ME6406 Machine Vision

Professor Kok-Meng Lee

Georgia Institute of Technology

George W. Woodruff School of Mechanical Engineering

Atlanta, GA 30332-0405

Email: kokmeng.lee@me.gatech.edu

Part 2 Model-based Vision
A. Hough Transform

http://kmlee.gatech.edu/me6406

1

Course Outline

- Introduction and low-level processing
  – Physics of digital images, histogram equalization, segmentation, edge detection, linear filters.
- **Model-based Vision**
  – **Hough transform, pattern representation, matching**
- Geometric methods
  – Camera model, calibration, pose estimation
- Neural network for machine vision
  – Basics, training algorithms, and applications
- Color images and selected topics
  – Physics, perception, processing and applications

Contents

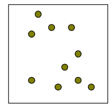
- **Hierarchical Feature Extraction**
- **Hough Transform (HT)**
  - Lines (without gradient)
  - Foot normal (line detection using gradient)
  - Circles and ellipses
  - Generalized HT

Hierarchical Feature Extraction

- **Most features are extracted by combining a small set of primitive features (edges, corners, regions)**
  - Grouping: which edges/corners/curves form a group? (Perceptual organization at the intermediate-level of vision)
  - Model Fitting: what structure best describes the group?
- **Consider a slightly simpler problem… From Edges to curves?**
  - Given local edge elements, can we organize these into more 'complete' structures, such as straight lines?
  - General idea:
    - Find an alternative space in which lines map to points.
    - Each edge element 'votes' for the straight line which it may be a part of.
    - Points receiving a high number of votes might correspond to actual straight lines in the image.
  - The idea behind the **Hough transform** is that a change in representation converts a point grouping problem into a peak detection problem.

Hough Transform

- A technique used to detect specific structural relationships among pixels in a binary digital image.
  - Mathematical equations; lines, circles and ellipses.
- Introduction: Given *n* points in an image, find a subsets of these points that lie on straight lines.



One possible solution (trivial) ,

Step 1: find all lines determined by every pairs of points,

$$\frac{n(n-1)}{2} \sim O(n^2) \text{ lines}$$

Step 2: find all subsets of points that are closed to particular lines.

$$(n-2)\left[\frac{n(n-1)}{2}\right] \sim O(n^3) \text{ operations}$$

Problems: Computationally prohibitive!

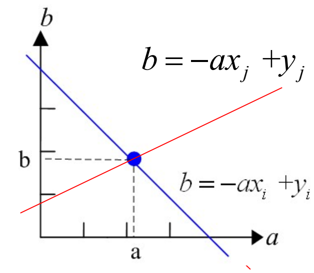
5

Hough's basic argument

- Alternatively, consider a point (x_i, y_i) and the general equation of a straight line:

$$y_i = ax_i + b \quad (1)$$

On the *xy* plane: Infinite number of lines pass through (x_i, y_i) but all satisfy Eq. (1) for every pair of (a, b).



- If we rewrite the equation

$$b = -ax_i + y_i \quad (2)$$

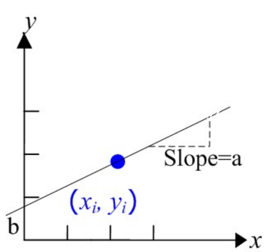
then we have a **single line** for a fixed pair of (x_i, y_i).

- We call the (a, b) plane as the **parameter** plane.
- Every point on the xy corresponds to a line in (a, b) plane.

6

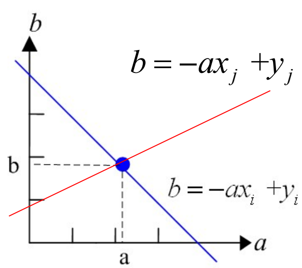
Comparison

Image (x, y) plane



$$y_i = ax_i + b \quad (1)$$

Parametric (a, b) plane

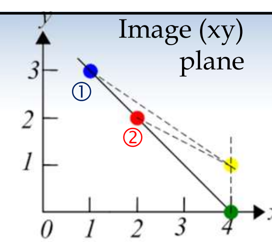


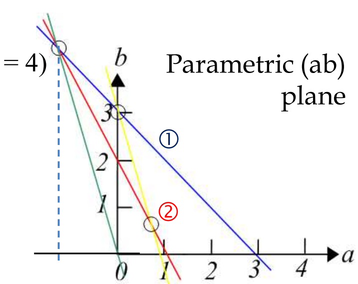
$$b = -ax_i + y_i \quad (2)$$

7

Example:

	x, y	$y = ax + b$	$b = -ax + y$
①	1, 3	$3 = a1 + b$	$b = -1a + 3$
②	2, 3	$2 = a2 + b$	$b = -2a + 2$
	4, 3	$3 = a4 + b$	$b = -4a + 3$
	4, 0	$0 = a4 + b$	$b = -4a + 0$





General Idea:

- The Hough space (ab) is a representation of every possible line segment in the plane.
- Make the Hough space *ab* discrete
- Let every edge point in the image 'vote for' any line it might belong to.

8

Line Detection Algorithm: Hough Transform

- Quantize a and b into appropriate 'buckets'.
  - Need to decide what's 'appropriate'
- Create accumulator array $H(a, b)$, *all* of whose elements are initially zero.
- For each point (i, j) in the edge image for which the edge magnitude is above a specific threshold, increment all points in $H(a, b)$ for all discrete values of a and b satisfying $b = -aj+i$.
  - Note that *H* is a two-dimensional histogram
- Local maxima in *H* corresponds to colinear edge points in the edge image.

9

Hough's basic argument

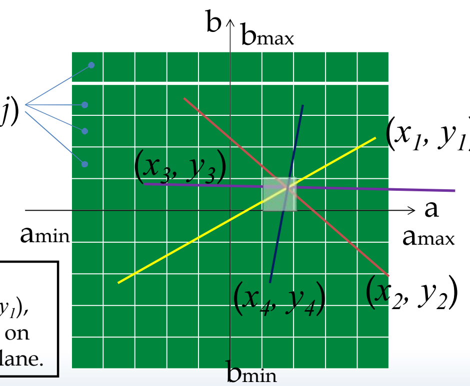
- If n points, $(x_i, y_i) \dots (x_j, y_j) \dots (x_n, y_n)$, are on the same line (with slope a and intercept b), then there will be n lines intercepting on the parametric space at (a, b).

$$b = -ax_i + y_i$$

Accumulator cell $H(i, j)$

$$a_{\min} \le a_k \le a_{\max}$$

$$b_{\min} \le b_k \le b_{\max}$$

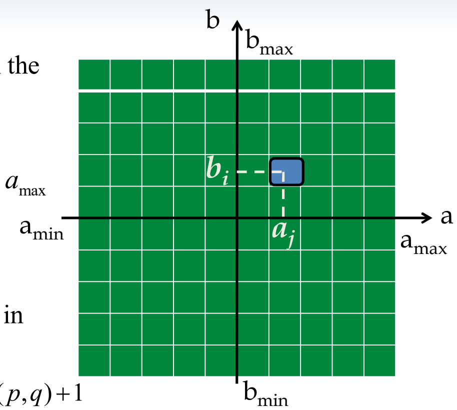


Example: $H(2,1) = 4$ implying that (x_1, y_1), (x_2, y_2), (x_3, y_3) and (x_4, y_4) are on the same line in the image plane.

10

Basic concept of Hough's algorithm

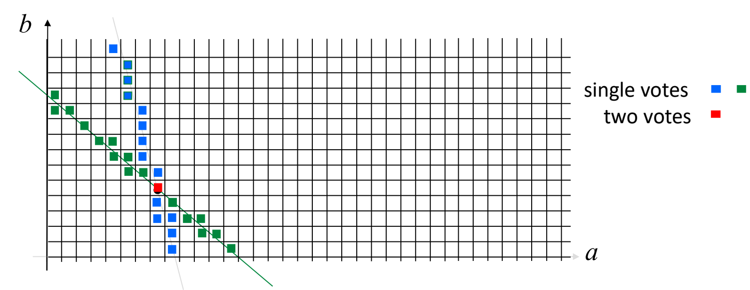
1. Initialize $H(i, j) = 0$
2. For every point (x_k, y_k) in the image plane
   - Let
     $a = a_i$ where $a_{\min} \le a \le a_{\max}$
     compute $b = -ax_k + y_k$
     round $b = b_j$
   - If a choice of a_p results in solution b_q
     Increment $H(p,q) = H(p,q) + 1$
3. At the end of this procedure, a value of M in $H(i, j)$ corresponds to M points in the image xy plane lying on the line.



11

Quantized Parameter Space

- Quantization



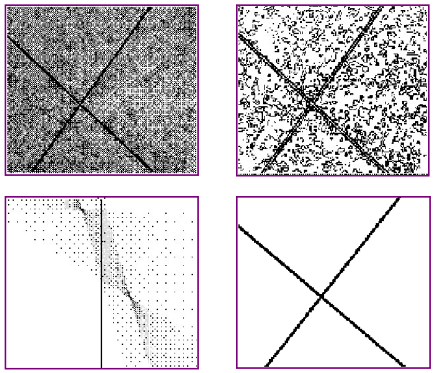
The problem of line detection in image space has been transformed into the problem of cluster detection in parameter space.

12

Example

Image

Edges



Accumulator Array

Result

13

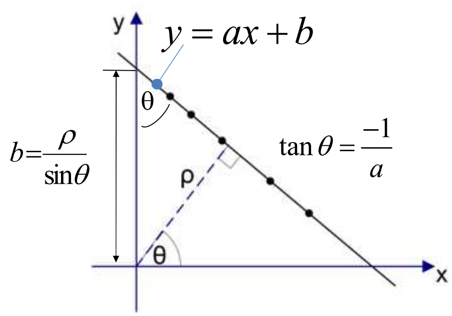
Hough Transform

- Computationally attractive.
  - If a is subdivided to *K* increment for every point (x_k, y_k), there are *K* possible values of *b* corresponding to *K* possible values of *a*.
  - For *n* image points, involving *nK* computations; linear in *n*.
- Accuracy of the co-linearity depends on the number of subdivisions in the ab plane.
- The range of the parameters (a, b) for the equation presents a problem:

$$y = ax + b \quad -\infty \leq a \leq +\infty$$ **(Vertical straight line)**

14

Alternative parameters (ρ, θ)



$$y = \underbrace{\left(-\frac{\cos\theta}{\sin\theta}\right)}_{a} x + \underbrace{\left(\frac{\rho}{\sin\theta}\right)}_{b} \Rightarrow \rho = x\cos\theta + y\sin\theta$$

$$0° \leq \theta \leq 180°$$

15

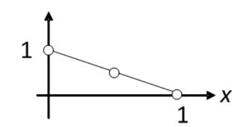
Example 1

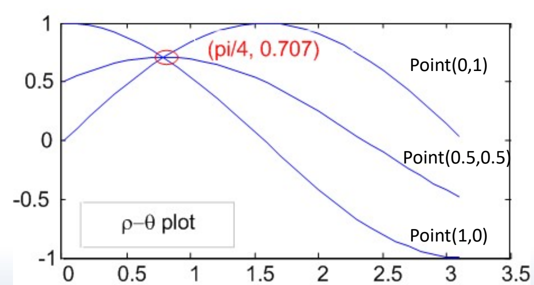
$$\rho = x\cos\theta + y\sin\theta$$

Point (0,1): $\rho = \sin\theta$

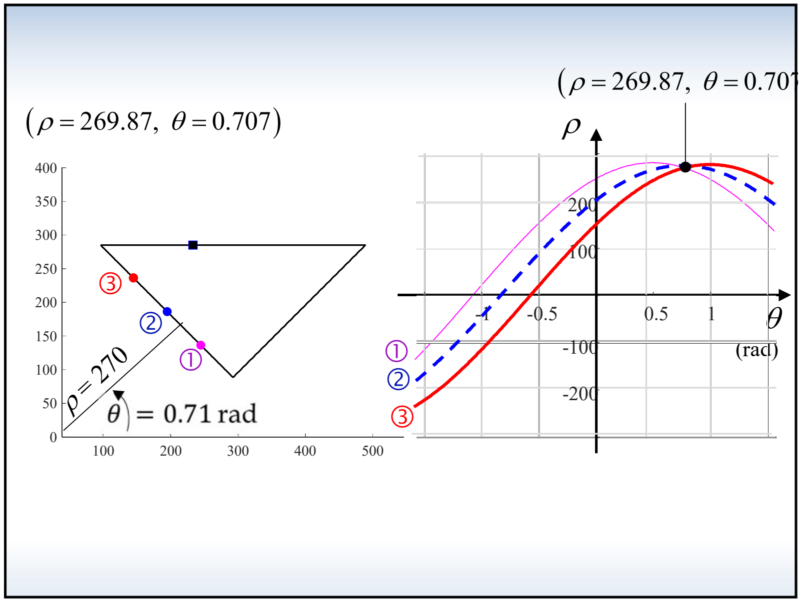
Point (0.5, 0.5): $\rho = 0.5(\cos\theta + \sin\theta)$

Point (1,0): $\rho = x\cos\theta$

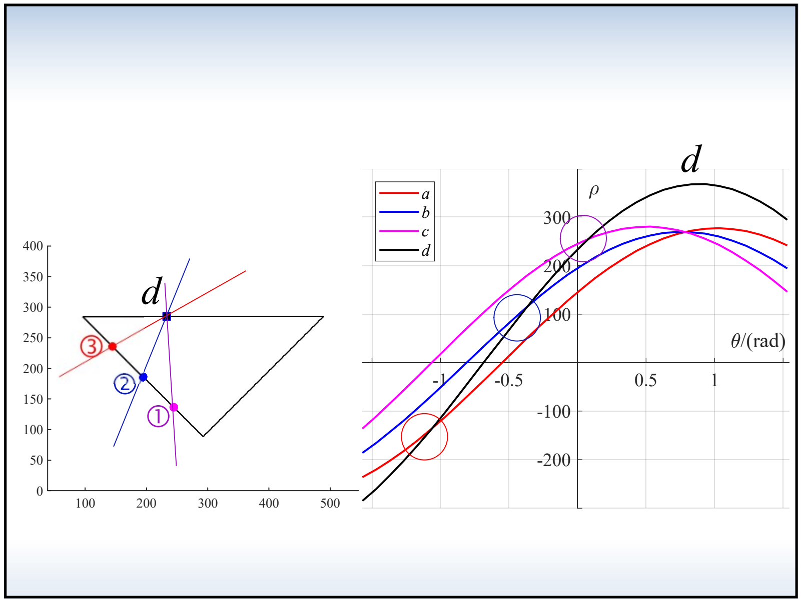




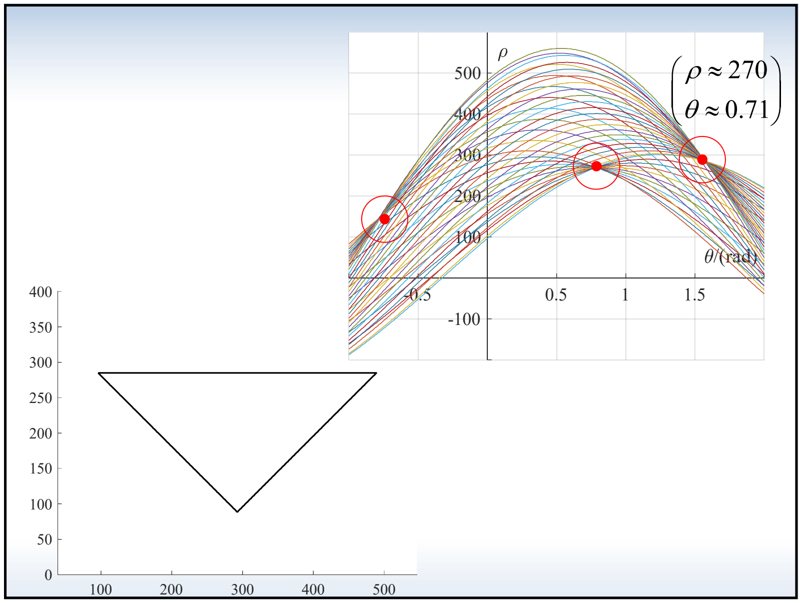
16



17

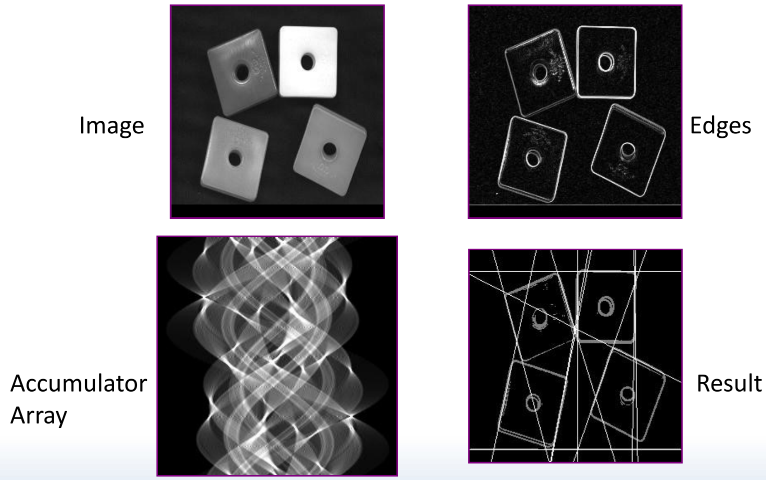


18



19

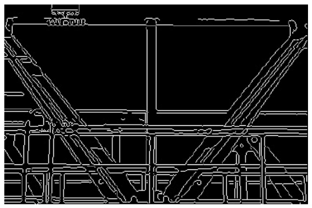
Example: Real objects



20

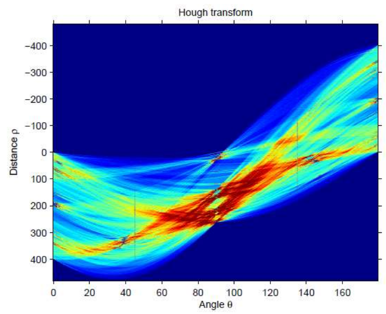
Example

Original Image



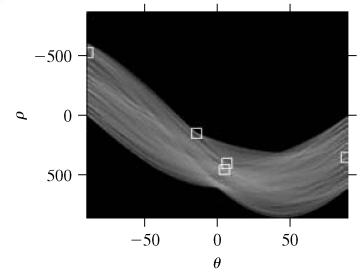
Edge Image

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



Jeppe Jensen, 2007.

21



a b

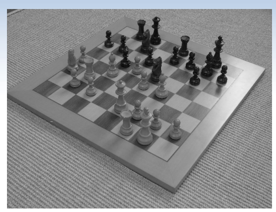
FIGURE 10.11
(a) Hough transform with five peak locations selected.
(b) Line segments corresponding to the Hough transform peaks.

Source: Gonzalez and Woods (2nd edition)

22

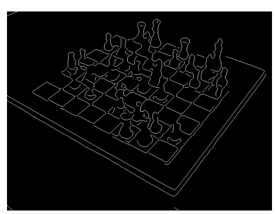
Example (Matlab)

David Young, University of Sussex, 2006



The image is read, converted to gray-level and then displayed in a new figure.

```
im = teachimage('chess1.bmp'); f1 = figure;
imshow(im);
```



Find edges (Canny edge detector; thresholds 0.1 and 0.2, and the smoothing constant 2 to remove the high spatial frequency texture in the background).

```
e = edge(im, 'canny', [0.1 0.2], 2); figure;
imshow(e);
```

Perform the (ρ,θ) Hough transform

```
[h, theta, rho] = hough(e); figure;
imshow(sqrt(h'), [ ]);
```

Find the peaks in the transform

The maximum number of peaks to find is 21, and peaks must not be closer together than 27 bins in rho and 11 bins in theta.

```
p = houghpeaks(h, 21, 'Threshold', 0, 'NhoodSize', [27 11]);
```

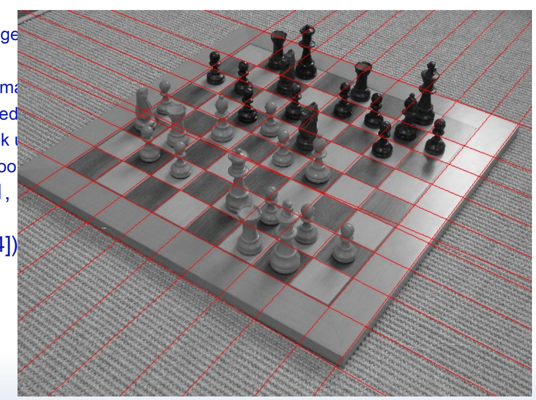
23

Example (Matlab) cont.

Plot the lines on the image

%convert the angles used from degrees to radians (because normal Matlab functions use radians – hough returns in degrees).

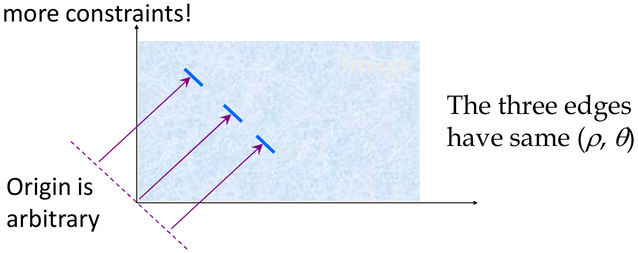
```
theta = (pi/180)*theta;
figure(f1); % original image
[nr, nc] = size(im);
hold on % plot on top of ima
for pr = p' % pr is assigned
    r = rho(pr(1)); % look
    t = theta(pr(2)); % loo
    l = line box(1, nc, 1,
    if ~isempty(l)
        plot(l([1 3]), l([2,4])
    end
end
hold off
```



24

Modifications

- Note that this technique only uses the fact that an edge exists at point (*i*, *j*). What about the orientation of the edge? Need more constraints!

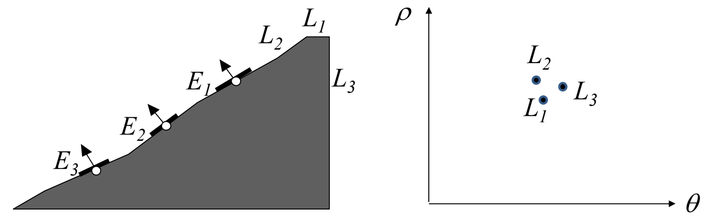


- Use estimate of edge orientation as θ.
- Each edge now maps to a point in Hough space.

25

Gradient Data

- Colinear edges in Cartesian coordinate space now form point clusters in (a, b) parameter space.



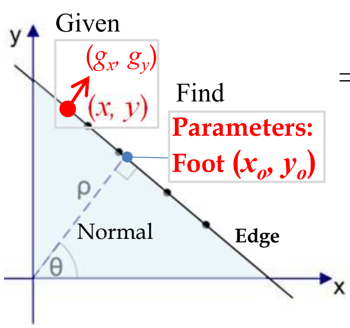
Need an edge 'connected components' algorithm.

- Sort the edges in one Hough cluster
  - rotate edge points according to θ
  - sort them by (rotated) x coordinate
- Look for Gaps using a "max gap" threshold
  - if two edges (in the sorted list) are more than max gap apart, break the line into segments
  - if there are enough edges in a given segment, fit a straight line to the points

26

Another alternative (foot of normal) $y = ax + b$

$G=\sqrt{g_x^2+g_y^2}$, $\tan\theta = \frac{g_y}{g_x} = \frac{y_o}{x_o}$



$$a = \frac{y-y_o}{x-x_o} = -\frac{x_o}{y_o} = -\frac{g_x}{g_y}$$

$y_o = g_y x_o / g$

$$\Rightarrow x_o(x-x_o)+y_o(y-y_o)=0$$

$$\Rightarrow \cancel{x_o}(x-x_o)+\frac{g_y \cancel{x_o}}{g_x}\left(y-\frac{g_y x_o}{g_x}\right)=0$$

$$\Rightarrow x_o\left(1+\frac{g_y^2}{g_x^2}\right)=\left(x+y\frac{g_y}{g_x}\right)$$

Similarly, solving for y_o

$$\boxed{\begin{bmatrix} x_o \\ y_o \end{bmatrix} = v\begin{bmatrix} g_x \\ g_y \end{bmatrix} \quad \text{where } v = \frac{xg_x + yg_y}{g_x^2+g_y^2}}$$

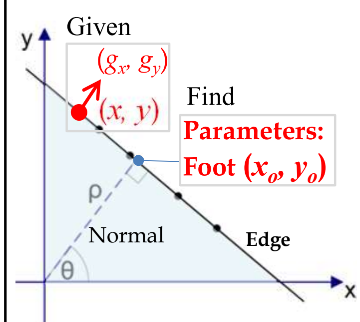
Advantages:
- Eliminate trigonometry functions
- Use of gradient information
- Compute both parameters simultaneously

Note: $\rho = \sqrt{x_o^2+y_o^2}$

27

Another alternative (foot of normal) $y = ax + b$

$G=\sqrt{g_x^2+g_y^2}$, $\tan\theta = \frac{g_y}{g_x} = \frac{y_o}{x_o}$



$$\begin{bmatrix} x_o \\ y_o \end{bmatrix} = v\begin{bmatrix} g_x \\ g_y \end{bmatrix} \quad \text{where } v = \frac{xg_x + yg_y}{g_x^2+g_y^2}$$

$$\theta = \tan^{-1}\frac{g_y}{g_x} = \tan^{-1}\frac{y_o}{x_o}$$

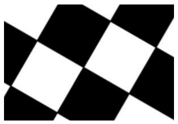
$$\rho = \sqrt{x_o^2+y_o^2} = b\sin\theta$$

$$a = \frac{-x_o}{y_o} \qquad b = \frac{\rho}{\sin\theta}$$

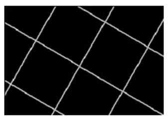
Advantages:
- Eliminate trigonometry functions
- Use of gradient information
- Compute both parameters simultaneously

28

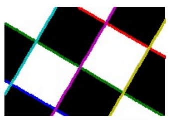
Example
(foot of normal)



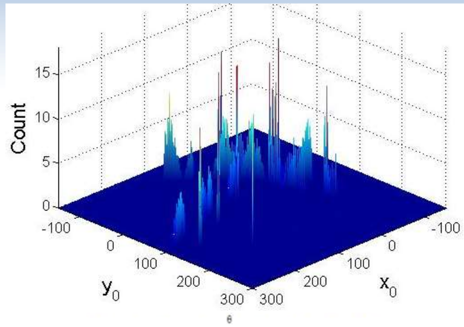
Rotated checker



Gradient



Plot limes on image



	x_0	y_0	a	b	ρ	$\theta°$
Line 1	–58	116	0.500	145	130	26.6
Line 2	–19	40	0.48	49.0	44	25.4
Line 3	25	–50	0.50	–62.5	56	25.6
Line 4	71	32	–2.22	189.5	78	–65.7
Line 5	159	80	–1.99	396.0	178	–63.3
Line 6	246	120	–2.05	624.3	274	–64.0

29

Generalizations

- Hough technique generalizes to any parameterized curve:

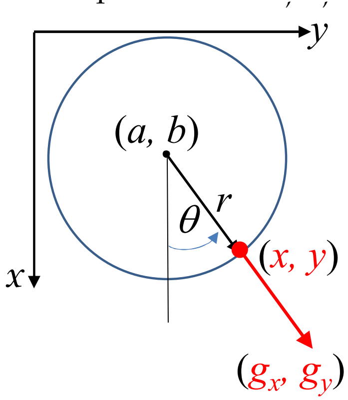
$$f\underbrace{(\mathbf{x},\mathbf{a})}_{\text{Parameter vector (axes in Hough space)}} = 0$$

- Success of technique depends upon the quantization of the parameters:
  - too coarse: maxima 'pushed' together
  - too fine: peaks less defined
- Note that exponential growth in the dimensions of the accumulator array with the the number of curve parameters restricts its practical application to curves with few parameters

30

Hough Transform for finding circles

Three parameters: a , b , r



$$(x-a)^2 + (y-b)^2 = r^2$$
$$a = x - r\cos\theta$$
$$b = y - r\sin\theta$$

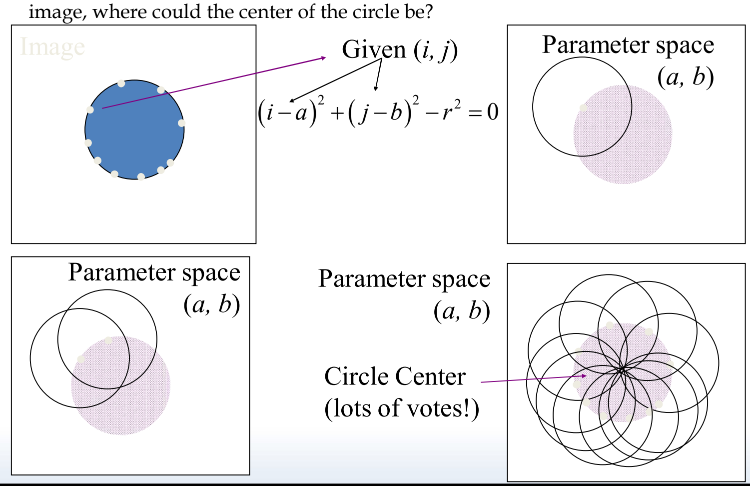
$$G = \sqrt{g_x^2 + g_y^2}, \quad \tan\theta = \frac{g_y}{g_x}$$

$$\cos\theta = \frac{g_x}{|g|} \qquad \sin\theta = \frac{g_y}{|g|}$$

31

Finding a Circle (Assume r is known)

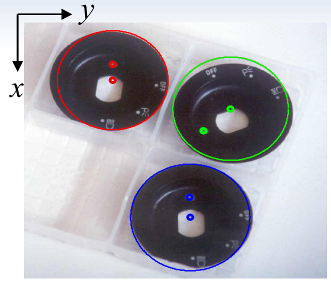
Example: Find the center of a circle given an edge image with no gradient direction information (edge location only). Given an edge point at (x,y) in the image, where could the center of the circle be?



32

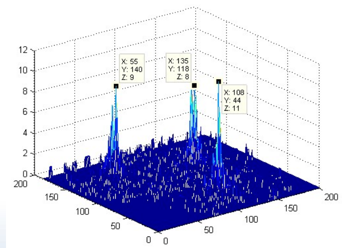
Finding a Circle (Assume *r* is unknown)

- If we don't know r, accumulator array is 3-dimensional.
- Computational complexity is reduced if edge directions are known; many methods.
- As an HT example finding the radius and location of the circles.



Matlab Implementation

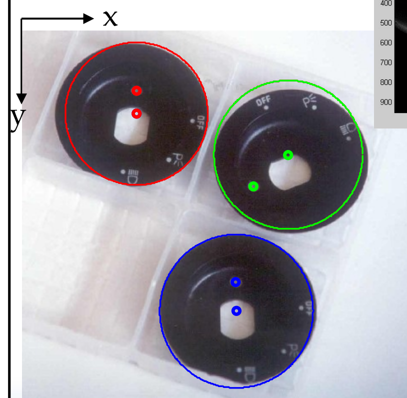
- Import grayscale image
- Find gradient (Sobel)
- Initialize Accumulator (x_c, y_c, r)
- Inspect each pixel,
  1) If gradient exist, calculate G_x and G_y
  2) For each r, calculate x_c and y_c.
  3) Update accumulator cell

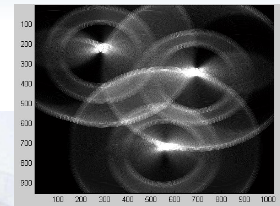


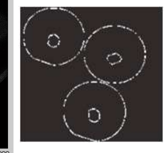
33

Example on HT for circles

Find the radius and location of the circles.



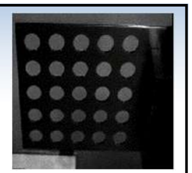


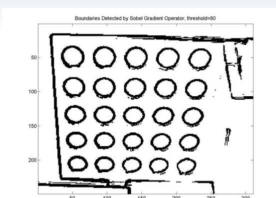


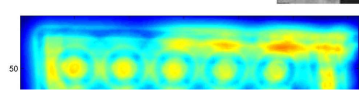
Big Circle	x_c	y_c	r
1	296	216	184
2	688	322	192
3	554	726	198
Small Circ	x_c	y_c	
1	296	156	
2	597	403	
3	551	651	

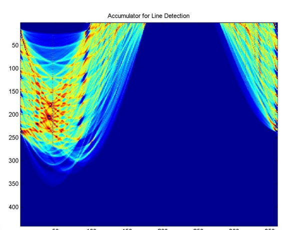
34

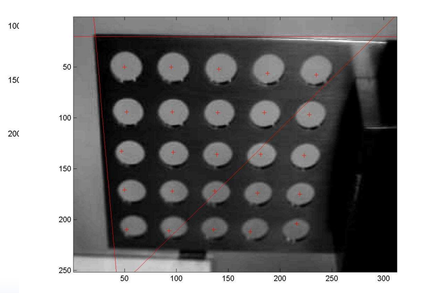
Example on HT for circles











35

Hough Transform for finding ellipses

For illustration, assume that the principal axis of the ellipse is parallel to the x axis

Four parameters (a, b, x_c, y_c): $$\frac{(x-x_c)^2}{a^2}+\frac{(y-y_c)^2}{b^2}=1$$

Take total derivative w. r. t. x: $$\frac{x-x_c}{a^2}+\frac{y-y_c}{b^2}\frac{dy}{dx}=0$$

Using edge operator (g_x/g_y).

$$x-x_c=-(a/b)^2(y-y_c)\tan\phi$$

$$\boxed{x_c=x\pm\frac{a}{\sqrt{1+(b/a)^2/\tan^2\phi}} \qquad y_c=y\pm\frac{b}{\sqrt{1+(a/b)^2\tan^2\phi}}}$$

36

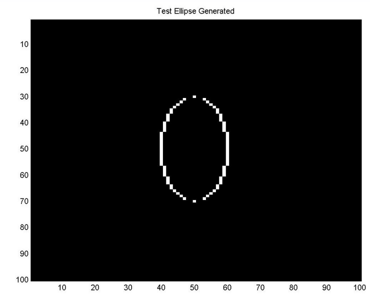
Hough Transform for finding ellipses

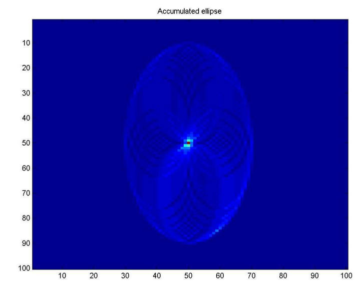
$$x_c = x \pm \frac{a}{\sqrt{1+(b/a)^2/\tan^2\phi}} \qquad y_c = y \pm \frac{b}{\sqrt{1+(a/b)^2\tan^2\phi^2}}$$

Algorithm for applying Hough technique to detect an ellipse from an image:

1. Quantize parameter space between appropriate maximum and minimum values for parameters (a, b, x_c, y_c).
2. Form an accumulator array whose elements are initially zero.
3. For every two discrete points of x and y, solve for the parameters .
4. Increment the point in parameter space .
5. Local maxima in the accumulator array now correspond to collinear points in the image array. The values of the accumulator array provide a measure of points on the ellipse.

Hough Transform for finding ellipses

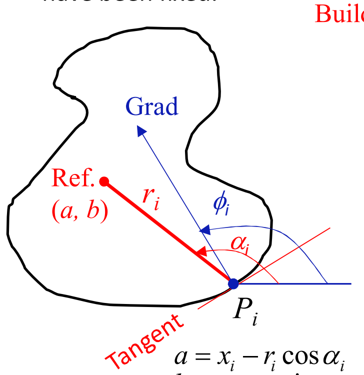




Generalizing the HT

The HT can be extended to finding curves that do not have a simple analytic form. For illustration, assume that scaling and rotation have been fixed.

Build a **Table** to serve as **Template**.



$$a = x_i - r_i \cos\alpha_i$$
$$b = y_i - r_i \sin\alpha_i$$

1. Pick a reference point (a, b)
2. For i = 1, … , n :
   a) Draw segment to P_i on the boundary.
   b) Measure its length r_i, and its orientation α_i.
   c) Write the ref coordinates (a, b) as a function of r_i and a_i
   d) Record the gradient orientation ϕ_i at P_i.
3. Build a table with the data, indexed by ϕ_i .

Generalized HT Algorithm for objects with no analytical form

R-Table (pre-stored template):	
ϕ_1	$(r_1^1,\alpha_1^1),(r_2^1,\alpha_2^1),\ldots(r_{n1}^1,\alpha_{n1}^1)$
ϕ_2	$(r_1^2,\alpha_1^2),(r_2^2,\alpha_2^2),\ldots(r_{n2}^2,\alpha_{n2}^2)$
…	
ϕ_m	$(r_1^m,\alpha_1^m),(r_2^2,\alpha_2^2),\ldots(r_{nm}^m,\alpha_{nm}^m)$

1. Form a R-Table
2. Initialize Accumulator (x, y). of possible reference point (a, b),
3. From each pixel point, do the following:
   a) Compute ϕ
   b) Calculate possible centers
   c) Increment Accumulator.
4. Increment the point in parameter space .
5. Local maxima in the accumulator array now correspond to collinear points in the object shape.

Find a **rotated, scaled** and **translated** version of the curve:

1. Form a H accumulator array (x, y, S, θ) of possible reference points (a, b), scaling factor S and rotation angle θ.
2. For each edge (x, y) in the image:
   1. Compute $\phi(x,y)$
   2. For each (r, a) corresponding to $\phi(x,y)$ do:

      For each S and θ:
      $$a = x_i + r(\phi) S \cos\left[\alpha(\phi)+\theta\right]$$
      $$b = y_i + r(\phi) S \sin\left[\alpha(\phi)+\theta\right]$$
      $$H(a, b, S, \theta)\text{ ++}$$
3. Find maxima of H.

41

Summary of Hough Transform (HT)

- HT is a "voting" scheme: points vote for a set of parameters describing a line or curve.
- The more votes for a particular set: the more evidence that the corresponding curve is present in the image.
- Can detect MULTIPLE curves in one shot.
- Computational cost increases with the number of parameters describing the curve.

42

Hough Transform (HT) Examples

Point	x_i	y_i	g_x	g_y
A	0	2	0	1
B	0.5	1.5	1	1
C	2	0	1	0
D	0.5	1.94	0.5	1.94

Points (A, B, D) are in a line, and points (A, C, D) are in a circle.

1) Use ρ-θ Hough Transformation on A, B, D.
2) Use Hough transforms on points A, C, D to find the parameters of the circle (x_c, y_c, R).

43

Hough Transform (HT) Examples

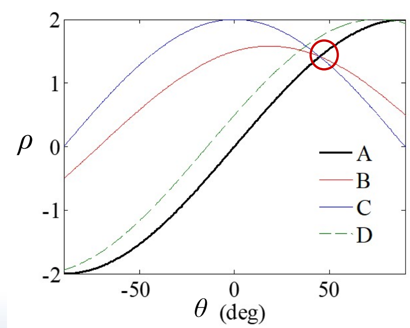
Point	x_i	y_i	g_x	g_y
A	0	2	0	1
B	0.5	1.5	1	1
C	2	0	1	0
D	0.5	1.94	0.5	1.94

Points (A, B, D) are in a line,

1) Use ρ-θ Hough Transformation on A, B, D.

$$\rho = x\cos\theta + y\sin\theta$$
$$A: \rho = 2\sin\theta$$
$$B: \rho = 0.5\cos\theta + 1.5\sin\theta$$
$$C: \rho = 2\cos\theta$$
$$D: \rho = 0.5\cos\theta + 1.94\sin\theta$$

$(\theta, \rho) = (45^\circ, 1.414)$



44

Hough Transform (HT) Examples

Point	x_i	y_i	g_x	g_y
A	0	2	0	1
B	0.5	1.5	1	1
C	2	0	1	0
D	0.5	1.94	0.5	1.94

Points (A, C, D) are in a circle.

2) Use Hough transforms on points A, C, D to find the circle parameters (x_c, y_c, R).

$$x_c = x - R\frac{g_x}{\sqrt{g_x^2 + g_y^2}}, \quad y_c = y - R\frac{g_y}{\sqrt{g_x^2 + g_y^2}}$$

Accumulator matrix

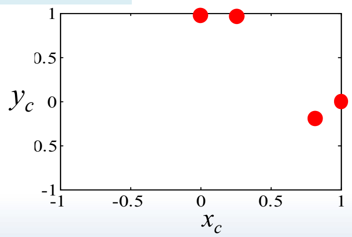
***R*=1:**

Point A: $x_c = 0$, $y_c = 1$,

Point B: $x_c = 0.793$, $y_c = -0.207$,

Point C: $x_c = 1$, $y_c = 0$,

Point D: $\boldsymbol{x_c} = 0.25$, $y_c = 0.969$



Hough Transform (HT) Examples

Point	x_i	y_i	g_x	g_y
A	0	2	0	1
B	0.5	1.5	1	1
C	2	0	1	0
D	0.5	1.94	0.5	1.94

Points (A, C, D) are in a circle.

2) Use Hough transforms on points A, C, D to find the circle parameters (x_c, y_c, R).

$$x_c = x - R\frac{g_x}{\sqrt{g_x^2 + g_y^2}}, \quad y_c = y - R\frac{g_y}{\sqrt{g_x^2 + g_y^2}}$$

Accumulator matrix

***R*=2:**

Point A: $x_c = 0$, $y_c = 0$,

Point B: $x_c = 0.086$, $y_c = -0.914$,

Point C: $x_c = 0$, $y_c = 0$,

Point D: $\boldsymbol{x_c} \approx 0$, $y_c = \approx 0$,