CS 6476 Computer Vision:

7. Fitting & Hough Transform

Humphrey Shi

9/12/2024

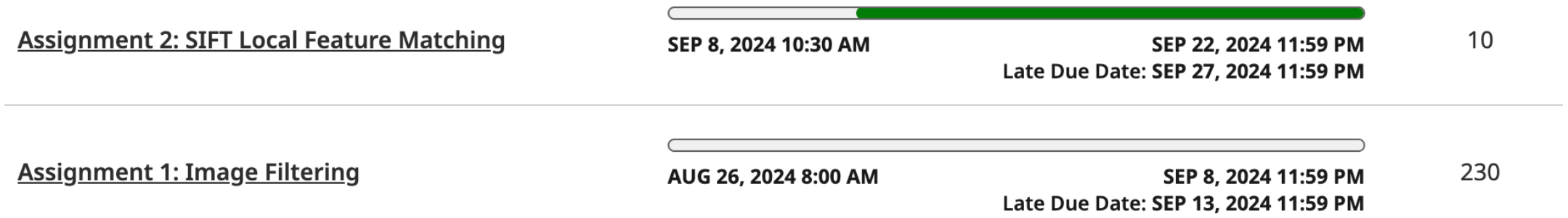
The follow slides are mostly based on the lecture slides of Judy Hoffman and Svetlana Lazebnik, with credits to Devi, Kristen, etc.

Outline

- Logistics & Course Recap
- Fitting & Hough Transform

Logistic & Course Recap

Assignments

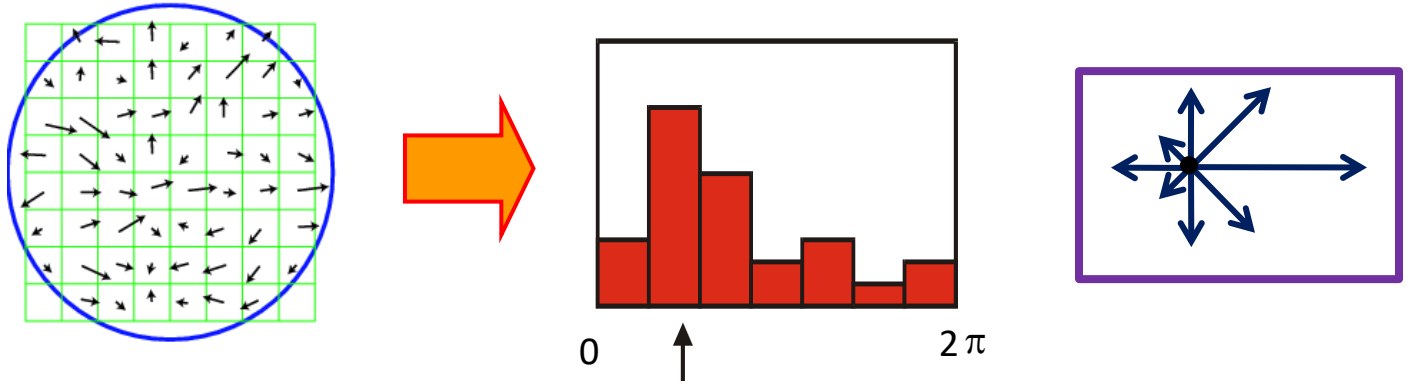


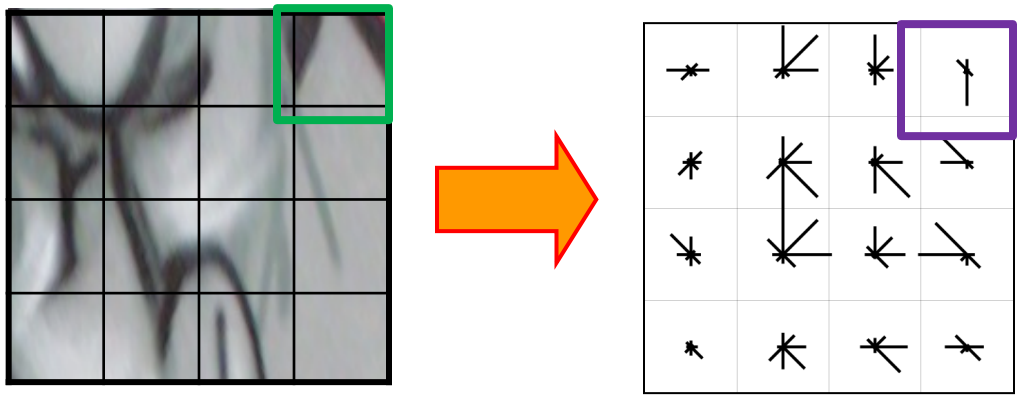
- You have five "late days" for the whole course without penalty.
- You will lose 10% each day for late projects, and late submissions will not be accepted after 1 week.

- There are extra credit points for several projects and the total score for those can be up to 110% of 14pts.

Recap: SIFT descriptor

- Use histograms to bin pixels within sub-patches according to their gradient orientation.





Why subpatches?

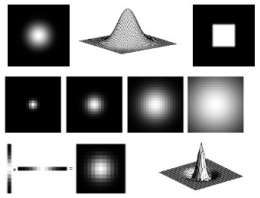
Why does SIFT have some illumination invariance?

Why scale/rotation invariant?

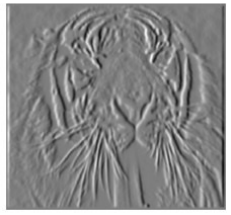
Fitting & Hough Transform

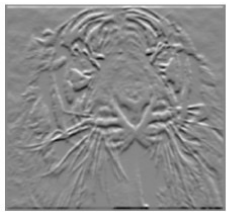
Our Vision Journey So Far

- Image Filtering

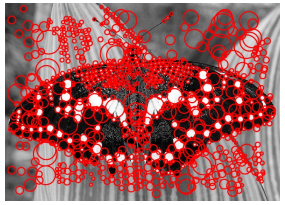


- Edge Detection





- Corner/Blob Detection

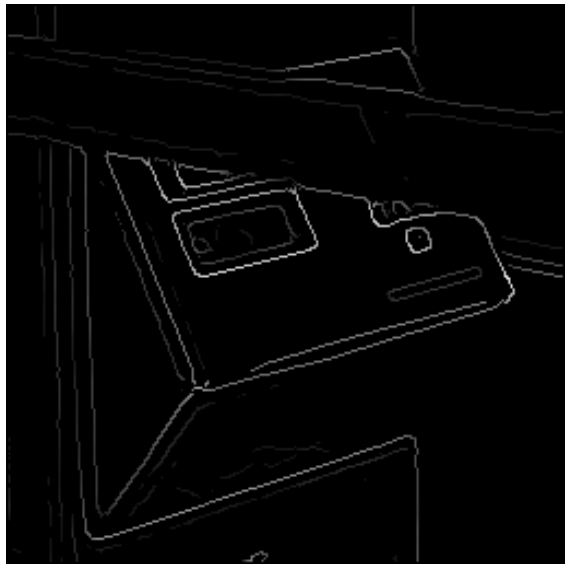
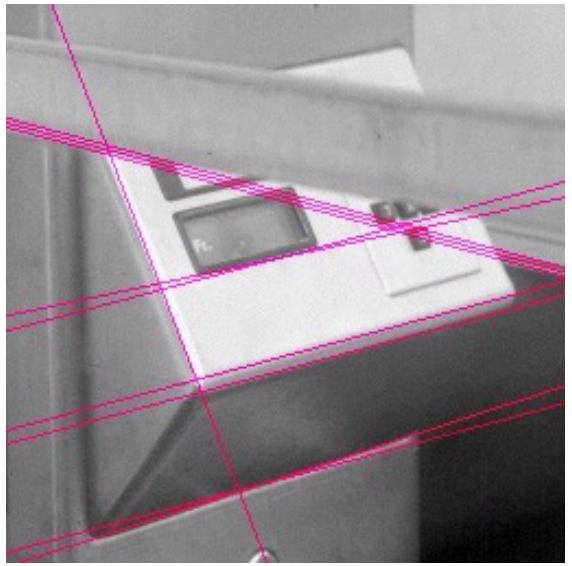


- SIFT

- What should we do next?

Fitting

- Want to associate a model with observed features

For example, the model could be a line, a circle, or an arbitrary shape.

[Fig from Marszalek & Schmid, 2007]

Fitting: Main idea

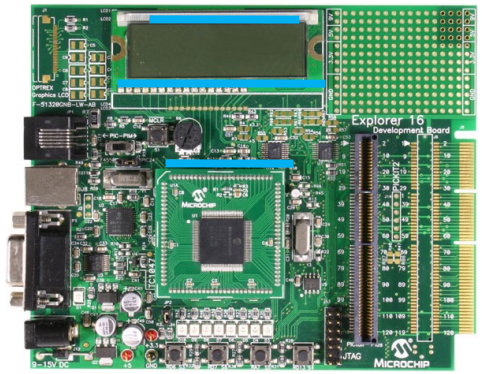
- Choose **a parametric model** to represent **a set of features**

- Membership criterion is not local
  - Can't tell whether a point belongs to a given model just by looking at that point
- Three main questions:
  - What model represents this set of features best?
  - Which of several model instances gets which feature?
  - How many model instances are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

Slide credit: L. Lazebnik

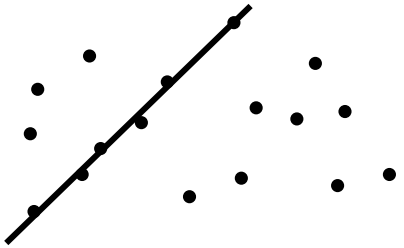
Example: Line Fitting

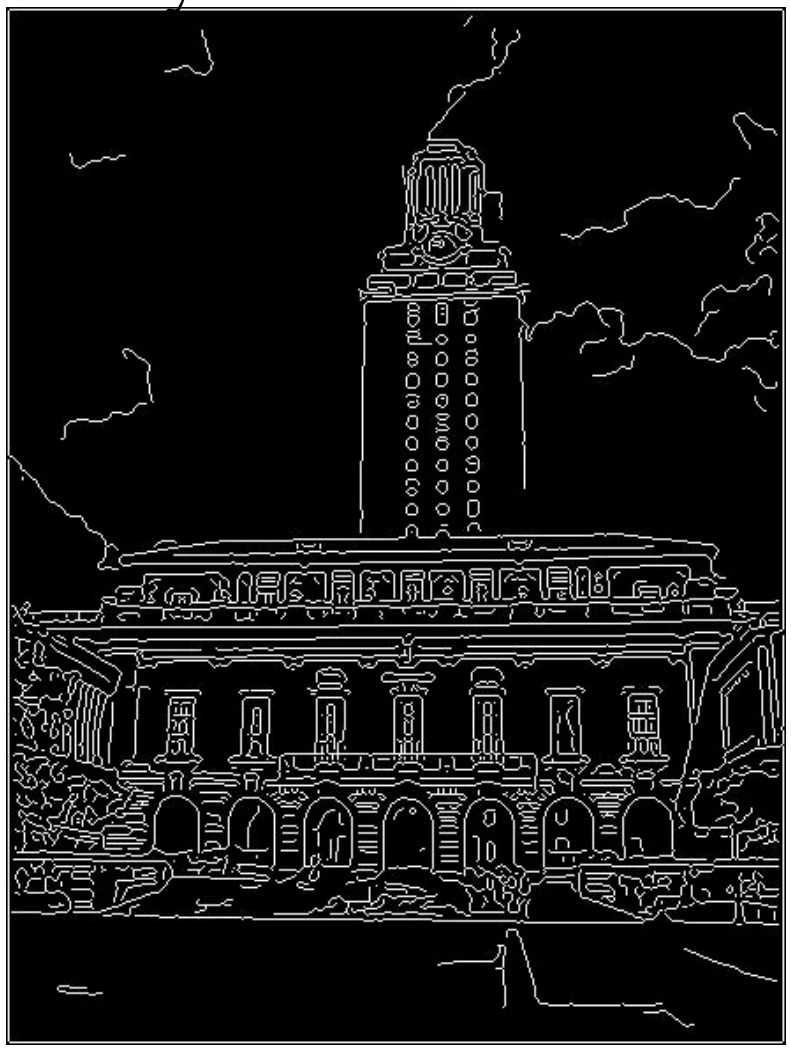
- Why fit lines? Many objects characterized by presence of straight lines



Wait, why aren't we done just by running edge detection?

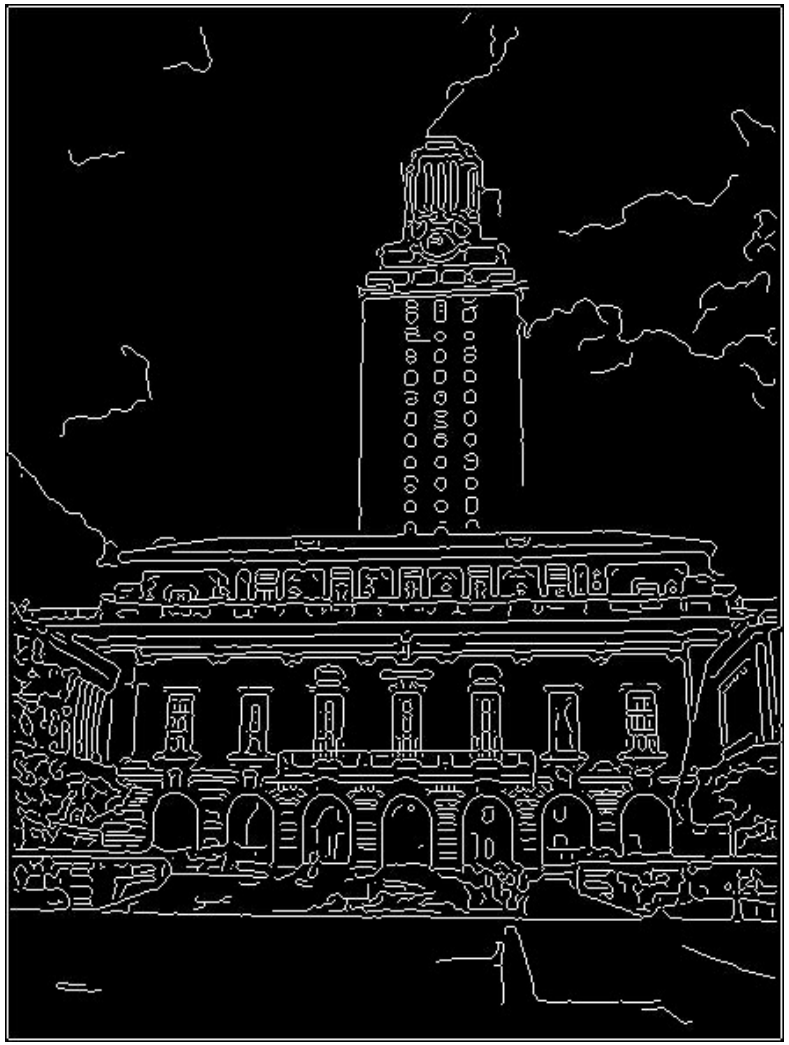
Difficulty of Line Fitting

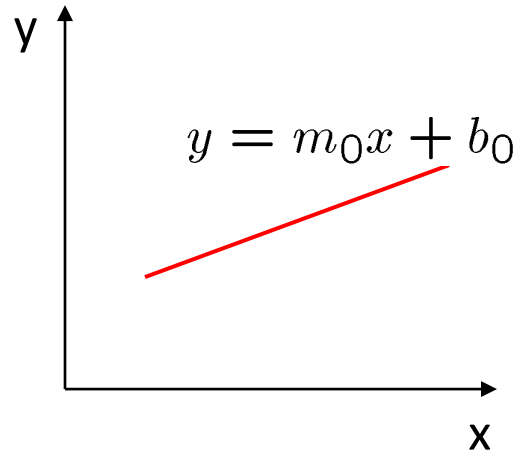




- **Extra** edge points (clutter), multiple models:
  - which points go with which line, if any?
- Only some parts of each line detected, and some parts are **missing:**
  - how to find a line that bridges missing evidence?
- **Noise** in measured edge points, orientations:
  - how to detect true underlying parameters?

How to Fit Lines to Images?





What are the challenges if we do this naively?

Voting

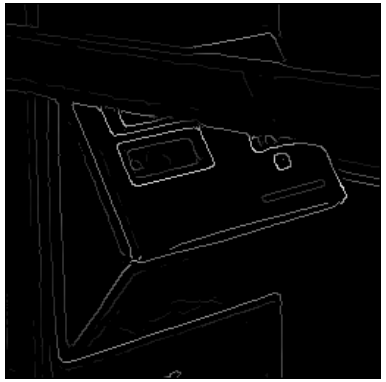
- It's not feasible to check all combinations of features by fitting a model to each possible subset.
- **Voting** is a general technique where we let the features *vote for all models that are compatible with it*.
  - Cycle through features, cast votes for model parameters.
  - Look for model parameters that receive a lot of votes.
- Noise & clutter features will cast votes too, *but* typically their votes should be inconsistent with the majority of "good" features.

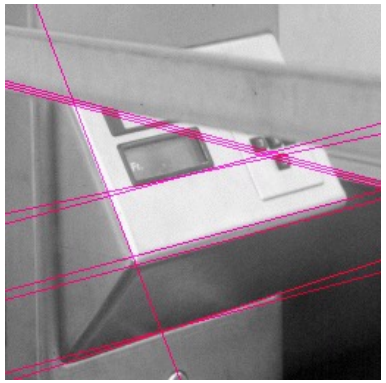
Fitting Lines: Hough transform

- Given points that belong to a line, what is the line?
- How many lines are there?
- Which points belong to which lines?
- **Hough Transform** is a voting technique that can be used to answer all of these questions.

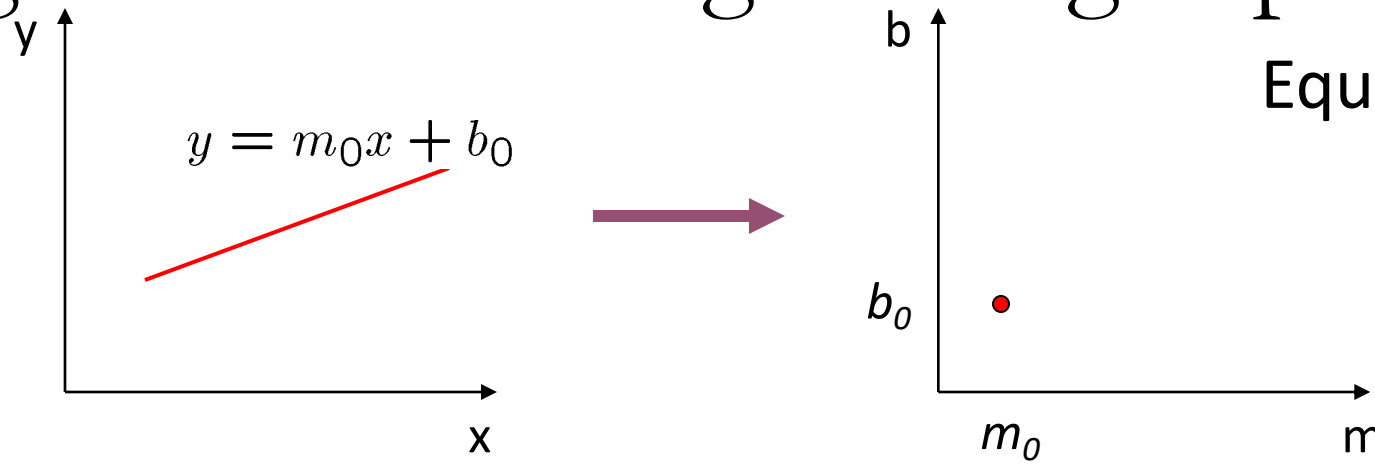
  Main idea:

  1. Record vote for each possible line on which each edge point lies.
  2. Look for lines that get many votes.





Finding lines in an image: Hough space



Equation of a line?

y = mx + b

image space

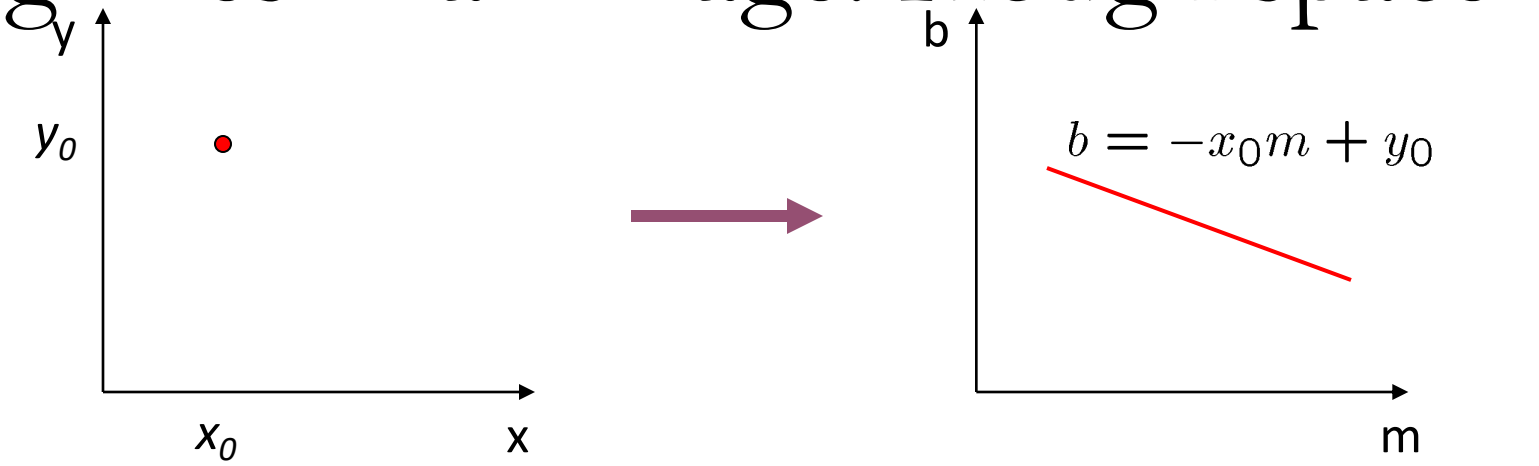
Hough (parameter) space

Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b

Slide credit: Steve Seitz

Finding lines in an image: Hough space

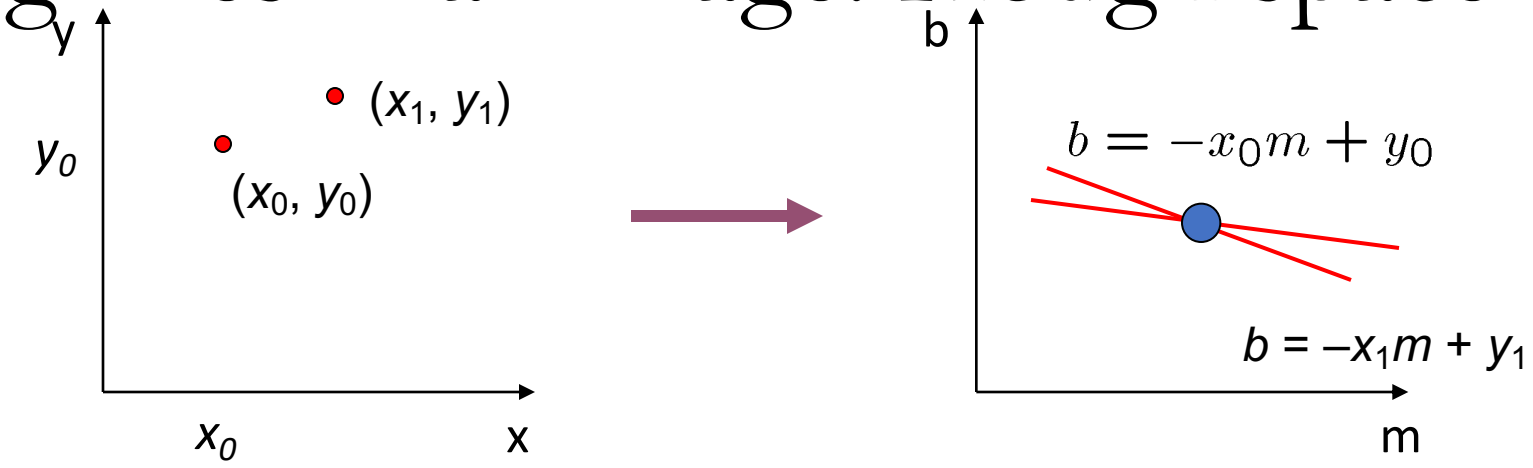


Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b
- What does a point (x_0, y_0) in the image space map to?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - this is a line in Hough space

Slide credit: Steve Seitz

Finding lines in an image: Hough space



What are the line parameters for the line that contains both (x_0, y_0) and (x_1, y_1)?

- It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$

Slide credit: Kristen Grauman

Finding lines in an image: Hough algorithm

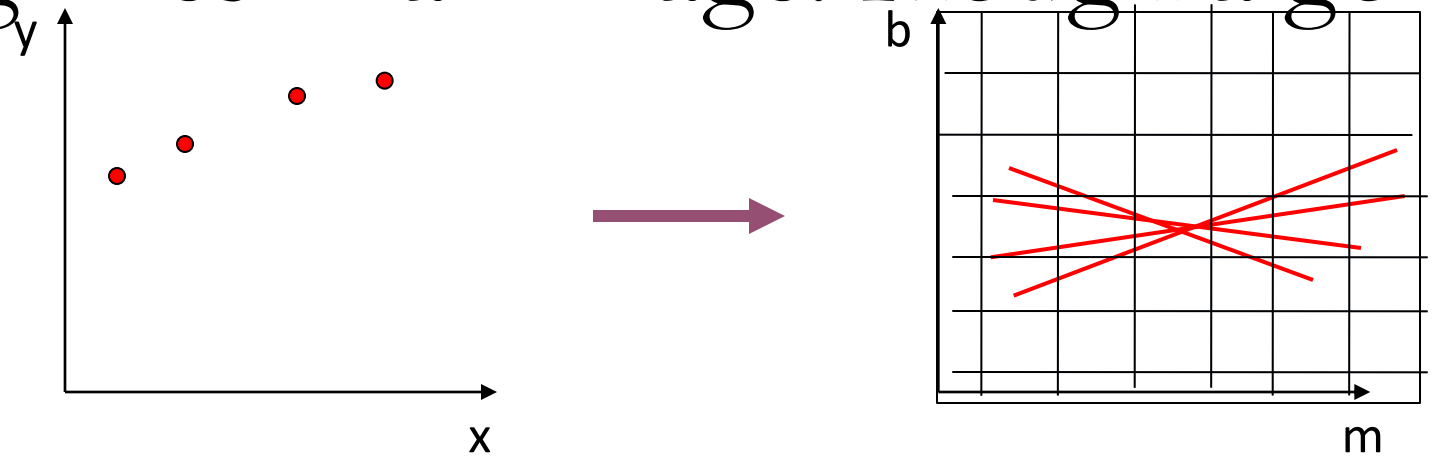


image space Hough (parameter) space

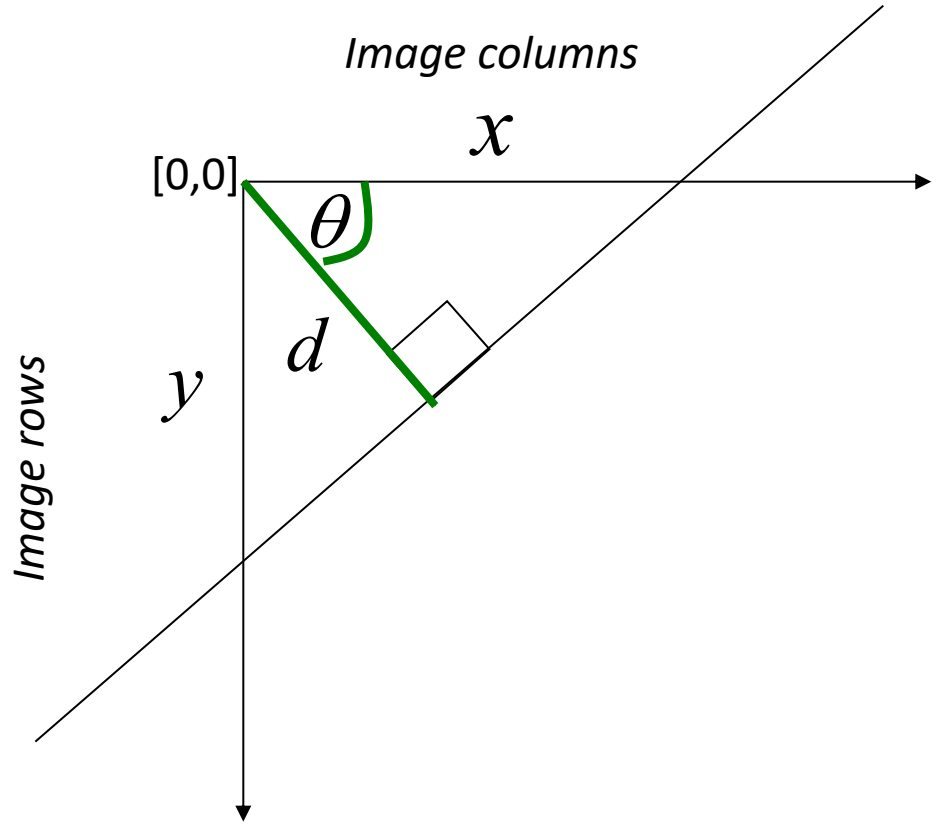
How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?

- Let each edge point in image space *vote* for a set of possible parameters in Hough space
- Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Slide credit: Kristen Grauman

Polar representation for lines

Issues with usual (*m,b*) parameter space: can take on infinite values, undefined for vertical lines.



d: perpendicular distance from line to origin

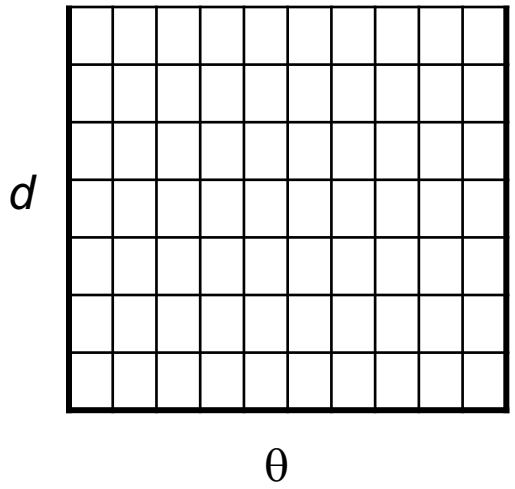
θ: angle the perpendicular makes with the x-axis

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

Point in image space → sinusoid segment in Hough space

Hough transform algorithm

H: accumulator array (votes)



Using the polar parameterization:

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

Basic Hough transform algorithm

1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image

   for θ = [θ_{min} to θ_{max}] // some quantization

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

   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by

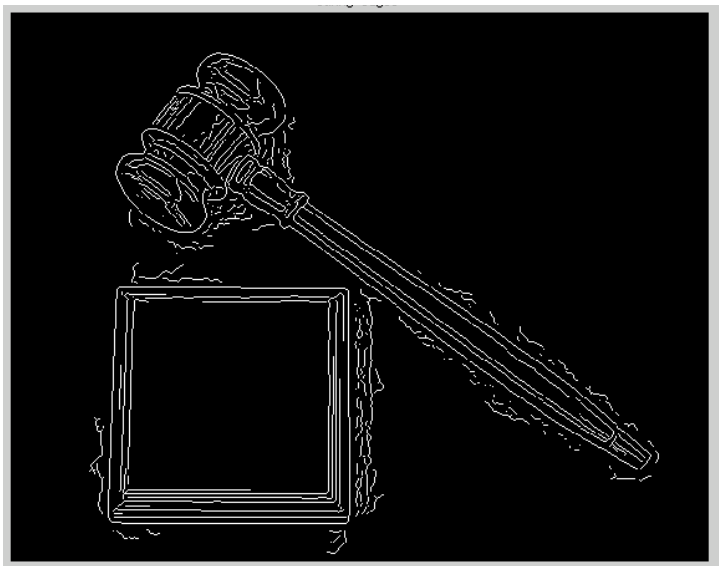
$$x\cos\theta + y\sin\theta = d$$

Source: Steve Seitz

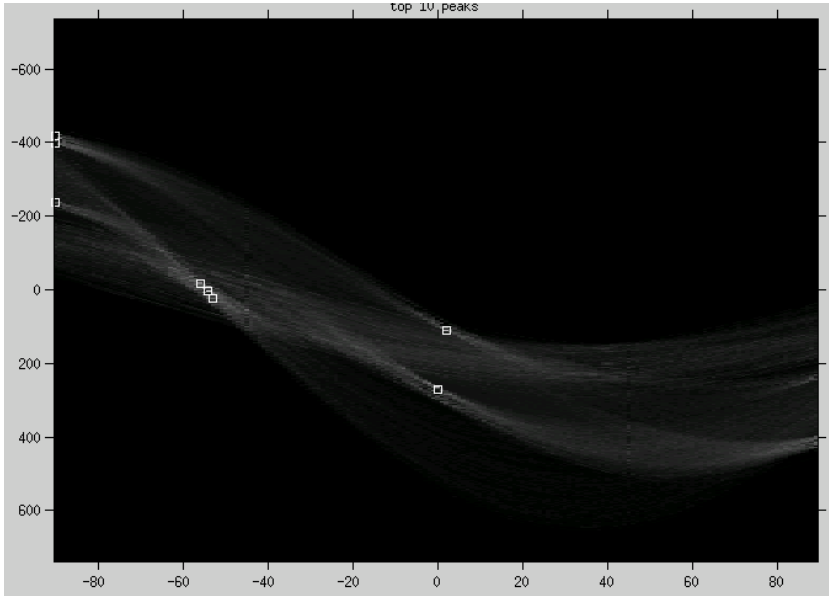
Hough Transform Example

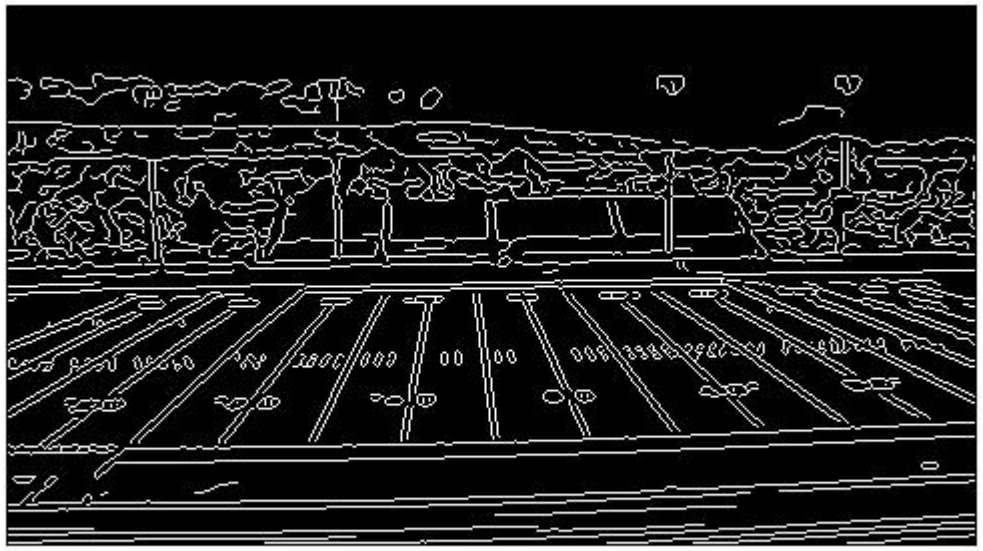
Original image

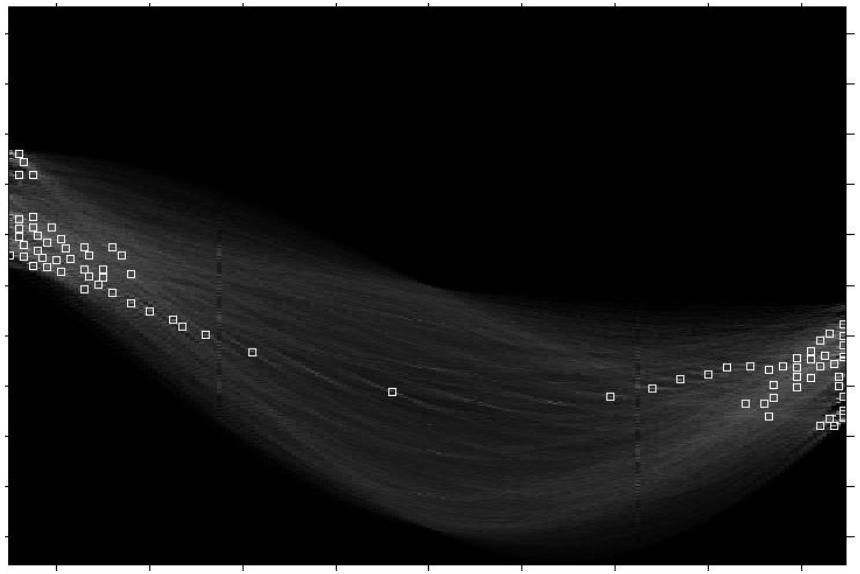
Canny edges

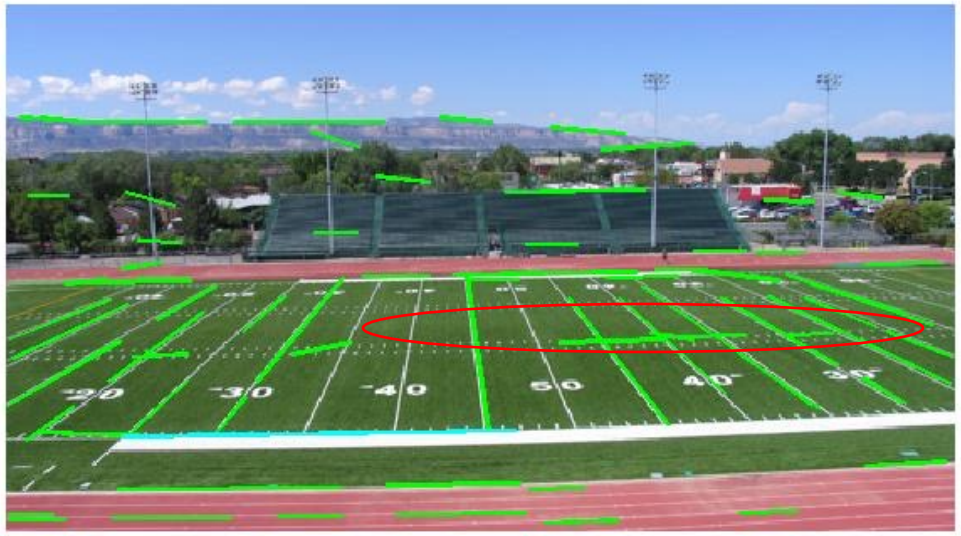


Vote space and top peaks









Showing longest segments found

Kristen Grauman

Impact of noise on Hough

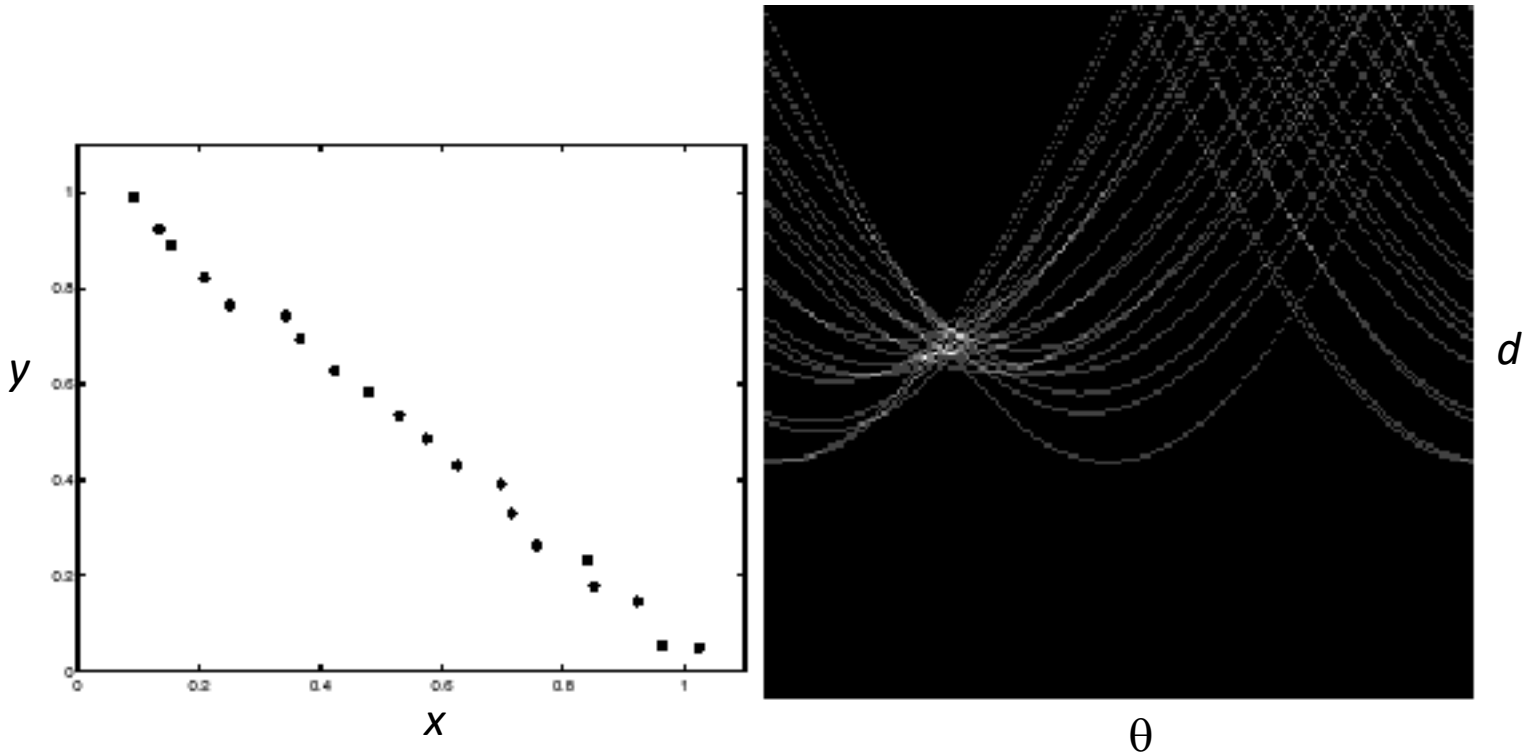


Image space
edge coordinates

Votes

What difficulty does this present for an implementation?

Slide credit: Kristen Grauman

Impact of noise on Hough

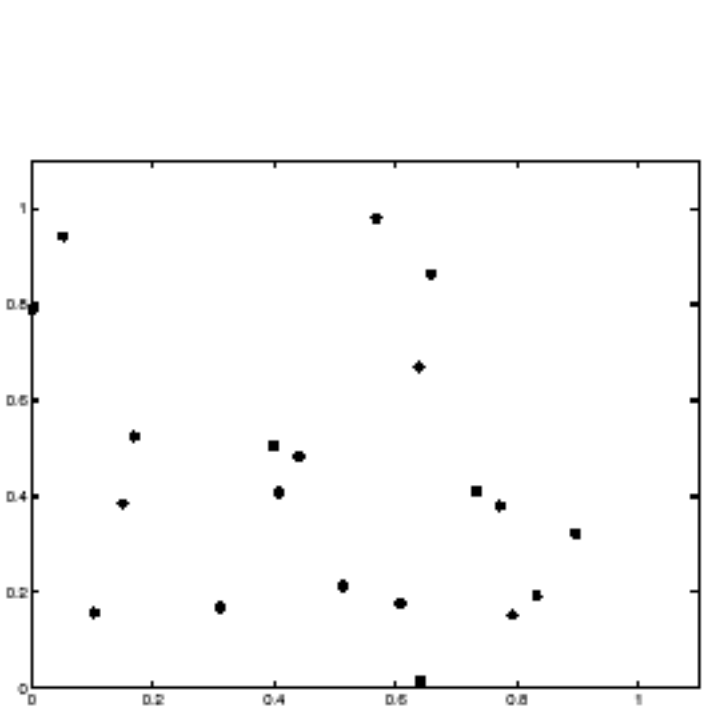
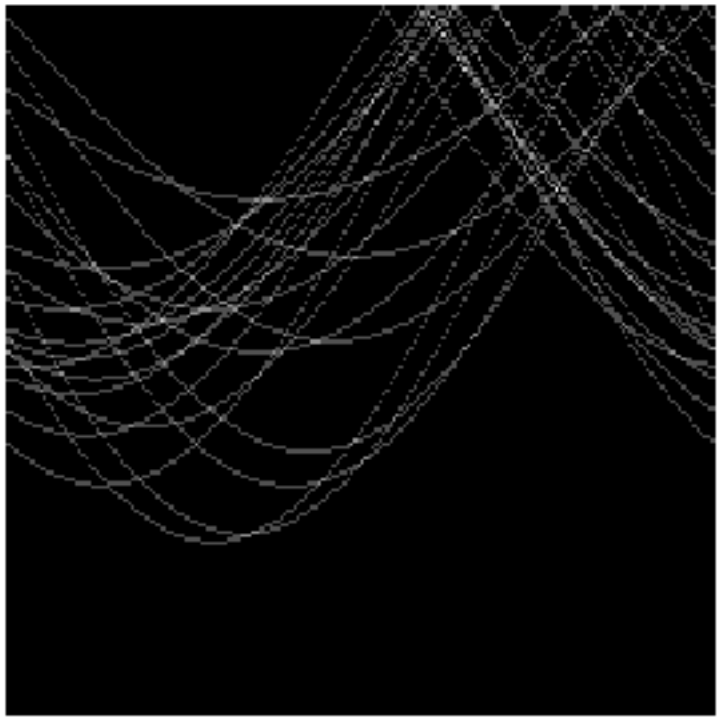


Image space
edge coordinates

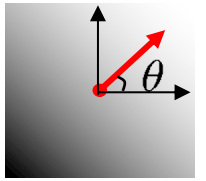


Votes

Here, everything appears to be “noise”, or random edge points, but we still see peaks in the vote space.

Slide credit: Kristen Grauman

Extensions



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

Extension 1: Use the image gradient

1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image
   θ = gradient at (x,y)
   $x \cos\theta + y \sin\theta = d$
   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by $x \cos\theta + y \sin\theta = d$

Reduces degree of freedom

Extension 2

- give more votes for stronger edges: H[d, θ] += magnitude(gradient)

Extension 3

- change the sampling of (d, θ) to give more/less resolution

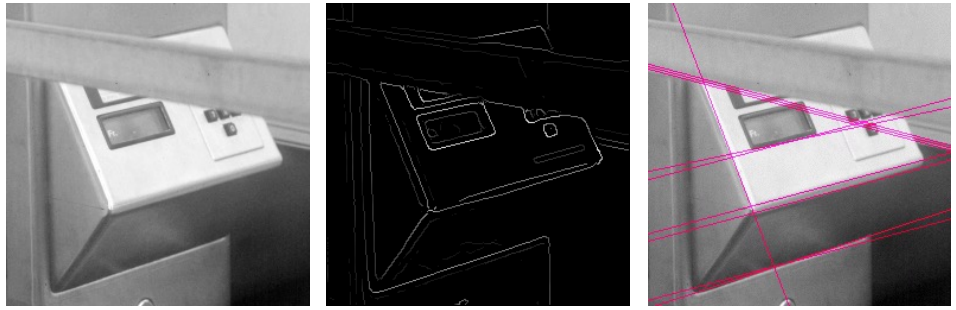
Extension 4

- The same procedure can be used with circles, squares, or any other shape…

Slide credit: Kristen Grauman

How to fit Circles or any Shapes?

- Want to associate a model with observed features



[Fig from Marszalek & Schmid, 2007]

For example, the model could be a line, a circle, or an arbitrary shape.

Hough transform for circles

- Circle: center (a,b) and radius r

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

Equation of circle?

- For a fixed radius r

Equation of set of circles that all pass through a point?

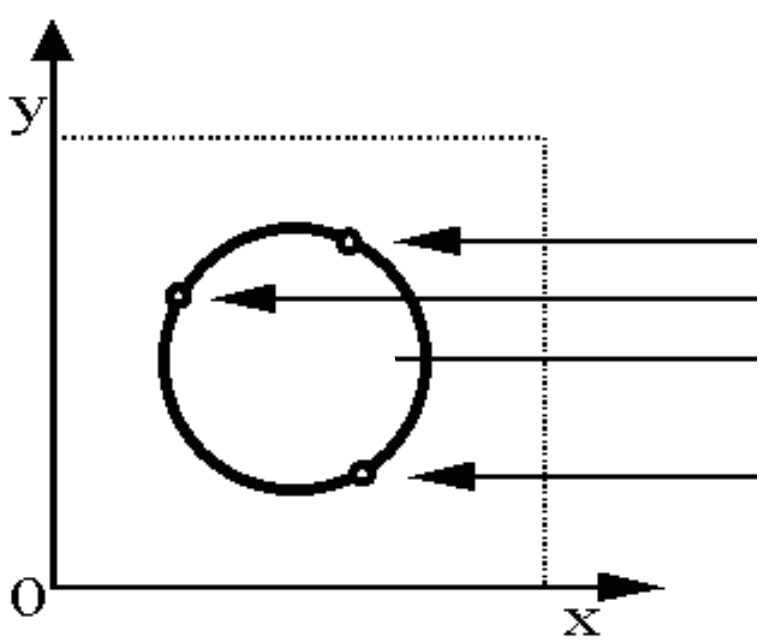
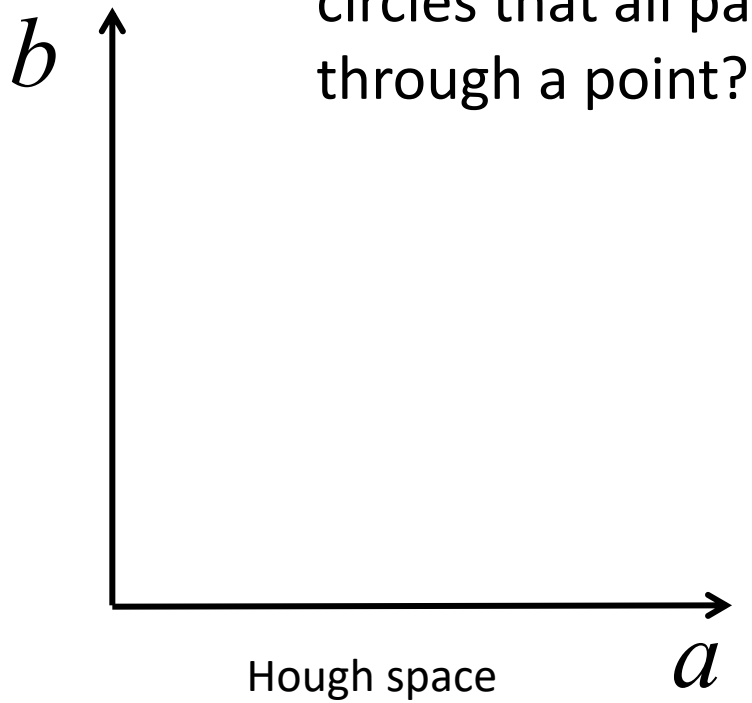


Image space



Hough space

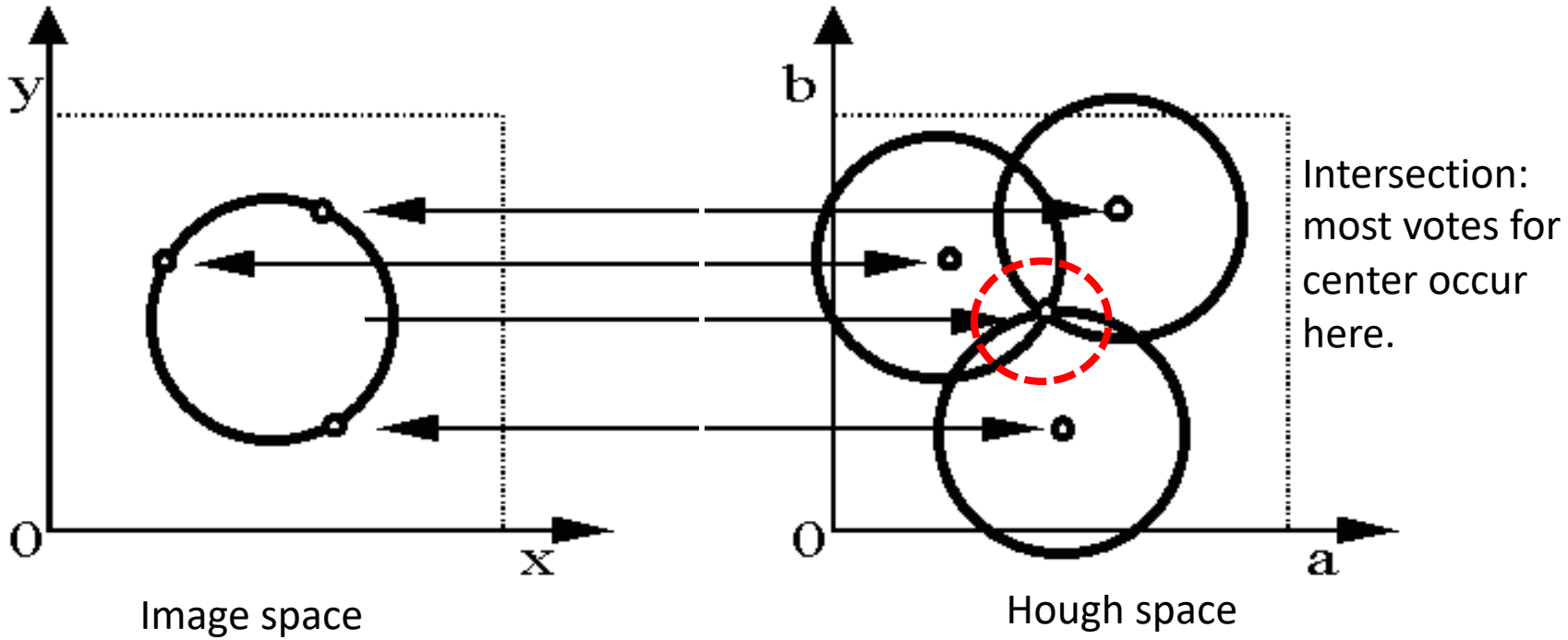
Adapted by Devi Parikh from: Kristen Grauman

Hough transform for circles

- Circle: center (a,b) and radius r

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

- For a fixed radius r

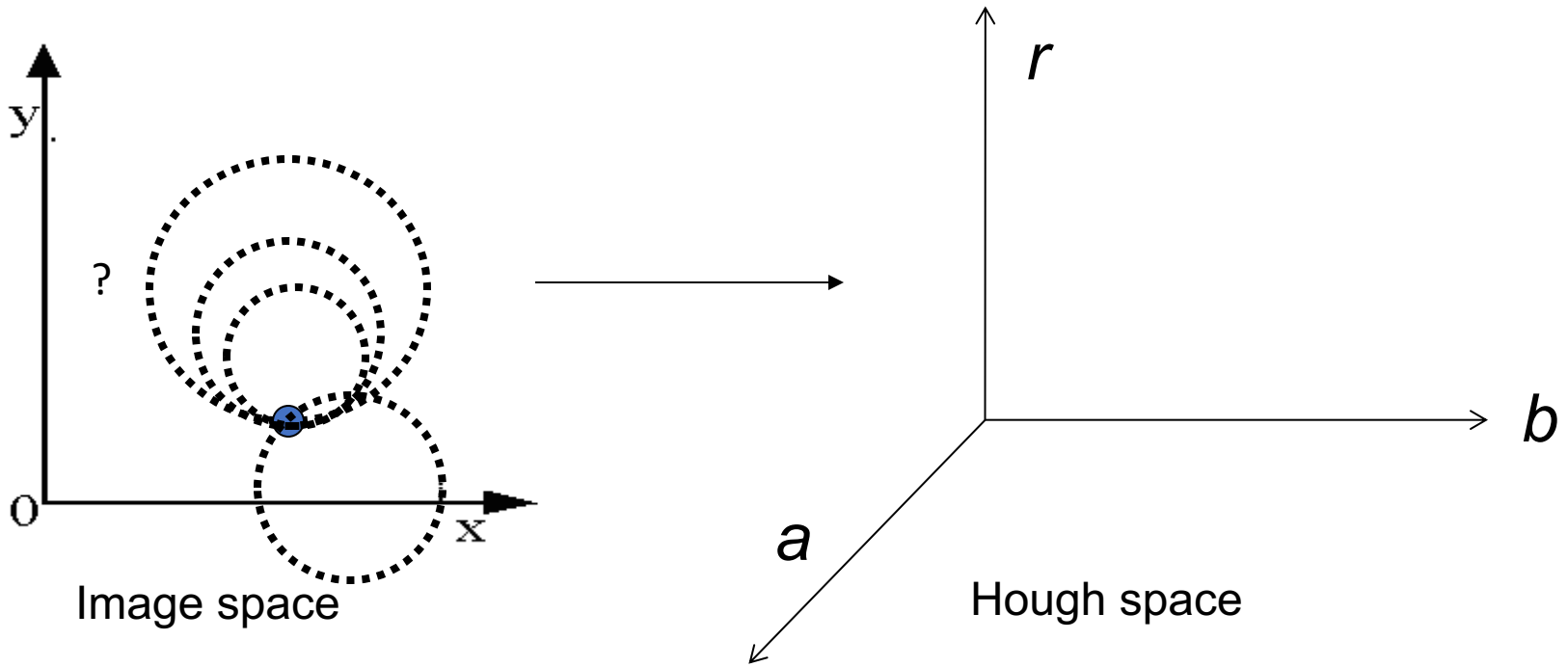


Hough transform for circles

- Circle: center (a,b) and radius r

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

- For an unknown radius r



Hough transform for circles

- Circle: center (a,b) and radius r

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

- For an unknown radius r

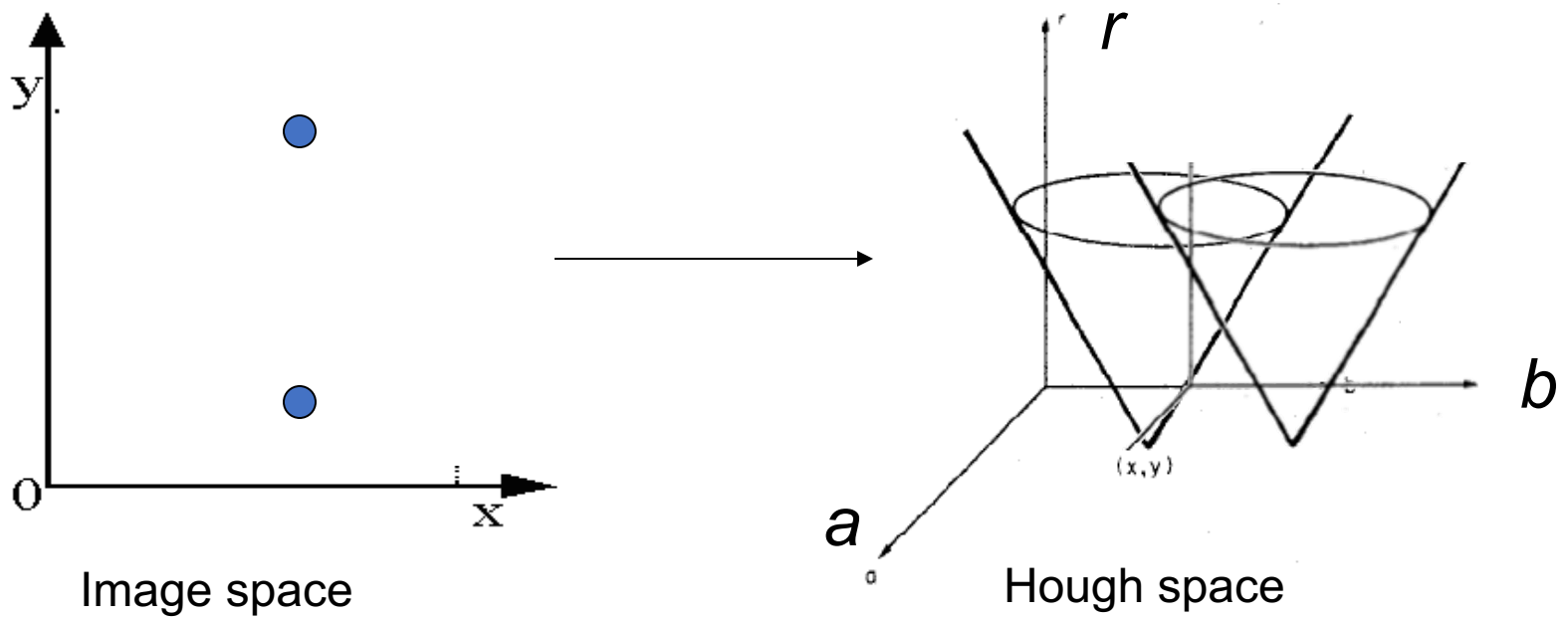


Image space Hough space

Hough transform for circles

- Circle: center (a,b) and radius r

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

- For an unknown radius r, **known** gradient direction

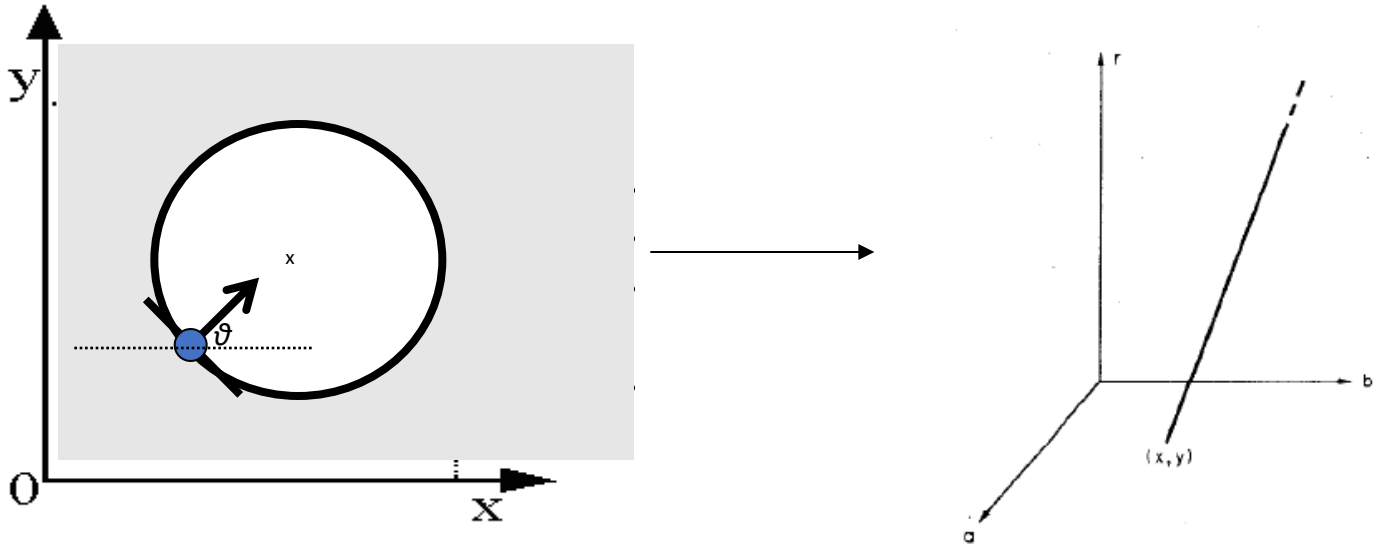


Image space

Hough space

Hough transform for circles

For every edge pixel (x,y) :

  For each possible radius value r:

    For each possible gradient direction θ:

      // *or use estimated gradient at* (x,y)

        $a = x - r\cos(\theta)$ // column

        $b = y - r\sin(\theta)$ // row

        H[a,b,r] += 1

  end

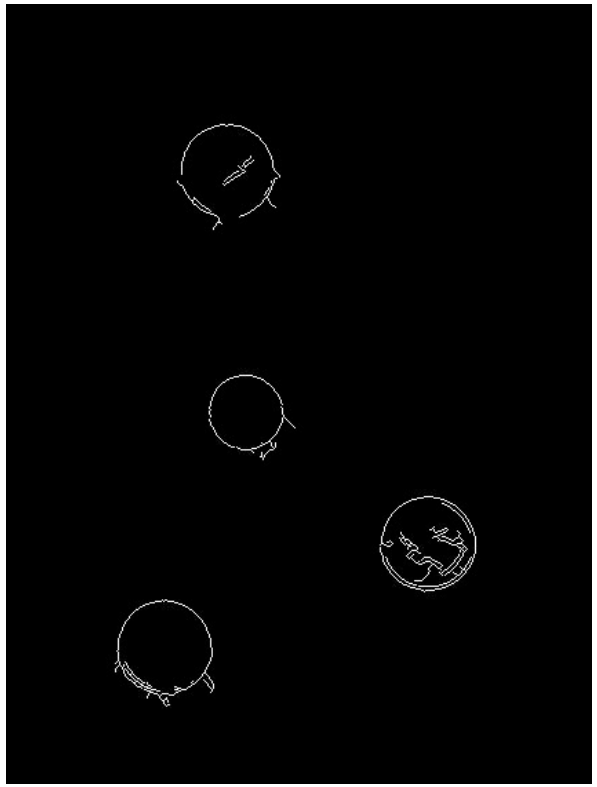
end

- Check out online demo : http://www.markschulze.net/java/hough/

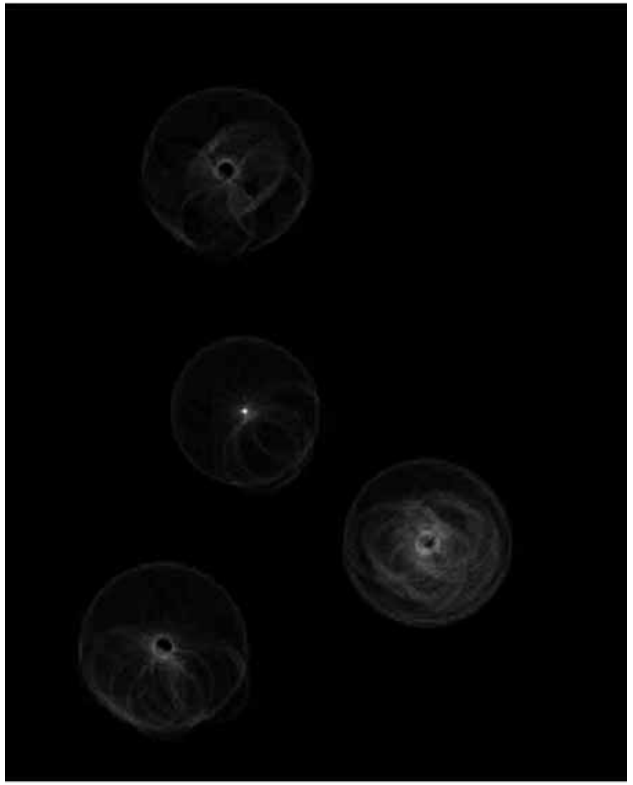
Example: detecting circles with Hough

Original

Edges



Votes: Penny

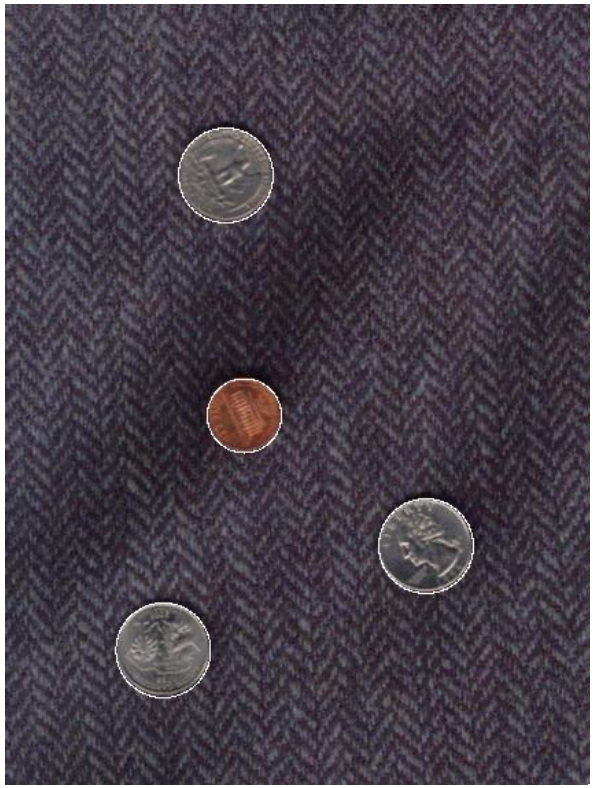


Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

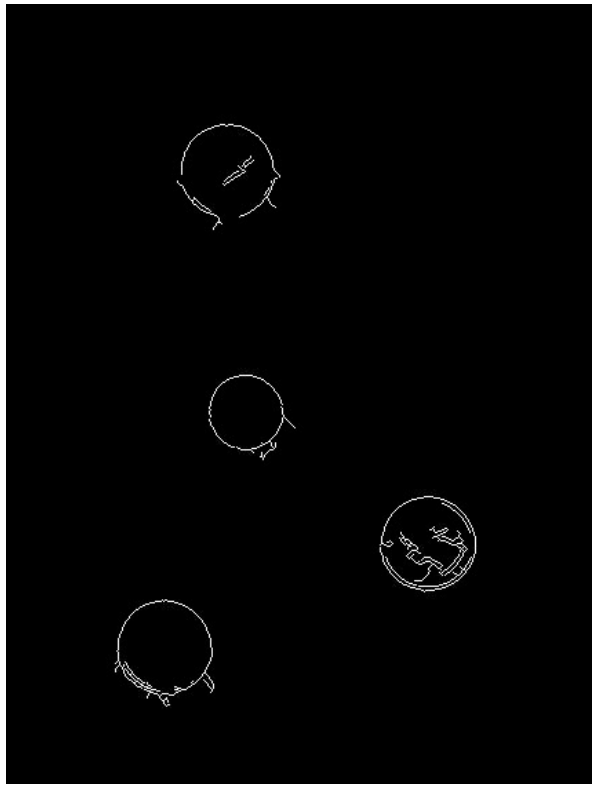
Slide credit: Kristen Grauman

Example: detecting circles with Hough

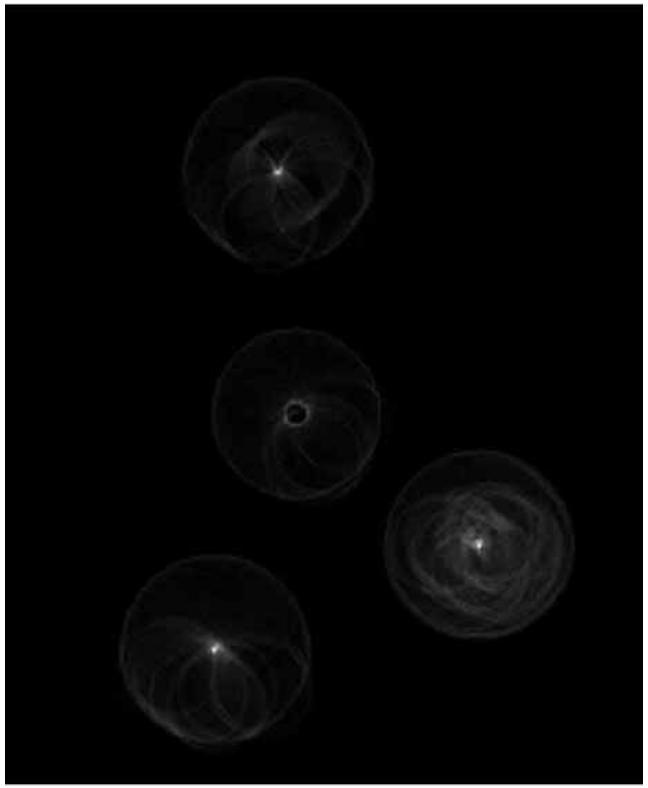
Original



Edges



Votes: Quarter



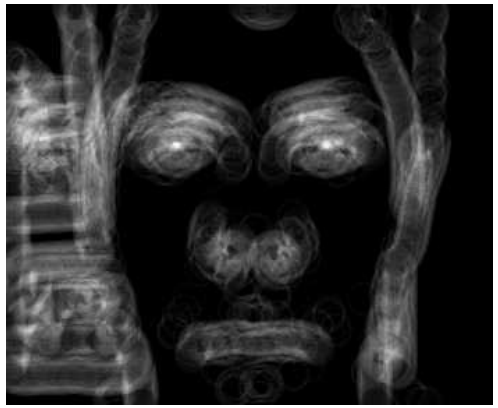
Combined detections

Slide credit: Kristen Grauman

Coin finding sample images from: Vivek Kwatra

Example: iris detection






Gradient+threshold

Hough space (fixed radius)

Max detections

- Hemerson Pistori and Eduardo Rocha Costa http://rsbweb.nih.gov/ij/plugins/hough-circles.html

Example: iris detection

Figure 2. Original image

Figure 3. Distance image Figure 4. Detected face region

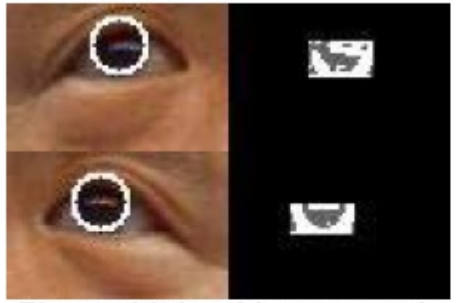


Figure 14. Looking upward

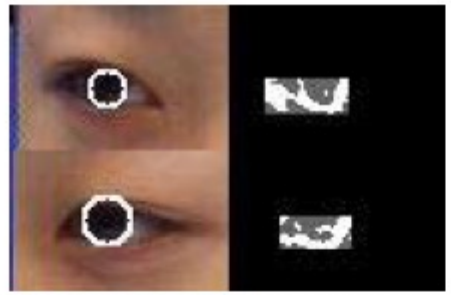


Figure 15. Looking sideways

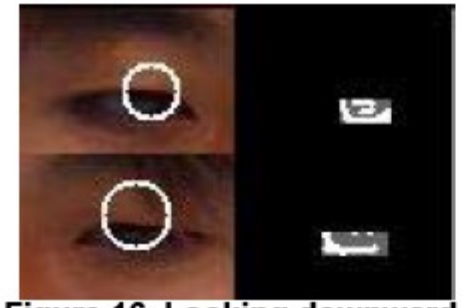
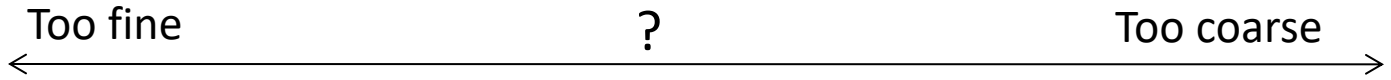


Figure 16. Looking downward

- An Iris Detection Method Using the Hough Transform and Its Evaluation for Facial and Eye Movement, by Hideki Kashima, Hitoshi Hongo, Kunihito Kato, Kazuhiko Yamamoto, ACCV 2002.

Voting: practical tips

- Minimize irrelevant tokens first
- Choose a good grid / discretization



- Vote for neighbors, also (smoothing in accumulator array)
- Use direction of edge to reduce parameters by 1
- Keep track of which points contributed to a vote
  - Allows you to quickly see which points correspond to a "winning" vote.

Hough transform: pros and cons

Pros

- All points are processed independently, so can cope with occlusion, gaps
- Some robustness to noise: noise points unlikely to contribute *consistently* to any single bin
- Can detect multiple instances of a model in a single pass

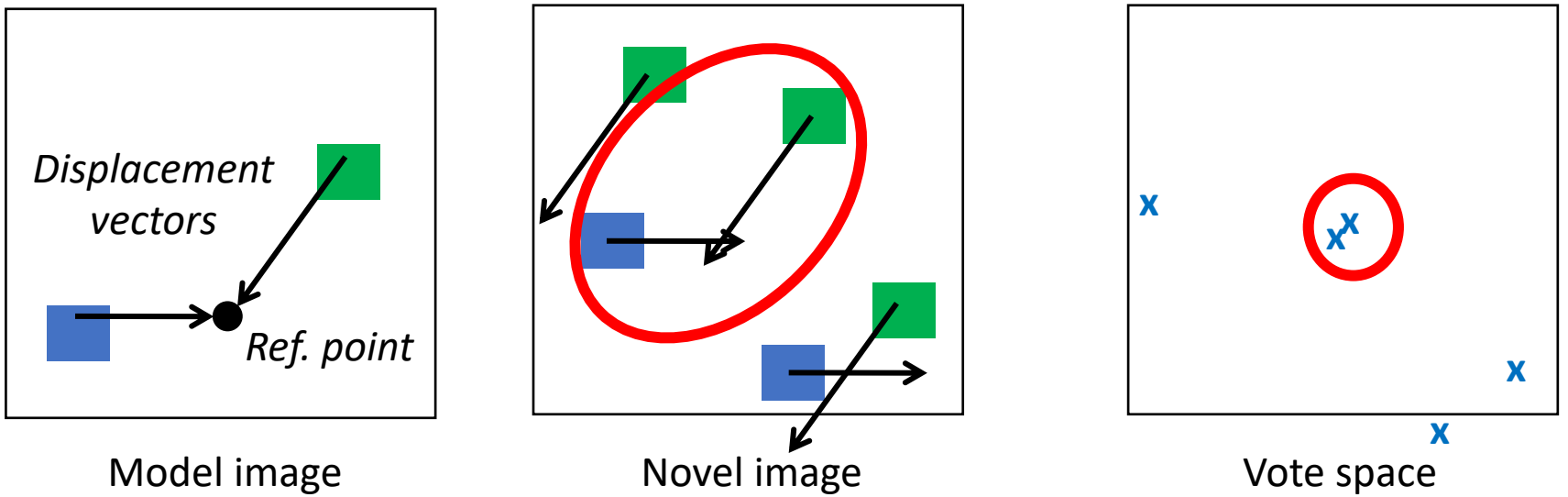
Cons

- Complexity of search time increases exponentially with the number of model parameters
- Non-target shapes can produce spurious peaks in parameter space
- Quantization: can be tricky to pick a good grid size

Generalized Hough Transform

What if we want to detect arbitrary shapes?

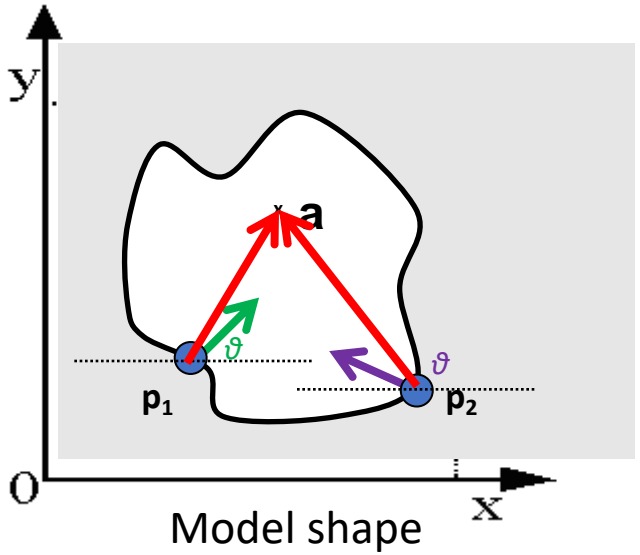
Intuition:



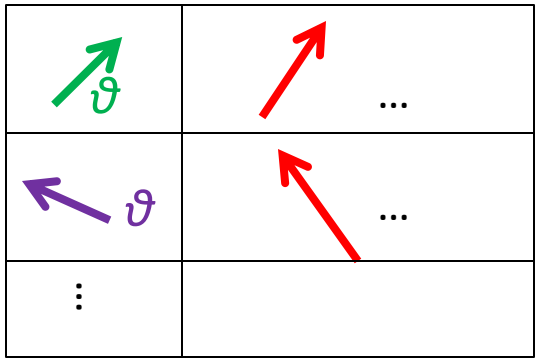
Now suppose those colors encode gradient directions…

Generalized Hough Transform

- **Define a model shape** by its boundary points and a reference point.



Model shape



Offline procedure:

At each boundary point, compute displacement vector: **r = a – p_i**.

Store these vectors in a table indexed by gradient orientation θ.

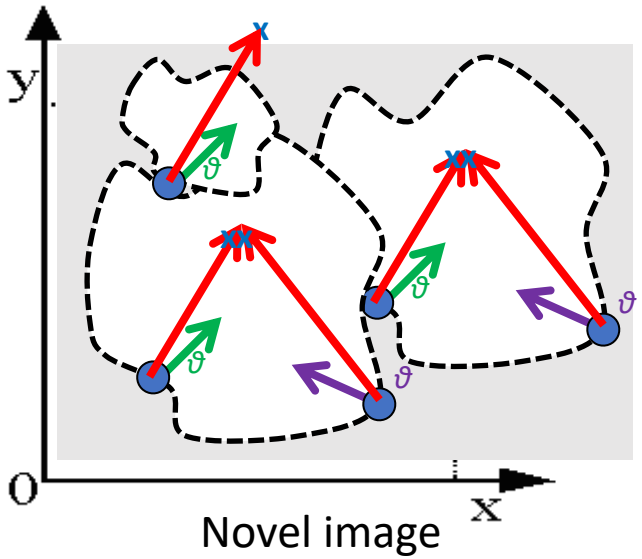
[Dana H. Ballard, Generalizing the Hough Transform to Detect Arbitrary Shapes, 1980]

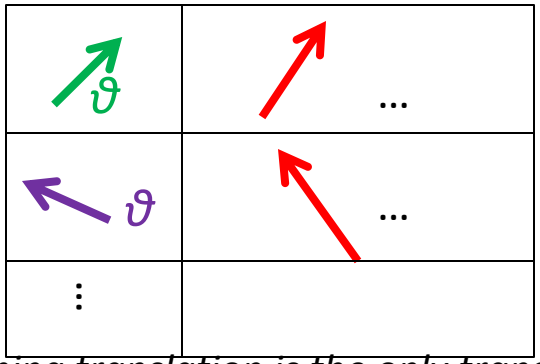
Generalized Hough Transform

Detection procedure:

For each edge point:

- Use its gradient orientation θ to index into stored table
- Use retrieved **r** vectors to vote for reference point

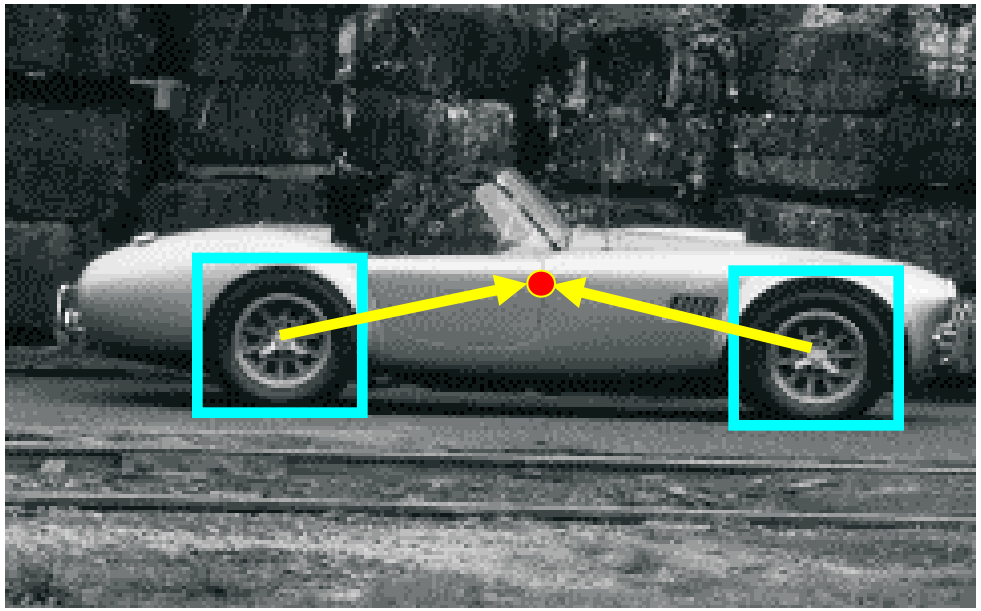




Assuming translation is the only transformation here, i.e., orientation and scale are fixed.

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by matched local patterns.



training image

“visual codeword” with displacement vectors

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

Generalized Hough for object detection

- Instead of indexing displacements by gradient orientation, index by "visual codeword"

test image

B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model, ECCV Workshop on Statistical Learning in Computer Vision 2004

Source: L. Lazebnik

Summary

- **Fitting** problems require finding any supporting evidence for a model, even within clutter and missing features.
  - associate features with an explicit model
- **Voting** approaches, such as the **Hough transform**, make it possible to find likely model parameters without searching all combinations of features.
  - Hough transform approach for lines, circles, …, arbitrary shapes defined by a set of boundary points, recognition from patches.

Questions?

See you next time!