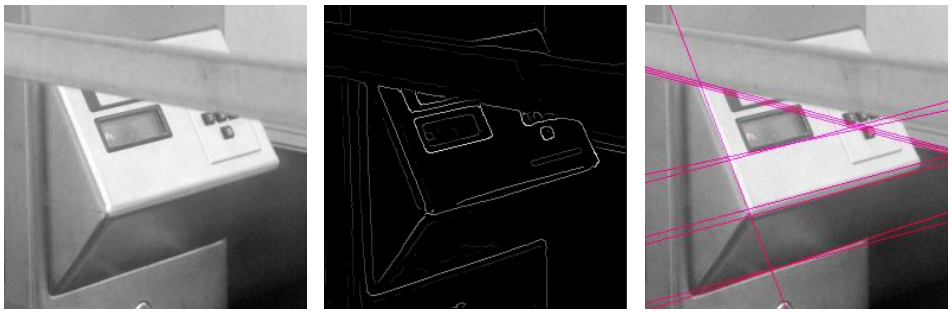
CMPE 362
Digital Image Processing

Fitting: Hough Transform

Fitting

- Associate a model with observed features (such as edges)



[Fig from Marszalek & Schmid, 2007]

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

Slide credit: Kristen Grauman

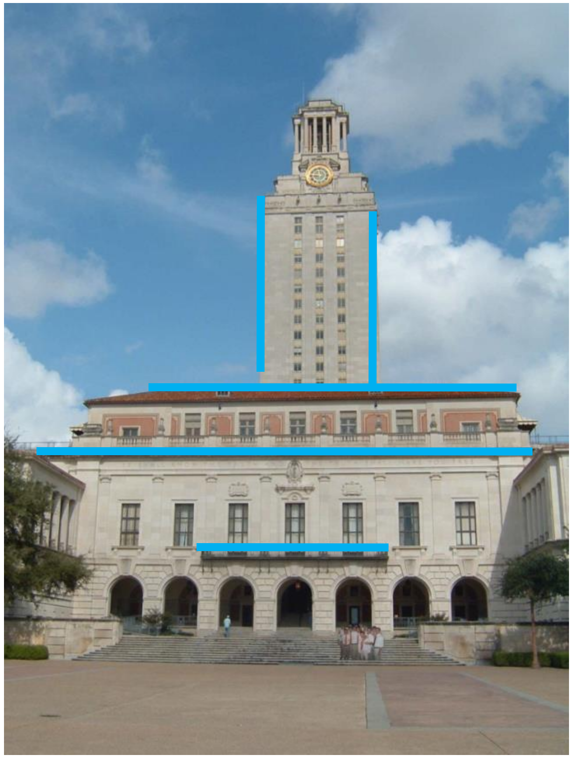
Fitting: Main idea

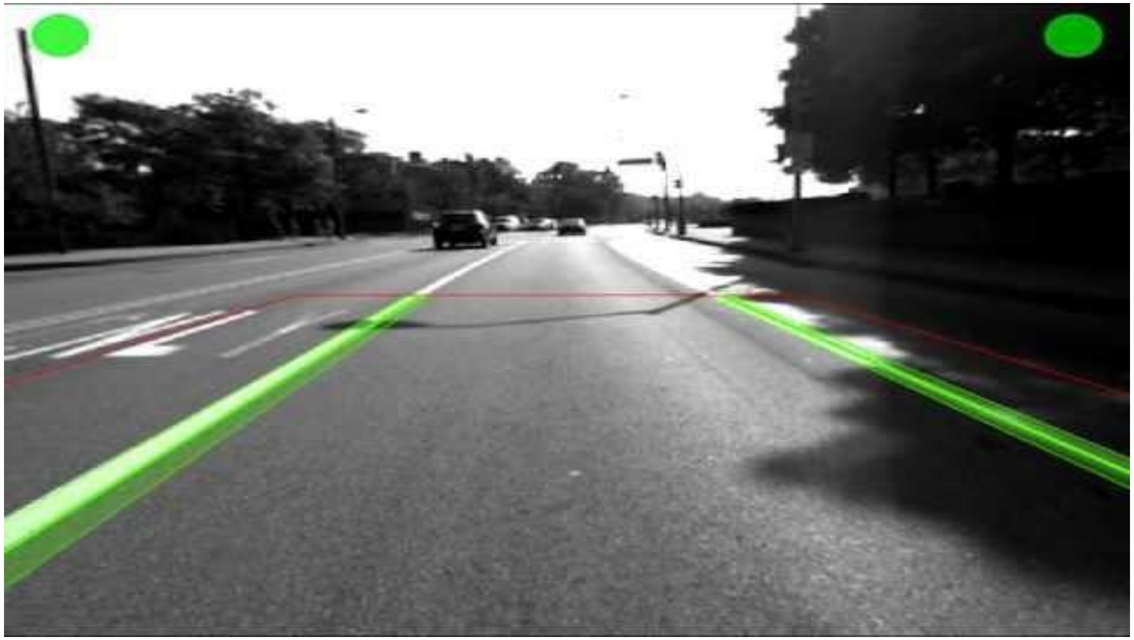
- Choose a parametric model to represent a set of features such as edges
- Membership criterion is not local
  - We can not tell whether a point belongs to a given model just by looking at that point
- Three main questions:
  - What model represents this set of features best?
  - Which of several model 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

Case study: Line fitting

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

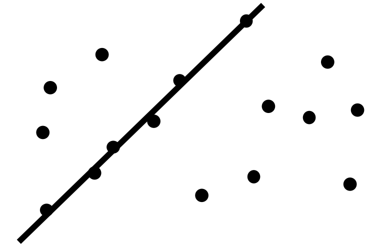


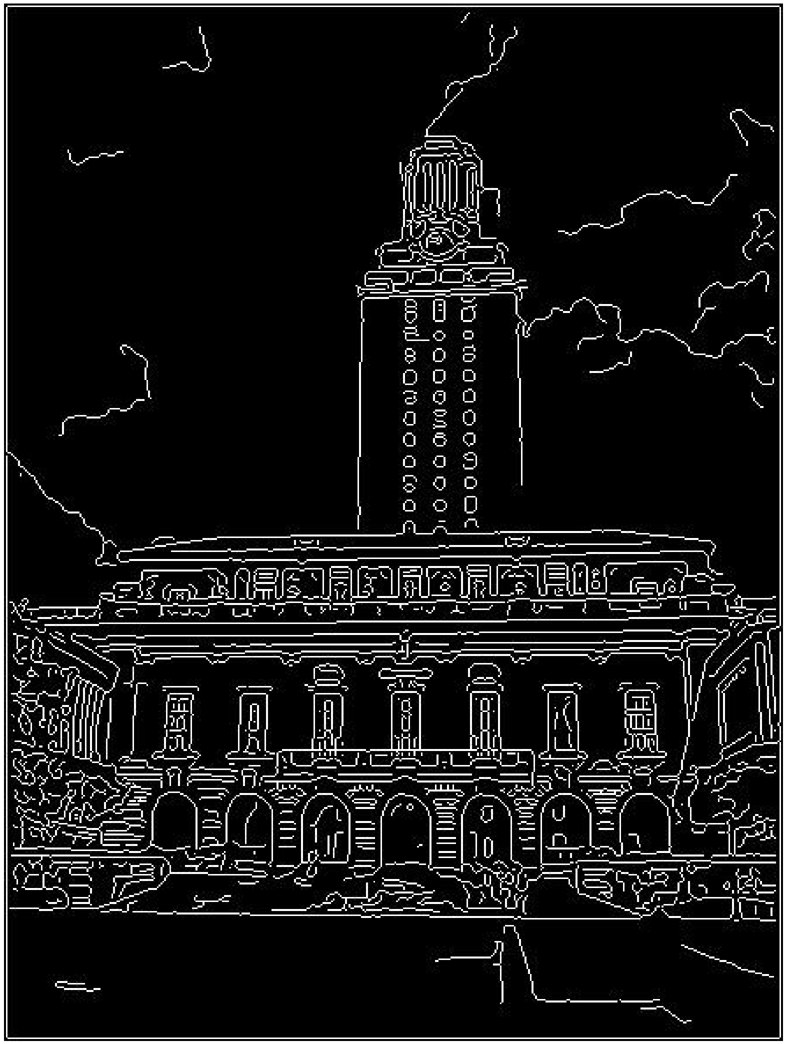


- Wait, why aren't we done just by running edge detection?

Slide credit: Kristen Grauman

Difficulties of line fitting





- **Extra** edge points (clutter), multiple models:
  - which points go with which line, if any?
- Only some parts of each line are 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?

Slide credit: Kristen Grauman

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 each feature 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.

Slide credit: Kristen Grauman

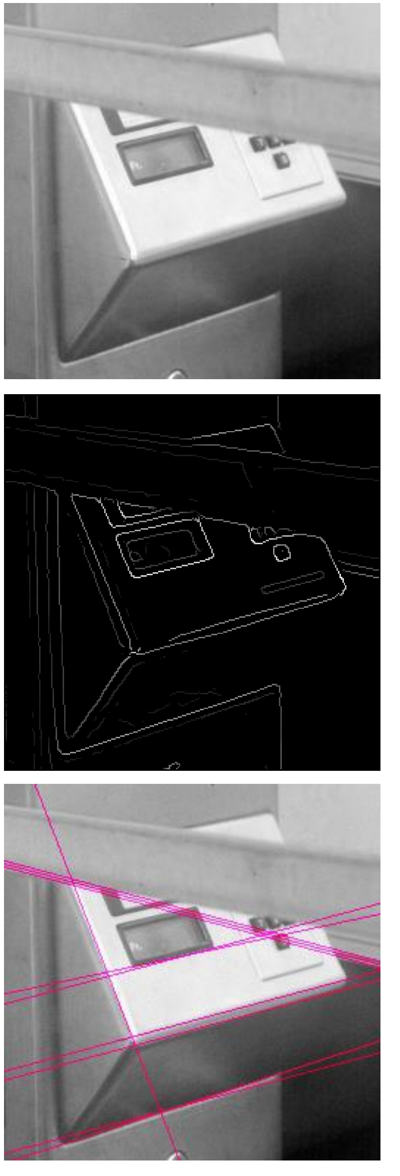
Fitting lines: Hough transform

- 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.



Slide credit: Kristen Grauman

Finding lines in an image: Hough space

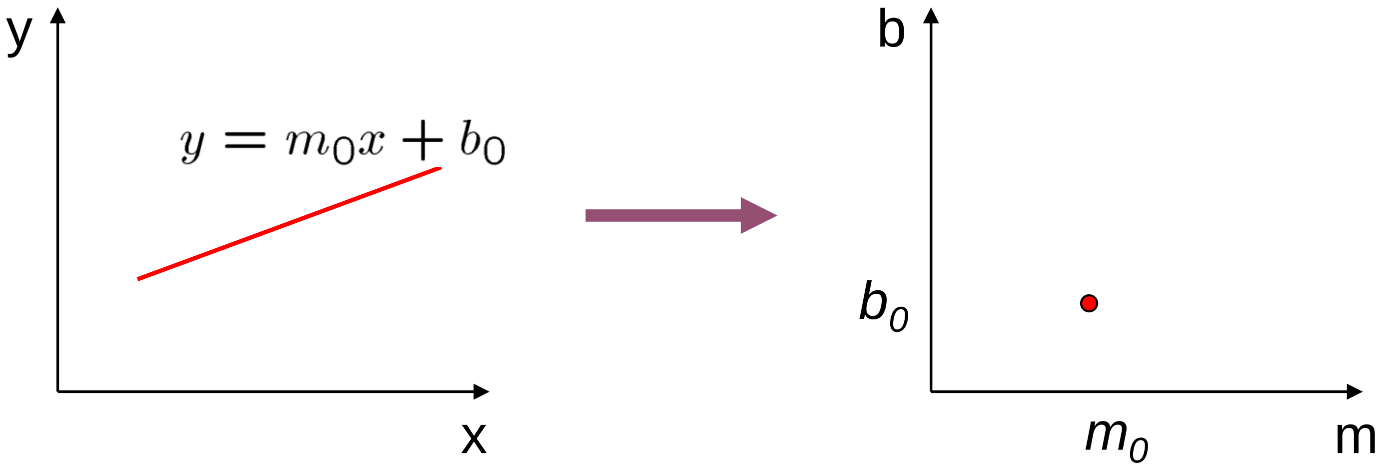


image space

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

Slide credit: Steve Seitz

Finding lines in an image: Hough space

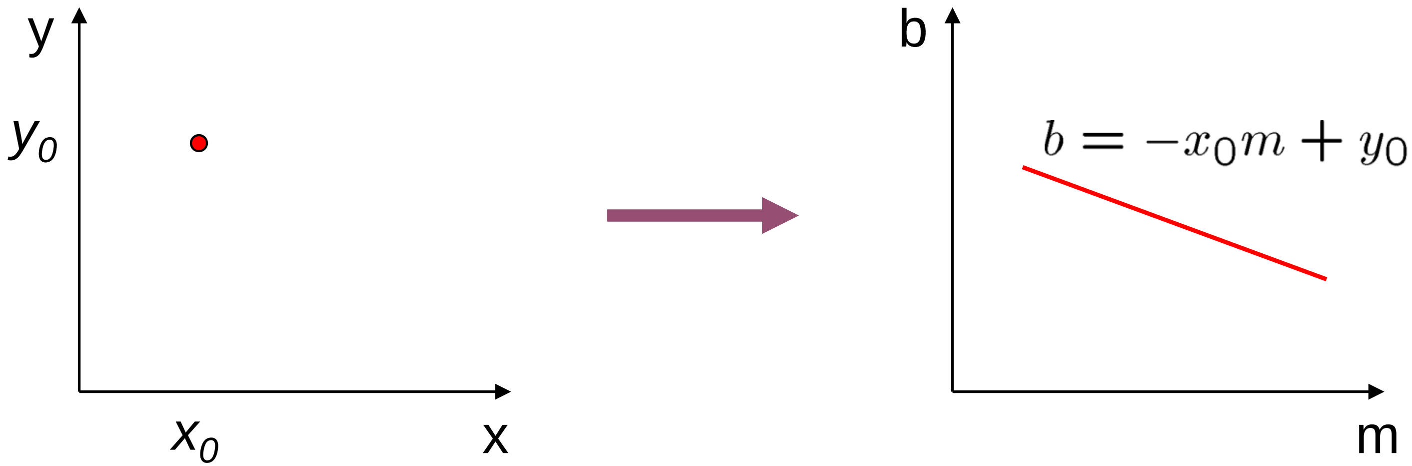


image space

Hough (parameter) space

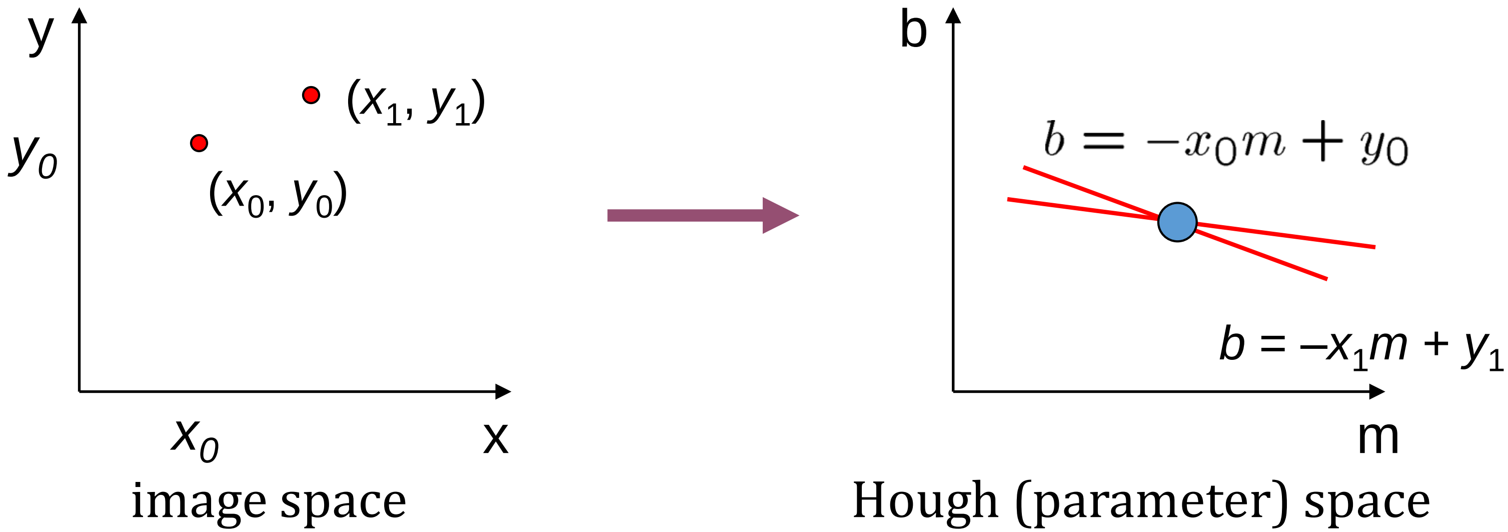
Connection between image (x, y) and Hough (m, b) spaces

- 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
- 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 $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$

Slide credit: Kristen Grauman

Finding lines in an image: Hough space

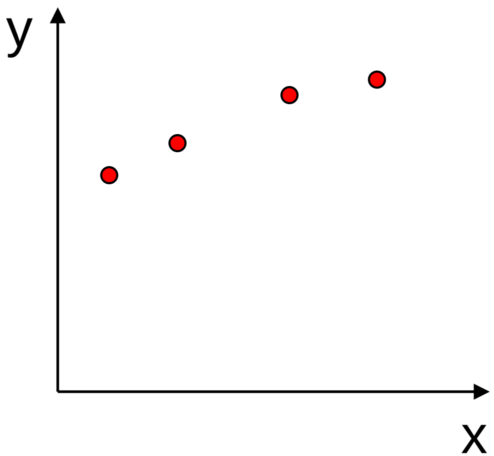
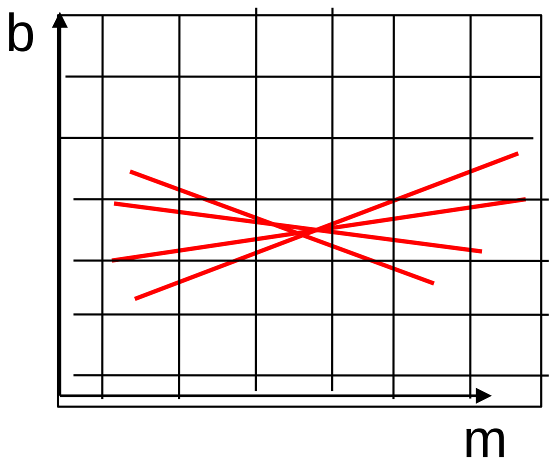


image space



Hough (parameter) space

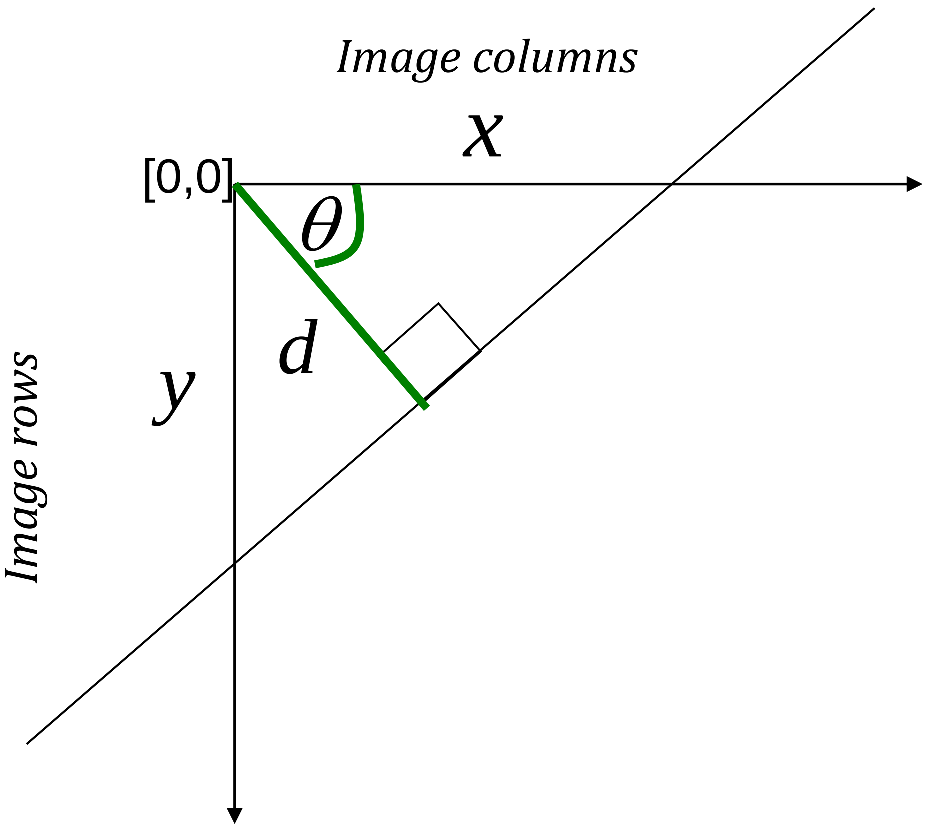
How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?

- 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 the most prominent lines 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 to line from origin

θ : angle the perpendicular makes with the x-axis

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

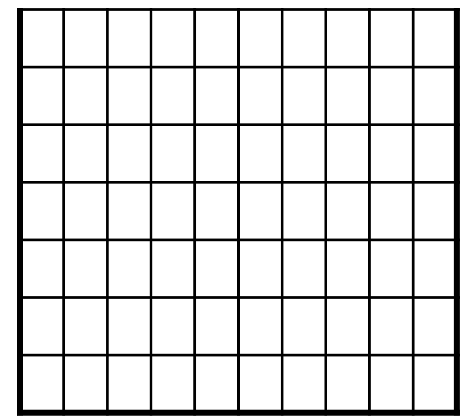
Point in image space → sinusoid segment in Hough space

Slide credit: Kristen Grauman

Hough transform algorithm

Using the polar parameterization:

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



Basic Hough transform algorithm

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

    for $\theta = [\theta_{min}$ to $\theta_{max}]$ // some quantization

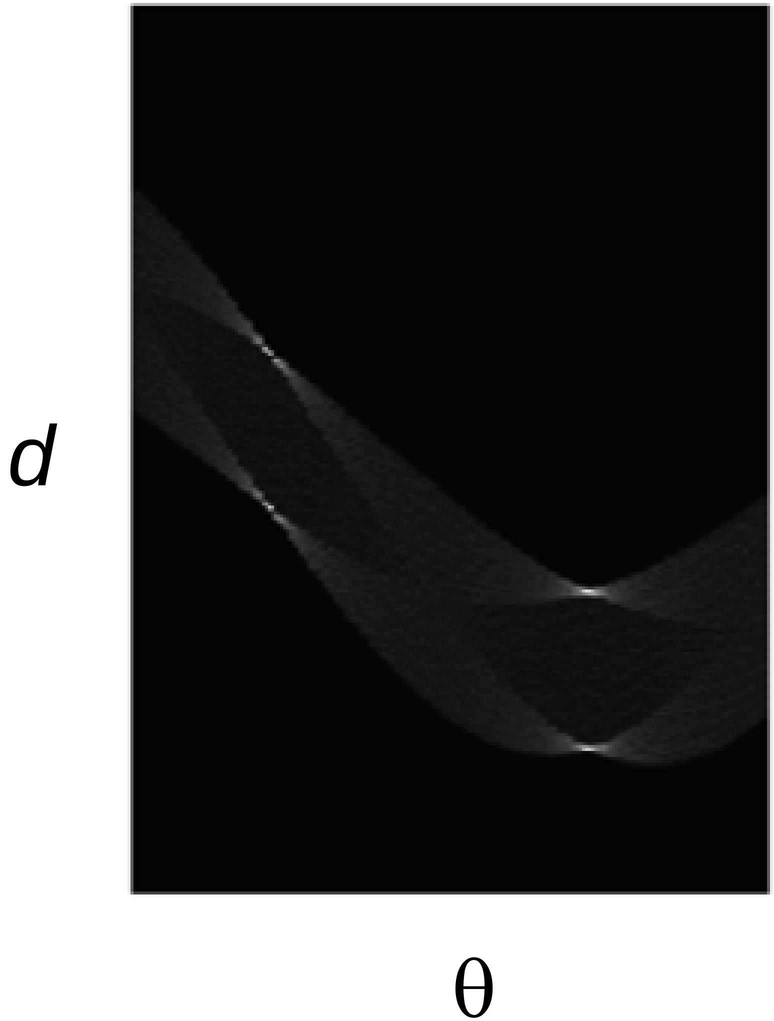
    $$d = x\cos\theta - y\sin\theta$$

    $H[d, \theta]$ += 1
3. Find the value or values of (d, θ) where $H[d, \theta]$ is maximum
4. The detected line in the image is given by $d = x\cos\theta - y\sin\theta$

Slide credit: Steve Seitz

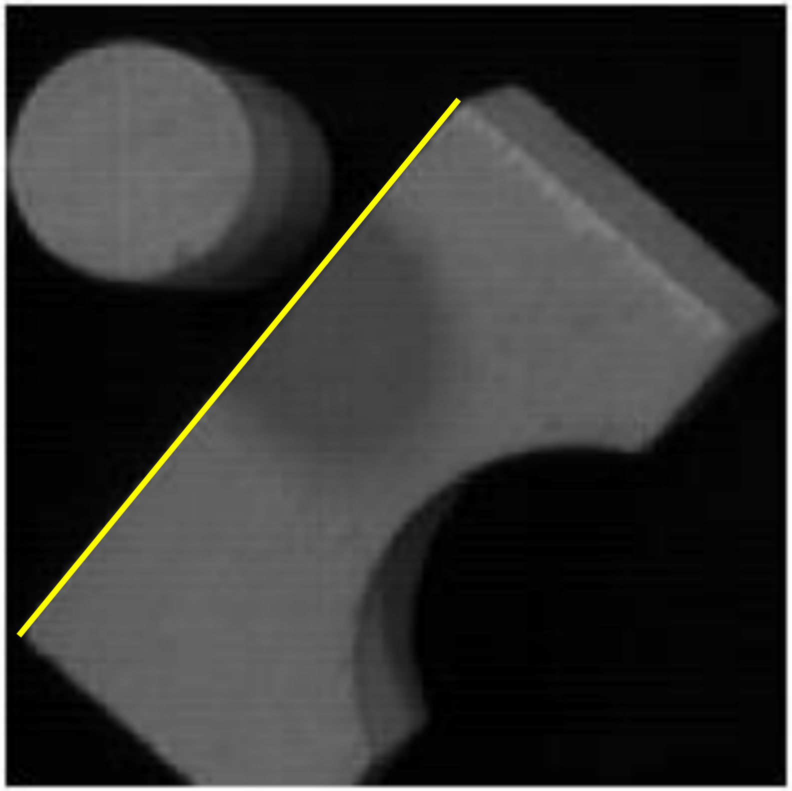
Example: What was the shape?

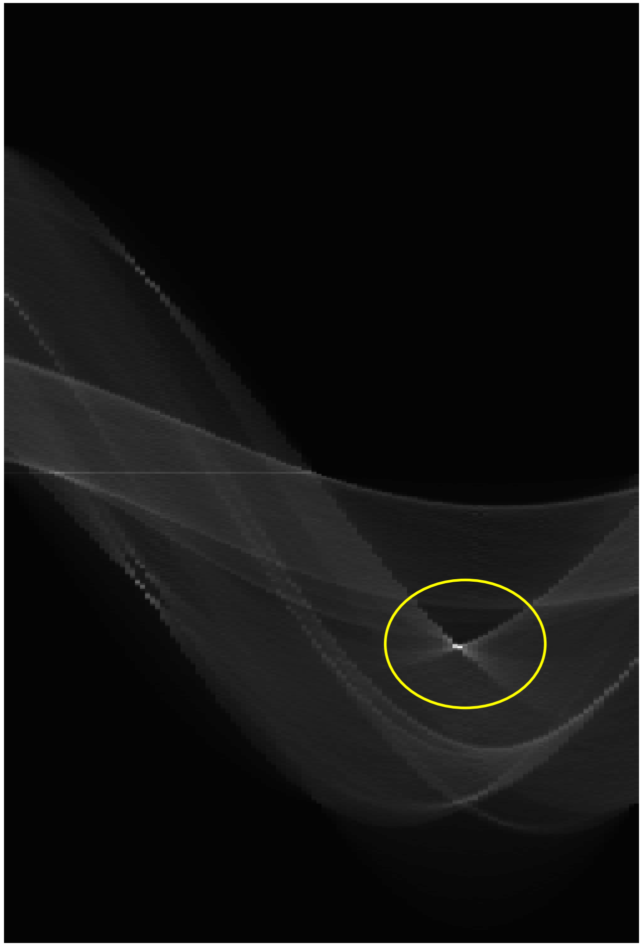
Square



Slide credit: Kristen Grauman

Example: Hough transform for straight lines



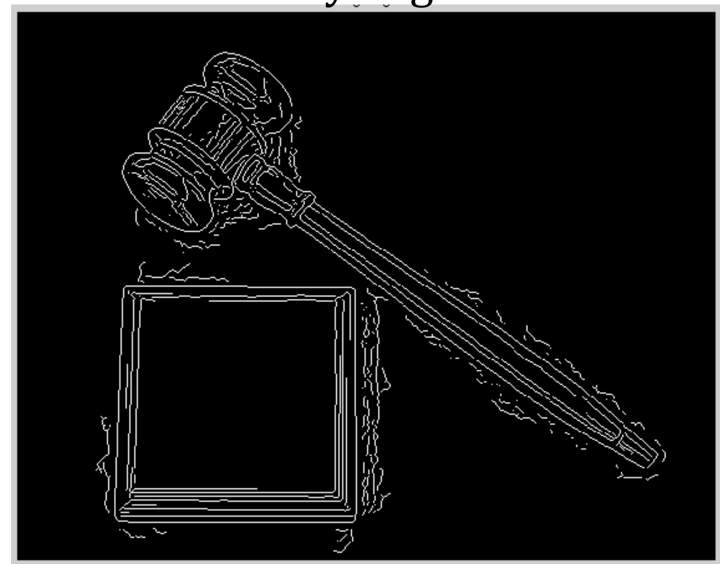


Which line generated this peak?

Slide credit: Kristen Grauman

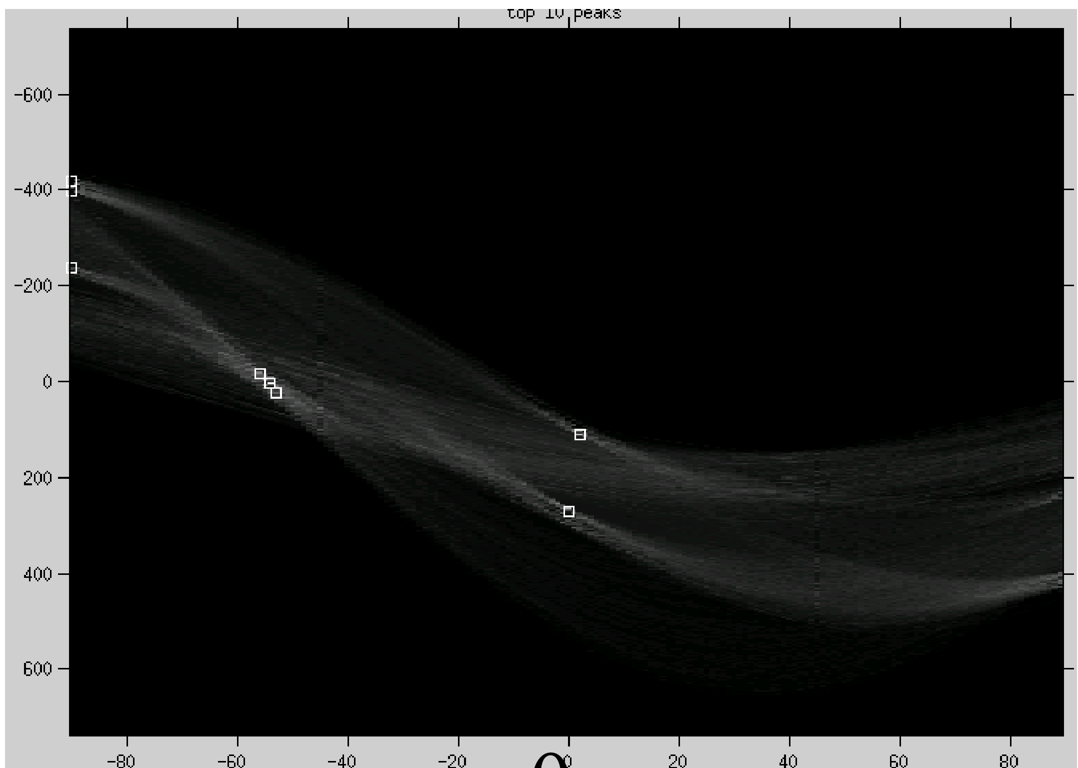
Original image

Canny edges



Decode the vote space.

d



θ

Slide credit: Kristen Grauman

Impact of noise on Hough

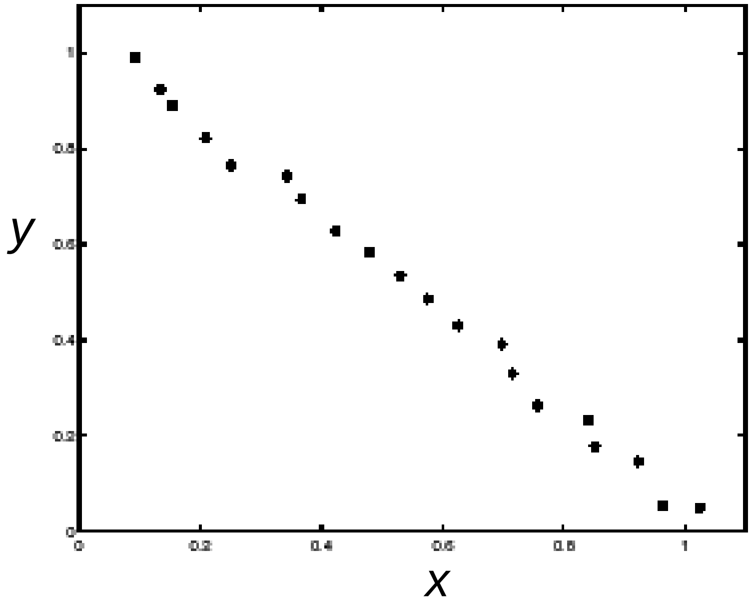
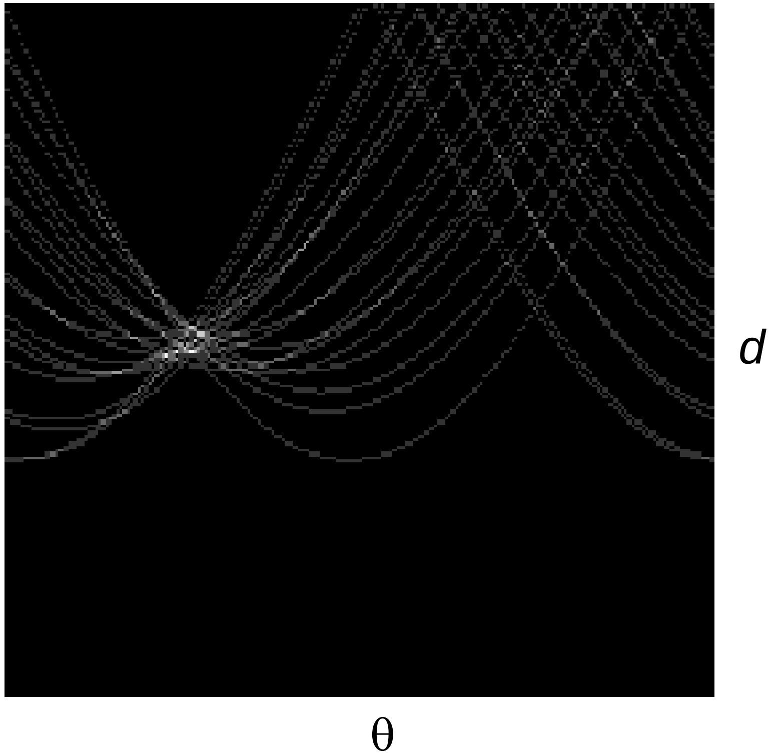


Image space
edge coordinates



Votes

What difficulty does this present for an implementation?

Slide credit: Kristen Grauman

Impact of noise on Hough

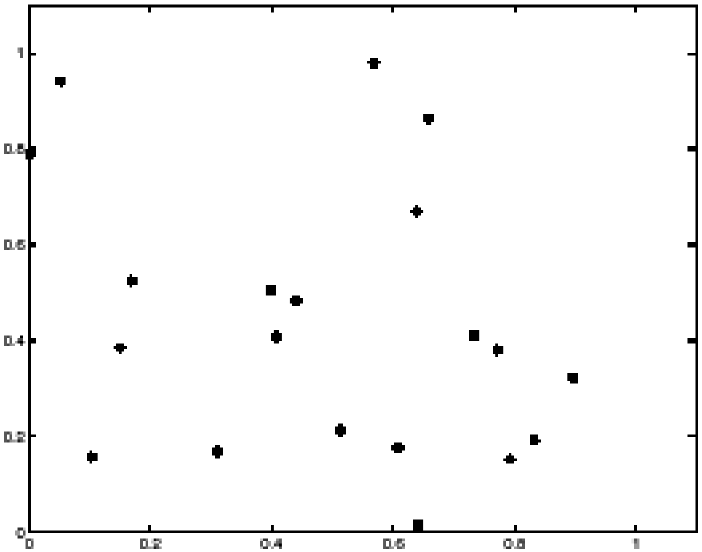
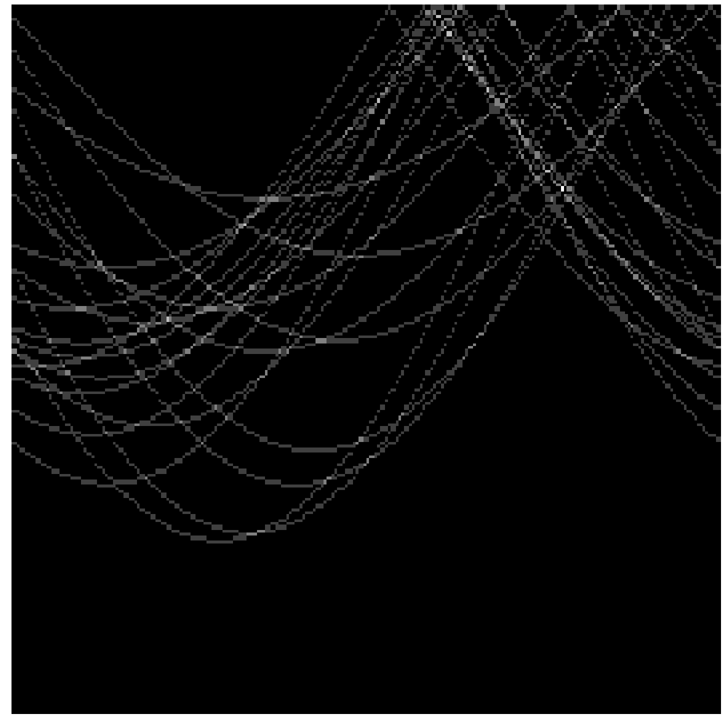


Image space
edge coordinates



Votes

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

Slide credit: Kristen Grauman

Voting: practical tips

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

Too fine ? Too coarse

- Vote for neighbors, also (smoothing in accumulator array)
- Use direction of edge to reduce parameters by 1
- To read back which points voted for "winning" peaks, keep tags on the votes.

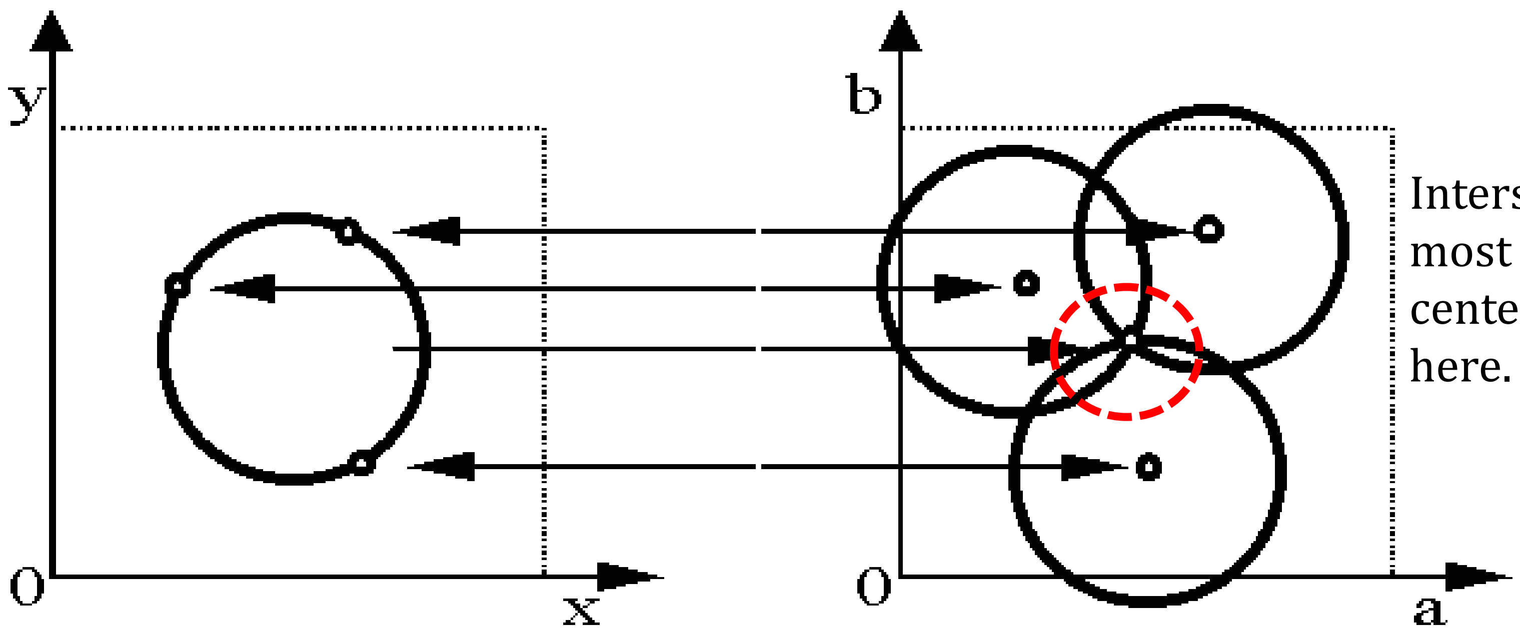
Slide credit: Kristen Grauman

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, unknown gradient direction



Intersection: most votes for center occur here.

Image space

Hough space

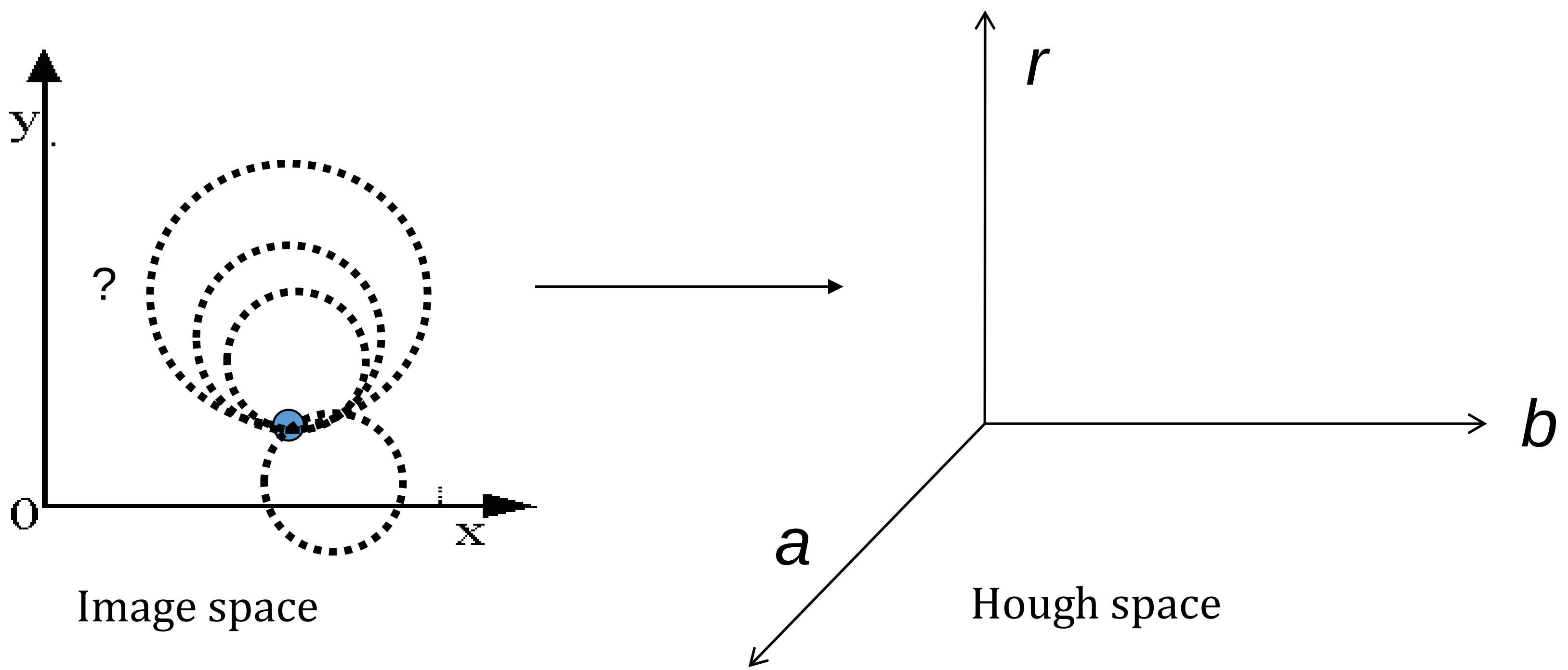
Slide credit: Kristen Grauman

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, unknown gradient direction



Slide credit: Kristen Grauman

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, unknown gradient direction

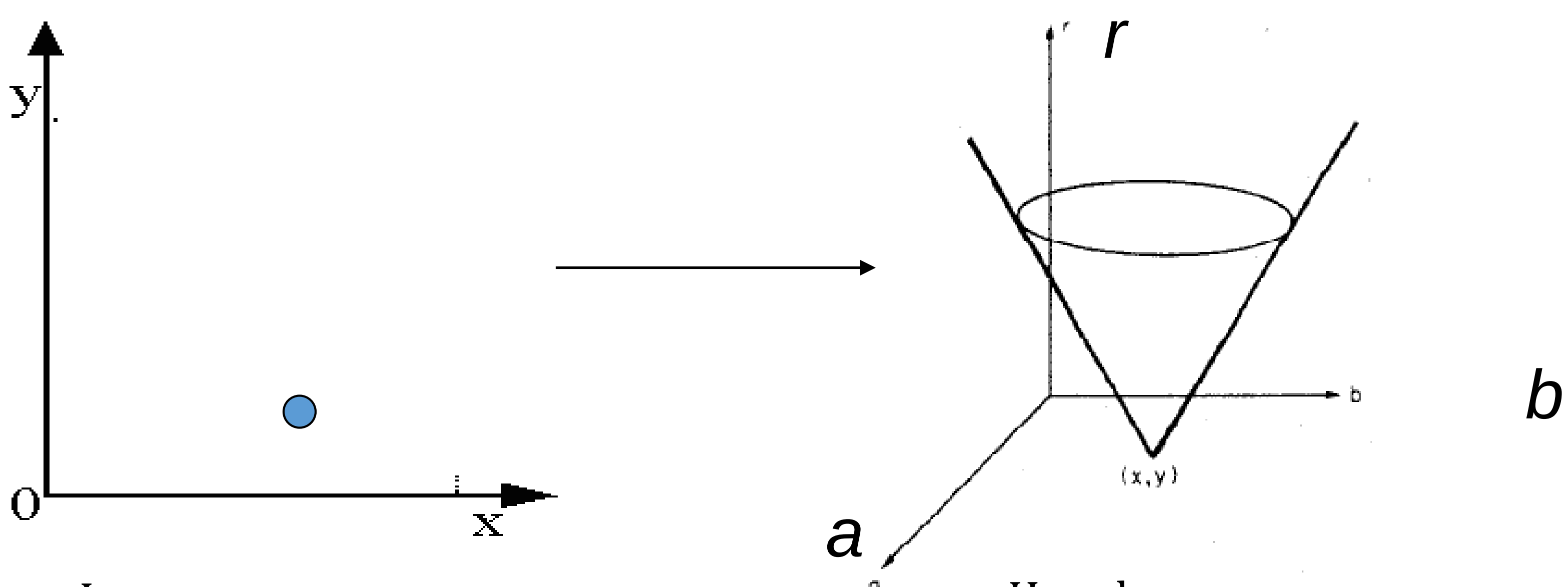


Image space

Hough space

Slide credit: Kristen Grauman

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

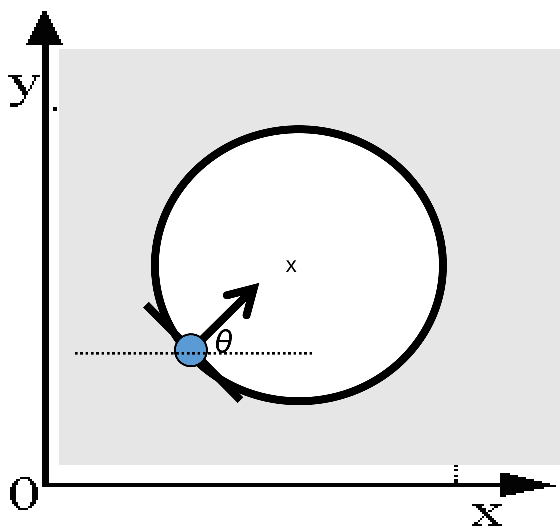
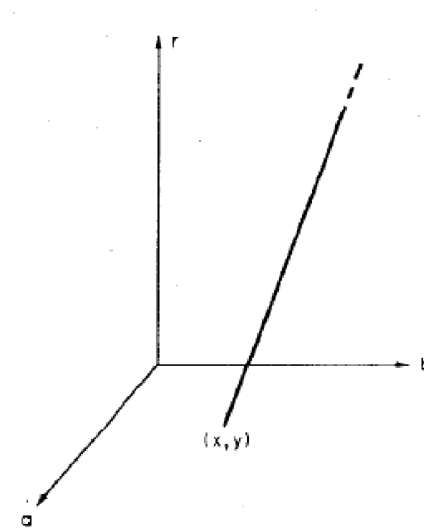


Image space



Hough space

Slide credit: Kristen Grauman

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

$H[a, b, r] \mathrel{+}= 1$

end

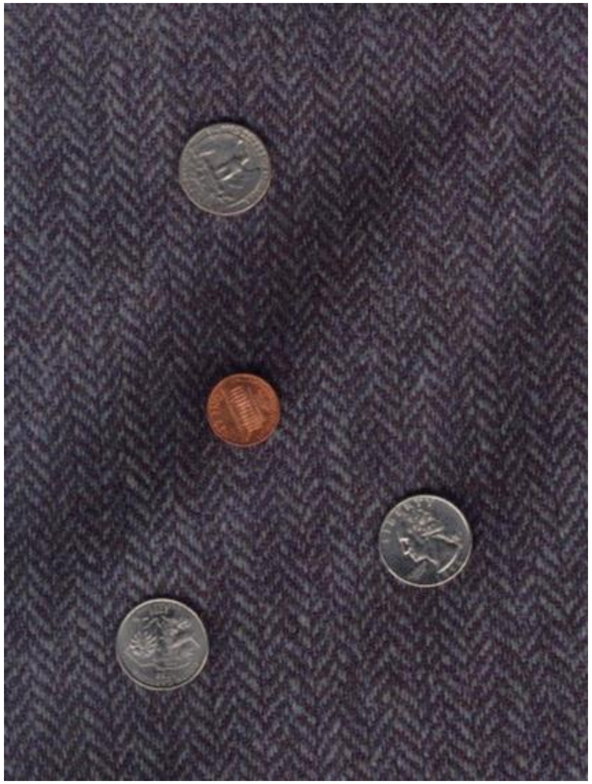
end

end

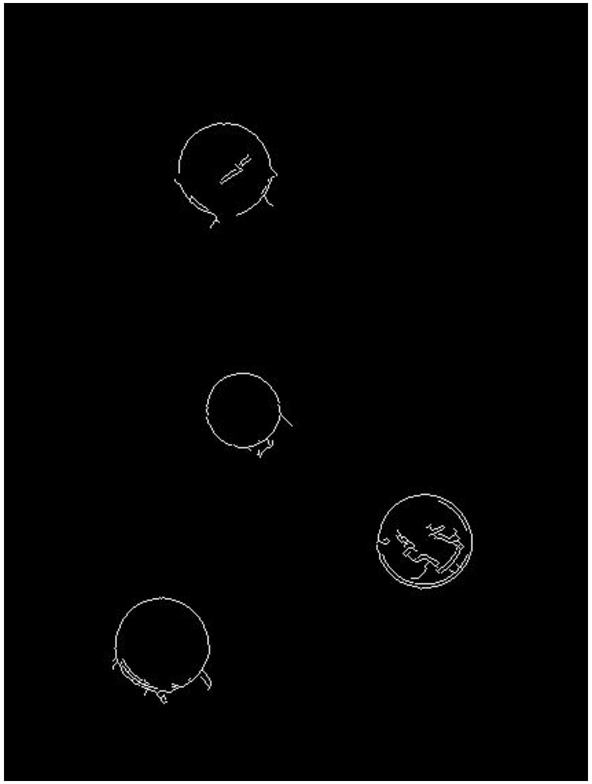
Slide credit: Kristen Grauman

Example: detecting circles with Hough

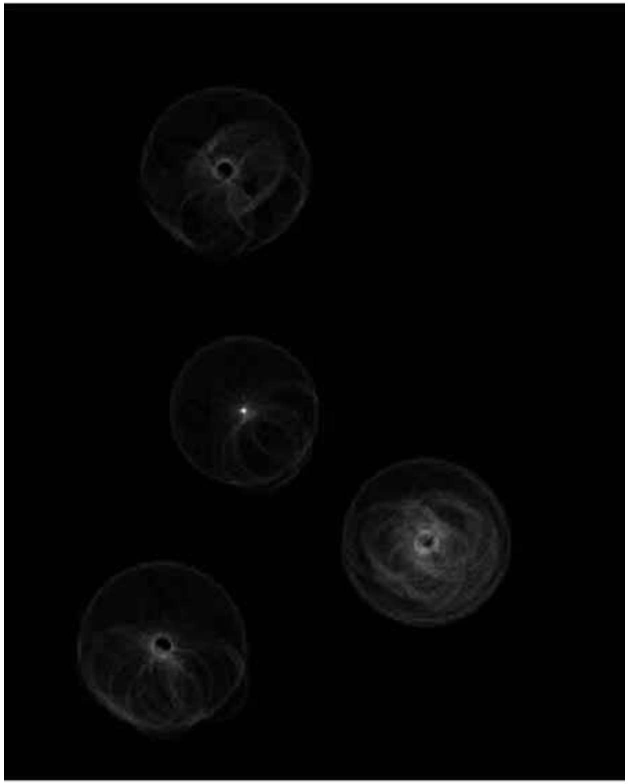
Original



Edges



Votes: Penny

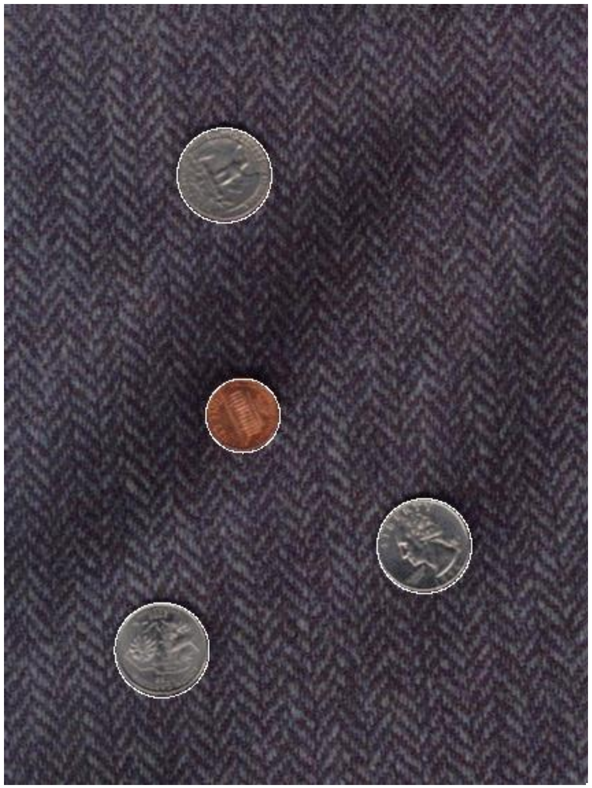


Slide credit: Kristen Grauman

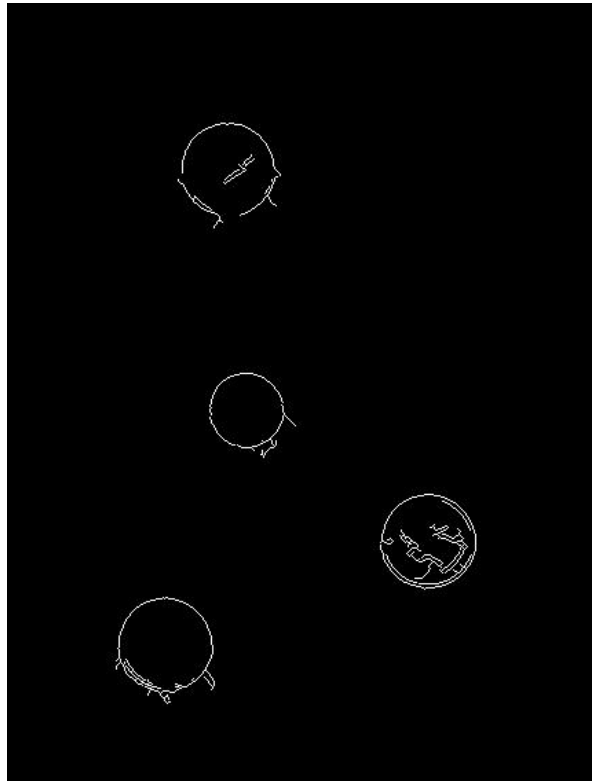
Example: detecting circles with Hough

Original

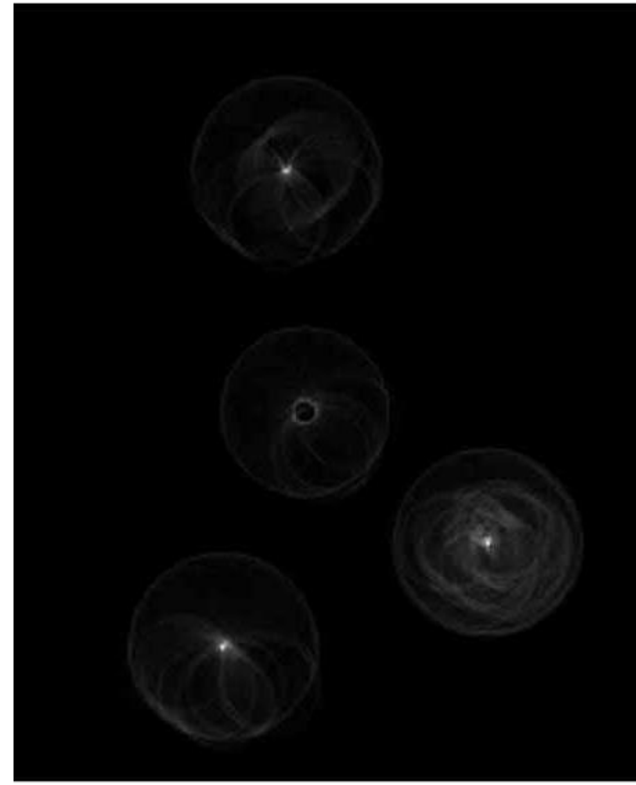
Combined detections



Edges



Votes: Quarter



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

Slide credit: Kristen Grauman

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

Slide credit: Kristen Grauman