Biomedical Imaging & Analysis

Lecture 6, Part 4. Fall 2014

Basic Image Processing / Filtering (IV) – Feature Extraction I

*[**Text**: Ch: 10, Gonzalez and Woods, Digital Image Processing (3rd Edition)]*

Prahlad G Menon, PhD

Assistant Professor

Sun Yat-sen University – Carnegie Mellon University (SYSU-CMU)

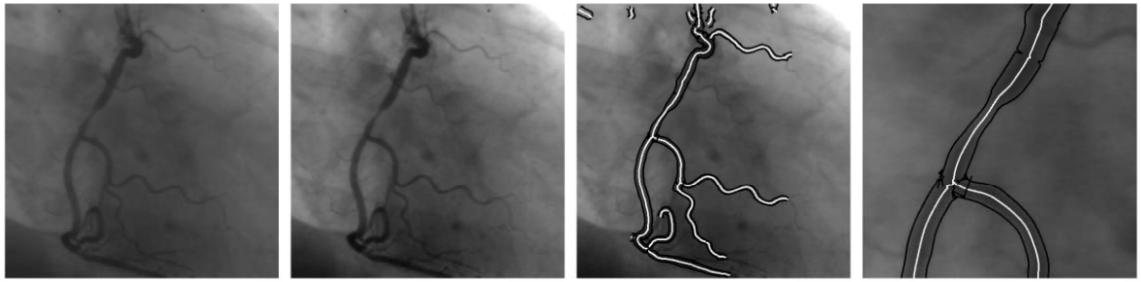
Joint Institute of Engineering

Image **Features**

- Local features
  - local regions with special properties, including points, edges, corners, lines, curves, regions with special properties, etc.
- Global features
  - global properties of an image, including intensity histogram, frequency domain descriptors, covariance matrix and high order statistics, etc.
  - Obtained from Principal Component Analysis (PCA), Laplace or Fourier Decomposition etc..
- Depending on applications, various features are useful.

Steger's Line/Curve Detection Algorithm

C. Steger, M. Ulrich, C. Wiedemann, Machine Vision Algorithms and Applications, Wiley-VCH, 2008 . http://ias.in.tum.de/people/steger



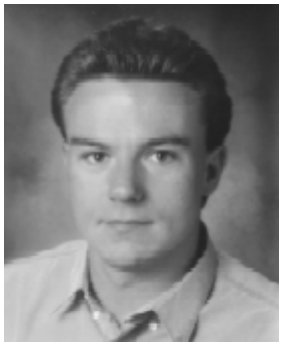
(a) (b) (c) (d)

Fig. 17. Lines detected in the coronary angiogram (a). Since this image has very low contrast, the results (c) extracted from (a) are superimposed onto a version of the image with better contrast (b). A three times enlarged detail of (c) is displayed in (d).

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 20, NO. 2, FEBRUARY 1998 113

An Unbiased Detector of Curvilinear Structures

Carsten Steger



Abstract—The extraction of curvilinear structures is an important low-level operation in computer vision that has many applications. Most existing operators use a simple model for the line that is to be extracted, i.e., they do not take into account the surroundings of a line. This leads to the undesired consequence that the line will be extracted in the wrong position whenever a line with different lateral contrast is extracted. In contrast, the algorithm proposed in this paper uses an explicit model for lines and their surroundings. By analyzing the scale-space behavior of a model line profile, it is shown how the bias that is induced by asymmetrical lines can be removed. Furthermore, the algorithm not only returns the precise subpixel line position, but also the width of the line for each line point, also with subpixel accuracy.

Index Terms—Feature extraction, curvilinear structures, lines, scale-space, contour linking, low-level processing, aerial images, medical images.

Steger's Line/Curve Detection Algorithm

Local Curvature of a 1D Function

- First derivative

$$f'(x_0) = 0$$

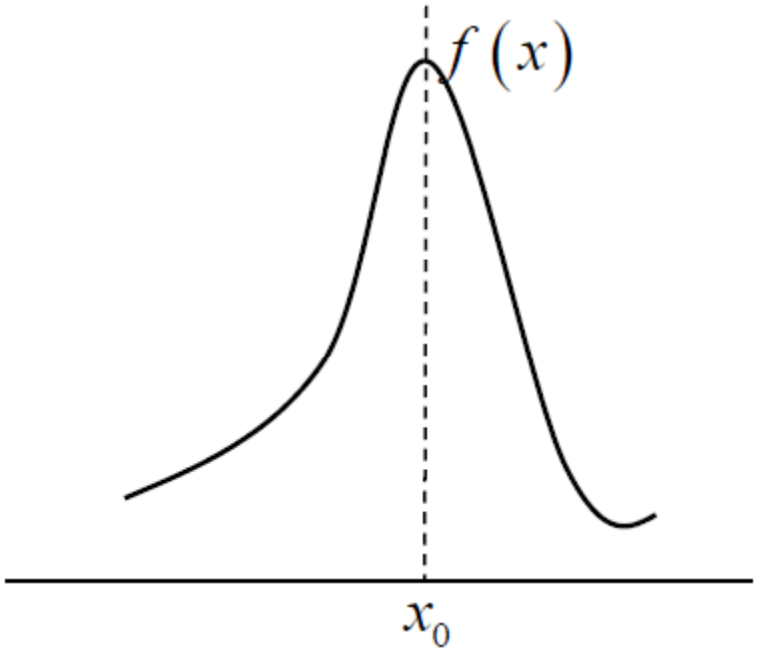
- Second derivative

$f''(x_0) > 0$ x_0 is a local minimum

$f''(x_0) < 0$ x_0 is a local maximum

- Curvature

$$K = \frac{|f''(x)|}{\left(1 + f'(x)^2\right)^{\frac{3}{2}}}$$



Steger's Line/Curve Detection Algorithm

Identify the center of the line by searching for the maximum of the second order directional derivative.

Directional Derivatives

- The gradient vector of function f defines the maximum rate of change and direction of change.

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- The first directional derivative of a function

$$f'_\beta = \lim_{h \to 0} \frac{f(x + h\cos\beta, y + h\sin\beta) - f(x, y)}{h}$$

$$= \frac{\partial f}{\partial x}\cos\beta + \frac{\partial f}{\partial y}\sin\beta$$

$$= \begin{bmatrix} \cos\beta & \sin\beta \end{bmatrix} \nabla f$$

Steger's Line/Curve Detection Algorithm

Identify the center of the line by searching for the maximum of the second order directional derivative.

Directional Derivatives

- The Hessian matrix

$$Hf = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 y}{\partial y^2} \end{bmatrix}$$

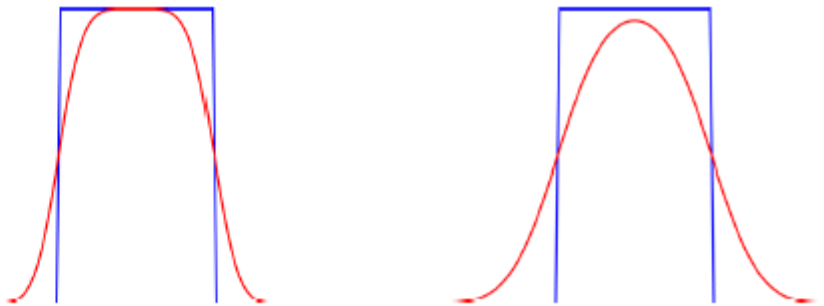
- The second directional derivative of a function

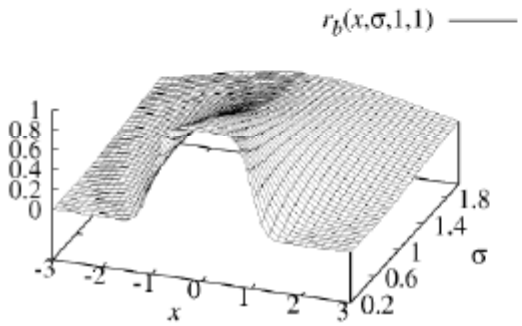
$$\begin{aligned} f''_\beta &= [\cos\beta \quad \sin\beta] H \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} \\ &= [\cos\beta \quad \sin\beta][v_1 \quad v_2] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \end{bmatrix} \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} \\ &= [\cos\gamma \quad \sin\gamma] \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \cos\gamma \\ \sin\gamma \end{bmatrix} \\ &= \lambda_1 \cos^2\gamma + \lambda_2 \sin^2\gamma \end{aligned}$$

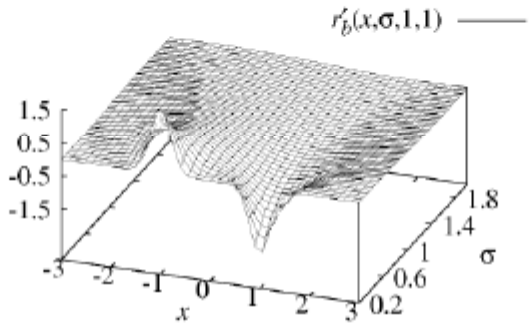
Steger's Line/Curve Detection (Step 0)

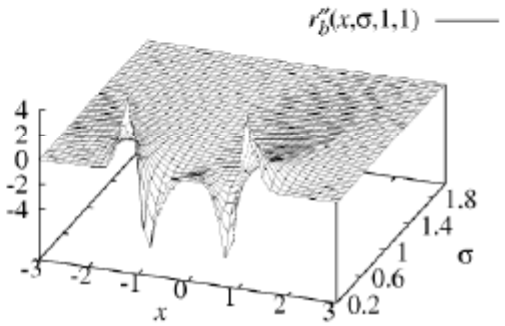
- Step 0: Identify maximal line width. Convolve image with a Gaussian kernel whose size satisfies:

$$\sigma \geq \frac{w}{\sqrt{3}}$$









Steger's Line/Curve Detection (Step 1-2)

- Step 1: For each pixel, calculate the Hessian matrix. Then determine the local direction of intensity search from the eigenvector corresponding to the eigenvalue of maximum magnitude.

$$\begin{pmatrix} n_x & n_y \end{pmatrix}$$

$\begin{bmatrix} n_x & n_y \end{bmatrix}$

- Direction perpendicular to the line;
- Eigenvector corresponding to the eigenvalue of maximal absolute magnitude

- Step 2: Calculate the first and second directional derivative

$$r' = \begin{bmatrix} n_x & n_y \end{bmatrix} \begin{bmatrix} f_x \\ f_y \end{bmatrix} = f_x n_x + f_y n_y$$

$$r'' = \begin{bmatrix} n_x & n_y \end{bmatrix} H \begin{bmatrix} n_x \\ n_y \end{bmatrix} = f_{xx} n_x^2 + 2 f_{xy} n_x n_y + f_{yy} n_y^2$$

Steger's Line/Curve Detection (Step 3)

- Step 3: Determine location of the local intensity maximum

$$p(x) = r + r'x + \frac{1}{2}r''x^2$$

$$x^* = -\frac{r'}{r''}$$

$$x^* = -\frac{r'}{r''} = -\frac{f_x n_x + f_y n_y}{f_{xx} n_x^2 + 2 f_{xy} n_x n_y + f_{yy} n_y^2}$$

Steger's Line/Curve Detection (Step 4-5)

- Step 4: Test whether it is a center line point

$$\begin{cases} \text{if } x^* \in \left[-\frac{1}{2} \quad \frac{1}{2}\right], \text{ the pixel is on the center line} \\ \text{otherwise} \end{cases}$$

$$\text{Declare a center line point if } x^* \in \left[-\frac{1}{2} \quad \frac{1}{2}\right]$$

- Step 5: Link individual center line pixels into a line/curve

Hough Transform

Great tutorial (Ch 10: Gonzalez & Woods, Digital Image Processing):
https://www.youtube.com/watch?v=kMK8DjdGtZo

- **Elegant method for direct object recognition**

  - STARTS with (Canny) Edge detection!
  - Detects line features.
  - Edges need not be connected

  - Complete object need not be visible

  - Key Idea: Edges “vote” for the possible model (known parametric shape / geometry)

    - Lines, Circles, Parabolas, Ellipses, etc..
    - More parameters, more computationally expensive!
    - Eg: http://docs.opencv.org/modules/imgproc/doc/feature_detection.html (OpenCV is a possible library to use for a project on “Feature Detection with OpenCV / Python” applied to biomedical images.)

Image and Parameter Spaces

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

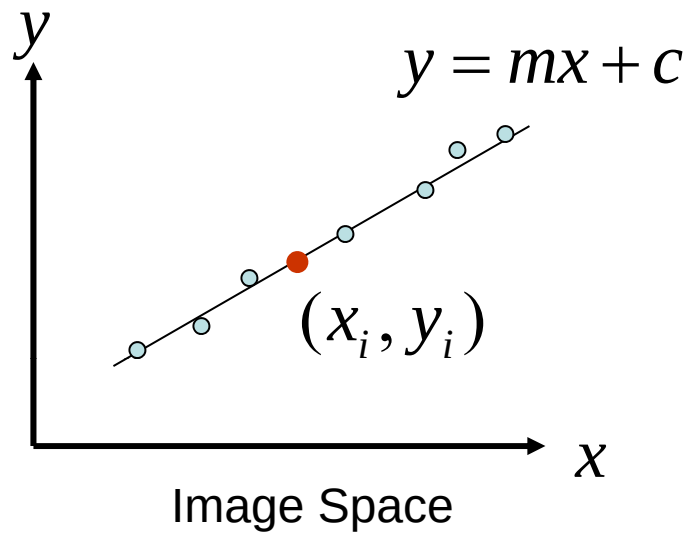
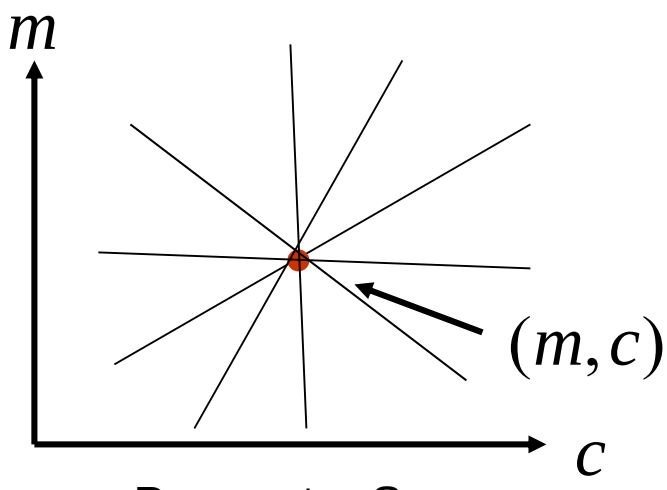


Image Space



Parameter Space

Line Detection by Hough Transform

Algorithm:

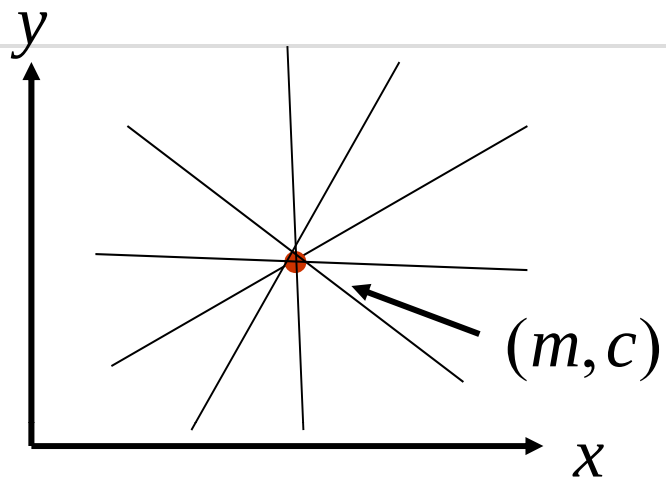
- Quantize Parameter Space (m, c)
- Create Accumulator Array $A(m, c)$
- Set $A(m, c) = 0 \quad \forall m, c$
- For each image edge (x_i, y_i) increment:

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

- 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: $-\infty \le m \le \infty$

Large Accumulator

More memory and computations

Improvement: (Finite Accumulator Array Size)

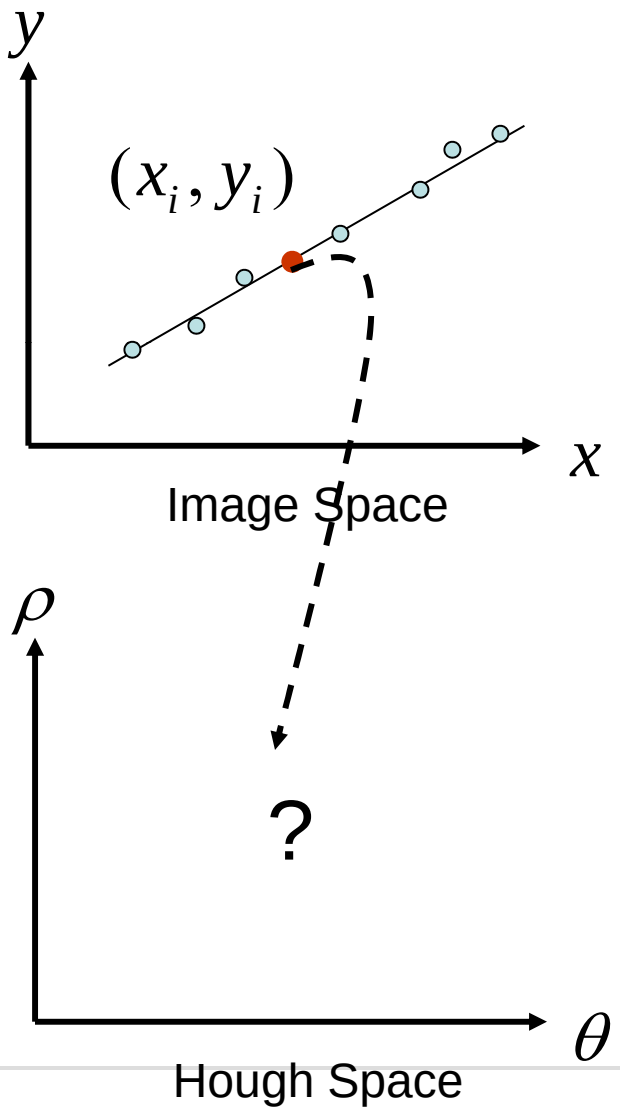
Line equation: $\rho = -x\cos\theta + y\sin\theta$

Here $0 \le \theta \le 2\pi$

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

Given points (x_i, y_i) find (ρ, θ)

Hough Space Sinusoid



Better Parameterization

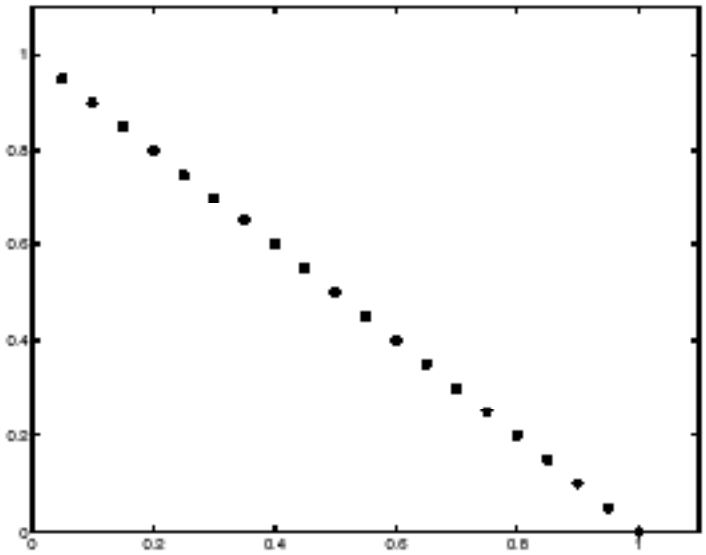
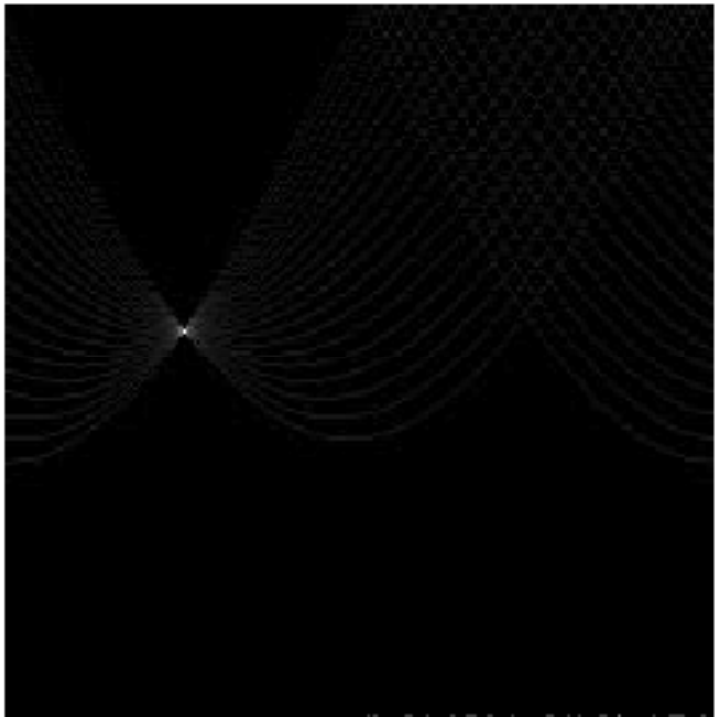


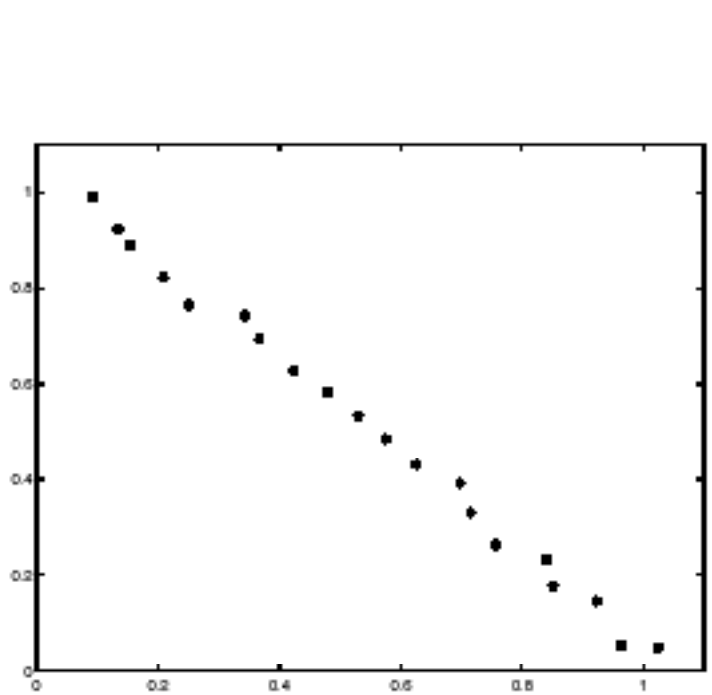
Image space



Votes

Horizontal axis is θ,
vertical is rho.

Better Parameterization



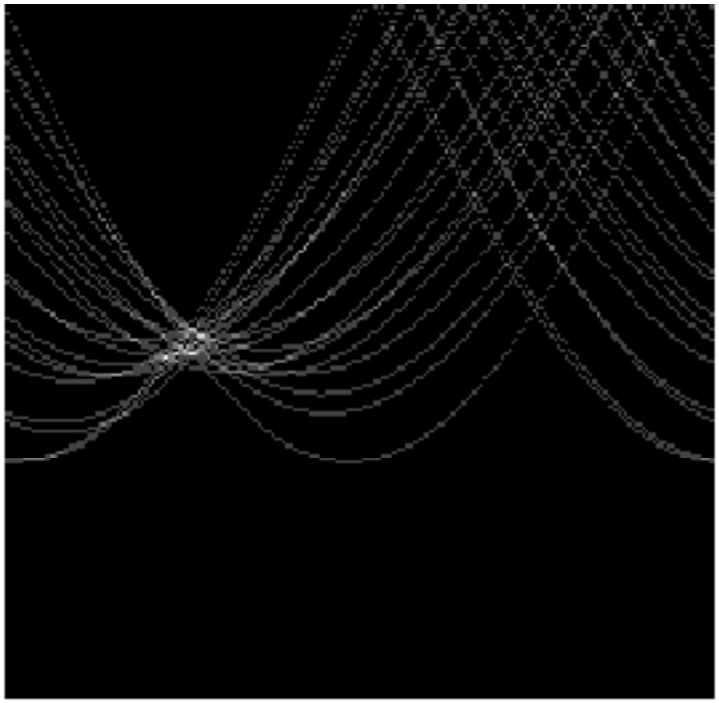
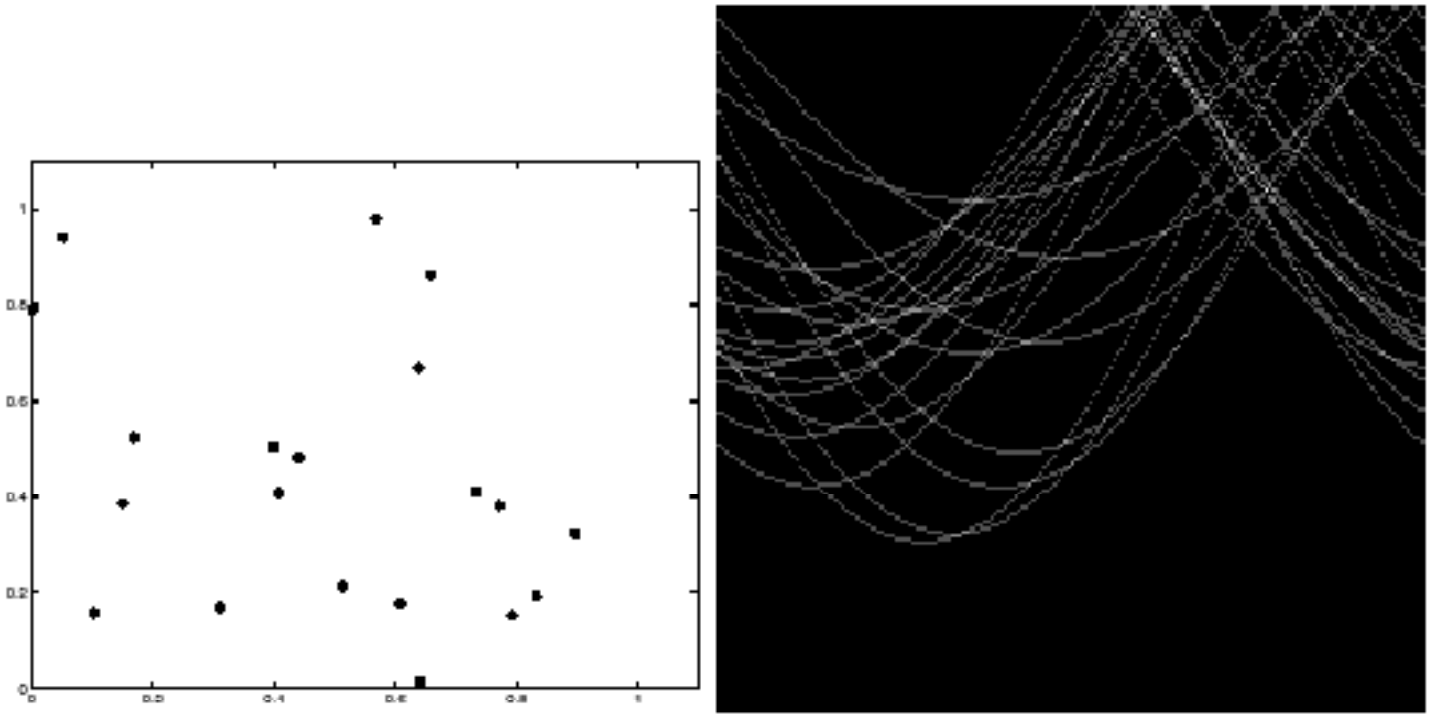


Image space

votes

Better Parameterization



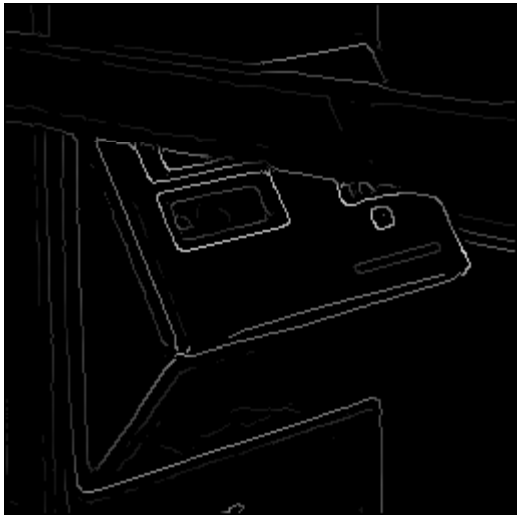
Lots of noise can lead to large peaks in the array.

Mechanics of the Hough transform

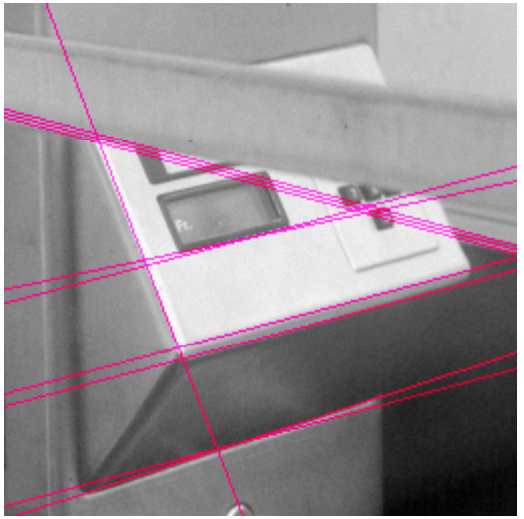
- Difficulties
  - how big should the cells be? (too big, and we merge quite different lines; too small, and noise causes lines to be missed)
  - Fewer votes land in a single bin when noise increases.
  - Adding more clutter increases number of bins with false peaks.
- How many lines?
  - Count the peaks in the Hough array
  - Treat adjacent peaks as a single peak
- Which points belong to each line?
  - Search for points close to the line
  - Solve again for line and iterate

Real World Example

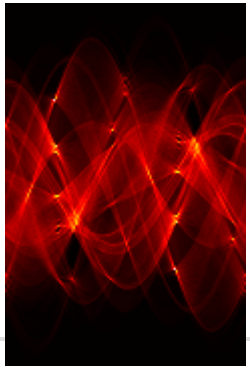
Original



Edge Detection



Found Lines



Parameter Space

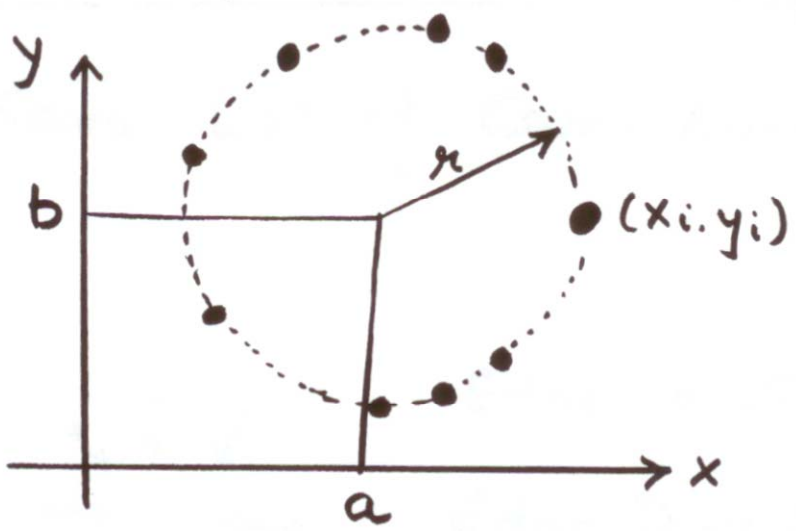
Finding Circles by Hough Transform

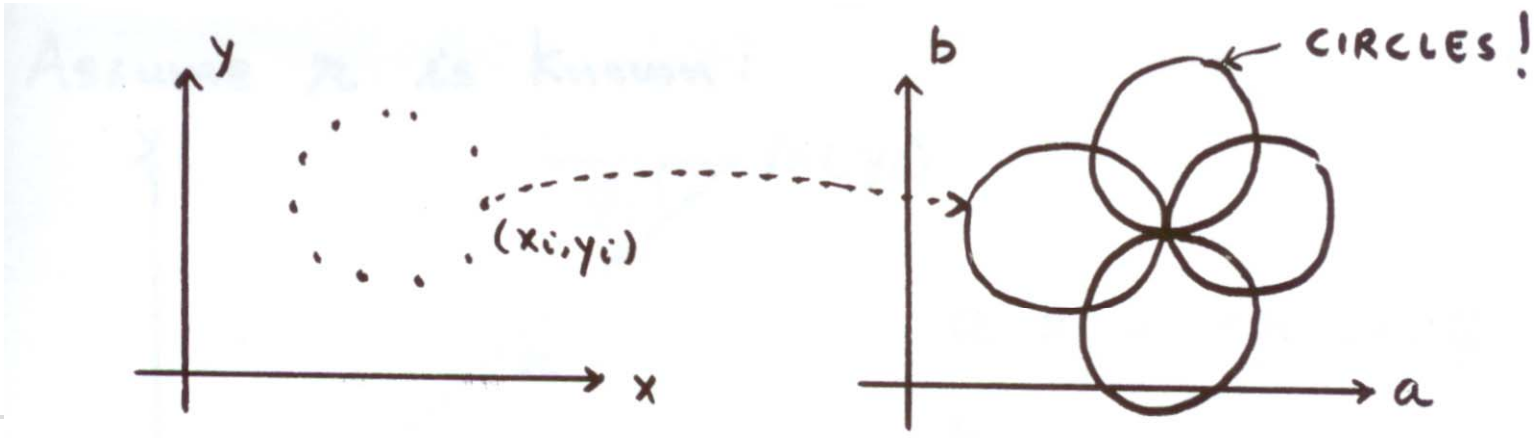
Equation of Circle:

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

If radius is known: (2D Hough Space)

Accumulator Array $A(a,b)$

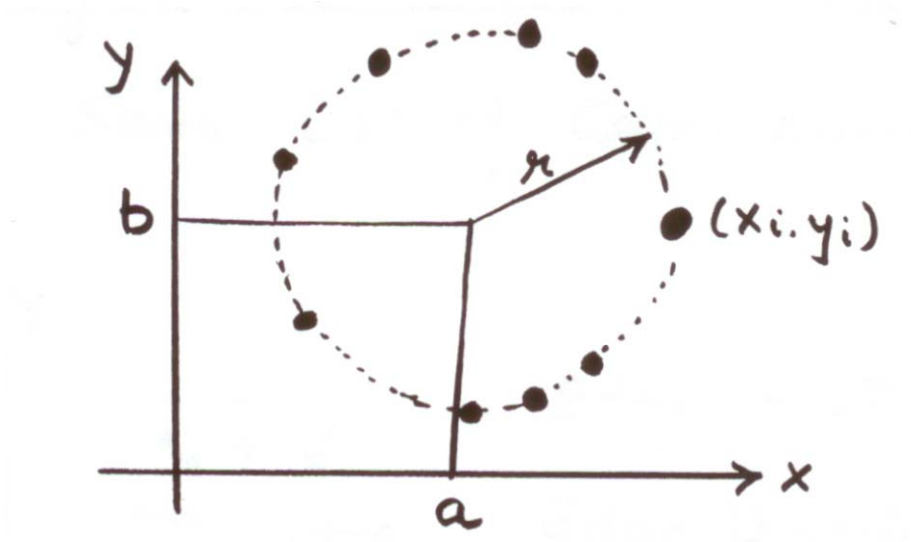




Finding Circles by Hough Transform

Equation of Circle:

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



If radius is not known: 3D Hough Space!

Use Accumulator array $A(a, b, r)$

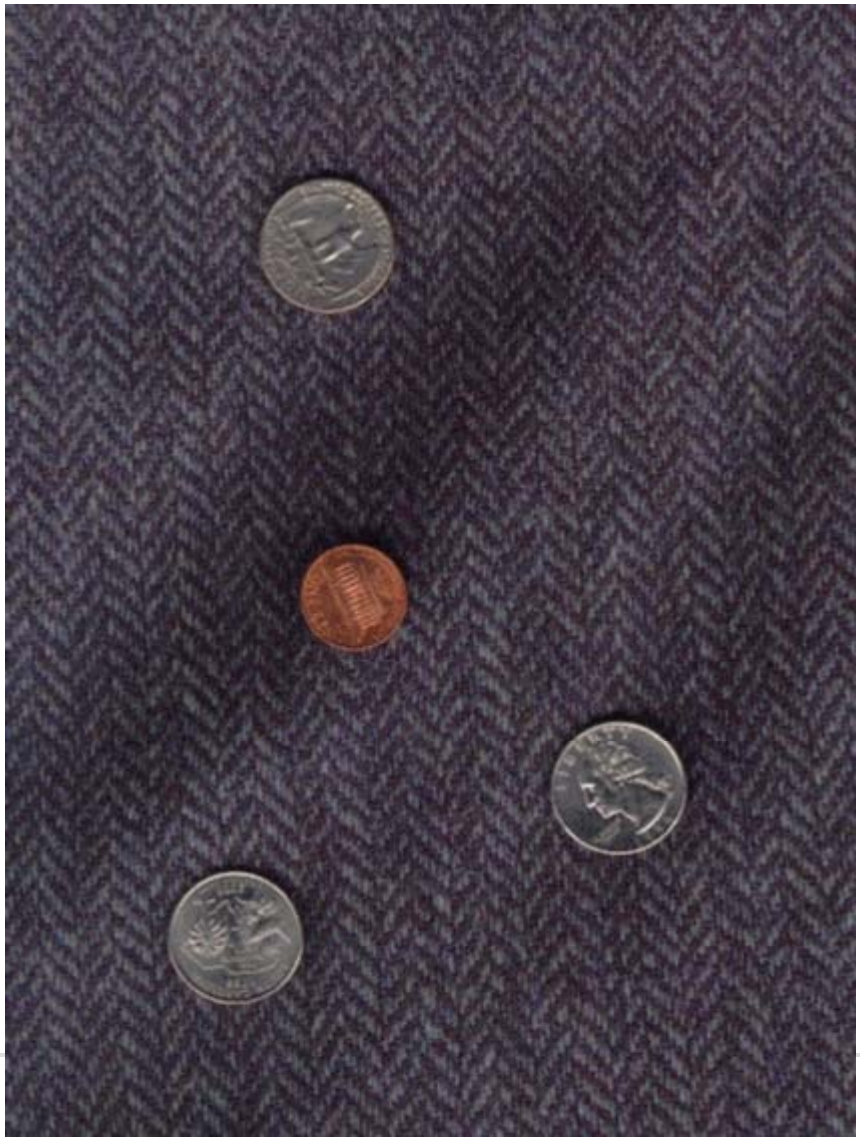
What is the surface in the hough space?

Real World Circle Examples

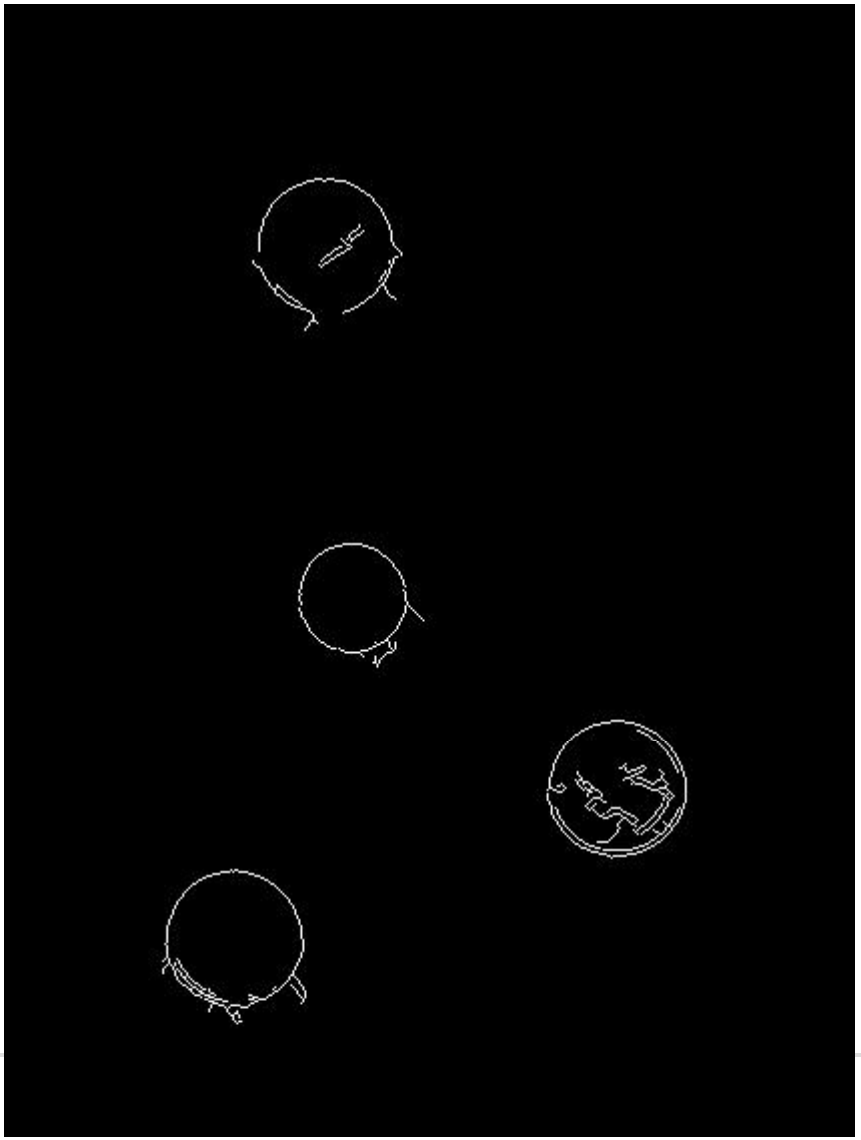
Crosshair indicates results of Hough transform, bounding box found via motion differencing.

Finding Coins

Original



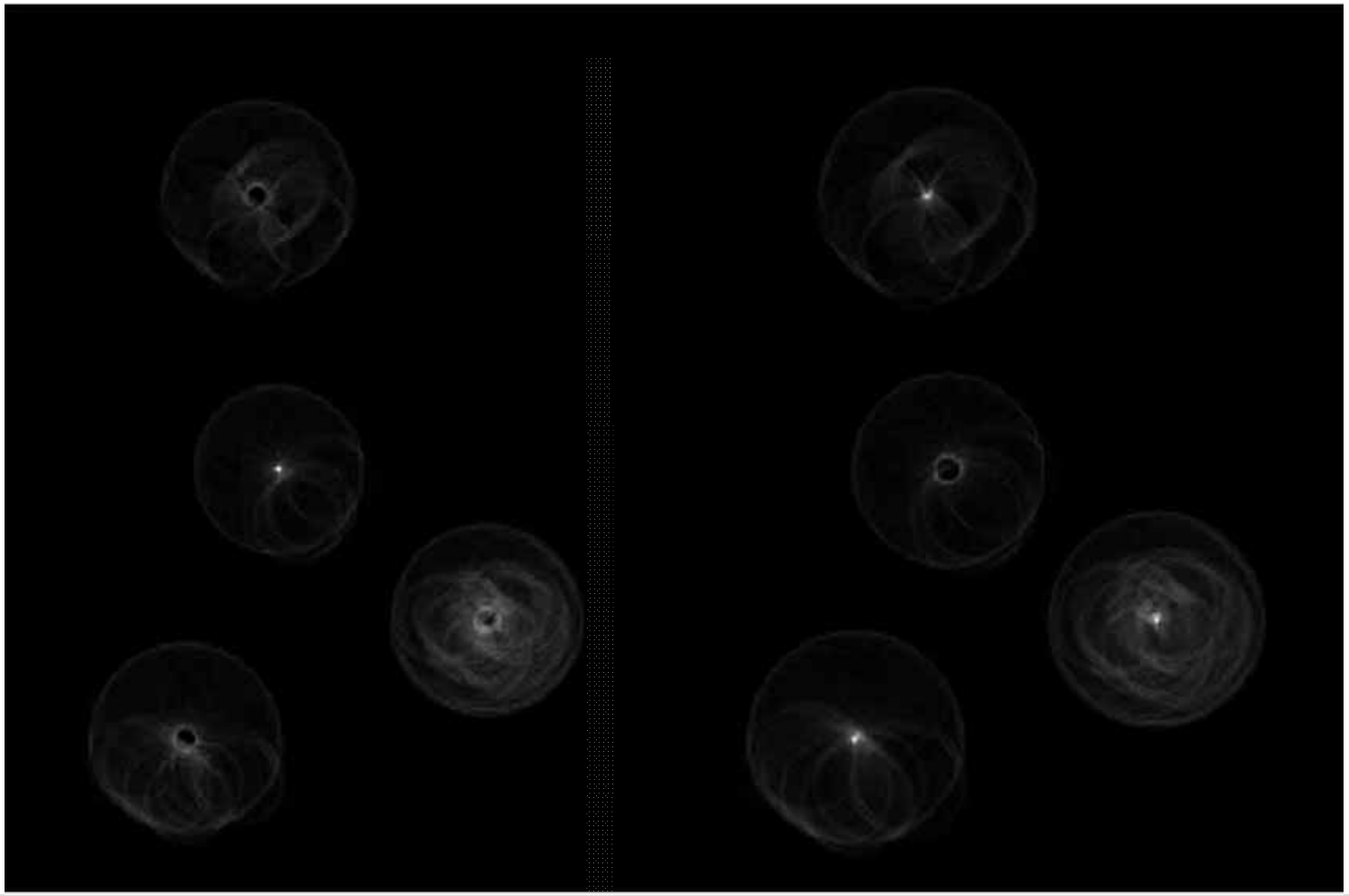
Edges (note noise)



Finding Coins (Continued)

Penny

Quarters

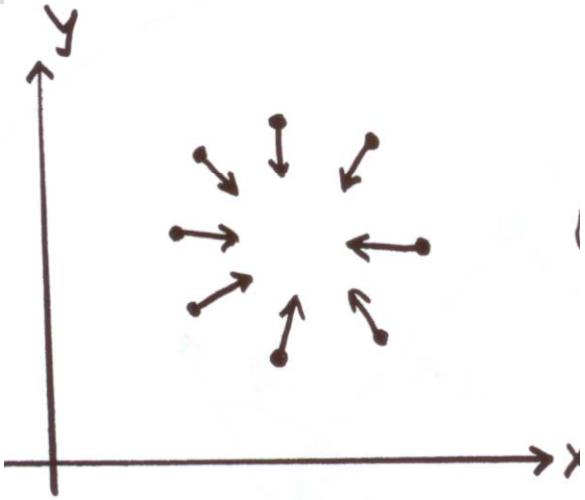


Using Gradient Information

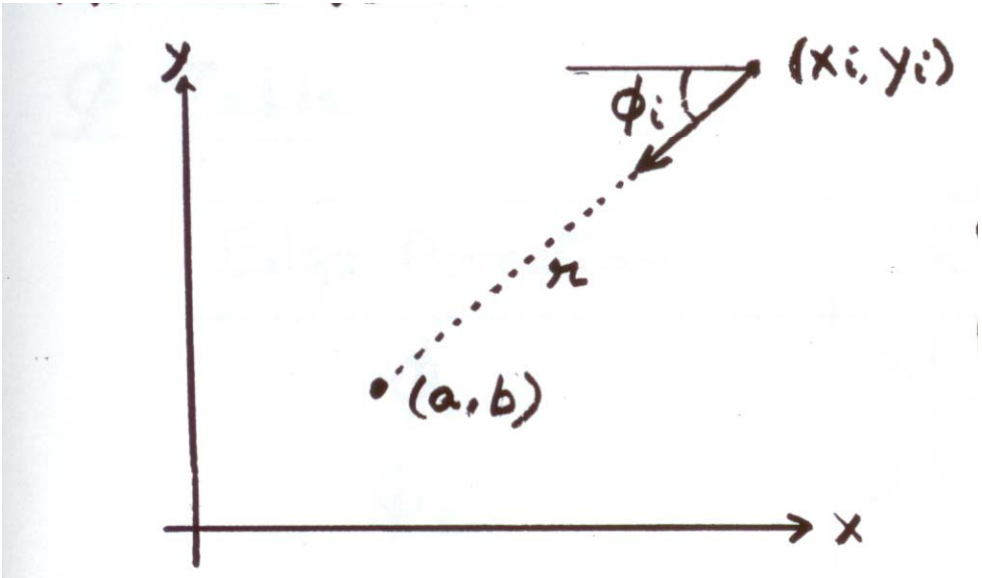
- Gradient information can save lot of computation:

  Edge Location (x_i, y_i)

  Edge Direction ϕ_i



Assume radius is known:



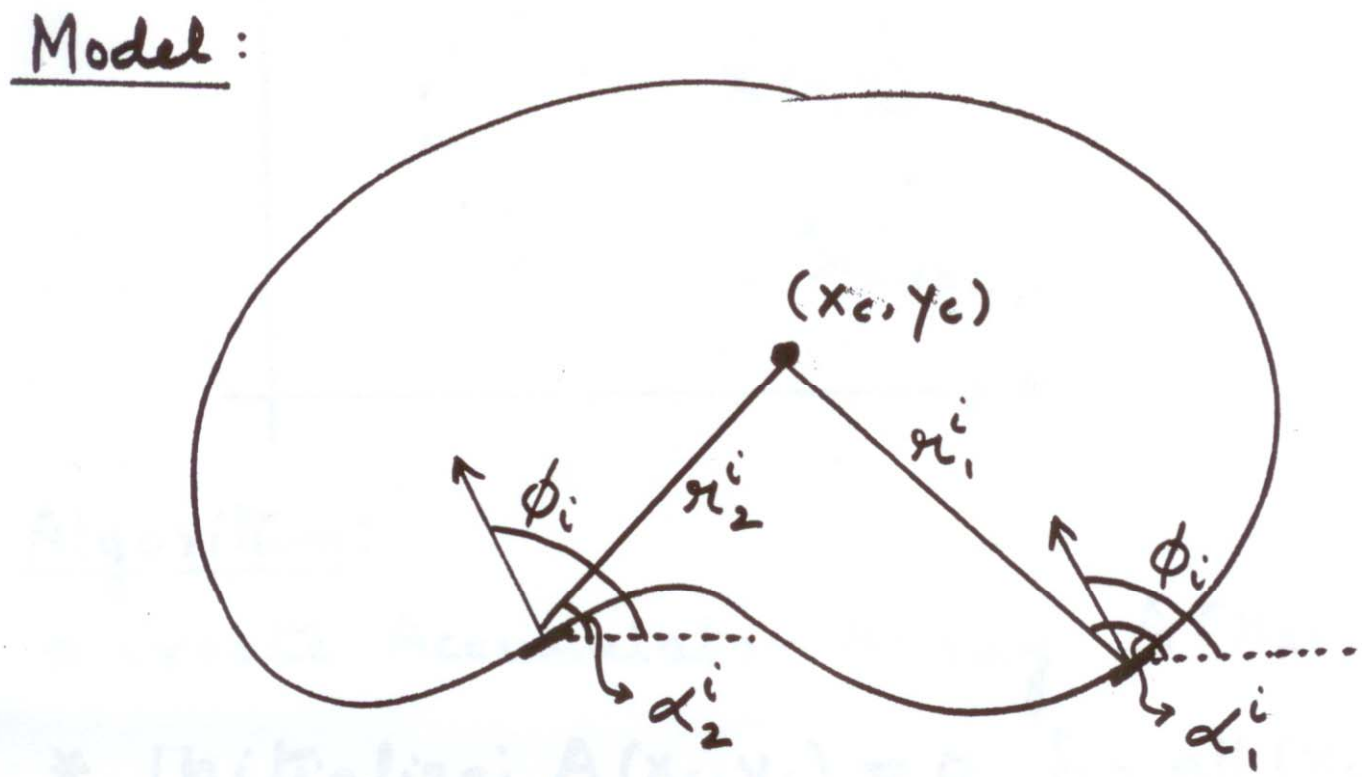
$$a = x - r\cos\phi$$

$$b = y - r\sin\phi$$

Need to increment only one point in Accumulator!!

Generalized Hough Transform

- Model Shape NOT described by equation



Generalized Hough Transform

- Model Shape NOT described by equation

ϕ - Table

Edge Direction	$\bar{r} = (r, \alpha)$
ϕ_1	$\bar{r}_1^1, \bar{r}_2^1, \bar{r}_3^1$
ϕ_2	$\bar{r}_1^2, \bar{r}_2^2$
ϕ_i	$\bar{r}_1^i \vdots \bar{r}_2^i$
ϕ_n	$\bar{r}_1^n, \bar{r}_2^n$

Generalized Hough Transform

Find Object Center (x_c, y_c) given edges (x_i, y_i, ϕ_i)

Create Accumulator Array $A(x_c, y_c)$

Initialize: $A(x_c, y_c) = 0 \quad \forall (x_c, y_c)$

For each edge point (x_i, y_i, ϕ_i)

For each entry $\overline{r}_k^i$ in table, compute:

$$x_c = x_i + r_k^i \cos \alpha_k^i$$

$$y_c = y_i + r_k^i \sin \alpha_k^i$$

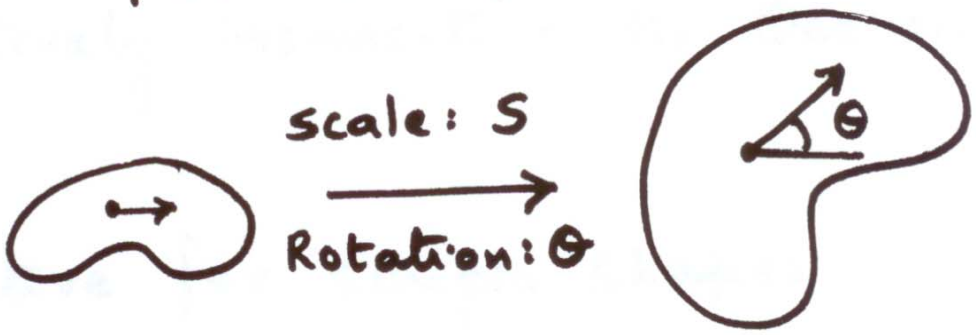
Increment Accumulator: $A(x_c, y_c) = A(x_c, y_c) + 1$

Find Local Maxima in $A(x_c, y_c)$

Scale & Rotation:

Use Accumulator Array:

$$A[x_c, y_c, S, \theta]$$



Use:

$$x_c = x_i + r^i_k S \cos(\alpha^i_k + \theta)$$

$$y_c = y_i + r^i_k S \sin(\alpha^i_k + \theta)$$

$$A(x_c, y_c, S, \theta) = A(x_c, y_c, S, \theta) + 1 .$$

Hough Transform: *Parting Remarks*

- Works on Disconnected Edges
- Relatively insensitive to occlusion
- Effective for simple shapes (lines, circles, etc)
- Trade-off between work in Image Space and Parameter Space
- Handling inaccurate edge locations:
    - Increment Patch in Accumulator rather than a single point

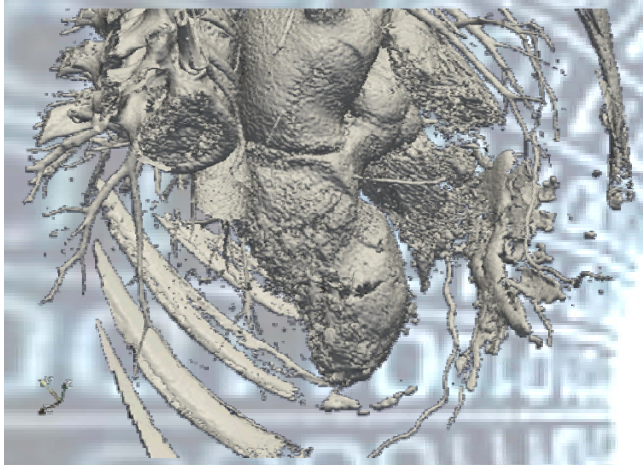
Global Features – a sneak preview / review

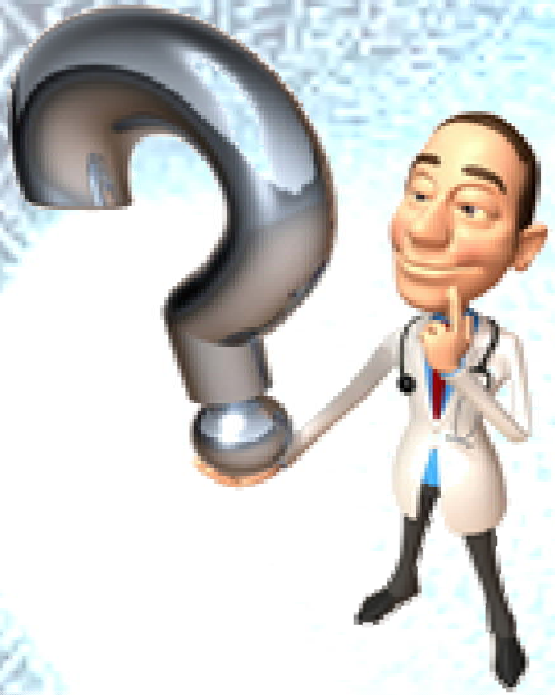
- Intensity Histograms used for thresholding / statistical decisions
- Features in feature-spaces obtained using series decompositions :

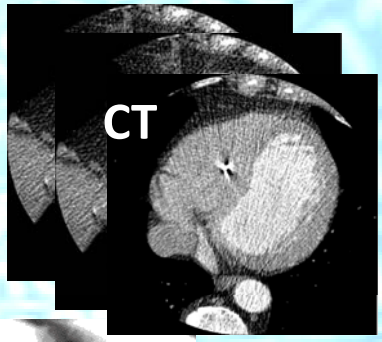
  - PCA, Laplace decomposition, Fourier decomposition

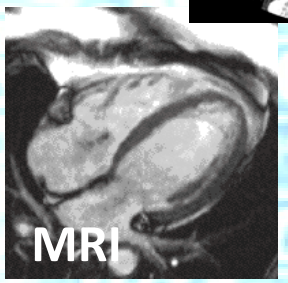
BIA 2014

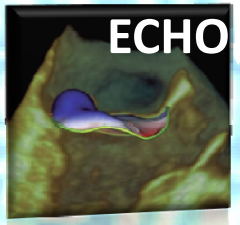
Carnegie Mellon University

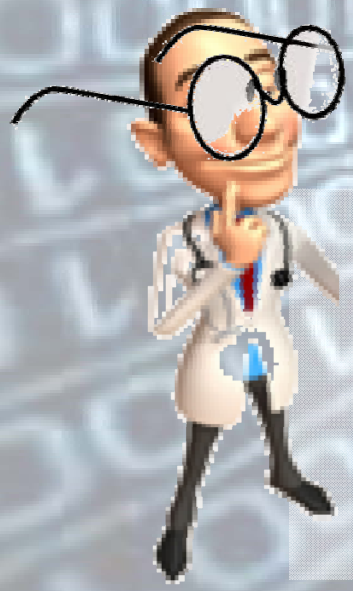


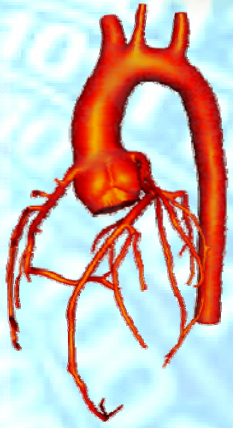












Prahlad G Menon, PhD

www.justcallharry.com

+1 412-259-3031

pgmenon@andrew.cmu.edu