CAP 5415 Computer Vision
Fall 2012

Hough Transform

Lecture-18

Sections 4.2, 4.3 Fundamentals of Computer Vision

Alper Yilmaz, Mubarak Shah, Fall 2011
UCF

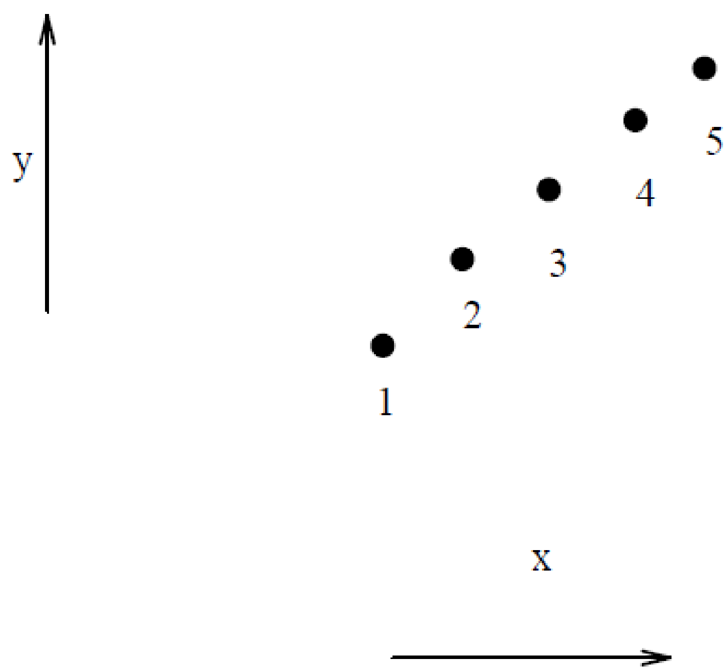
Image Feature Extraction

- Edges (edge pixels)
  - Sobel, Roberts, Prewit
  - Laplacian of Gaussian
  - Canny
- Interest Points
  - Harris
  - SIFT
- Descriptors
  - SIFT
  - HOG

Shape Features

- Straight Lines
- Circles and Ellipses
- Arbitrary Shapes

How to Fit A Line?



$$y = mx + c$$

How to Fit A Line?

- Least squares Fit (over constraint)
- RANSAC (constraint)
- Hough Transform (under constraint)

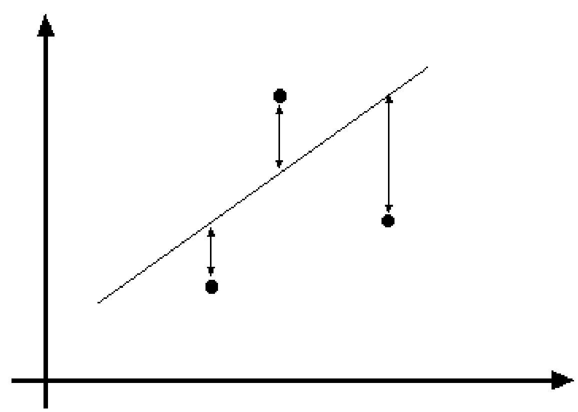
Least Squares Fit

- Standard linear solution to estimating unknowns.
  - If we know which points belong to which line
  - Or if there is only one line

$$y = mx + c = f(x, m, c)$$

$$\text{Minimize } E = \sum_i \left[y_i - f(x_i, m, c) \right]^2$$

Take derivatives wrt *m* and *c* set them to 0



Line Fitting

$$y = mx + c$$

$$y_1 = mx_1 + c$$
$$y_2 = mx_2 + c$$
$$\vdots$$
$$y_n = mx_n + c$$

$$\underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}_{B} = \underbrace{\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & 1 \\ x_n & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} m \\ c \end{bmatrix}}_{D} \quad \Rightarrow B = AD$$

$$A^T B = A^T A D$$

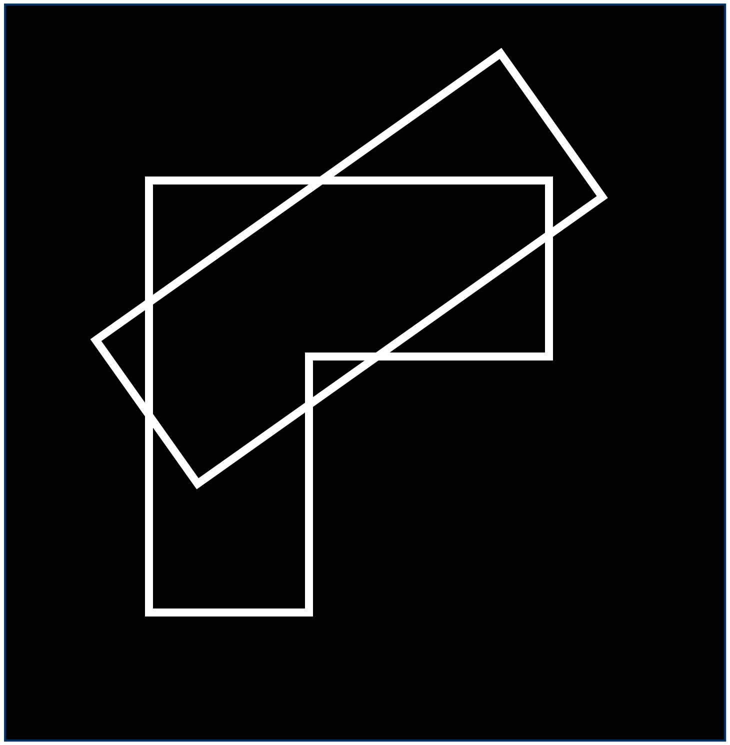
$$\left(A^T A\right)^{-1} A^T B = \left(A^T A\right)^{-1} \left(A^T A\right) D$$

$$D = \left(A^T A\right)^{-1} A^T B$$

RANSAC: Random Sampling and Consensus

1. Randomly select two points to fit a line
2. Find the error between the estimated solution and all other points. If the error is less than tolerance, then quit, else go to step (1).

Line Fitting: Segmentation



- Several Lines
- How do we Know which points belong to which lines?

Alper Yilmaz, Mubarak Shah, Fall 2011
UCF

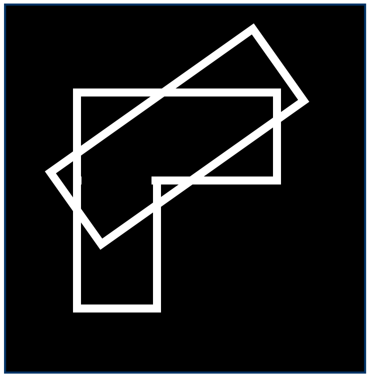
Hough Transform

- **METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS,** Paul V. C. Hough et al
  - **Inventors**: Paul V. C. Hough, Paul V. C. Hough **Current U.S. Classification**: 382/281; 342/176; 342/190; 382/202
  - http://www.google.com/patents?vid=3069654

Line Fitting: Hough Transform

- Line equation

$y = mx + c \qquad m \text{ is slope, } c \text{ is } y\text{-intercept}$



- Rewrite this equation

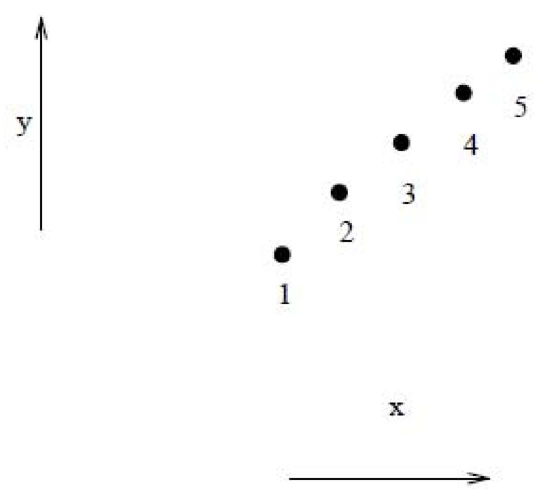
$c = (-x)m + y$

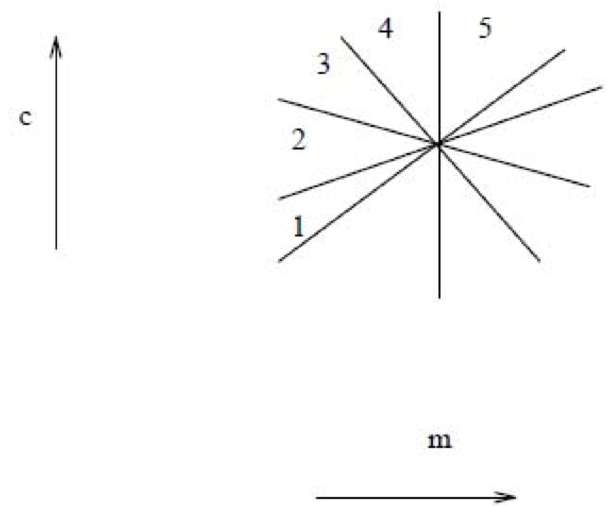
- For particular edge point *i* this becomes

  $c = (-x_i)m + y_i$

- This is an equation of a line in (*c*,*m*) space.

Line Fitting: Hough Transform





$$c = (-x_i)m + y_i$$

Alper Yilmaz, Mubarak Shah, Fall 2011
UCF

Hough Transform Algorithm for Fitting Straight Lines

1. Quantize the parameter space $P[c_{min}, \ldots\ldots, c_{max}, m_{min}, \ldots\ldots, m_{max}]$.

2. For each edge point (x, y) do
   for $(m = m_{min}, m \leq m_{max}, m++)$ do
   $c = (-x)m + y$,
   $P[c, m] = P[c, m] + 1$.

3. Find the local maxima in the parameter space.

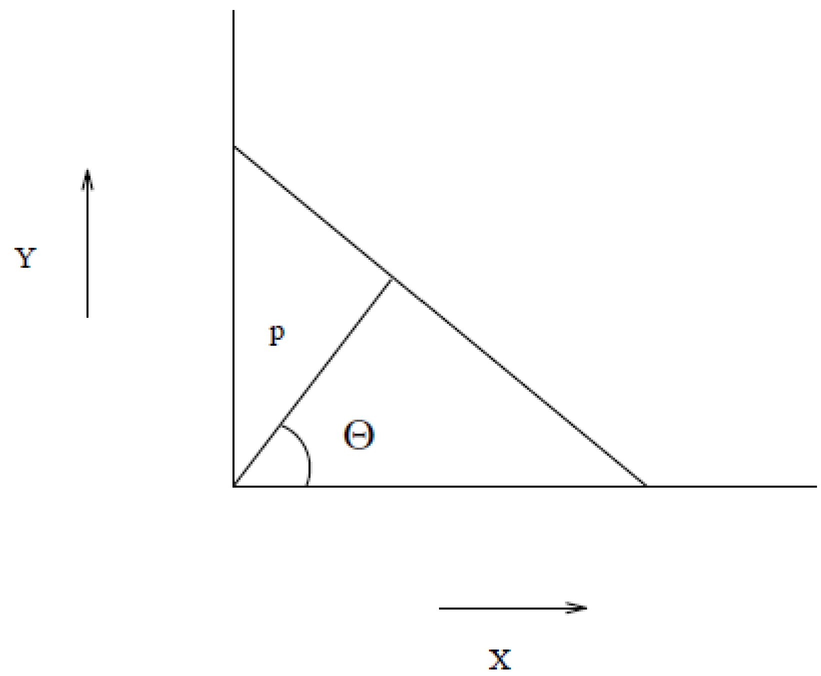
Figure 4.2: Hough transform algorithm for fitting straight lines.

Polar Form of Equation of Line

$$c_i = (-x)m_j + y$$

Problematic for vertical lines

m and c grow to infinity



$$p = x\cos\theta + y\sin\theta$$

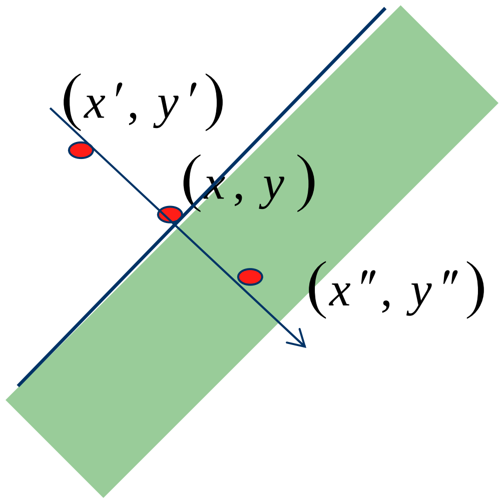
Use θ from gradient

Image Gradient

(S_x, S_y) Gradient Vector

magnitude $= \sqrt{(S_x^2 + S_y^2)}$

direction $= \theta = \tan^{-1} \dfrac{S_y}{S_x}$



Hough Transform for Polar Form of Equation of Line

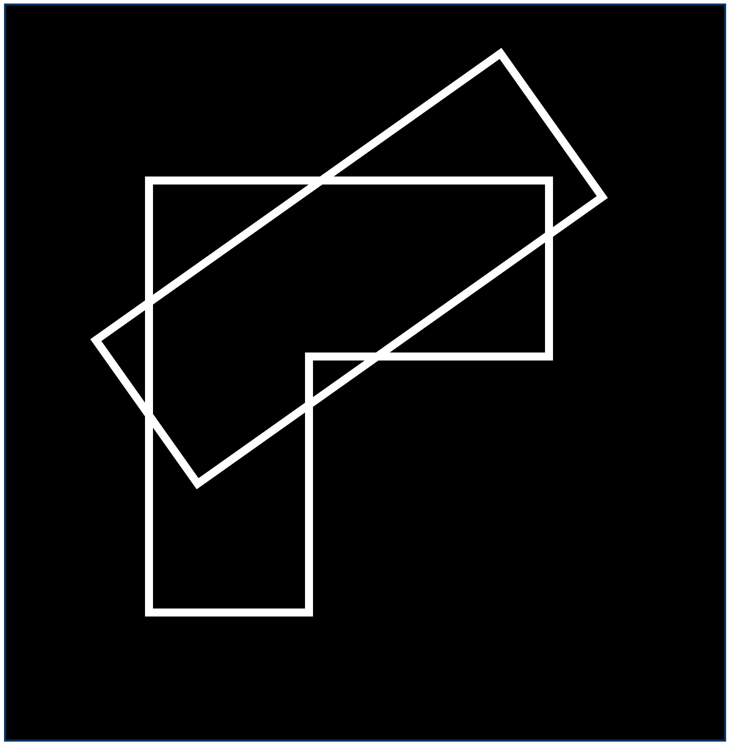
1. Quantize the parameter space $P[\theta_{min}, \ldots\ldots, \theta_{max}, p_{min}, \ldots\ldots, p_{max}]$.

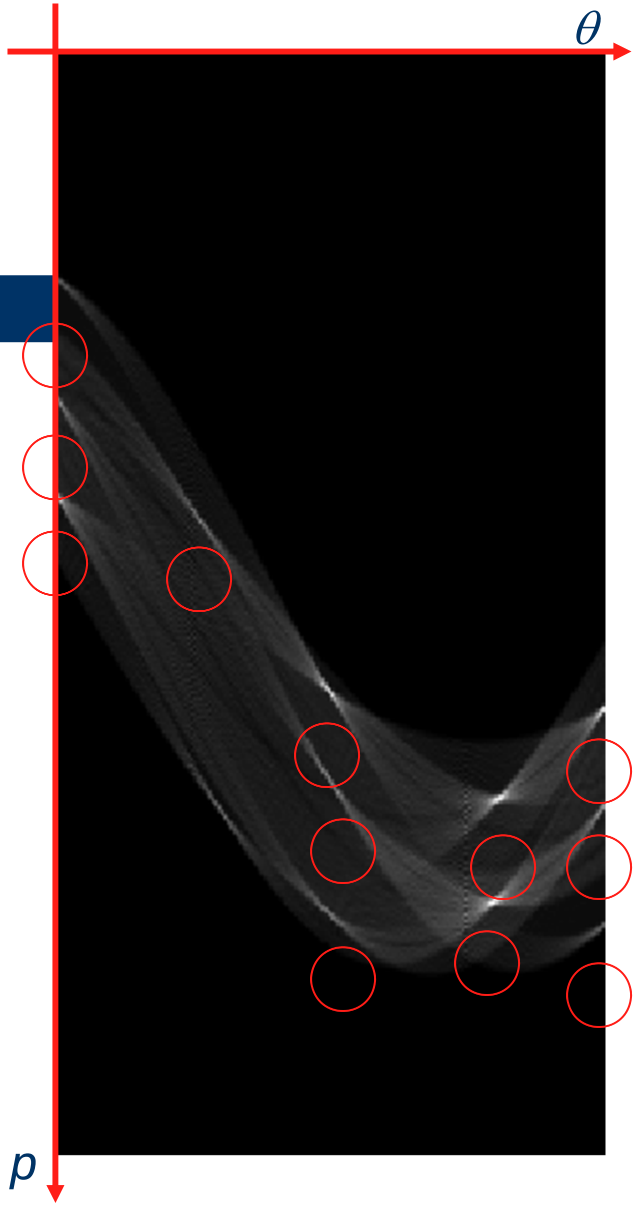
2. For each edge point (x, y) do
   $p = x\cos\theta + y\sin\theta$,
   $P[\theta, p] = P[\theta, p] + 1$.

3. Find the local maxima in the parameter space.

Figure 4.4: Hough transform algorithm using polar form of equation of straight line.

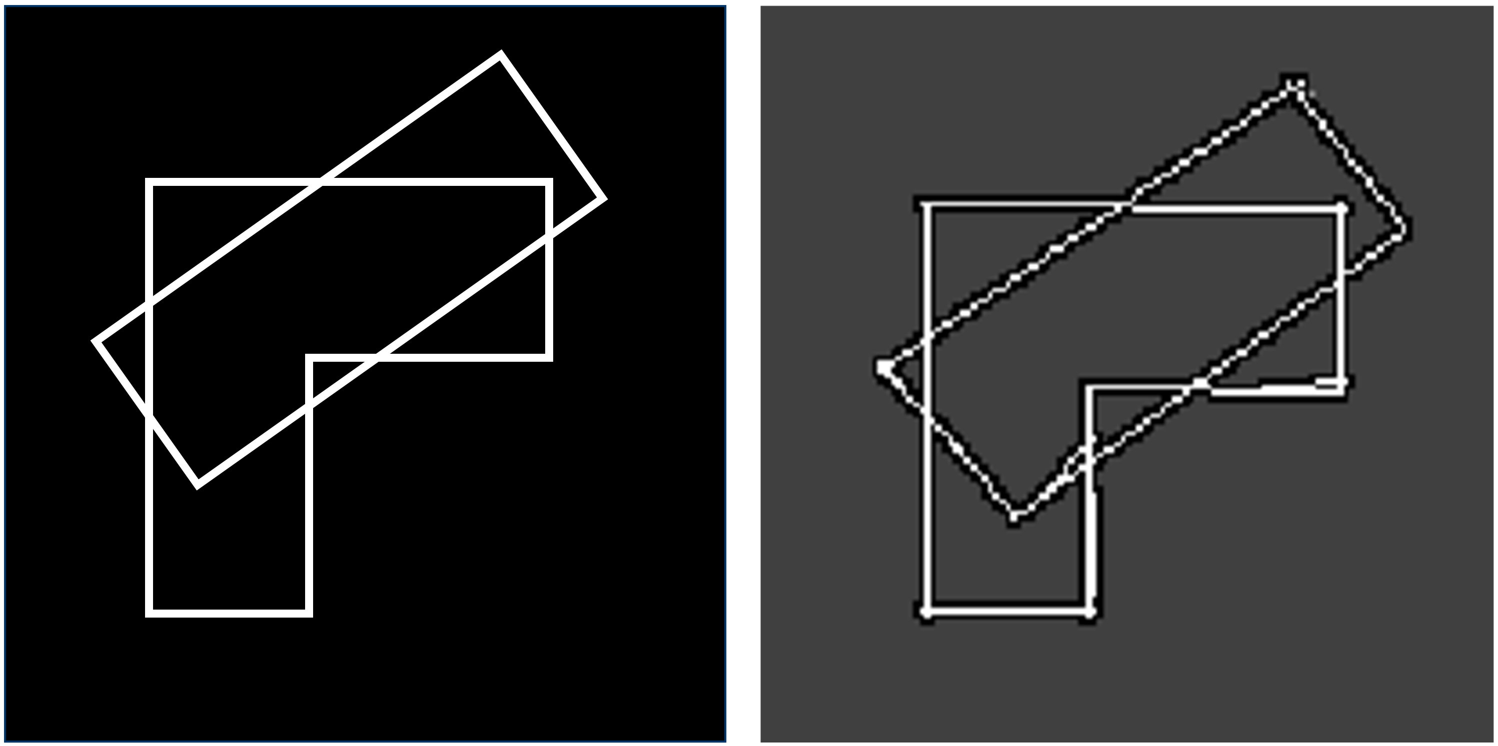
Line Fitting





Alper Yilmaz, Mubarak Shah, Fall 2011 UCF

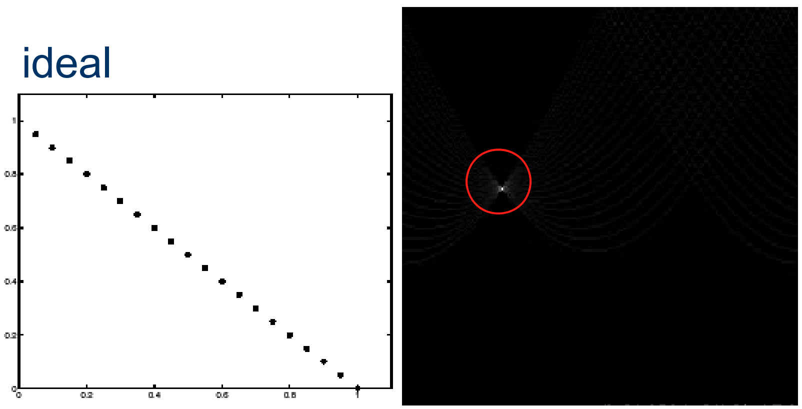
Line Fitting



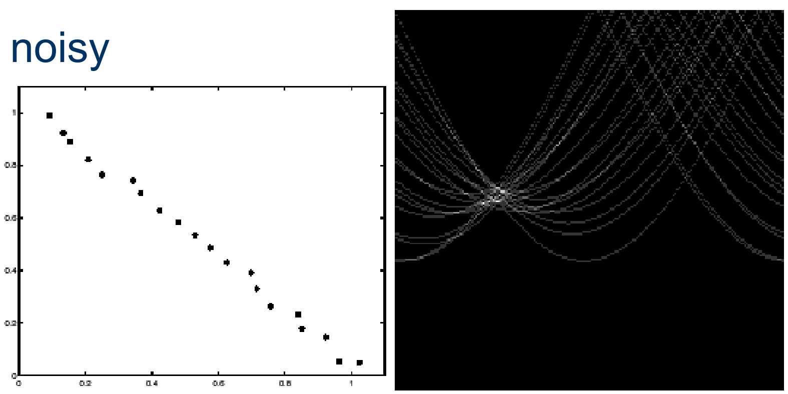
Alper Yilmaz, Mubarak Shah, Fall 2011
UCF

Line Fitting Examples

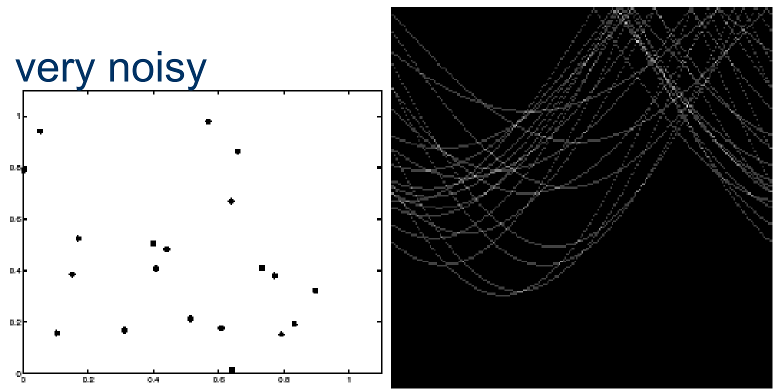
ideal



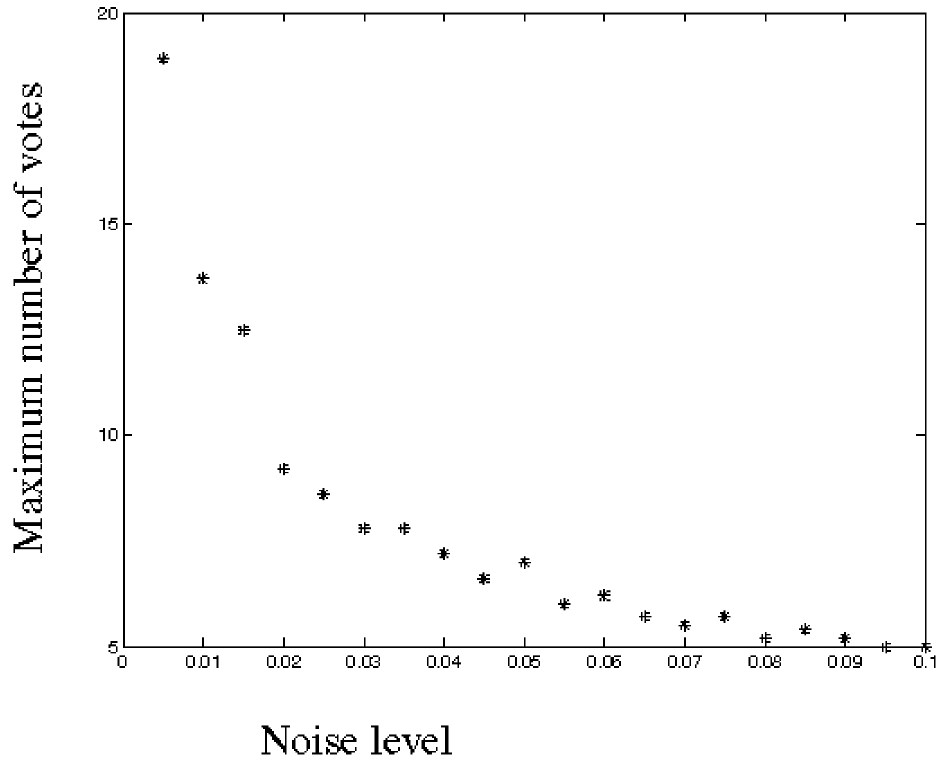
noisy



very noisy

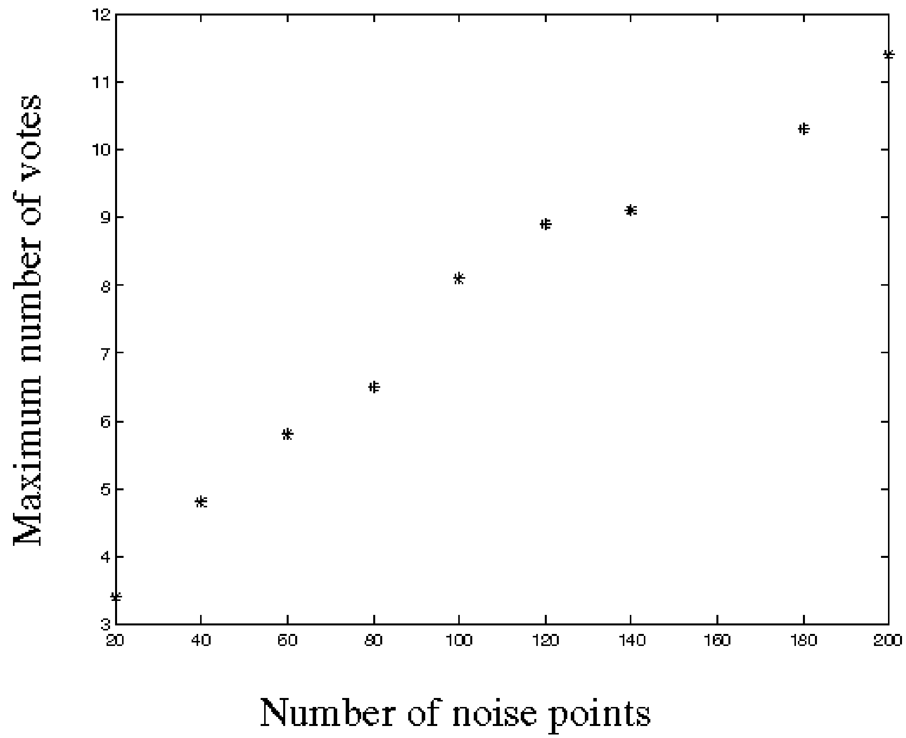


Noise Factor



This is the number of votes that the real line of 20 points gets with increasing noise

Noise Factor



as the noise increases in a picture without a line, the number of points in the max cell goes up, too

Difficulties

- What is the increments for θ and *p*.
  - too large? We cannot distinguish between different lines
  - too small? noise causes lines to be missed

Circle Fitting

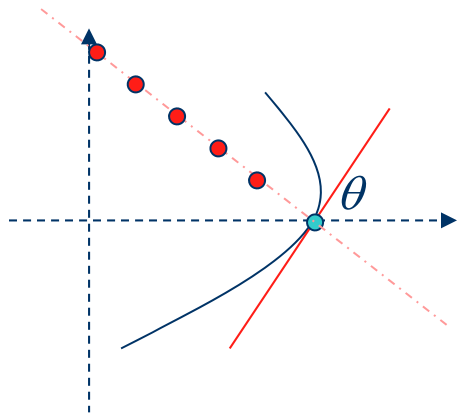
- Similar to line fitting
  - Three unknowns

$$(x - x_o)^2 + (y - y_o)^2 - r^2 = 0$$

- Construct a 3D accumulator array **A**
  - Dimensions: x_0, y_0, r
- Fix one of the parameters and loop for the others
- Increment corresponding entry in **A**.
- Find the local maxima in **A**

More Practical Circle Fitting

- Use the tangent direction θ at the edge point



- Compute x_0, y_0 given x, y, r

$$x_0 = x - r\cos\theta$$
$$y_0 = y - r\sin\theta$$

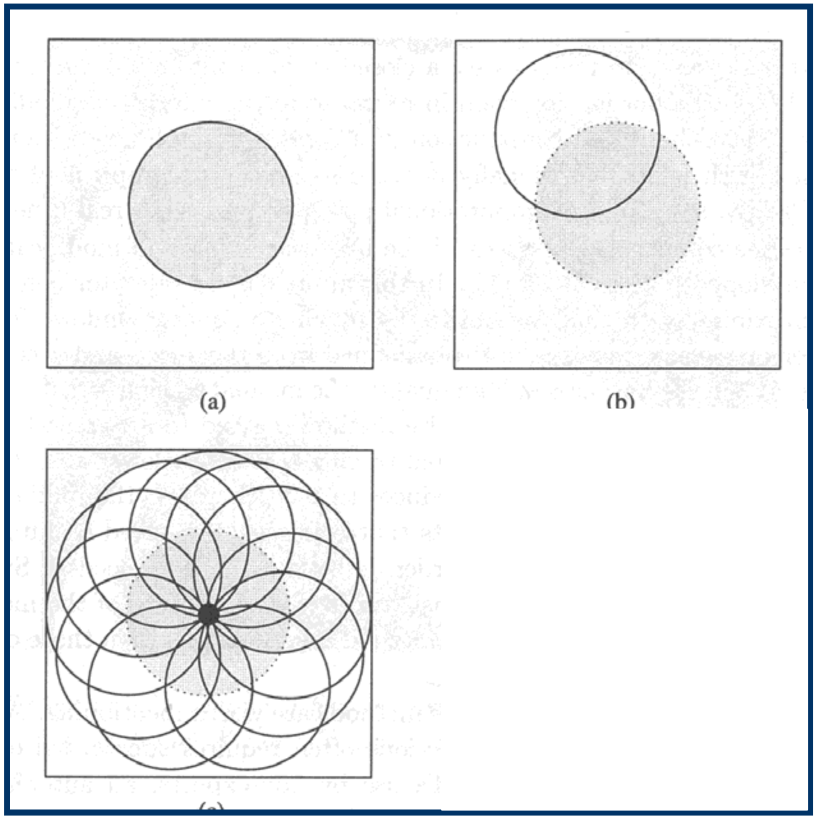
1. Quantize the parameter space
   $P[x_{0_{min}}, \ldots, x_{0_{max}}, y_{0_{min}}, \ldots, y_{0_{max}}, r_{min}, \ldots, r_{max}]$.

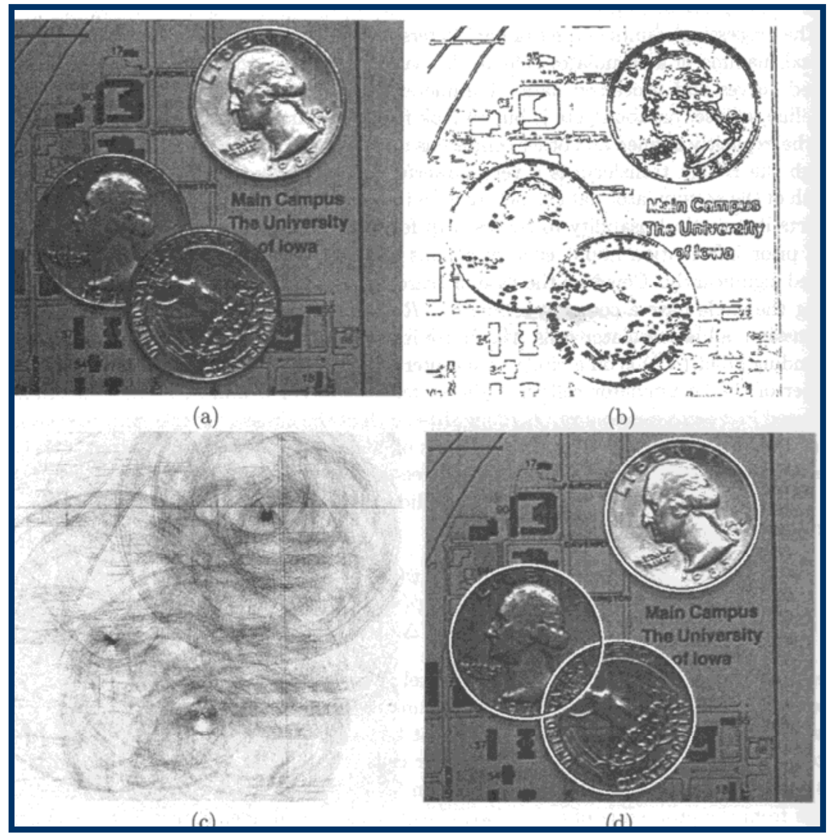
2. For each edge point (x, y) do
   For $(r = r_{min}, r \leq r_{max}, r++)$
   $x_0 = x - r\cos\theta$
   $y_0 = y - r\sin\theta$
   $P[x_0, y_0, r] = P[x_0, y_0, r] + 1$.

3. Find the local maxima in the parameter space.

Figure 4.6: Hough transform algorithm for fitting circle using polar form of equation of a circle.

Examples





Generalized Hough Transform

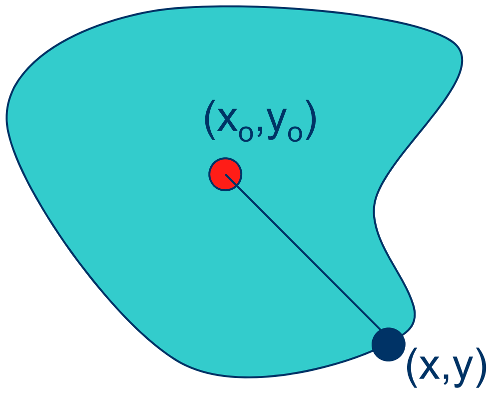
- Used for shapes with ***no*** analytical expression
- Requires training
  - Object of known shape
  - Generate model
    - R-table
- Similar approach to line and circle fitting during detection

Generating R-table

- Compute centroid
- For each edge compute its distance to centroid

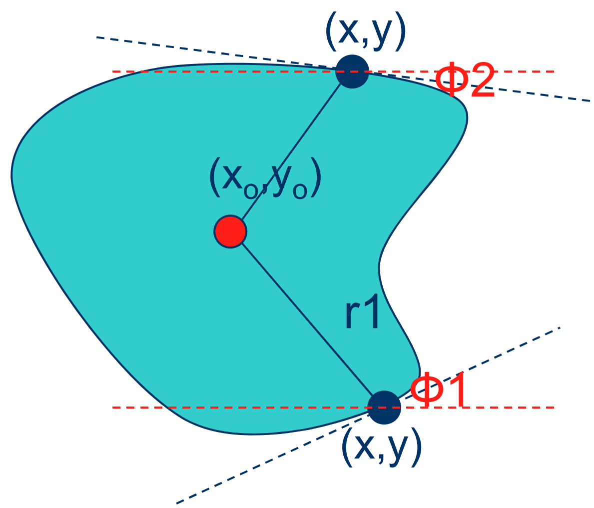
$$r = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x - x_0 \\ y - y_0 \end{bmatrix}$$

- Find edge orientation (gradient angle)
- Construct a table of angles and *r* values



Generating R-table

Φ1	r1, r2, r3 …
Φ2	r14, r21, r23 …
Φ3	r41, r42, r33 …
Φ4	r10, r12, r13 …



Detecting shape

- known
  - Edge points (x,y)
  - Gradient angle at every edge point θ
  - R-table of the shape needs to be determined
- For each edge point find θ store it in corresponding row of R-table
- Create an accumulator array of 2D (x,y)

1. Quantize the parameter space $P[x_{cmin}, \ldots\ldots, x_{cmax}, y_{cmin}, \ldots\ldots, y_{cmax}]$.

2. For each edge point (x, y) do
   compute $\phi(x, y)$
   for each table entry for ϕ do

$$x_c = x + x' \quad (4.13)$$

$$y_c = y + y' \quad (4.14)$$

   $P[x_c, y_c] = P[x_c, y_c] + 1$.

3. Find the local maxima in the parameter space.

Figure 4.8: Generalized Hough transform algorithm.

Rotation and Scale Invariance

- Rotation around Z-axis

$$x' = x\cos\alpha - y\sin\alpha$$

$$y' = x\sin\alpha + y\cos\alpha$$

- Scaling

$$x' = sx$$

$$y' = sy$$

- Rotation+scaling

$$x' = s(x\cos\alpha - y\sin\alpha)$$

$$y' = s(x\sin\alpha + y\cos\alpha)$$

Rotation and Scale Invariance

- Replace equations 4.13 and 4.14 in Algorithm 4.8 by following and loop for scale and rotation angles.

$$\begin{aligned} x_c &= x + s_x(x' \cos\theta + y' \sin\theta) \\ y_c &= y + s_y(-x' \sin\theta + y' \cos\theta) \end{aligned}$$

1. Quantize the parameter space $P[x_{cmin}, \ldots\ldots, x_{cmax}, y_{cmin}, \ldots\ldots, y_{cmax}]$.

2. For each edge point (x, y) do
   compute $\phi(x, y)$
   for each table entry for ϕ do

$$x_c = x + x' \quad (4.13)$$

$$y_c = y + y' \quad (4.14)$$

   $P[x_c, y_c] = P[x_c, y_c] + 1.$

3. Find the local maxima in the parameter space.

Figure 4.8: Generalized Hough transform algorithm.