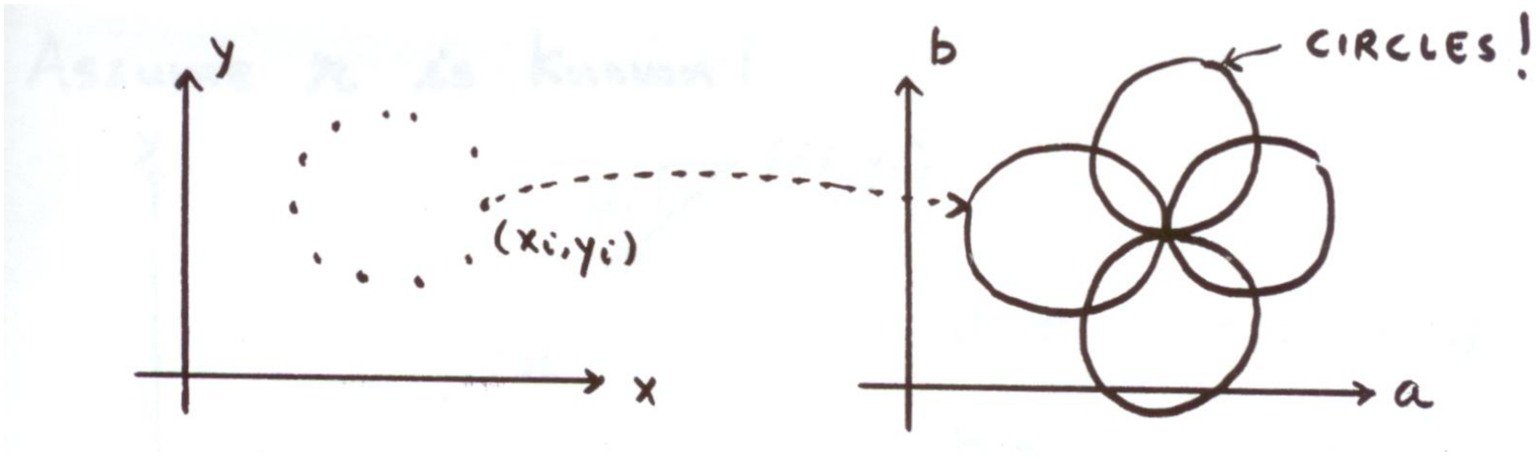
Equation of Circle:

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$



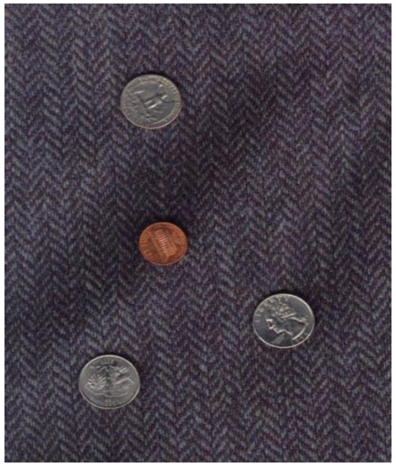
If radius is known: (2D Hough Space)

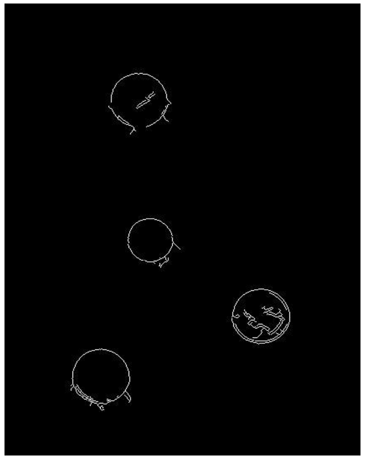
Accumulator Array $A(a,b)$

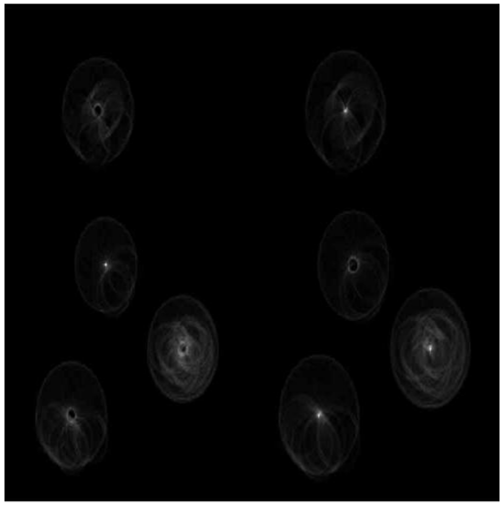


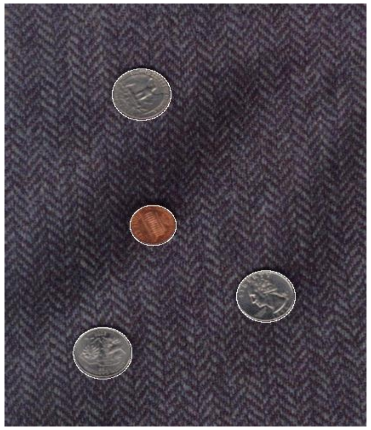
Hough Transform for Circles

Images taken from Gonzalez & Woods, Digital Image Processing (2002)


Original image


Canny


Hough space


Detected circles

Generalized Hough Transform

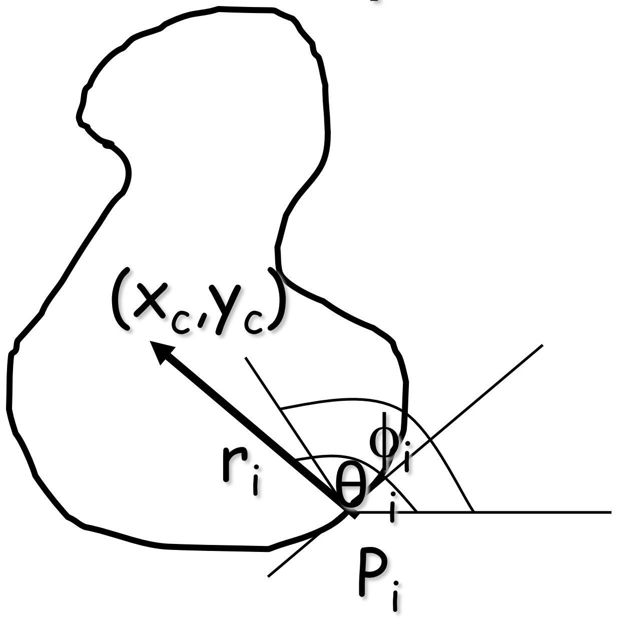
- The Generalized Hough transform can be used to detect arbitrary shapes.
- Complete specification of the exact shape of the target object is required in the form of the R-Table.
- Information that can be extracted are:
  - Location
  - Size
  - Orientation
  - Number of occurrences of that particular shape

Generalized Hough Transform

- Algorithm
  - Choose a reference point.
  - Draw a vector from the reference point to an edge point on the boundary.
  - Store the information of the vector against the gradient angle in the R-Table.
  - There may be more than one entry in the R-Table corresponding to a gradient value.

Generalized Hough Transform

- The H.T. can be used even if the curve has no simple analytic form!



1. Pick a reference point (x_c,y_c)
2. For i = 1,…,n :
   - Draw segment to P_i on the boundary.
   - Measure its length r_i, and its orientation θ_i.
   - Write the coordinates of (x_c,y_c) as a function of r_i and θ_i
   - Record the gradient orientation f_i at P_i.
3. Build a table with the data, indexed by f_i .

$$x_c = x_i + r_i\cos(\theta_i)$$

$$y_c = y_i + r_i\sin(\theta_i)$$

Generalized Hough Transform

- Form an Accumulator array to hold the candidate locations of the reference point.
- For each point on the edge:
  - Compute the gradient direction and determine the row of the R-Table it corresponds to
  - For each entry on the row calculate the candidate location of the reference point

$$x_c = x_i + r\cos\theta$$

$$y_c = y_i + r\sin\theta$$

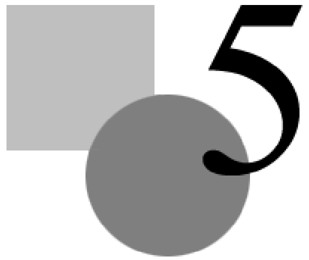
  - Increase the Accumulator value for that point
- The reference point location is given by the highest value in the accumulator array.

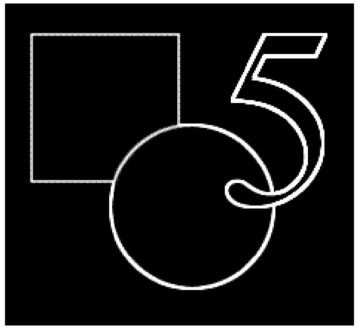
Generalized Hough Transform

- The size and orientation of the shape can be found out by simply manipulating the R-Table.
- For scaling by factor *S* multiply the R-Table vectors by *S*.
- For rotation by angle θ, rotate the vectors in the R-Table by angle θ.

Edge Detection

- Edge - Area of significant change in the image intensity / contrast.
- Edge Detection – Locating areas with strong intensity contrasts.
- Use of Edge Detection – Extracting information about the image, e.g. location of objects present in the image, their shape, size, image sharpening and enhancement.

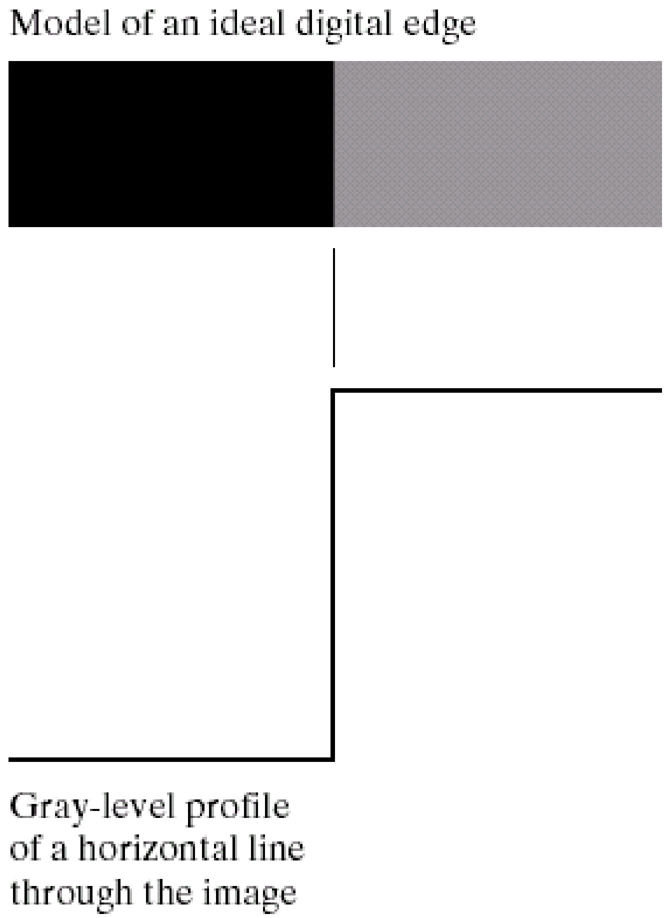


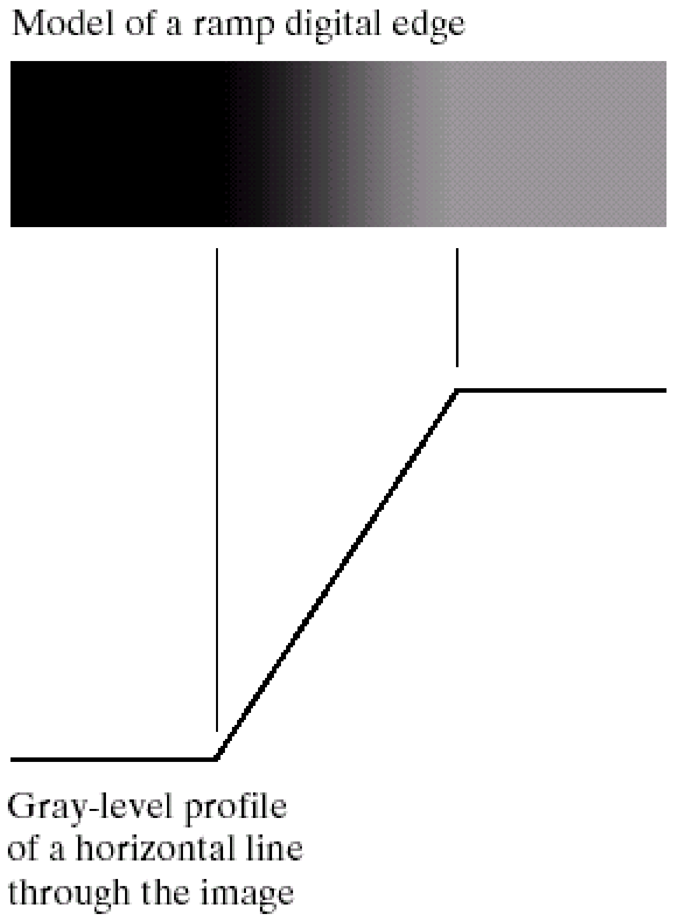


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Edge Detection

•An edge is a set of connected pixels that lie on the boundary between two regions.


Model of an ideal digital edge
Gray-level profile of a horizontal line through the image

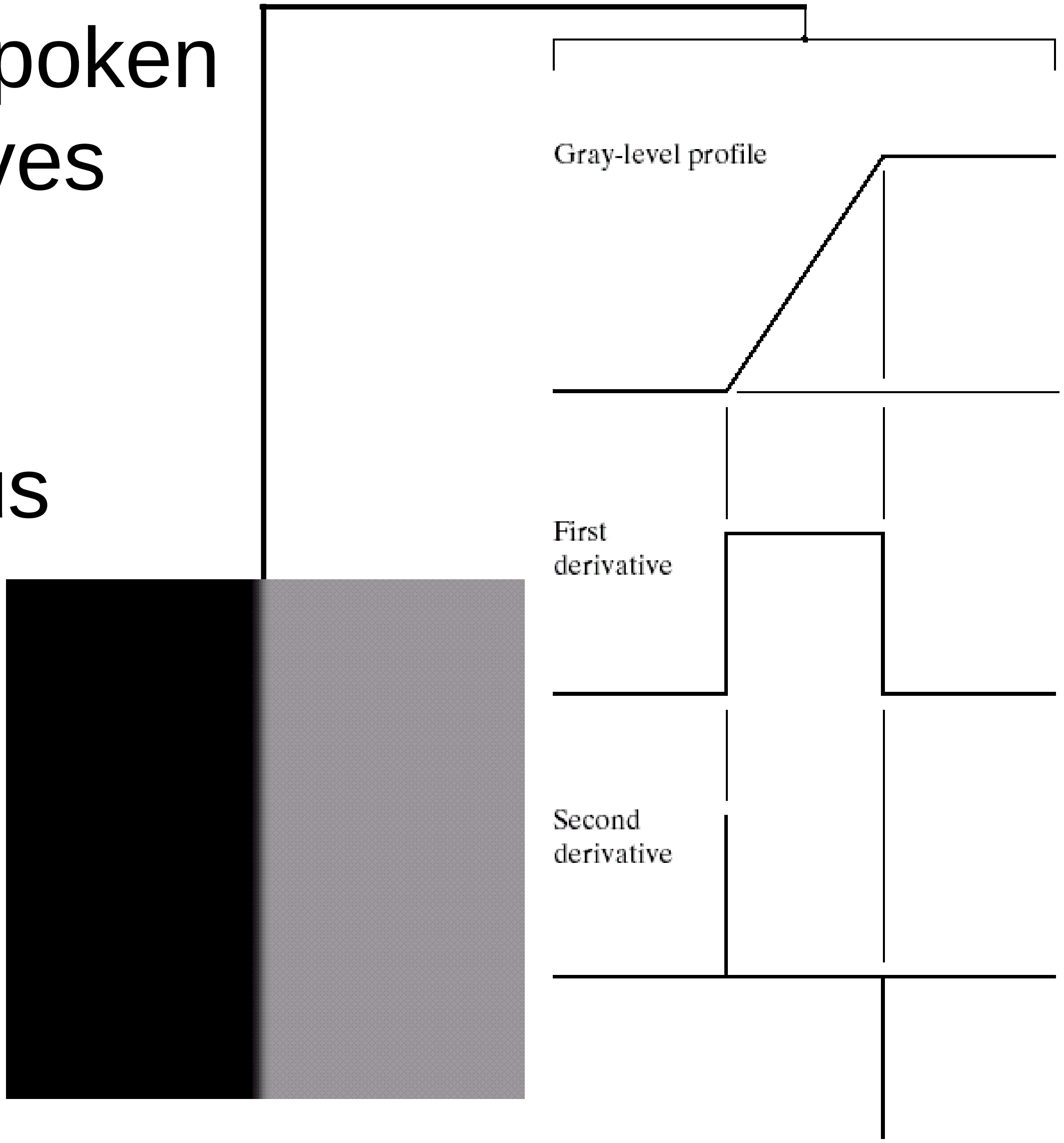

Model of a ramp digital edge
Gray-level profile of a horizontal line through the image

Edges & Derivatives

We have already spoken about how derivatives are used to find discontinuities

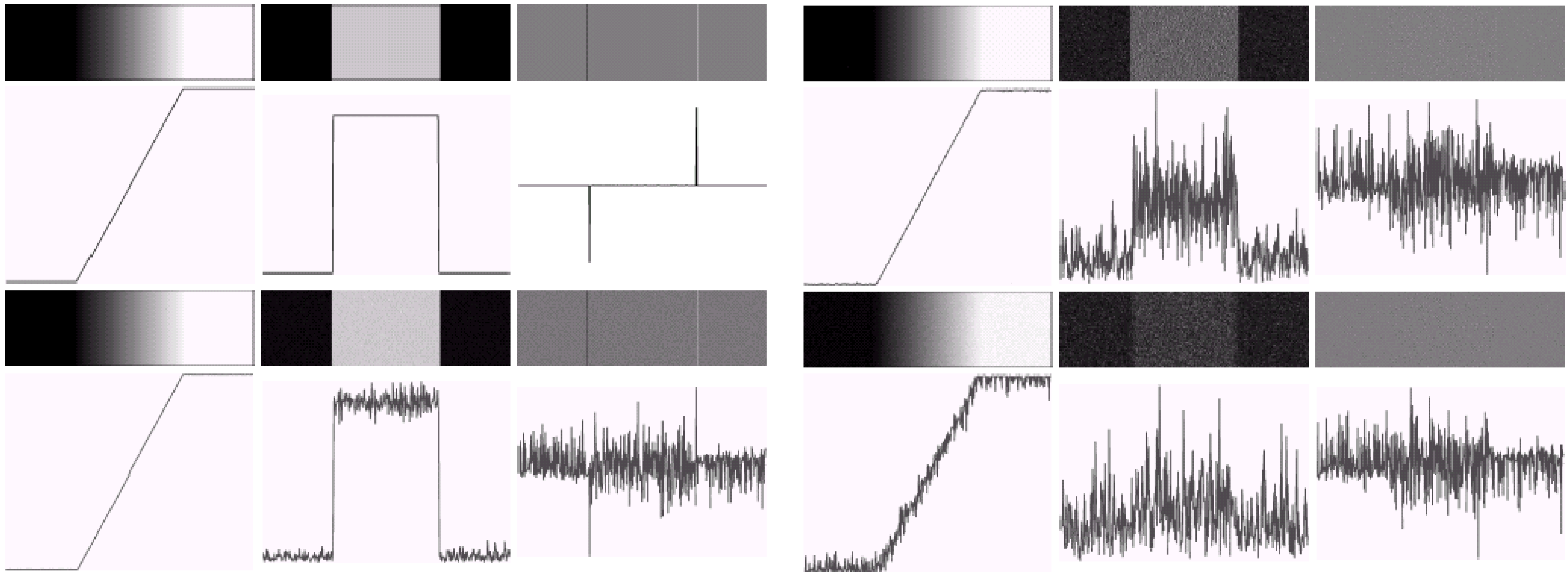
1st derivative tells us where an edge is

2nd derivative can be used to show edge direction



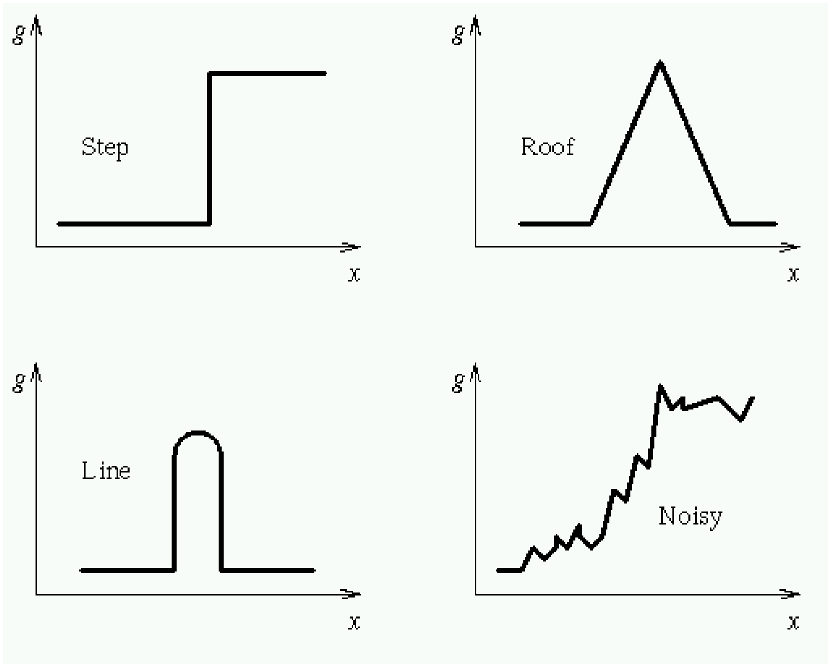
Derivatives & Noise

Derivative based edge detectors are extremely sensitive to noise



Edge Detection

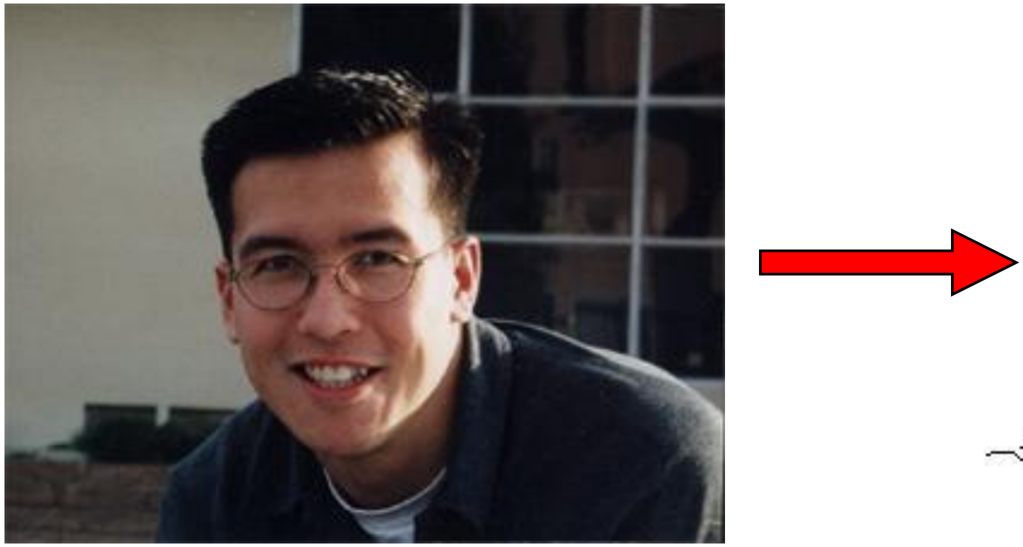
- Types of edges: Variation of Intensity / Gray Level
    - Step Edge
    - Ramp Edge
    - Line Edge
    - Roof Edge



Typical edge profiles.

Edge Detection

- Our goal is to extract a “line drawing” representation from an image.
- Useful for recognition: edges contain shape information.



Edge Detection

- Edges are locations with high image gradient or derivative.
- Estimate derivative using finite difference.

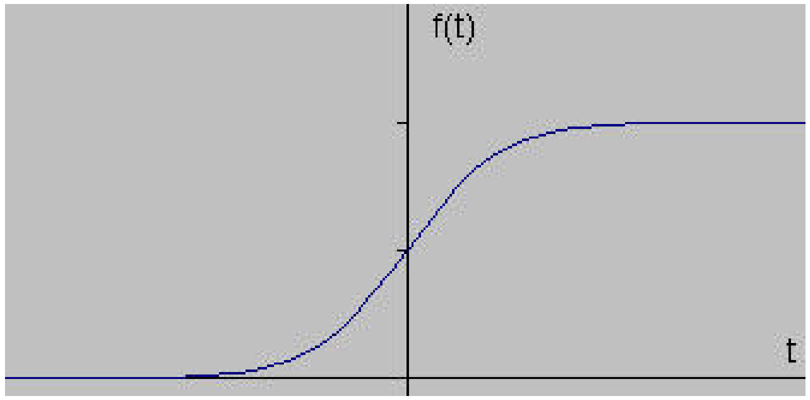
$$\frac{\partial}{\partial x} I(x_0, y_0) \approx I(x_0 + 1, y_0) - I(x_0, y_0)$$

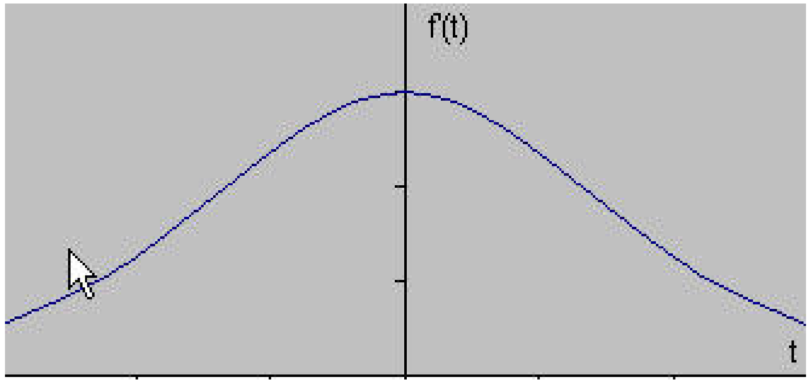
Edges & Derivatives

- First Order Derivative / Gradient Methods
  - Roberts Operator
  - Sobel Operator
  - Prewitt Operator
- Second Order Derivative
  - Laplacian
  - Laplacian of Gaussian
  - Difference of Gaussian
- Optimal Edge Detection
  - Canny Edge Detection

First Derivative

- At the point of greatest slope, the first derivative has maximum value.





First Derivative

- For a continuous two dimensional function gradient is defined as:

$$G[f(x,y)] = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

$$|G| = \sqrt{Gx^2 + Gy^2} \approx |Gx| + |Gy|$$

$$\theta = \tan^{-1}\left(\frac{Gy}{Gx}\right)$$

Gradient

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \approx f(x+1) - f(x) \;\; (h{=}1)$$

$$\frac{\partial f}{\partial x} = \frac{f(x+h_x, y) - f(x,y)}{h_y} = f(x+1, y) - f(x,y), \;\; (h_x{=}1)$$

$$\frac{\partial f}{\partial y} = \frac{f(x, y+h_y) - f(x,y)}{h_y} = f(x, y+1) - f(x,y), \;\; (h_y{=}1)$$

Gradient

- Convolution Mask:
  - Gx = $\begin{array}{|c|c|}\hline -1 & 1 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|}\hline -1 & 1 \\ \hline -1 & 1 \\ \hline \end{array}$
  - Gy = $\begin{array}{|c|}\hline 1 \\ \hline -1 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|}\hline 1 & 1 \\ \hline -1 & -1 \\ \hline \end{array}$

Common Edge Detectors

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

•Given a 3*3 region of an image, the following edge detection filters can be used:

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	−1	−1
0	0	0
1	1	1

−1	0	1
−1	0	1
−1	0	1

Prewitt

−1	0
0	1

0	−1
1	0

Roberts

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

Roberts Operator

- Provides an approximation to the gradient

$$G[f(i,j)] = |Gx| + |Gy| = |f(i,j) - f(i+1, j+1)| + |f(i+1, j) - f((i, j+1)|$$

- Convolution Mask:
  - Gx =

1	0
0	-1

  - Gy =

0	-1
1	0

Sobel Operator

- The 3x3 convolution mask smoothes the image by some amount , hence it is less susceptible to noise.
  - But it produces thicker edges. So edge localization is poor.

- Convolution Mask:

Gx =

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

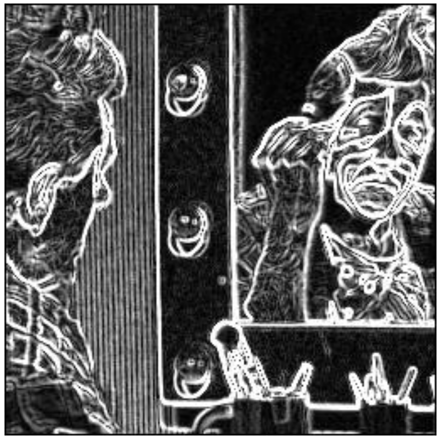
Gy =

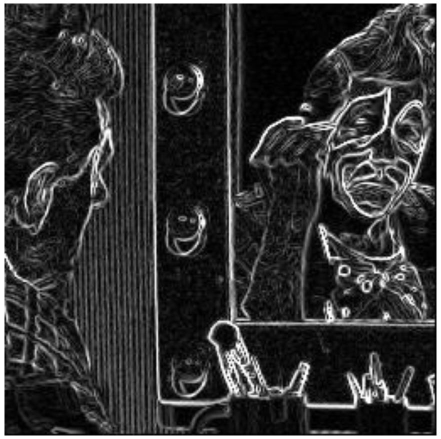
1	2	1
0	0	0
-1	-2	-1

- The differences are calculated at the center pixel of the mask.

Comparison

- Compare the output of the Sobel Operator with that of the Roberts Operator:
  - The spurious edges are still present but they are relatively less intense compared to genuine lines.
  - Roberts operator has missed a few edges.
  - Sobel operator detects thicker edges.


Sobel


Roberts

Prewitt Operator

- It is similar to the Sobel operator but uses slightly different masks.
- Convolution Mask:

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

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

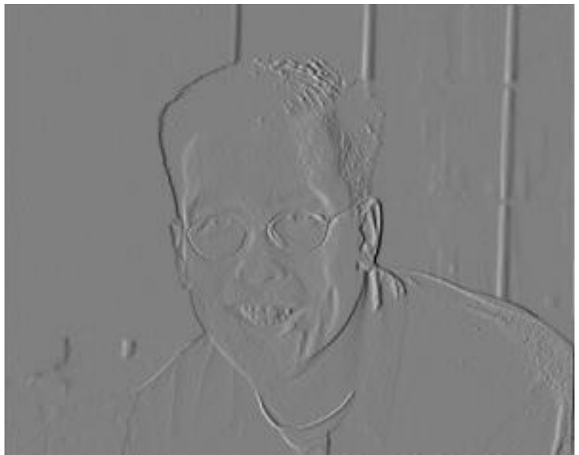
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

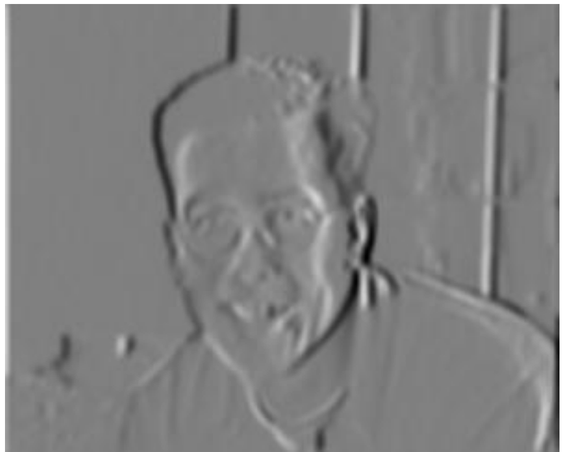
Derivatives & Noise

- Reduce image noise by smoothing with a Gaussian.

$$J = G * I$$

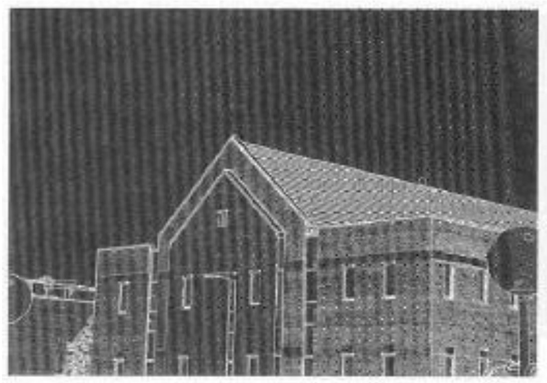
$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$





Edge Orientation

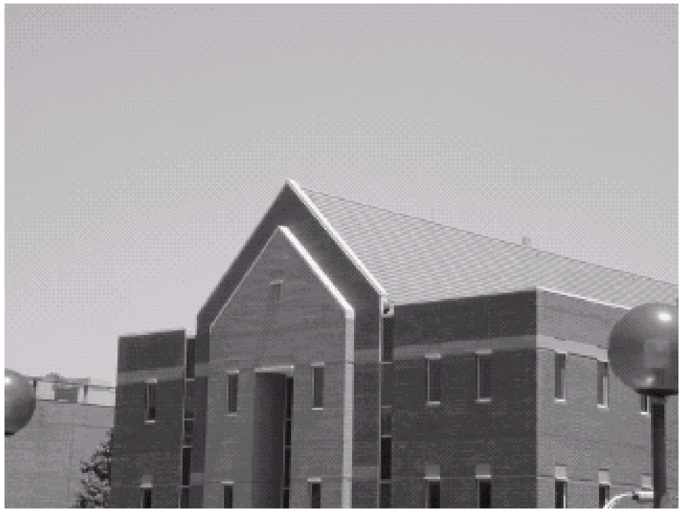
- Would like gradients in all directions.
- Approximate:
  - Compute smoothed derivatives in *x*,*y* directions.
  - Edge strength $$e_s(i,j) = \sqrt{J_x^2(i,j) + J_y^2(i,j)}$$
  - Edge normal $$e_o(i,j) = atan\frac{J_y}{J_x}$$



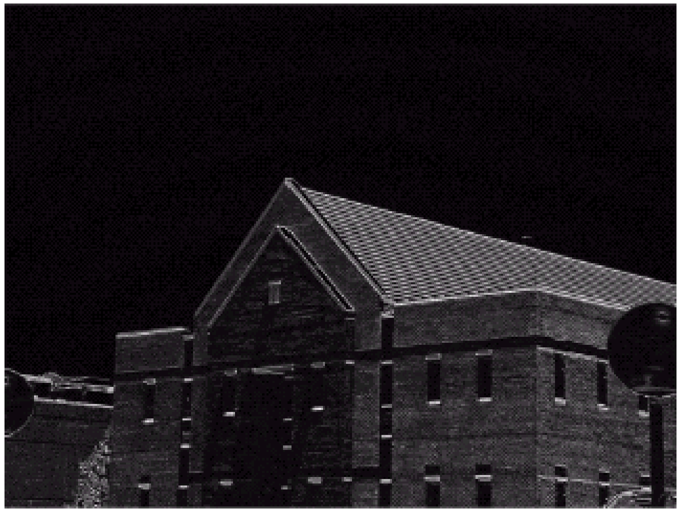
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

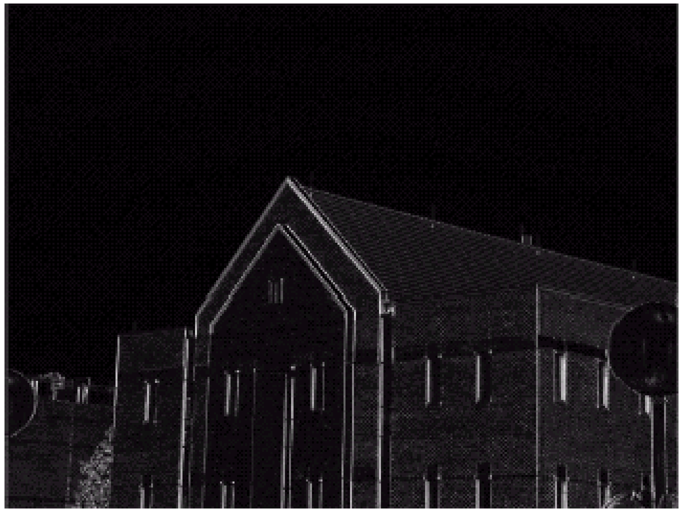
Edge Detection Example

Original Image

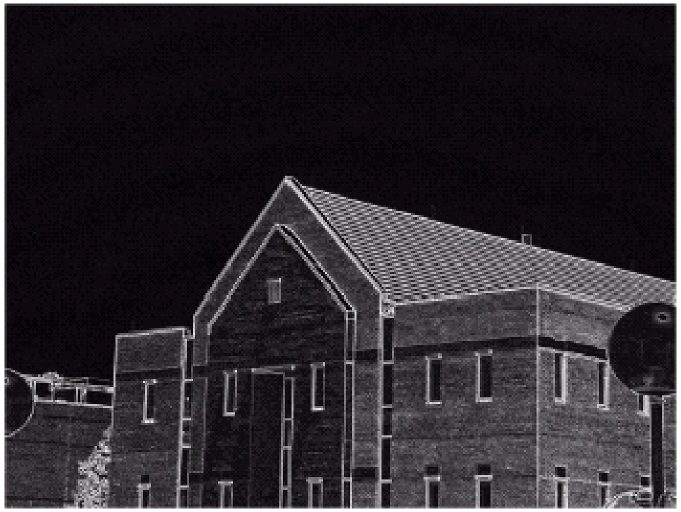


Horizontal Gradient Component



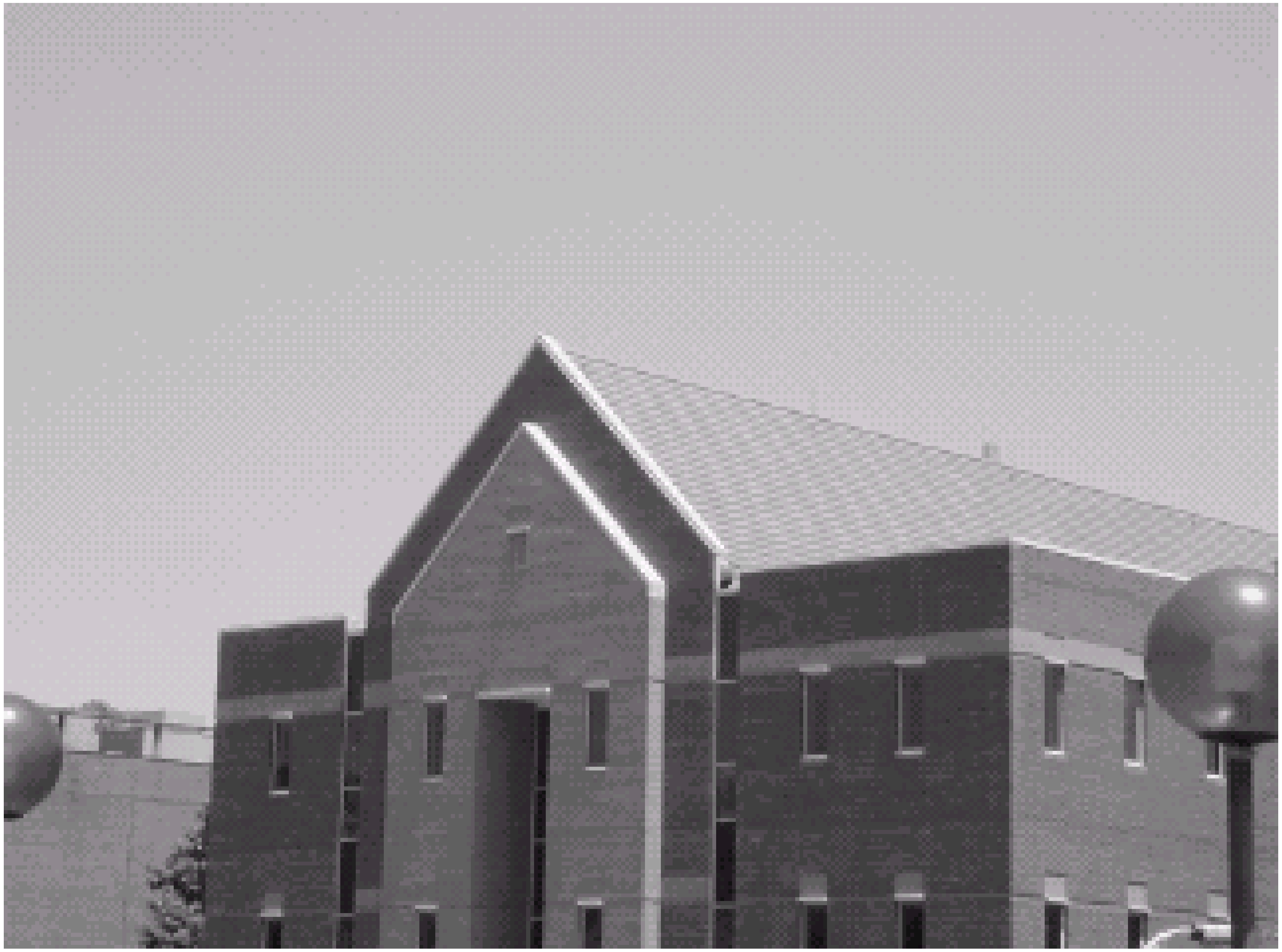


Vertical Gradient Component

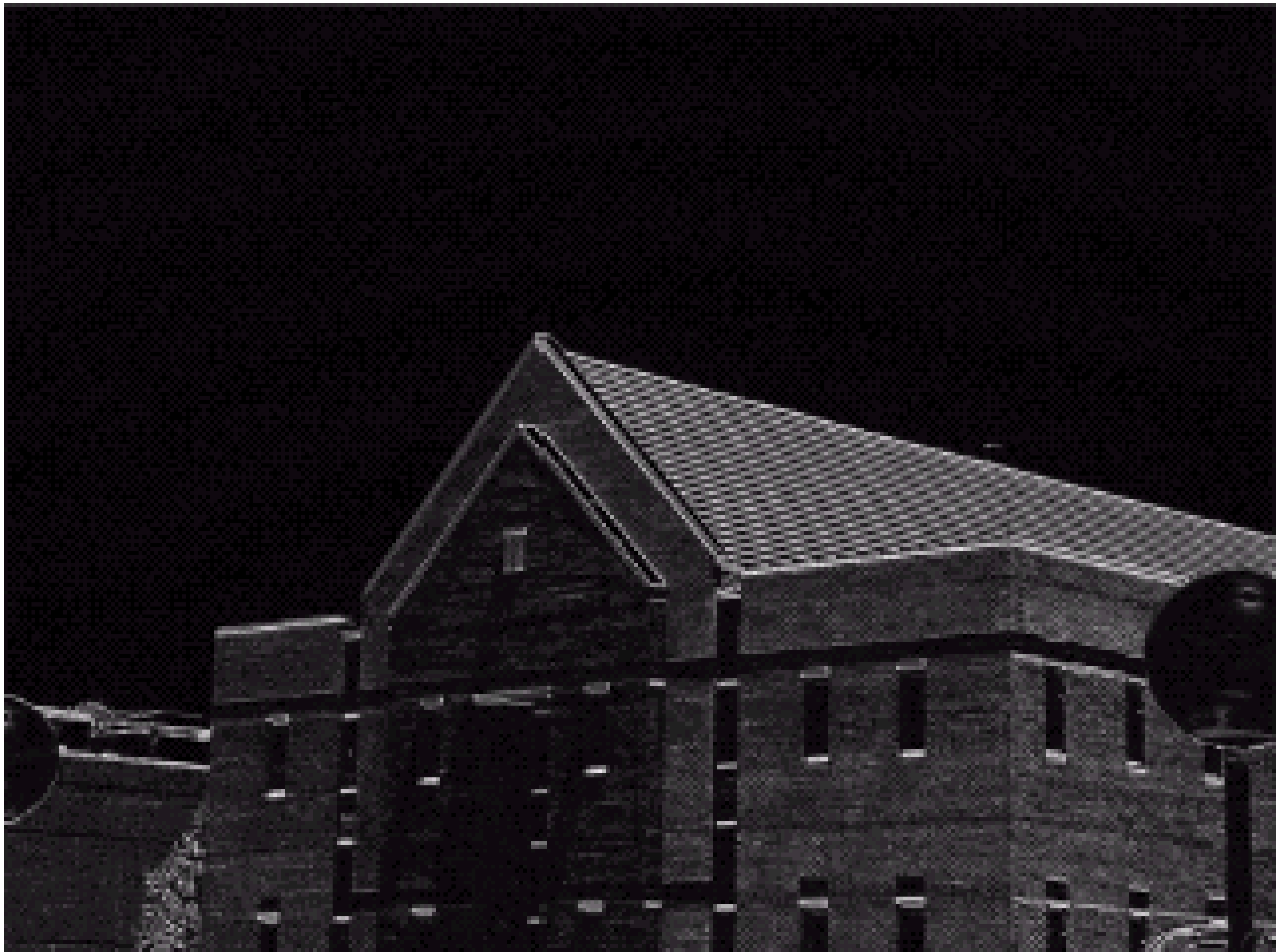


Combined Edge Image

Edge Detection Example

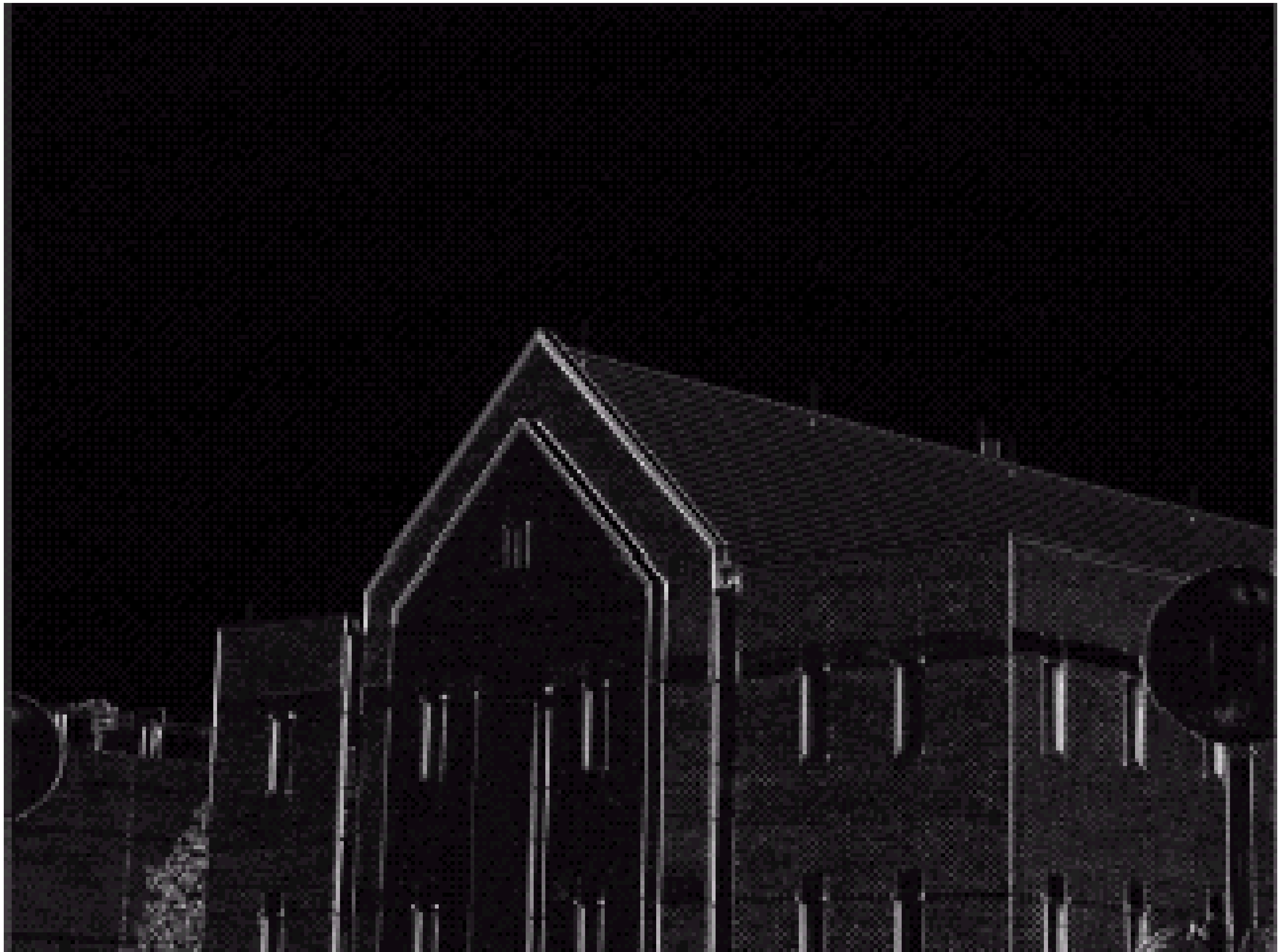


Edge Detection Example



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

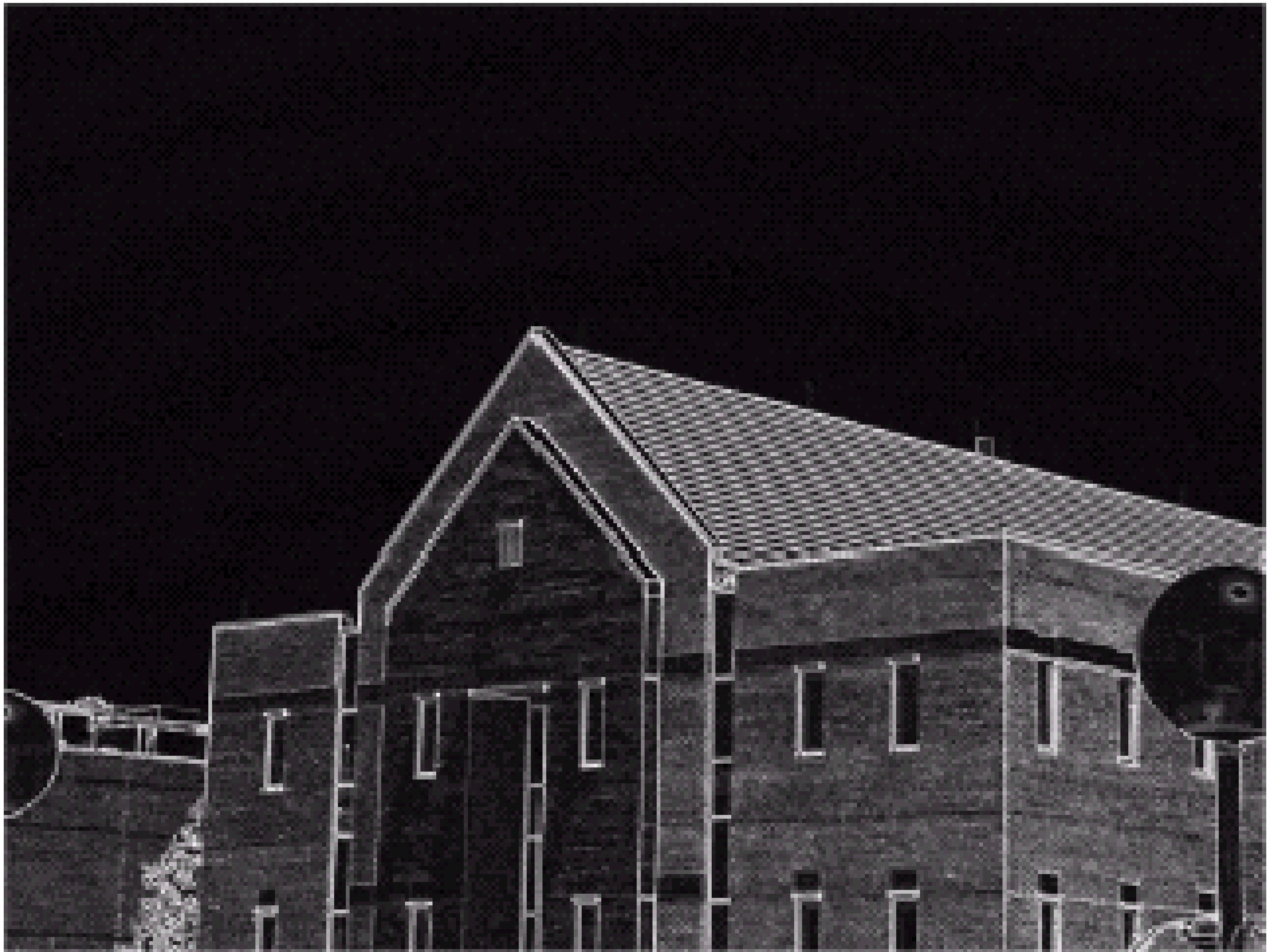
Edge Detection Example



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Edge Detection Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

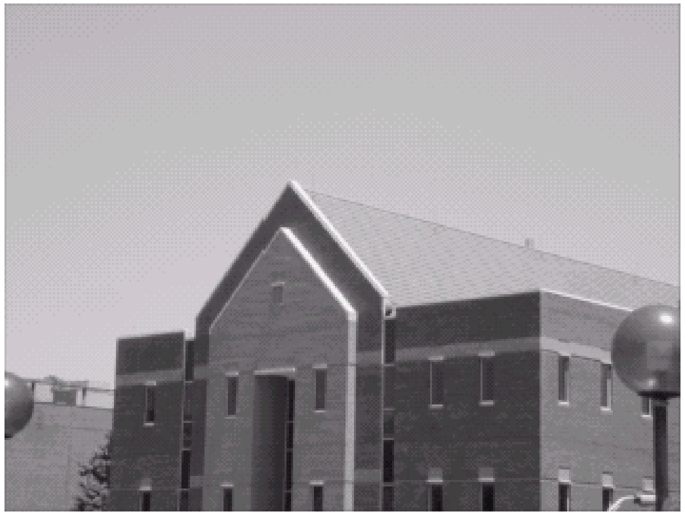


Edge Detection Problems

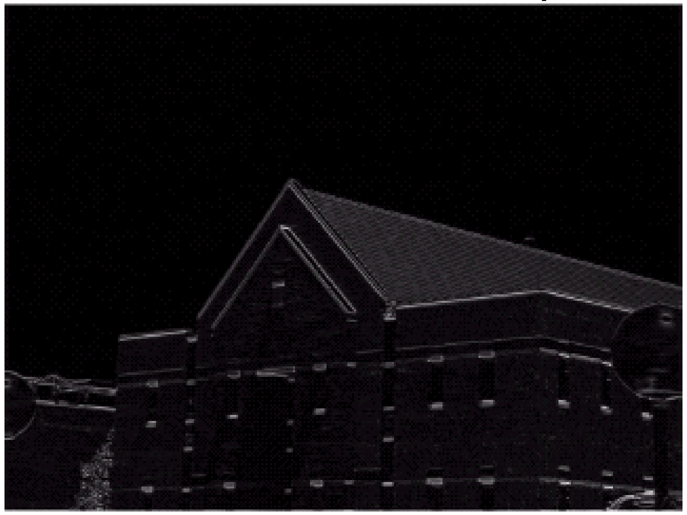
- Often, problems arise in edge detection when there is too much detail.
  - For example, the brickwork in the previous example.
- One way to overcome this is to smooth images prior to edge detection.

Edge Detection Example With Smoothing

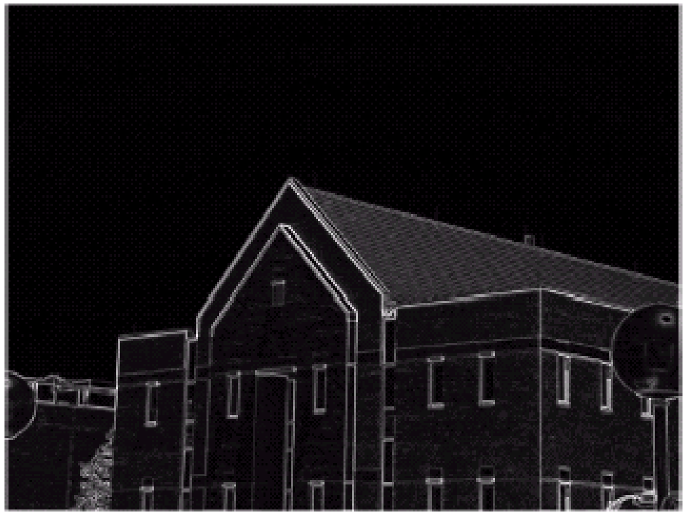
Original Image



Horizontal Gradient Component



Vertical Gradient Component



Combined Edge Image

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Second Order Derivatives

$$f''(x) = \lim_{h \to 0} \frac{f'(x+h) - f'(x)}{h} \approx f'(x+1) - f'(x) =$$

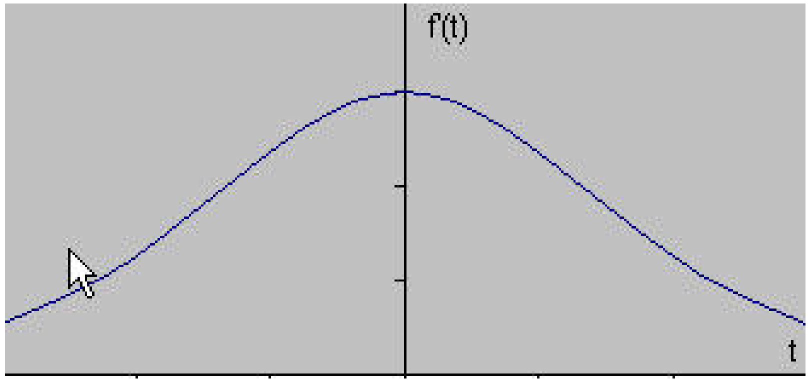
$$f(x+2) - 2f(x+1) + f(x) \quad (h=1)$$

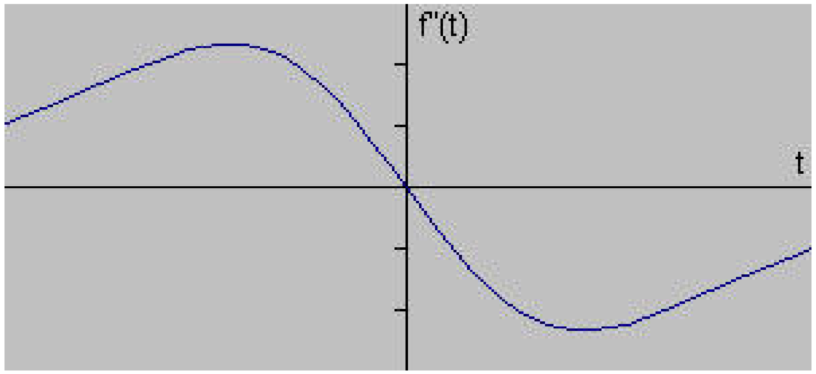
$$f''(x) \approx f(x+1) - 2f(x) + f(x-1)$$

mask: $[\,1 \quad -2 \quad 1\,]$

Second Order Derivatives

- Zero crossing of the second derivative of a function indicates the presence of a maxima.





Laplacian Edge Detection

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

- Laplacian is a 2nd-order derivative based filter defined as:

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

- Mask:

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

  – The Laplacian is typically not used by itself as it is too sensitive to noise. Usually, the Laplacian is combined with a smoothing Gaussian filter.

Laplacian of Gaussian (LoG)

- Also called Marr-Hildreth Edge Detector.
- Steps:
  - Smooth the image using Gaussian filter.
  - Enhance the edges using Laplacian operator.
  - Zero crossings denote the edge location.
  - Use linear interpolation to determine the sub-pixel location of the edge.

Laplacian of Gaussian

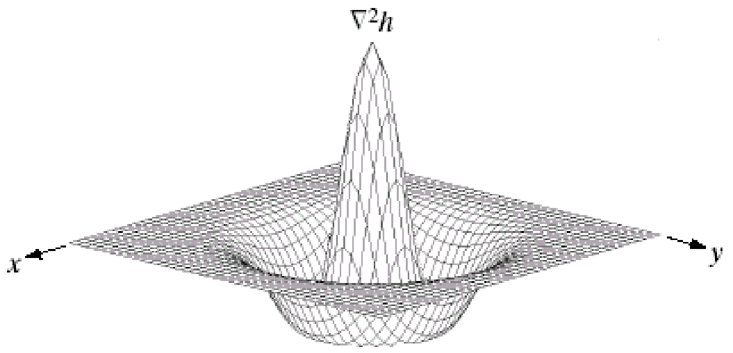
- Defined as:

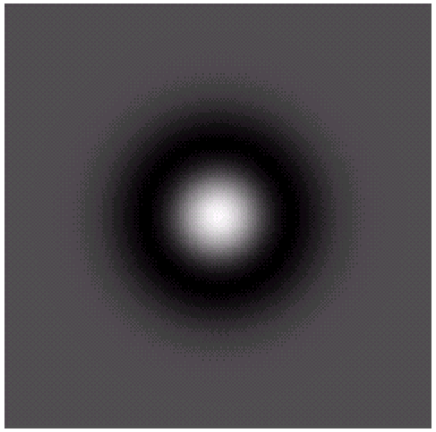
$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- Greater the value of σ (standard deviation), broader is the Gaussian filter, more is the smoothing.
- Too much smoothing may make the detection of edges difficult.

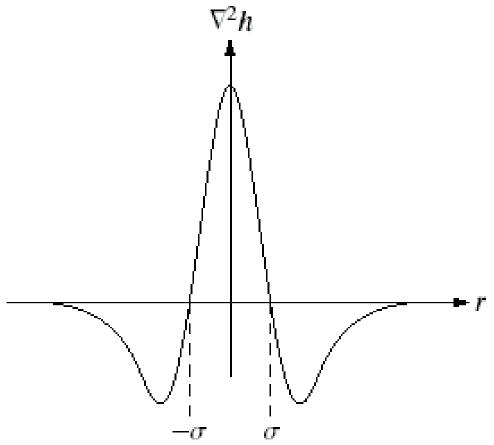
Laplacian Of Gaussian

- The LoG filter uses the Gaussian for noise removal and the Laplacian for edge detection.



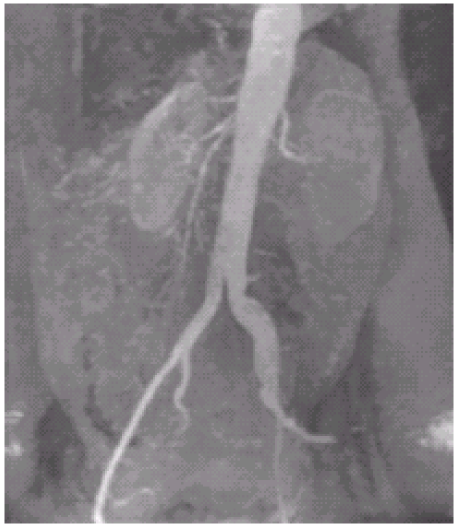
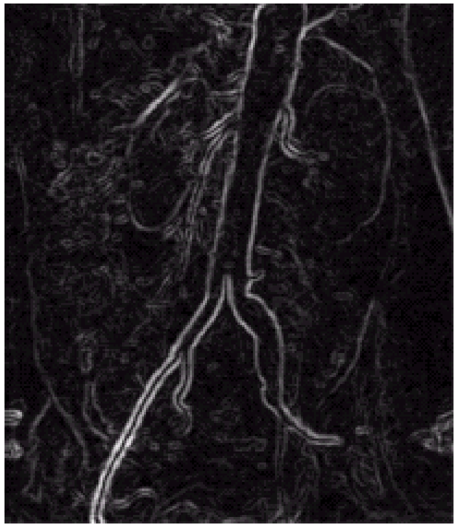
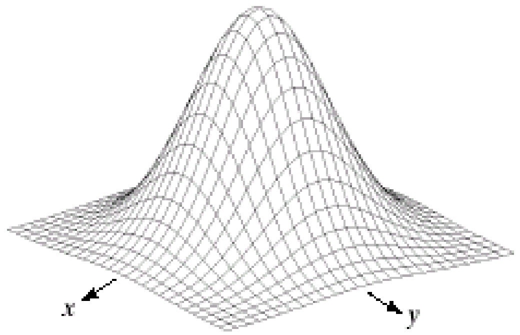


0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

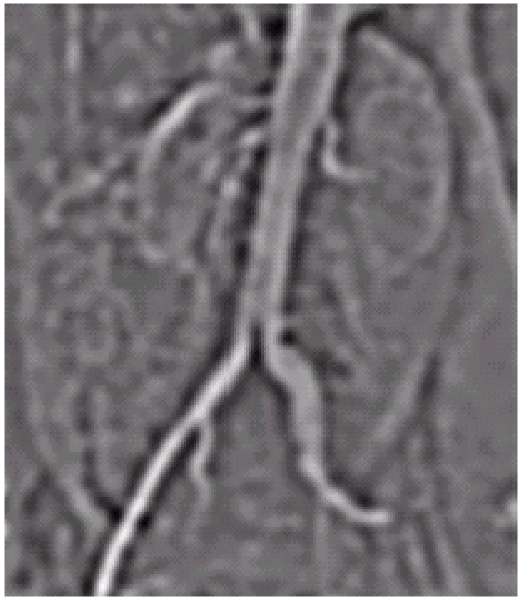
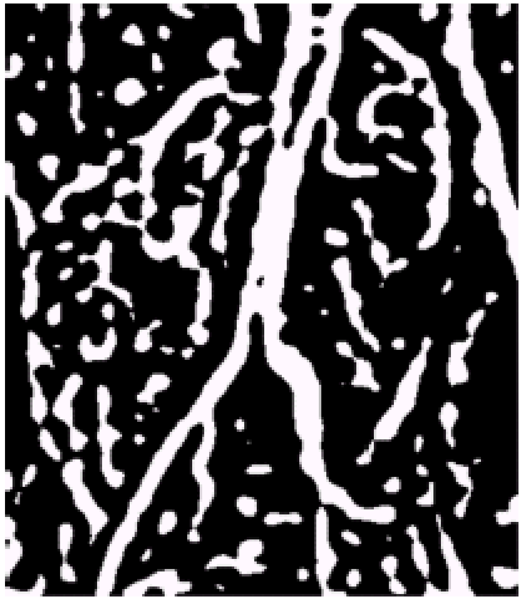
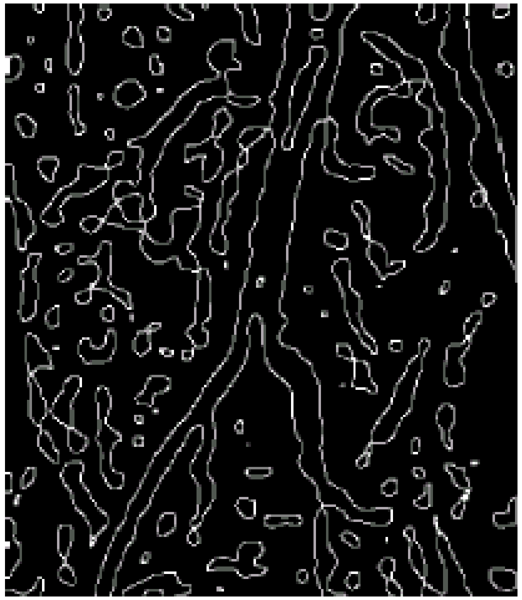


Laplacian Of Gaussian Example

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

−1	−1	−1
−1	8	−1
−1	−1	−1

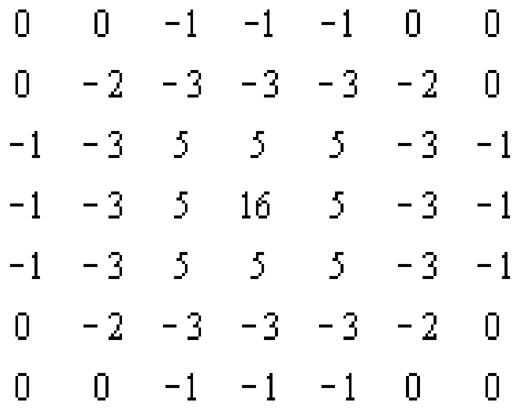
Difference Of Gaussian (DoG)

- LoG requires large computation time for a large edge detector mask.
- To reduce computational requirements, approximate the LoG by the difference of two LoG – the DoG.

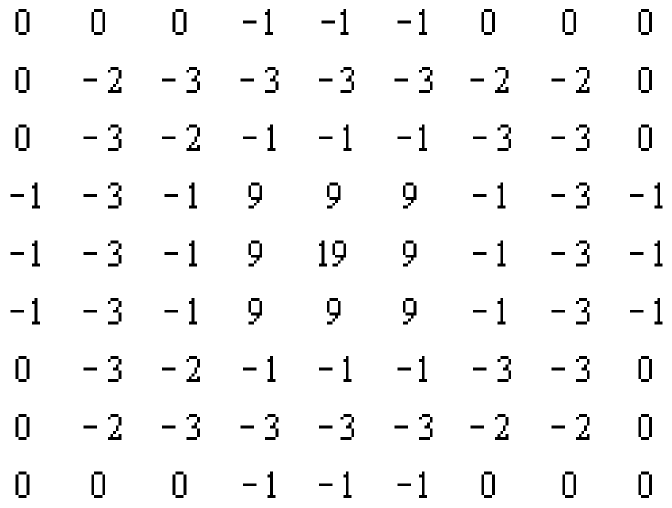
$$DoG(x,y)=\frac{e^{-(\frac{x2+y2}{2\pi\sigma_1{}^2})}}{2\pi\sigma_1{}^2}-\frac{e^{-(\frac{x2+y2}{2\pi\sigma_2{}^2})}}{2\pi\sigma_2{}^2}$$

Difference Of Gaussian (DoG)

- Advantage of DoG:
  - Close approximation of LoG.
  - Less computation effort.
  - Width of edge can be adjusted by changing $\sigma 1$ and $\sigma 2$.



0	0	-1	-1	-1	0	0
0	-2	-3	-3	-3	-2	0
-1	-3	5	5	5	-3	-1
-1	-3	5	16	5	-3	-1
-1	-3	5	5	5	-3	-1
0	-2	-3	-3	-3	-2	0
0	0	-1	-1	-1	0	0



0	0	0	-1	-1	-1	0	0	0
0	-2	-3	-3	-3	-3	-2	-2	0
0	-3	-2	-1	-1	-1	-3	-3	0
-1	-3	-1	9	9	9	-1	-3	-1
-1	-3	-1	9	19	9	-1	-3	-1
-1	-3	-1	9	9	9	-1	-3	-1
0	-3	-2	-1	-1	-1	-3	-3	0
0	-2	-3	-3	-3	-3	-2	-2	0
0	0	0	-1	-1	-1	0	0	0

Optimal Edge Operator

- Optimal edge detector depending on:
  - Low error rate – edges should not be missed and there must not be spurious responses.
  - Localization – distance between points marked by the detector and the actual center of the edge should be minimum.
  - Response – Only one response to a single edge.
- One dimensional formulation:
  - Assume that 2D images have constant cross section in some direction.

Canny Edge Operator

- Step 1
  - Noise is filtered out – usually a Gaussian filter is used.
  - Width is chosen carefully.
- Step 2
  - Edge strength is found out by taking the gradient of the image.
  - A Roberts mask or a Sobel mask can be used.

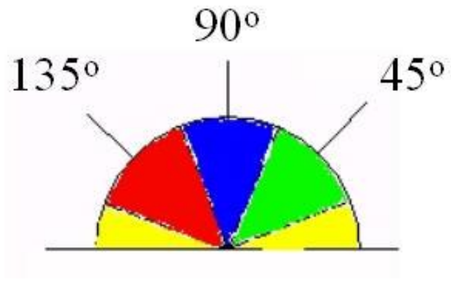
$$|G| = \sqrt{Gx^2 + Gy^2} \approx |Gx| + |Gy|$$

Canny Edge Operator

- Step 3
  - Find the edge direction:

$$\theta = \tan^{-1}\left(\frac{Gy}{Gx}\right)$$

- Step 4
  - Resolve edge direction:

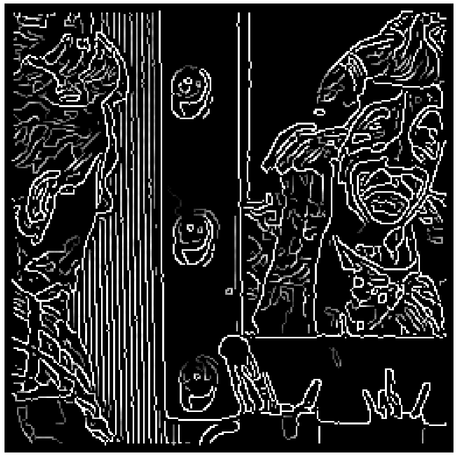


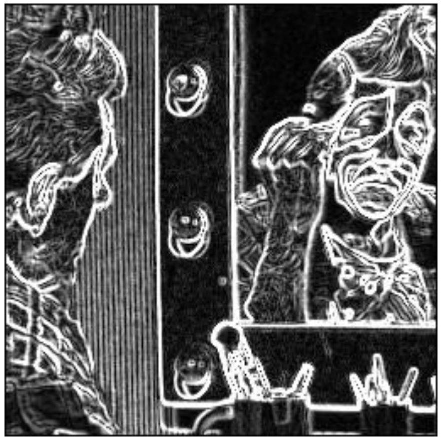
Canny Edge Operator

- Step 5
  - Non-maxima suppression: trace along the edge direction and suppress any pixel value not considered to be an edge. Gives a thin line for edge.
- Step 6
  - Use double / hysterisis thresholding to eliminate streaking.

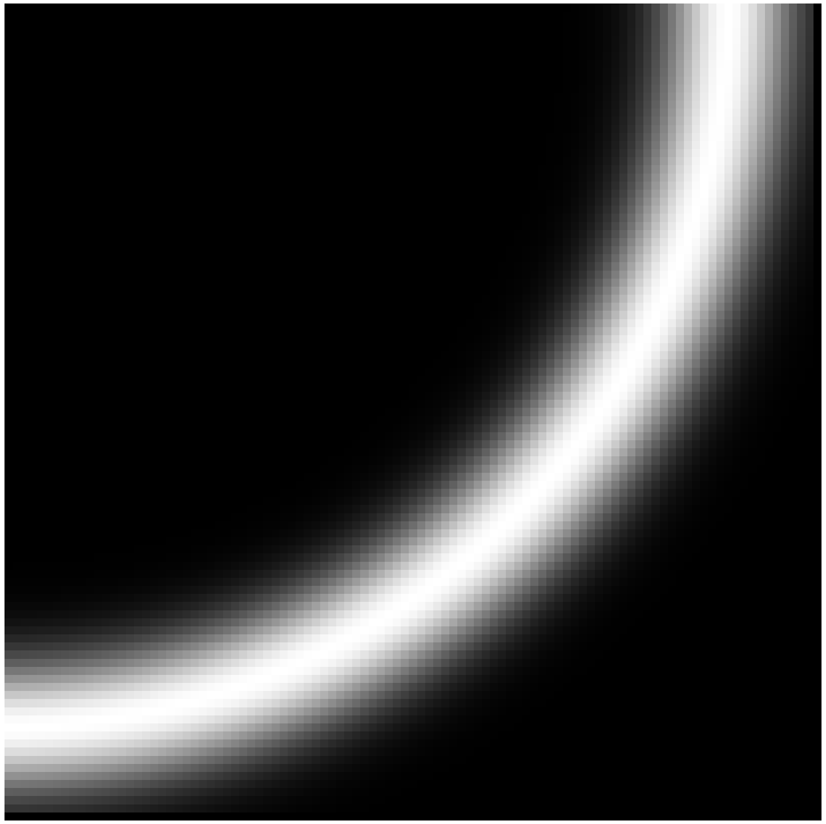
Comparison

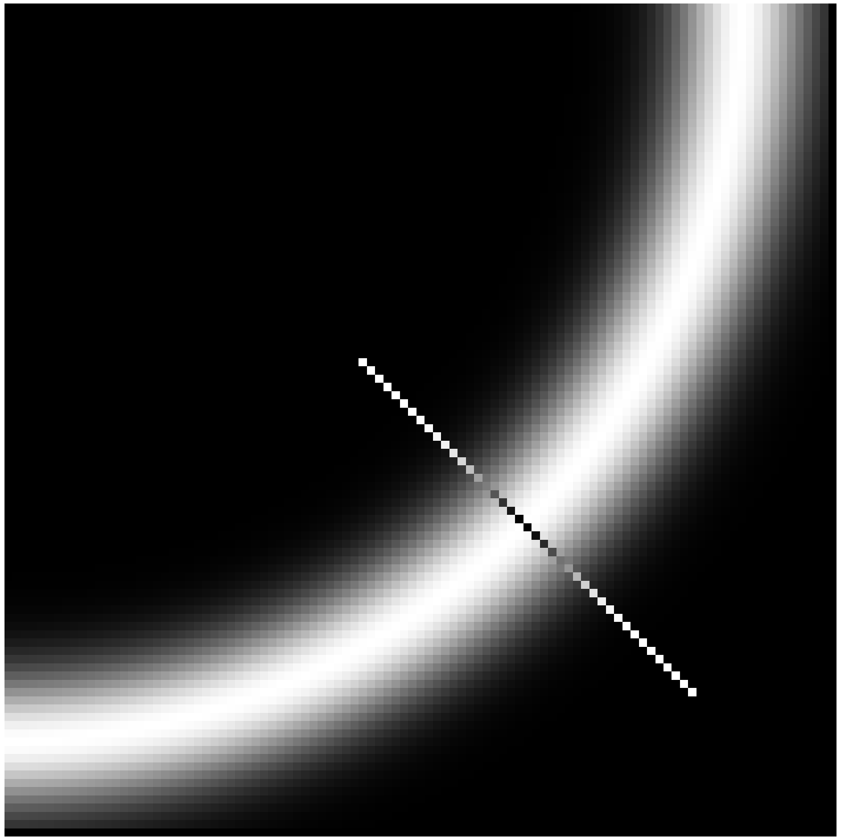
- Compare the results of Sobel and Canny.


Canny


Sobel

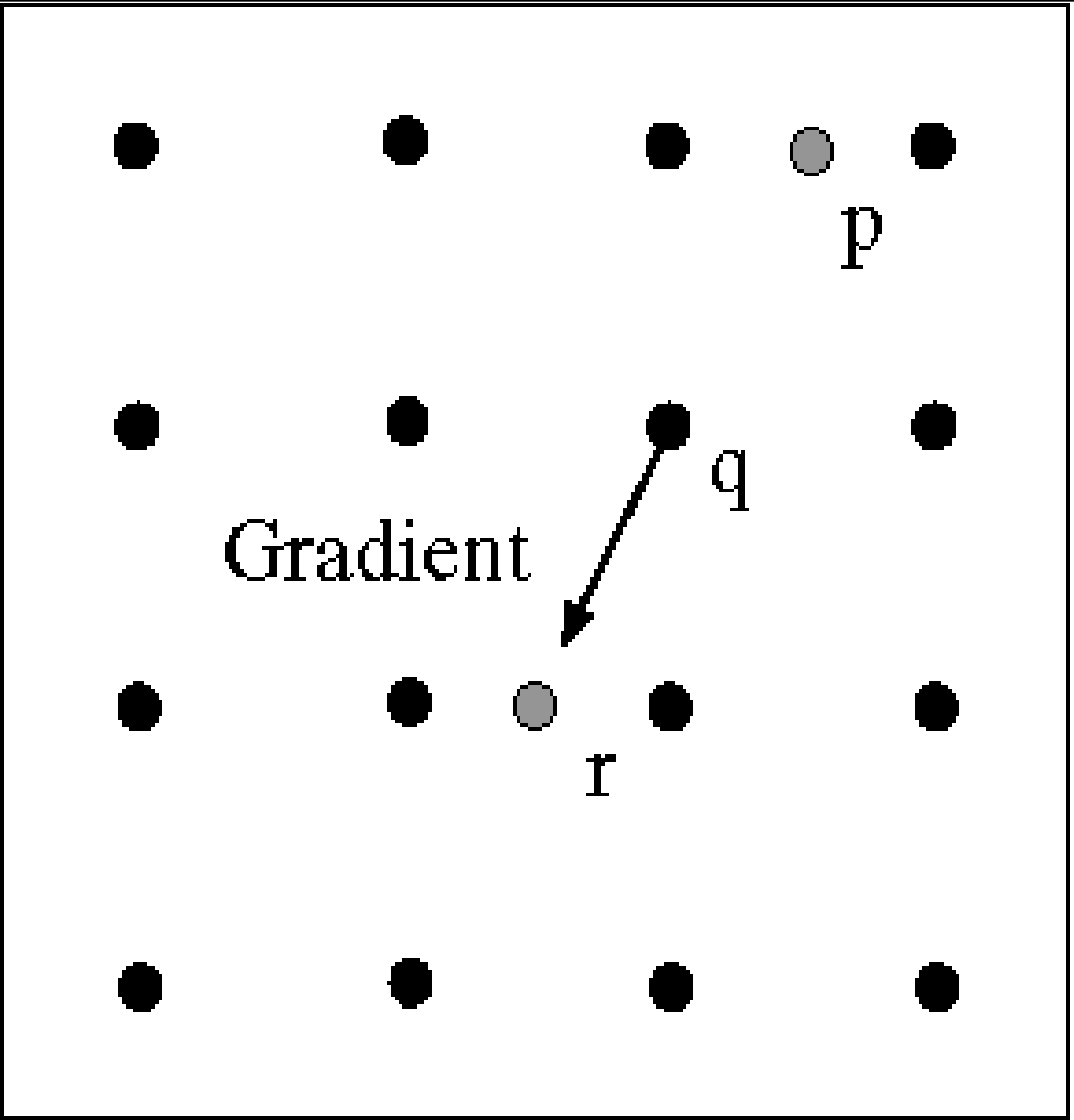
Canny Edge Detection





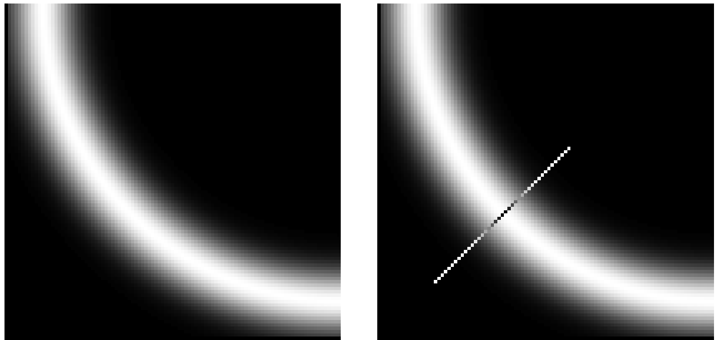
- Non-maximum suppression:
  - Select the single maximum point across the width of an edge.

Canny Edge Detection

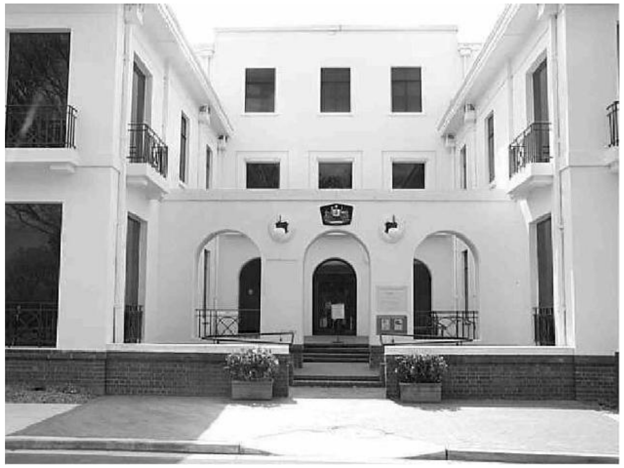


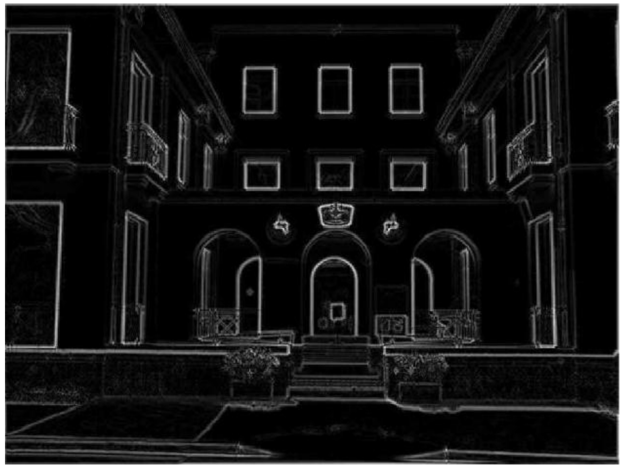
Non-maximum suppression:

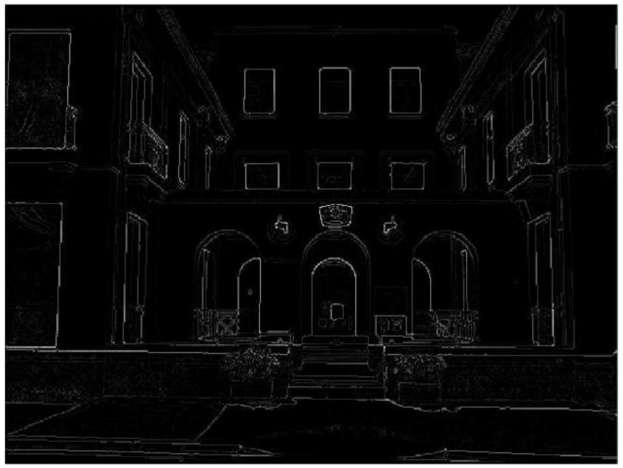
At *q*, the value must be larger than values interpolated at *p* or *r*.

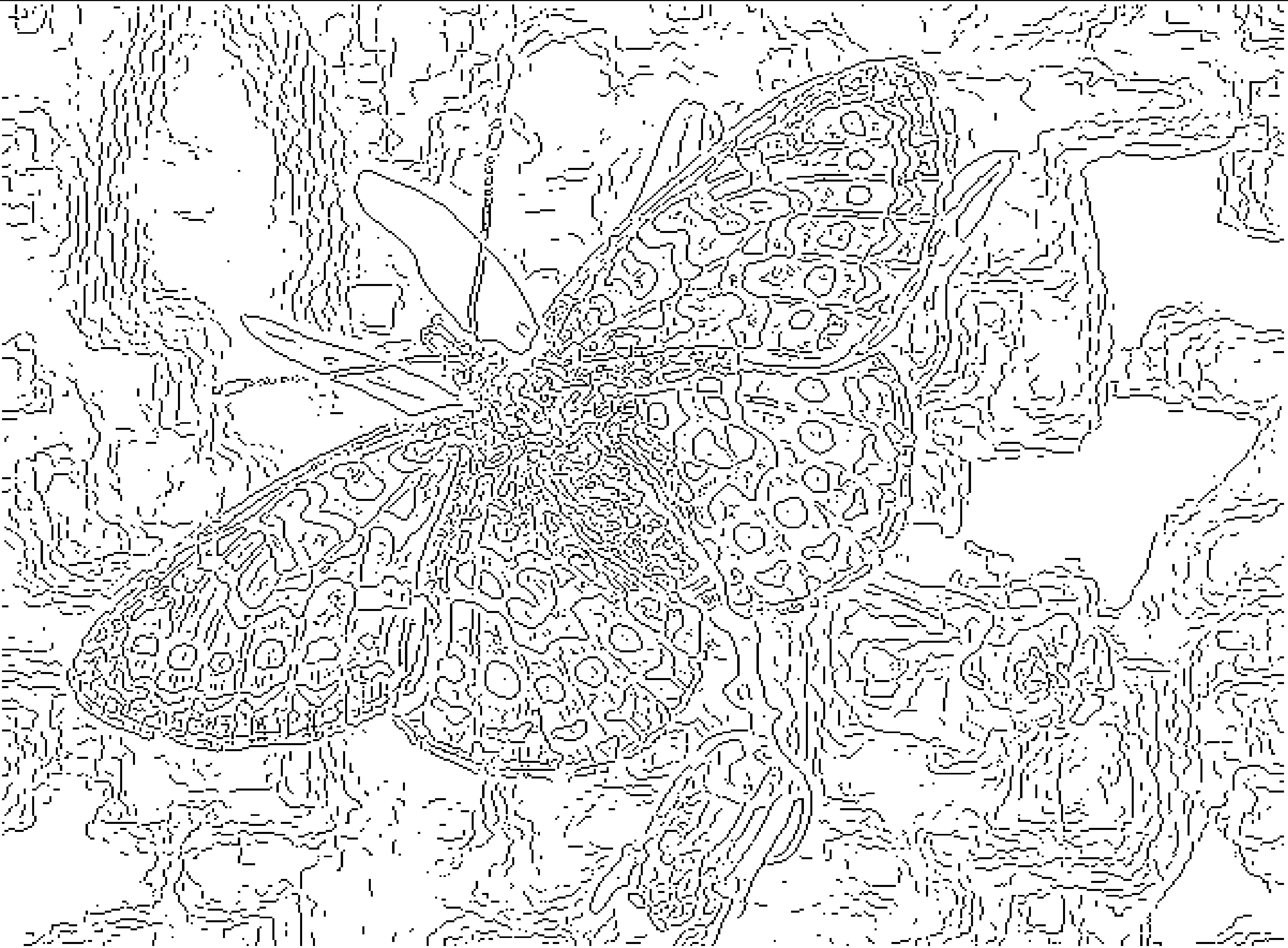


Non-maximum Suppression: Examples

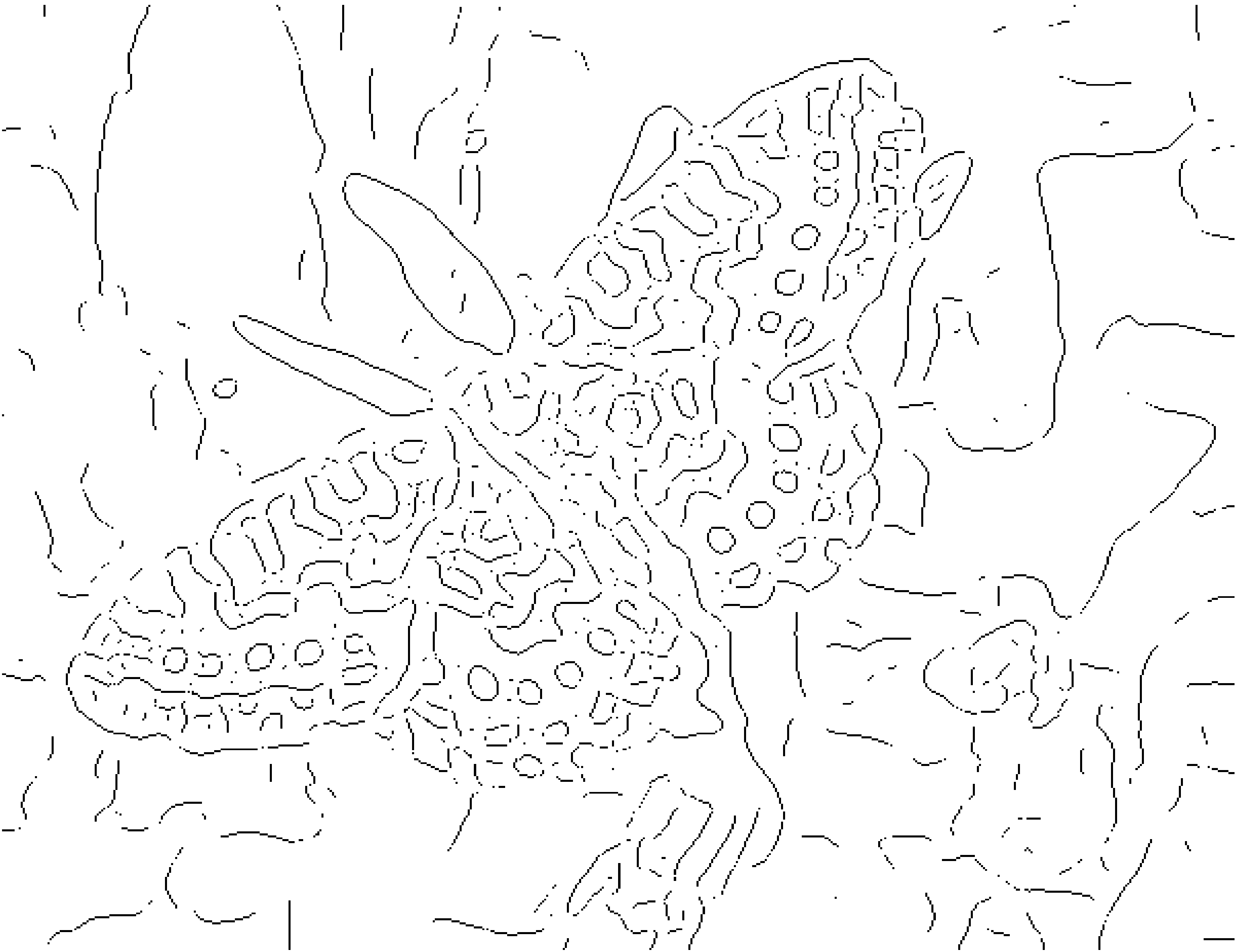

Original image


Gradient magnitude


Non-maxima suppressed



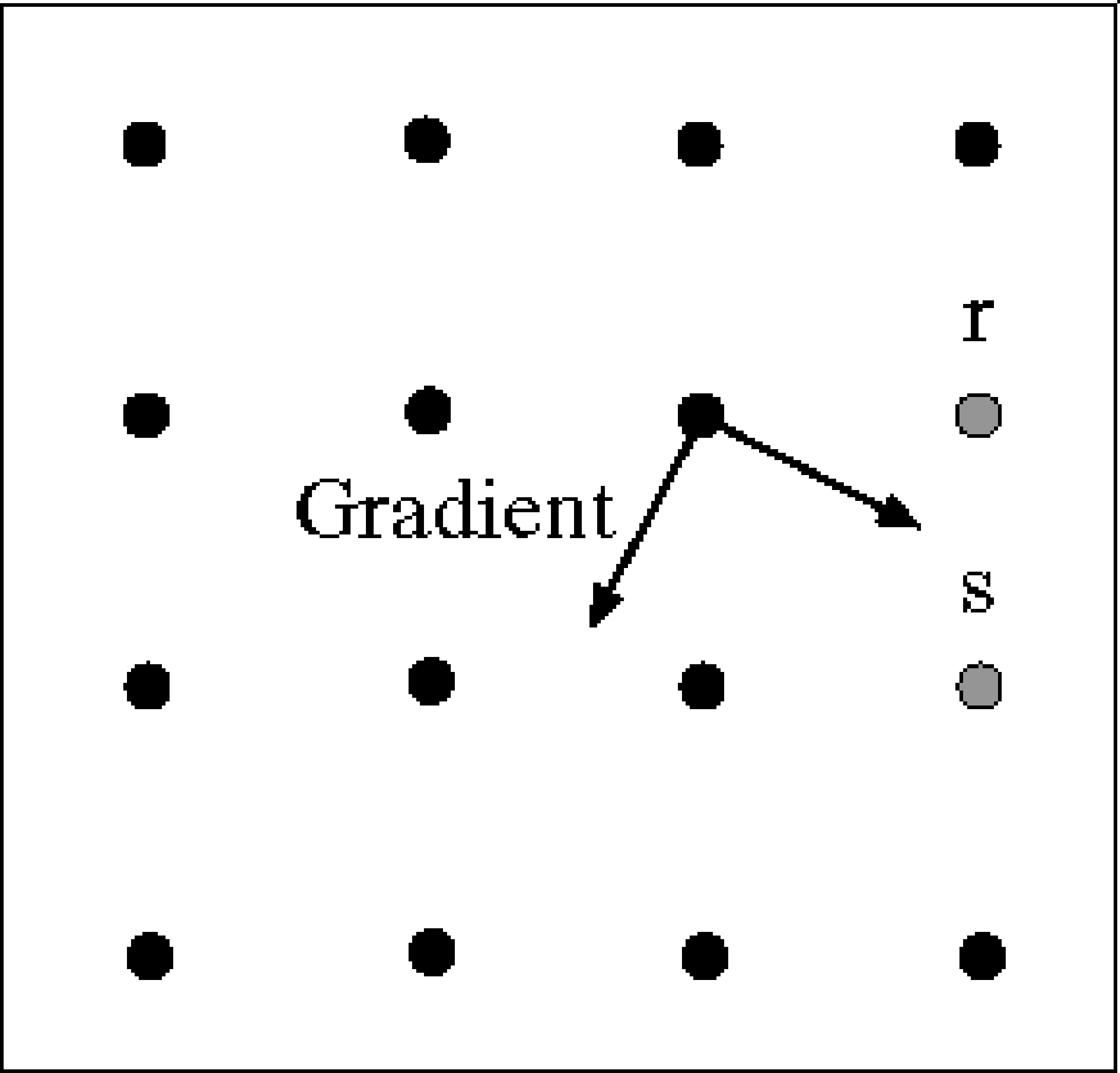
Fine scale;
High threshold



Coarse scale;

High threshold

Canny Edge Detection



Linking to the next edge point:

Assume the marked point is an edge point.
Take the normal to the gradient at that point and use this to predict continuation points (either *r* or *s*).

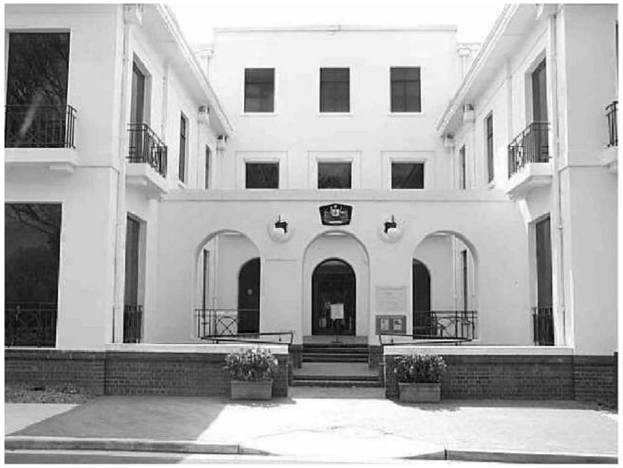
Edge Hysteresis

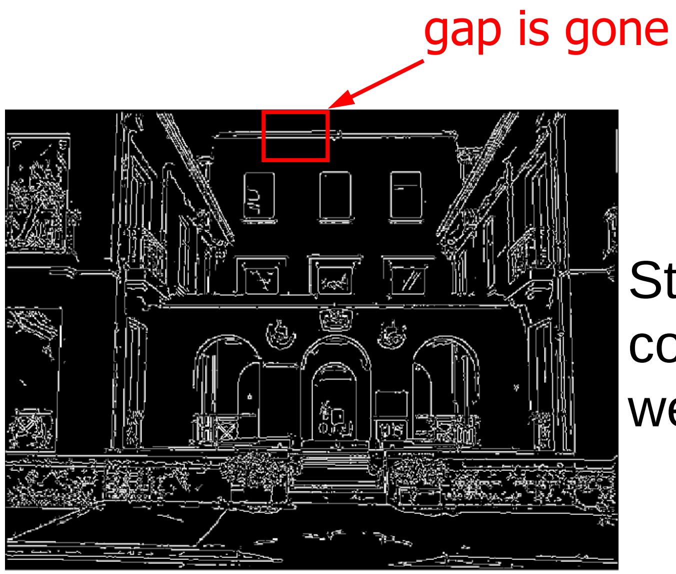
- Hysteresis: A lag or momentum factor.
- Maintain two thresholds k_{high} and k_{low}
  - Use k_{high} to find strong edges to start edge chain.
  - Use k_{low} to find weak edges which continue edge chain.
- Typical ratio of thresholds is roughly:

$$k_{high} / k_{low} = 2$$

Canny Edge Detection: Example

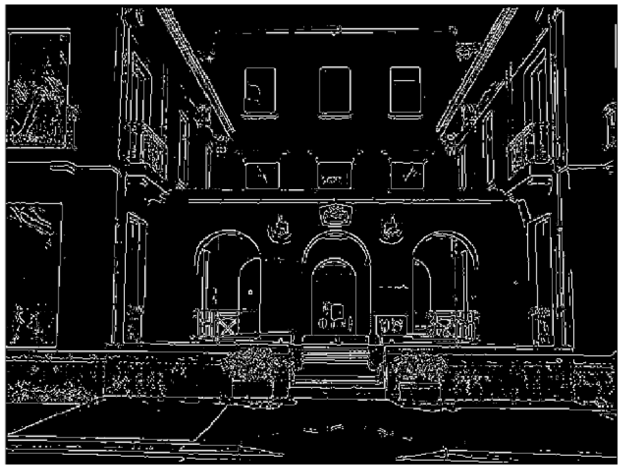
Original image

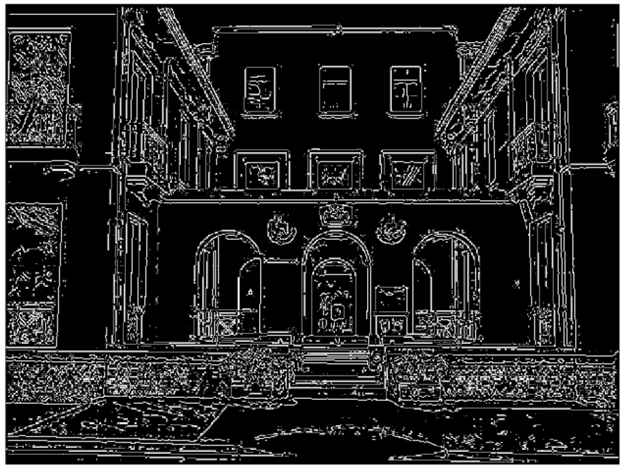




Strong + connected weak edges

Strong edges only





Weak edges

Canny Edge Detection: Example

Edge map of remote sensed image using Canny

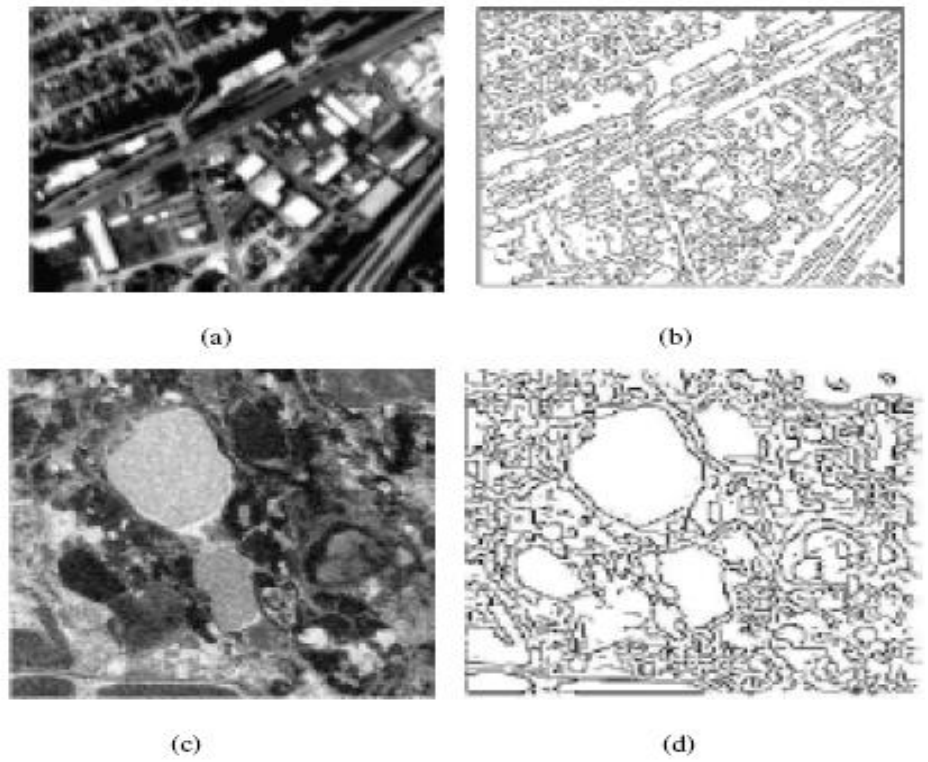


Fig. 3. (a) Original Arial Urban remote sensing image (b) Edge map using Canny (c) Original remote sensing image (d) Edge map using Canny

Histogram Thresholding

- Histogram are constructed by splitting the range of the data into equal-sized bins (called classes).
- For each bin, the number of points from the data set that fall into each bin are counted.
- Vertical axis: Frequency (i.e., counts for each bin).
- Horizontal axis: Response variable.
  - In image histograms, the pixel intensities form the horizontal axis.
  - In Matlab, histograms for images can be constructed using the imhist command.

Histogram Thresholding

- Suppose that the gray-level histogram corresponds to an image, f(x,y):
  - composed of dark objects in a light background.
  - object and background pixels have gray levels grouped into two dominant modes.
- To extract objects from background is to select a threshold T:
  - Any point (x,y) for which f(x,y) > T is called an object point.
  - otherwise, the point is called a background point.

Histogram Thresholding

- If two dominant modes characterize the image histogram, it is called a bimodal histogram.
  - Only one threshold is enough for partitioning the image.

Histogram Thresholding

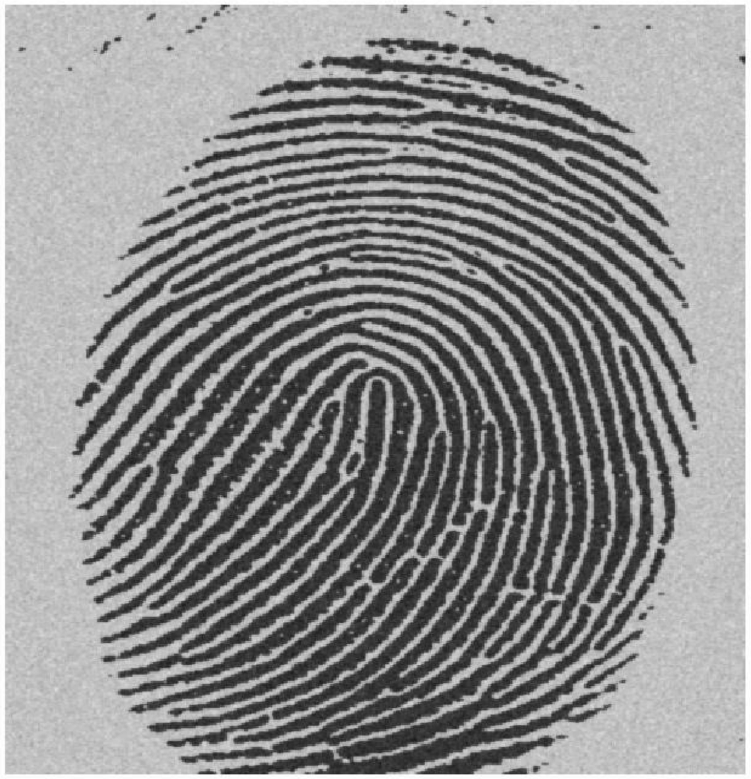


Image of a Finger Print with light background

Bimodal Histogram

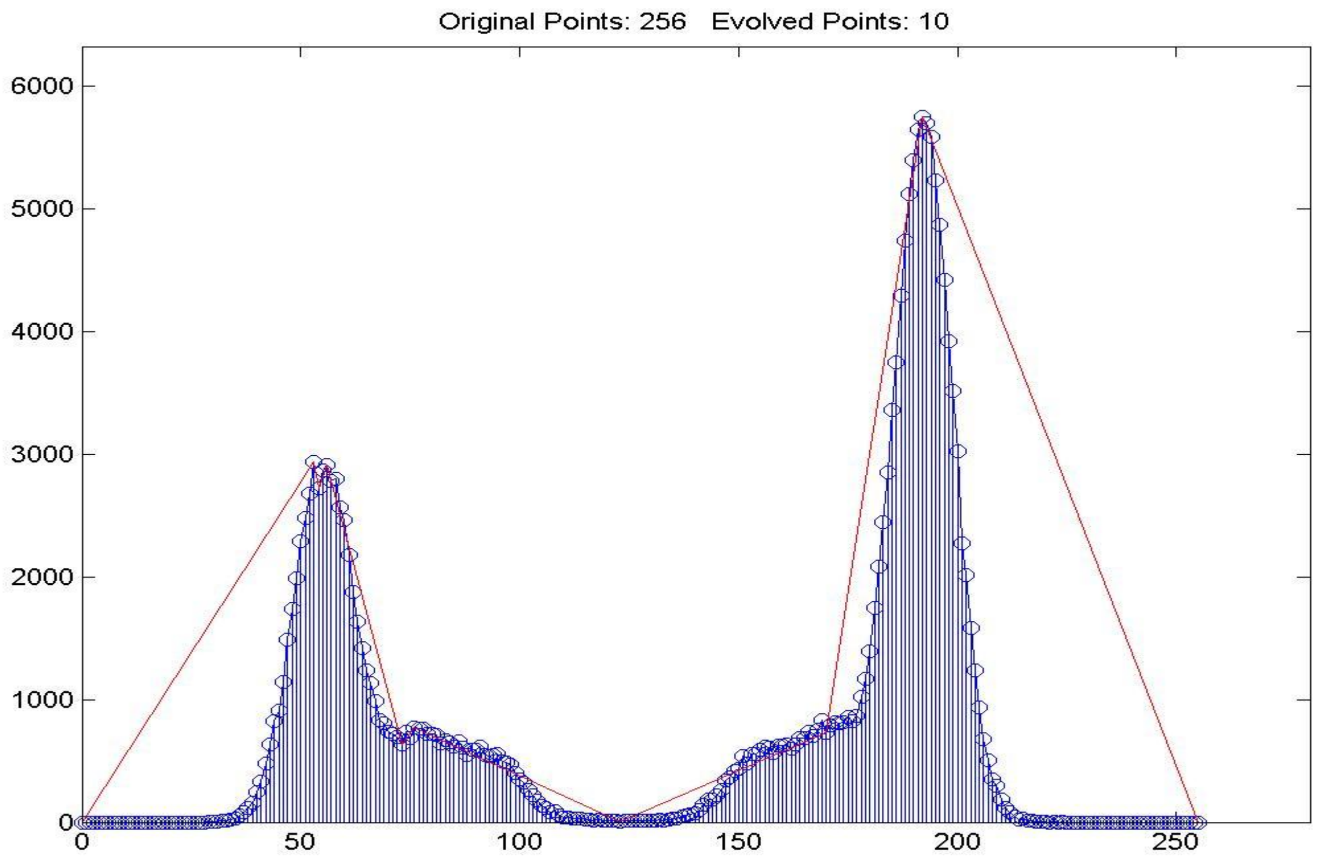
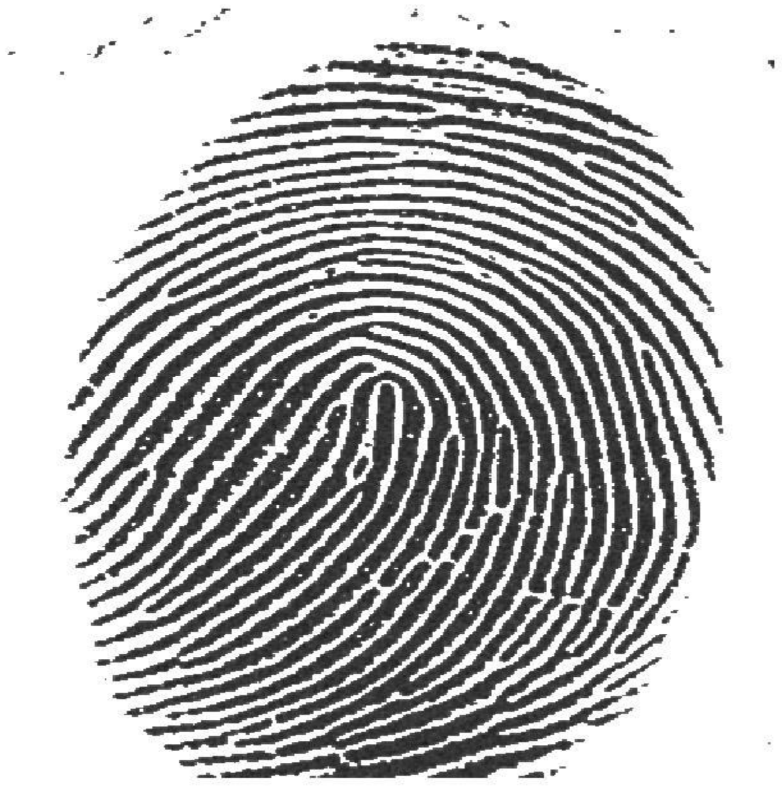


Image histogram

Histogram Thresholding



Segmented image

Histogram Thresholding

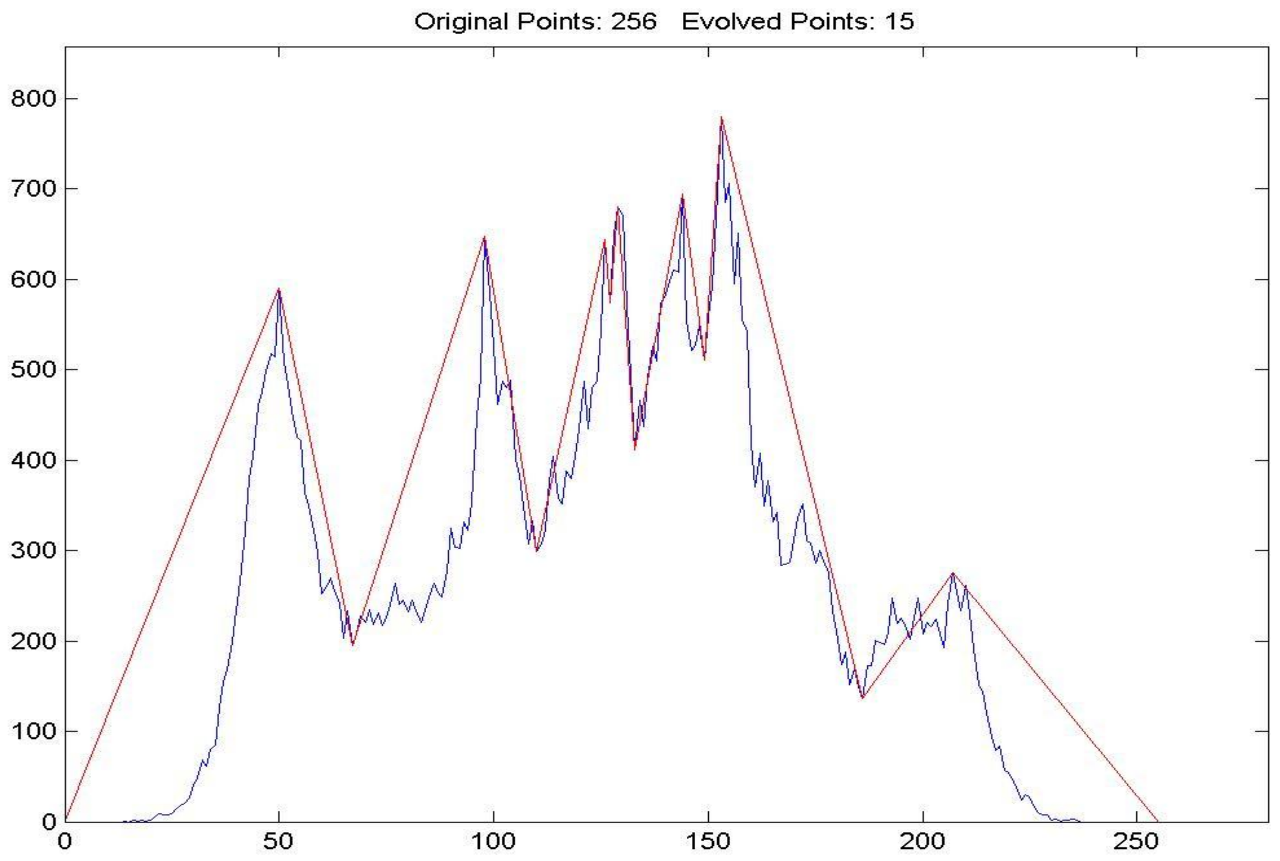
- If an image is composed of two types of light objects on a dark background:
  - three or more dominant modes characterize the image histogram.
  - histogram has to be partitioned by multiple thresholds classifying a point (*x*,*y*) as belonging to:

one object class:	if $T1 < (x,y) <= T2$,
to the other object class	if $f(x,y) > T2$,
and to the background	if $f(x,y) <= T1$

Histogram Thresholding

Original image of Lena

Multimodal Histogram



Histogram of Lena

Histogram Thresholding

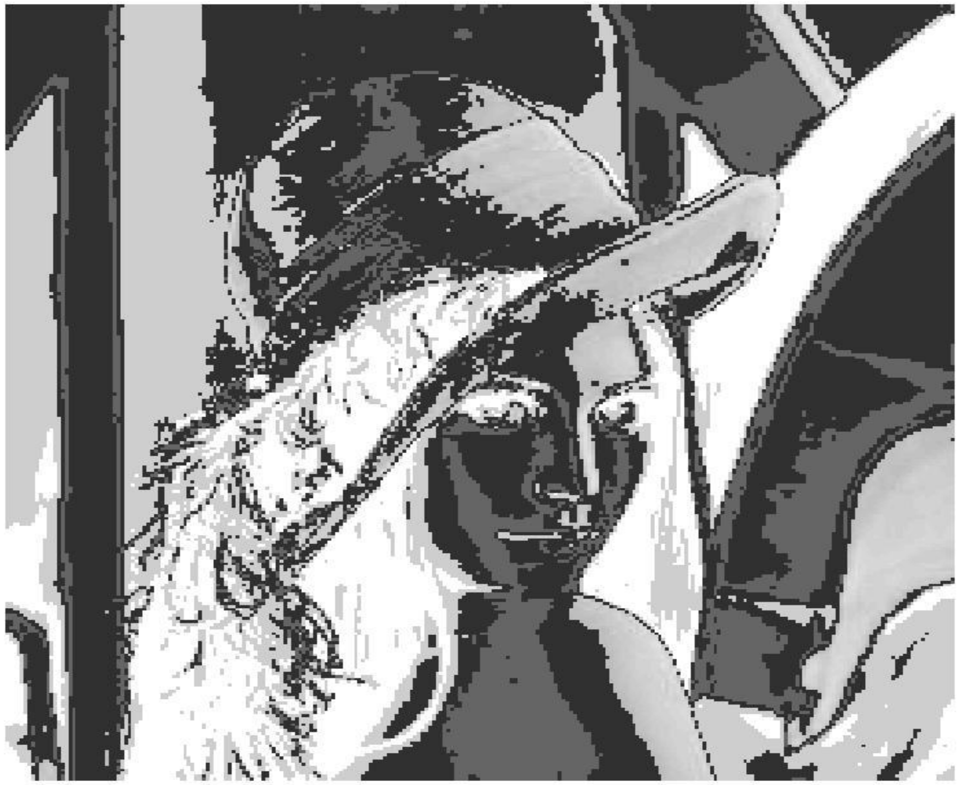


Image after segmentation – we get an outline of her face, hat, shadow etc

Histogram Thresholding

- Basic Global Thresholding:
  1. Select an initial estimate for *T* and segment image using *T*.
  2. This will produce two groups of pixels:
     - G1 consisting of all pixels with gray level values >*T*
     - G2 consisting of pixels with values <=*T*
  3. Compute the average gray level values *mean1* and *mean2* for the pixels in regions G1 and G2.
  4. Compute a new threshold value

     $$T = (1/2)(mean1 + mean2)$$
  5. Repeat steps 2 through 4 until difference in *T* in successive iterations is smaller than a predefined parameter *T0*.

Histogram Thresholding

- Basic Adaptive Thresholding:
  - Images having uneven illumination makes it difficult to segment using histogram,
  - this approach is to divide the original image into sub images and use the above said thresholding process to each of the sub images.

THE END