10/02/13

Straight Lines and Hough Transform
Sec. 4.3 Szeliski

Many slides from James Hays, Larry Davis, Derek Hoiem, Lana Lazebnik, Steve Seitz, David Forsyth, David Lowe, Fei-Fei Li

Canny edge detector

1. Filter image with x, y derivatives of Gaussian
2. Find magnitude and orientation of gradient
3. Non-maximum suppression:
   - Thin multi-pixel wide "ridges" down to single pixel width
4. Thresholding and linking (hysteresis):
   - Define two thresholds: low and high
   - Use the high threshold to start edge curves and the low threshold to continue them

- MATLAB: edge(image, 'canny')

Source: D. Lowe, L. Fei-Fei

Finding straight lines

- One solution: try many possible lines and see how many points each line passes through

- Hough transform provides a fast way to do this

Outline of Hough Transform

1. Create a grid of parameter values

2. Each point votes for a set of parameters, incrementing those values in grid

3. Find maximum or local maxima in grid

Finding lines using Hough transform

- Using m,b parameterization
- Using r, theta parameterization
  - Using oriented gradients
- Practical considerations
  - Bin size
  - Smoothing
  - Finding multiple lines
  - Finding line segments

Hough transform

- An early type of voting scheme
- General outline:
  - Discretize parameter space into bins
  - For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  - Find bins that have the most votes

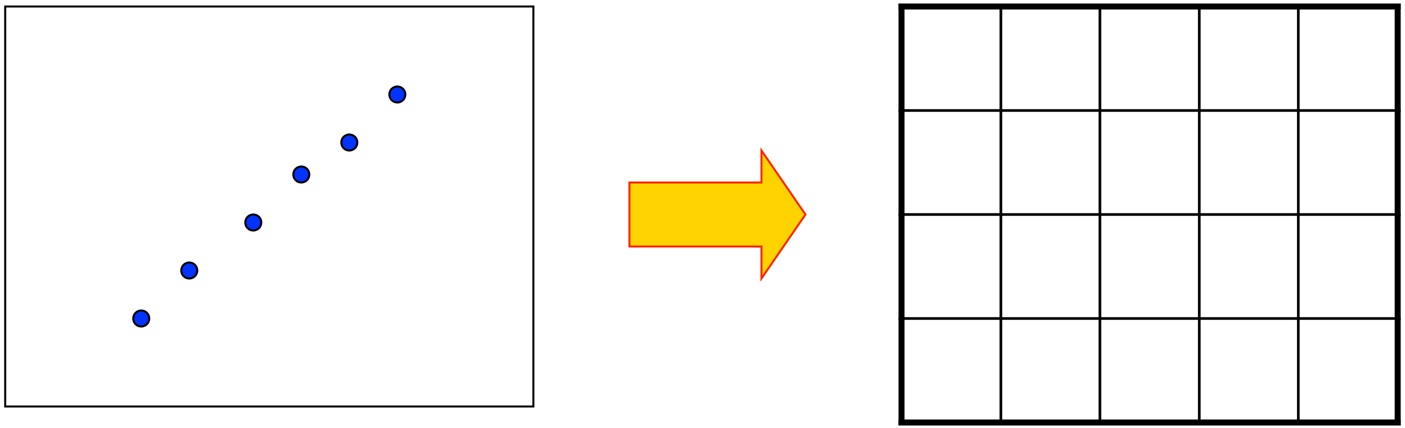


Image space

Hough parameter space

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

Parameter space representation

- A line in the image corresponds to a point in Hough space

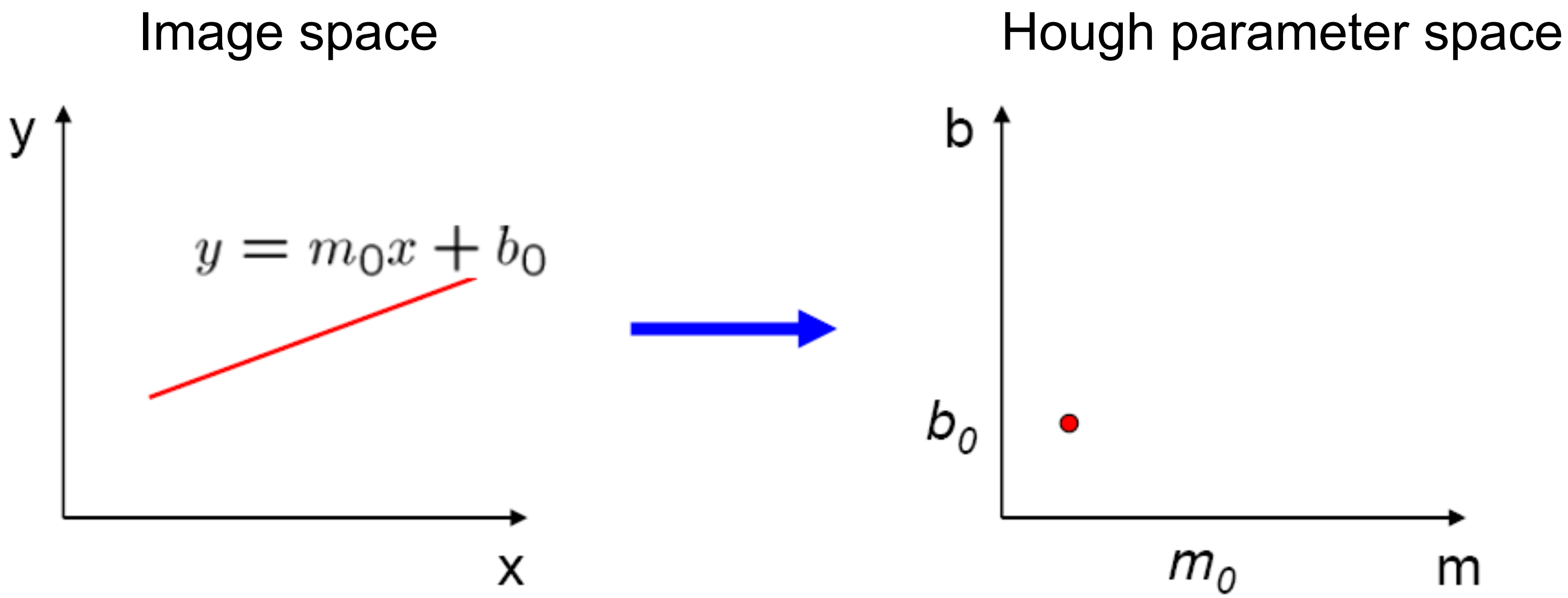


Image space

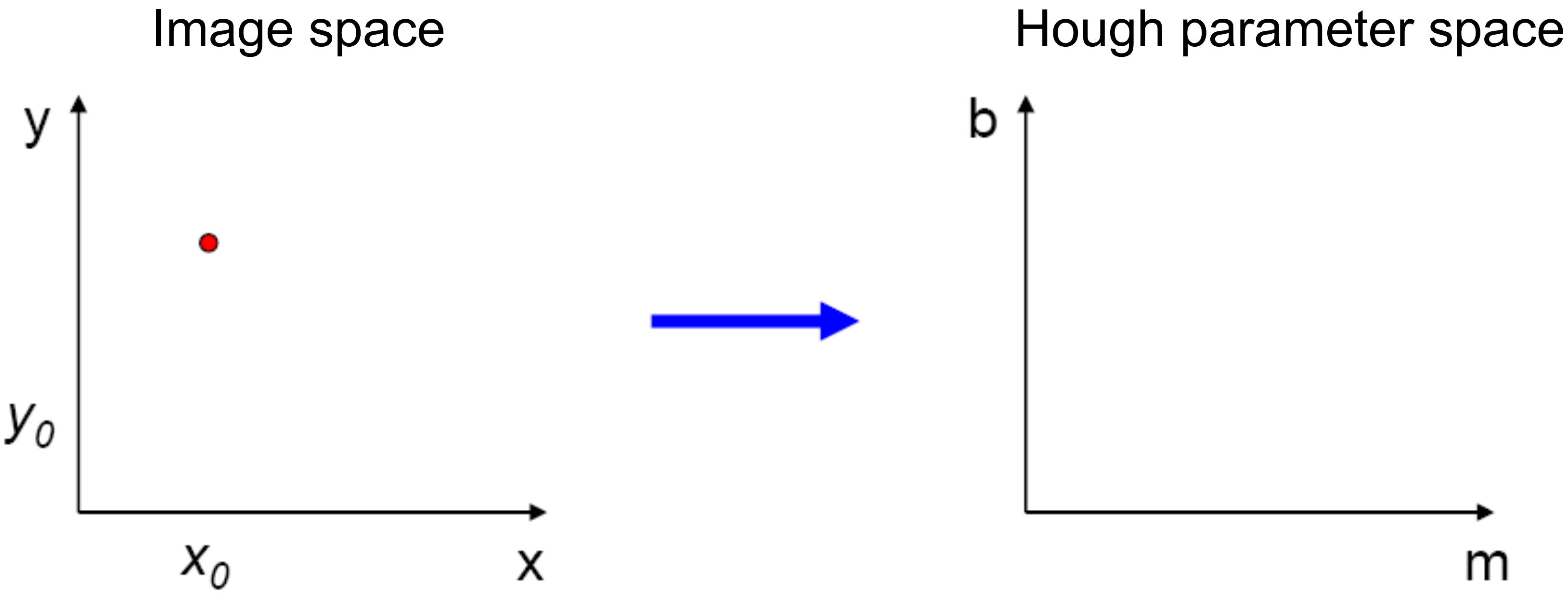
Hough parameter space

$y = m_0 x + b_0$

Source: S. Seitz

Parameter space representation

- What does a point (x_0, y_0) in the image space map to in the Hough space?



Parameter space representation

- What does a point (x_0, y_0) in the image space map to in the Hough space?
  - Answer: the solutions of $b = –x_0m + y_0$
  - This is a line in Hough space

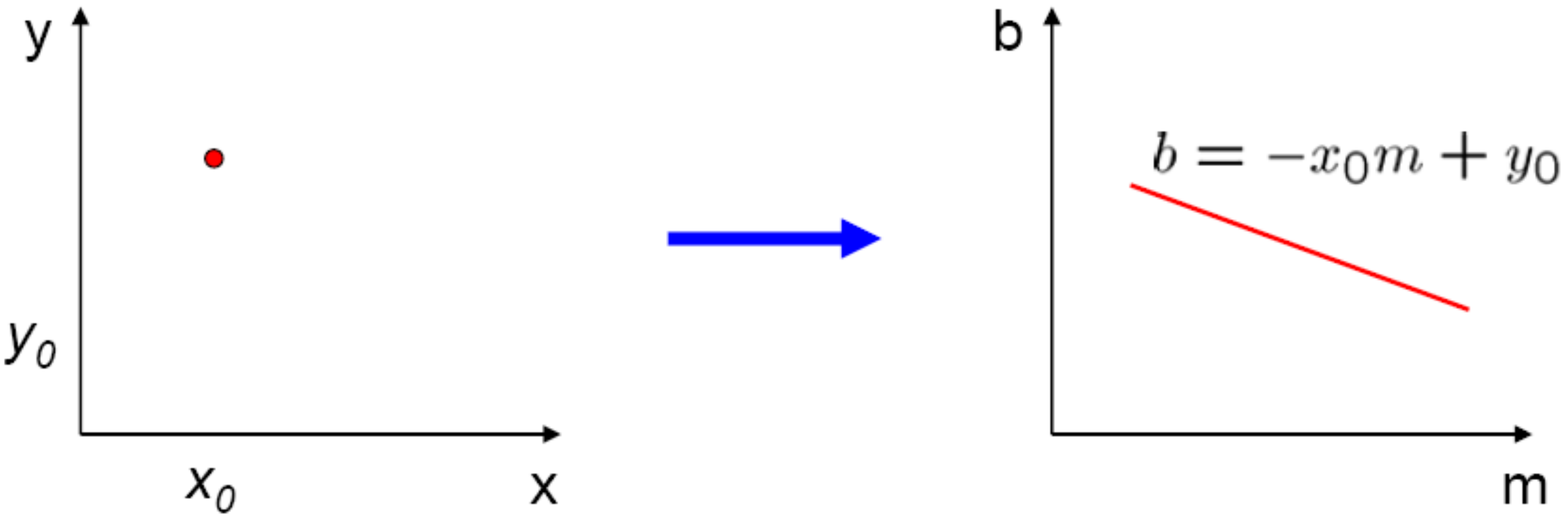


Image space

Hough parameter space

Parameter space representation

- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?

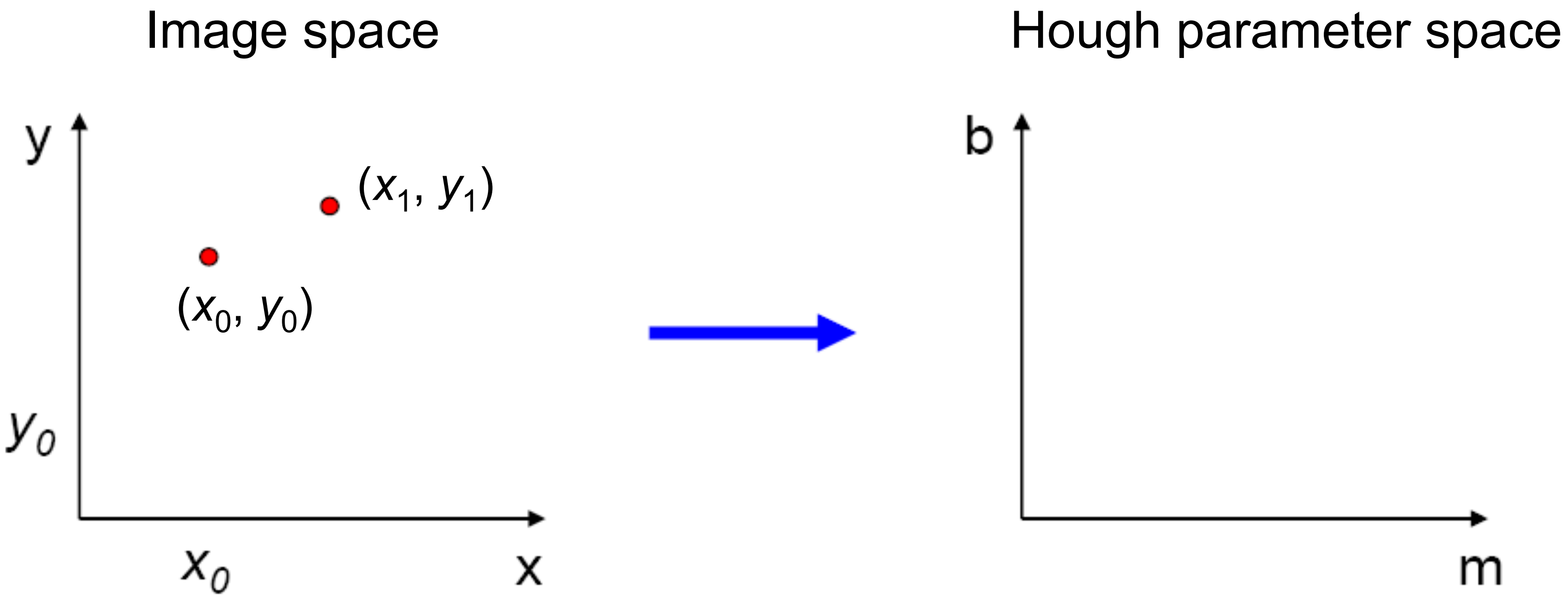


Image space

Hough parameter space

Parameter space representation

- Where is 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 $b = -x_1 m + y_1$

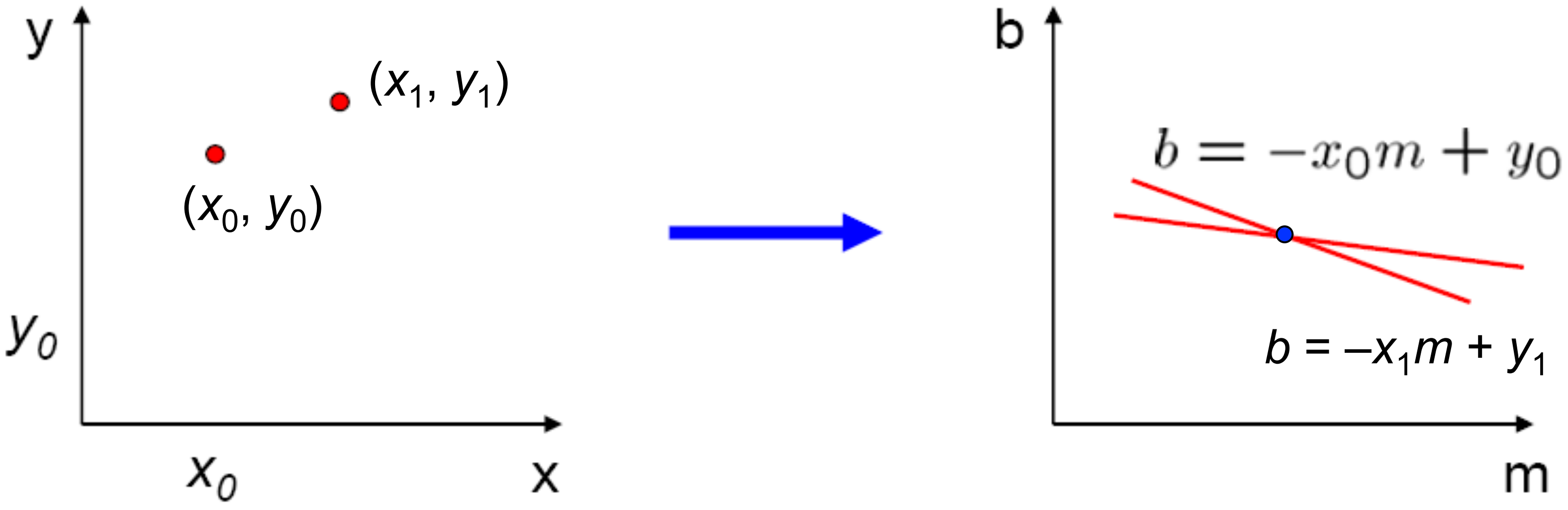


Image space

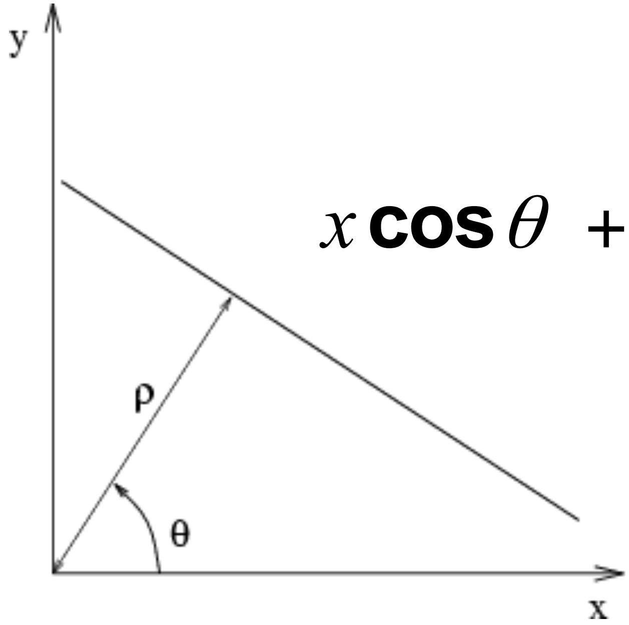
Hough parameter space

Parameter space representation

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m

Parameter space representation

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m
- Alternative: polar representation



$$x\,\mathbf{cos}\,\theta + y\,\mathbf{sin}\,\theta = \rho$$

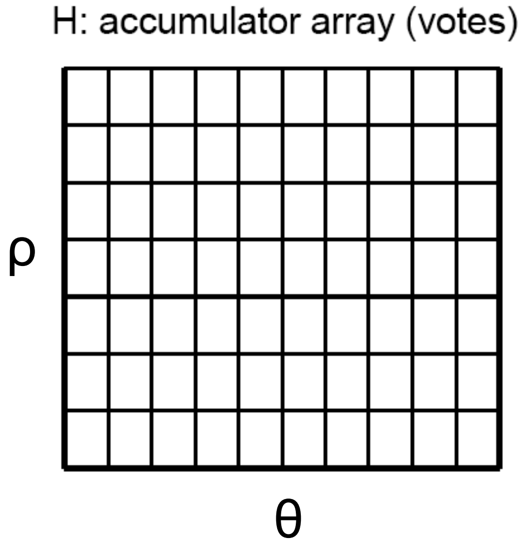
Each point will add a sinusoid in the (θ,ρ) parameter space

Algorithm outline

- Initialize accumulator H to all zeros
- For each edge point (x,y) in the image

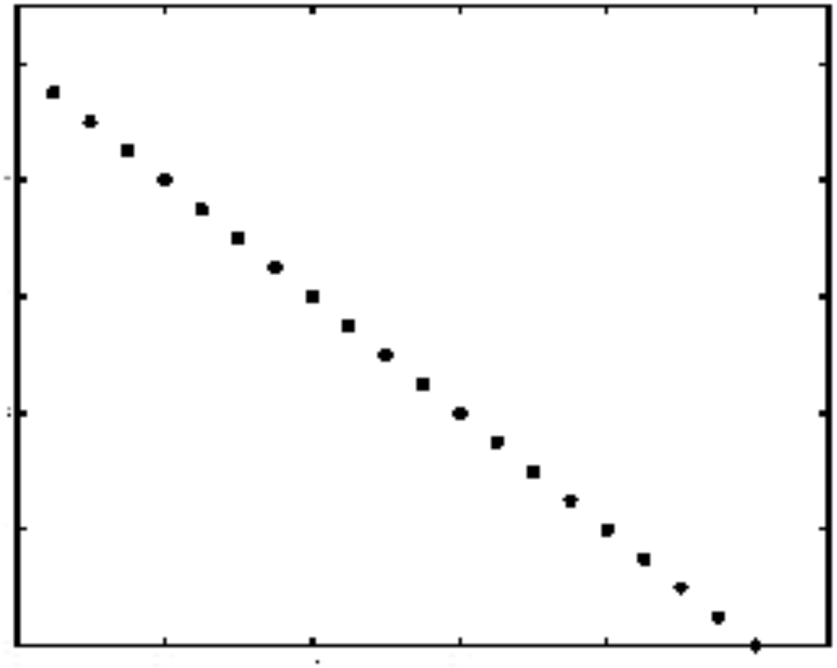
```
For $\theta$ = 0 to 180
    $\rho = x \cos \theta + y \sin \theta$
    $H(\theta, \rho) = H(\theta, \rho) + 1$
end
```

  end

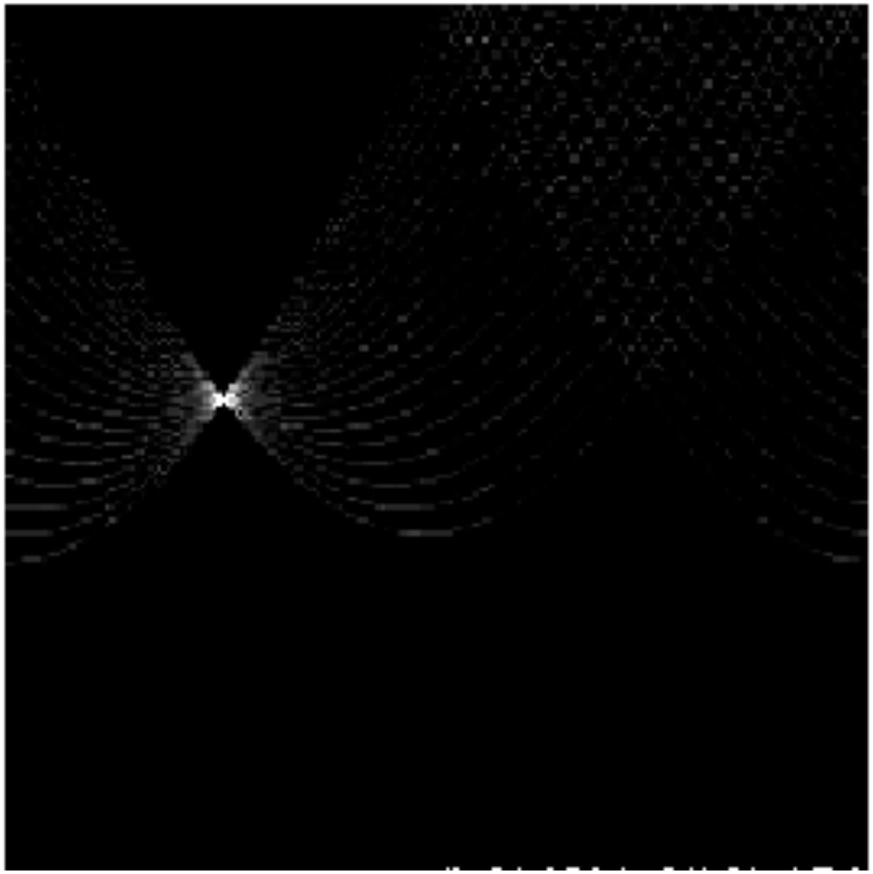


- Find the value(s) of (θ, ρ) where $H(\theta, \rho)$ is a local maximum
  - The detected line in the image is given by $\rho = x \cos \theta + y \sin \theta$

Basic illustration



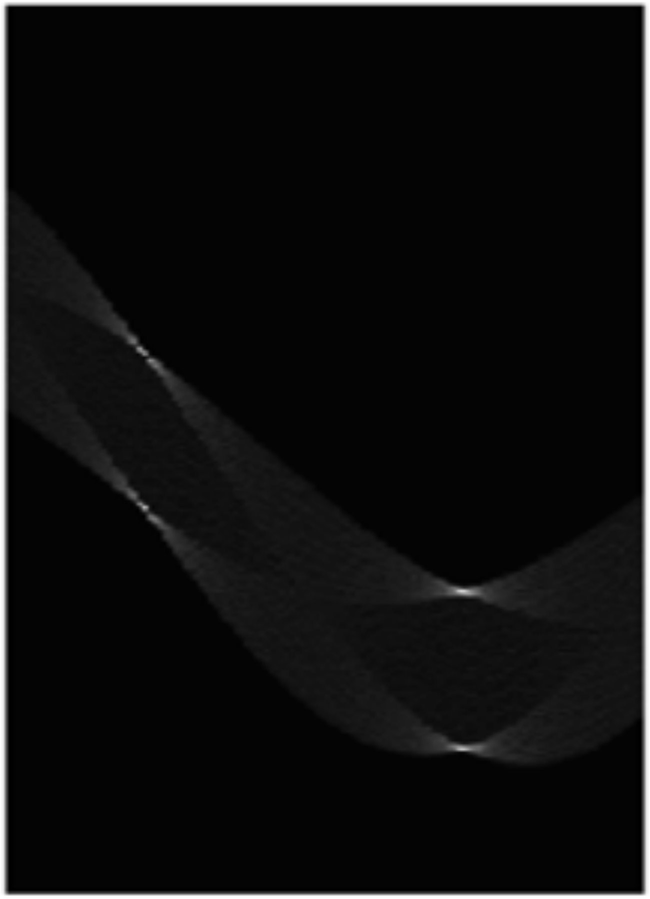
features



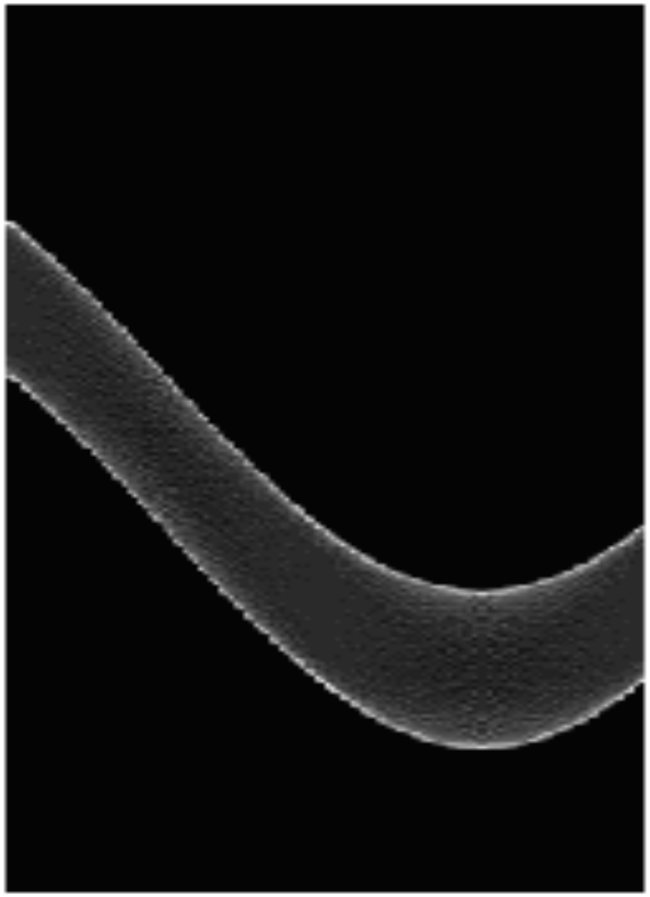
votes

Other shapes

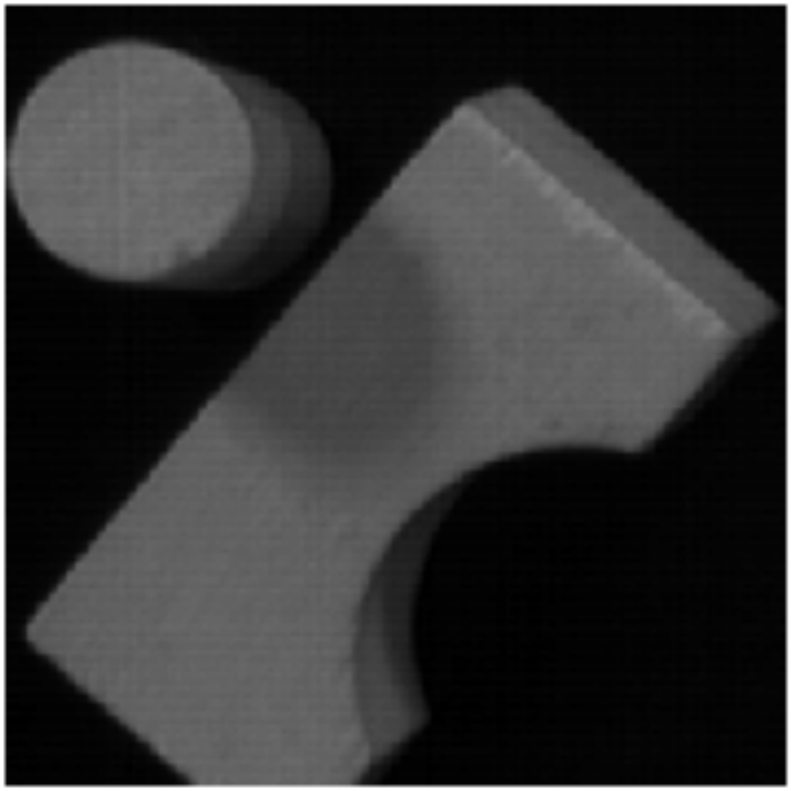
Square

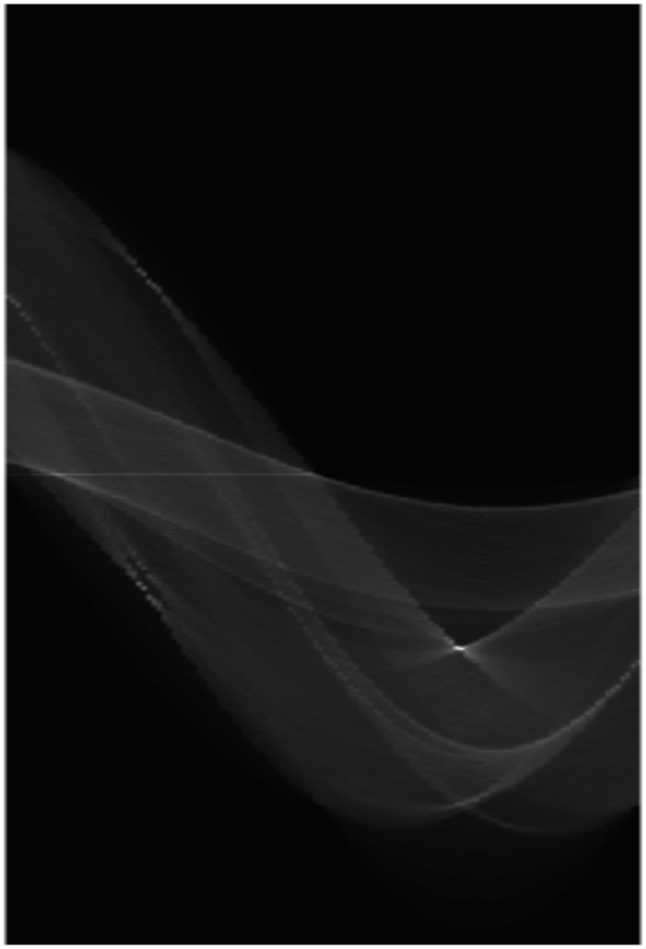


Circle

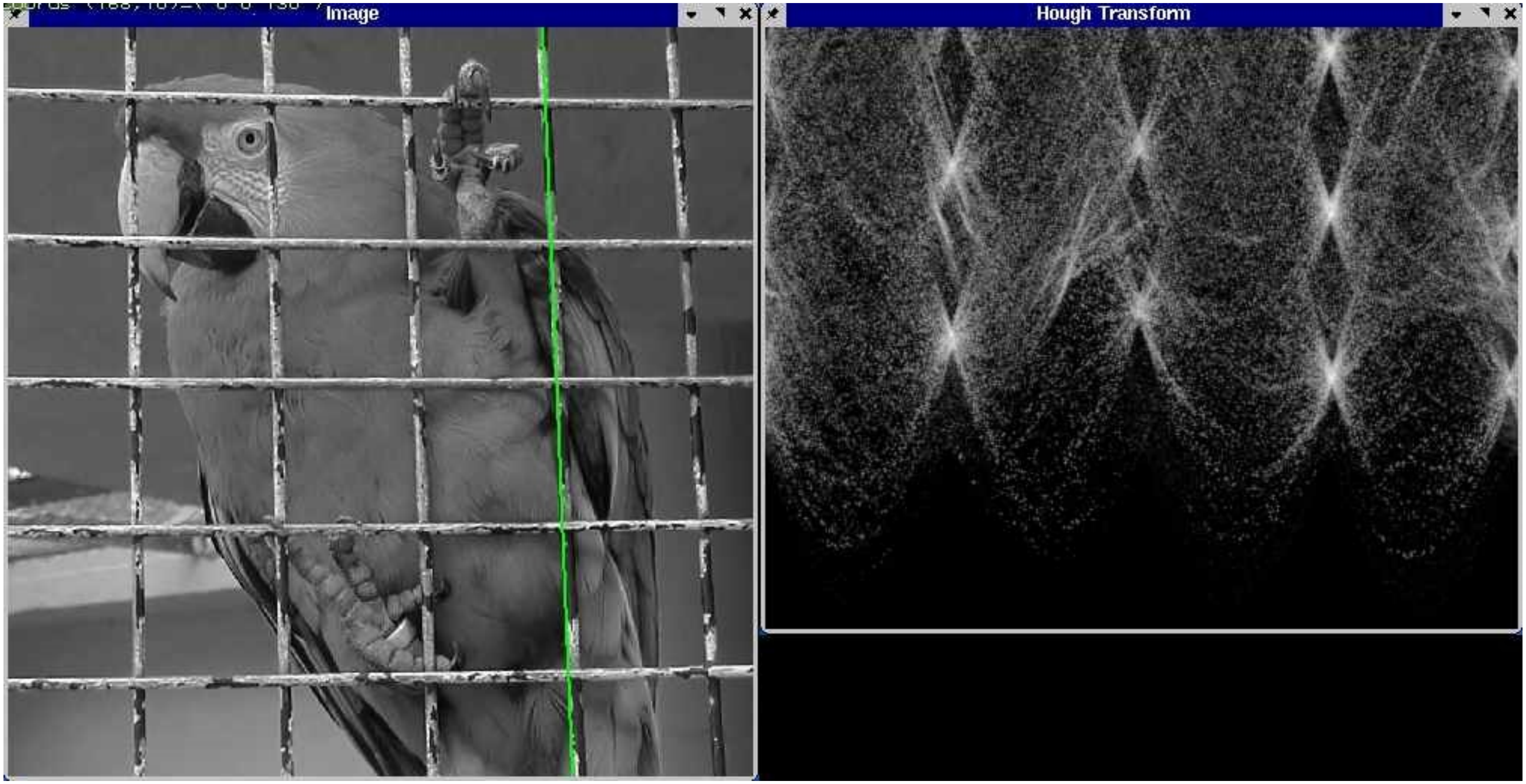


Several lines



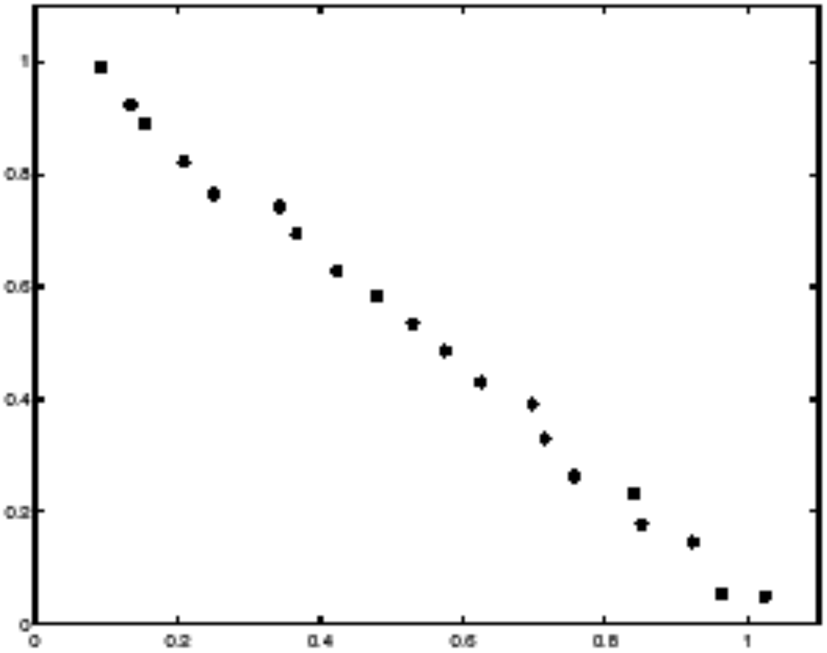


A more complicated image



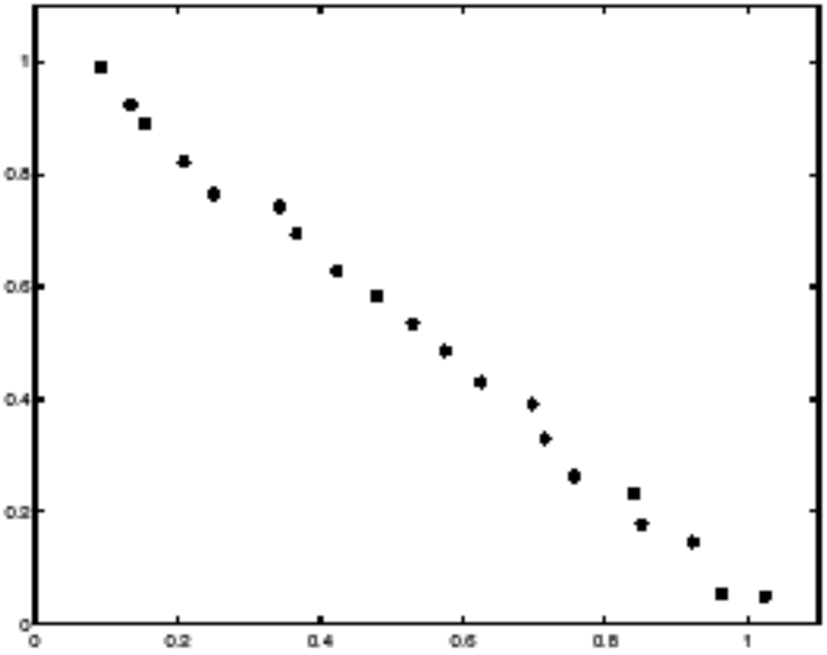
http://ostatic.com/files/images/ss_hough.jpg

Effect of noise

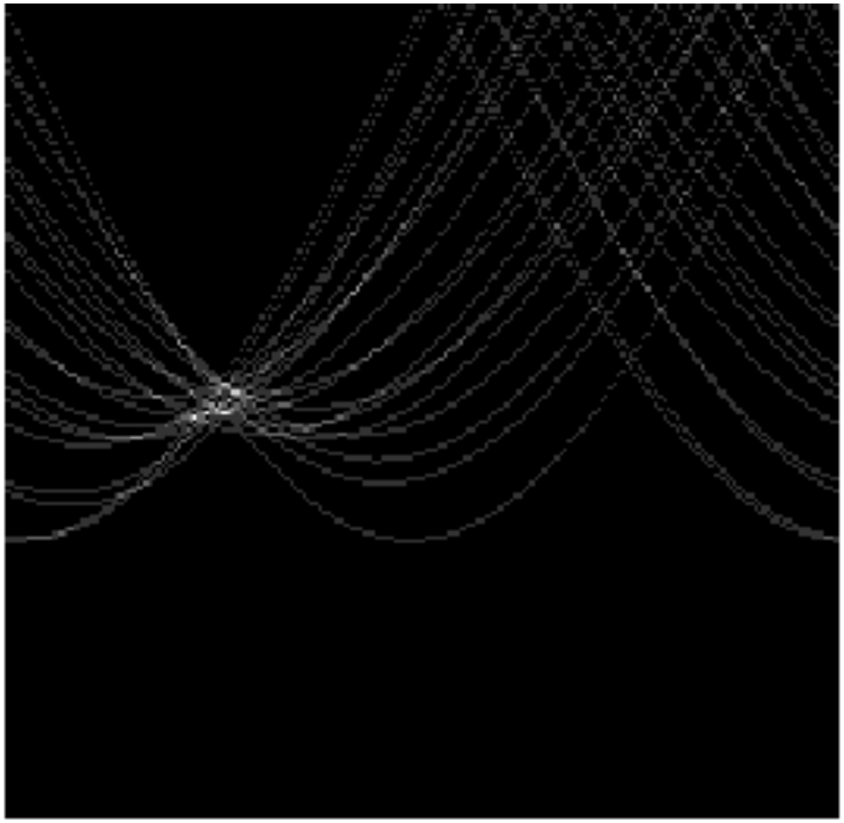


features

Effect of noise



features

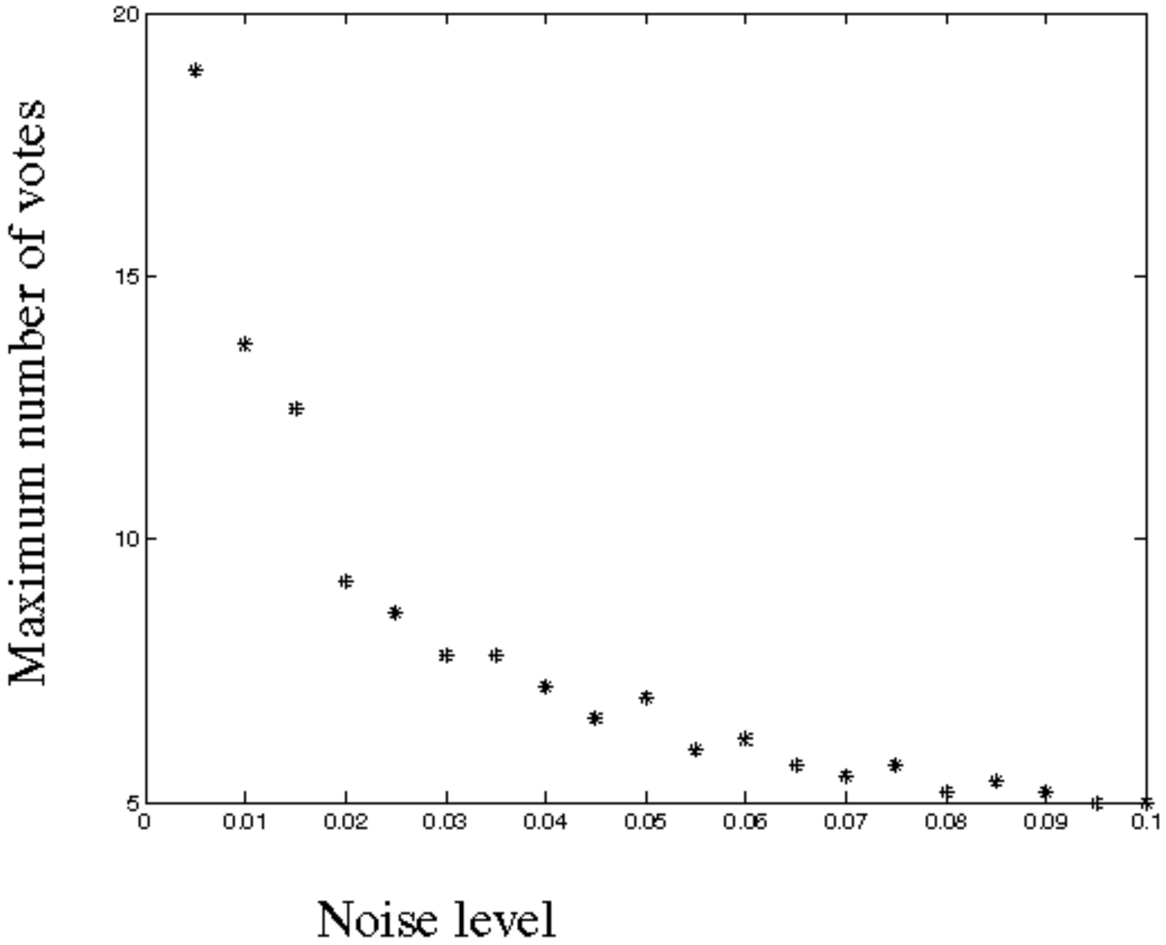


votes

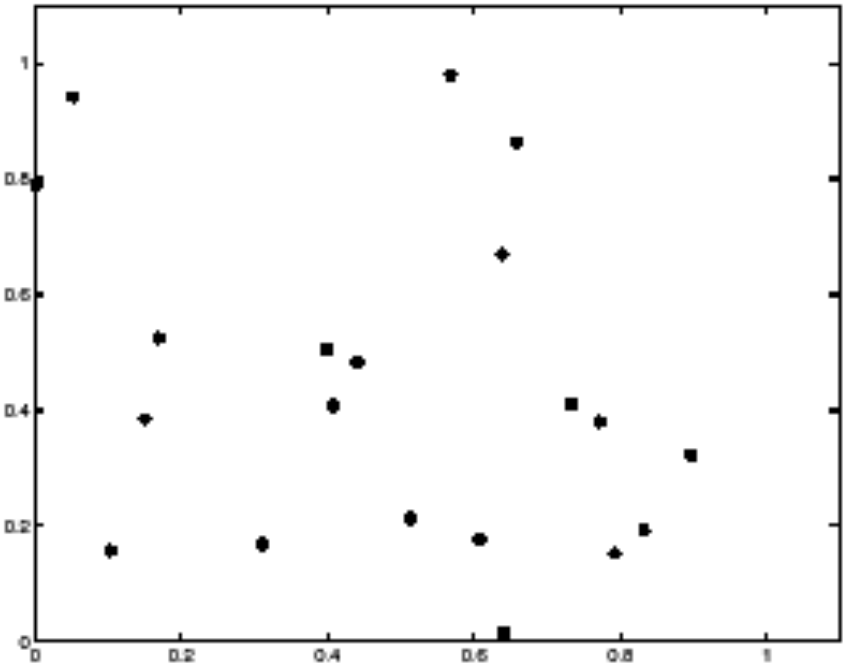
Peak gets fuzzy and hard to locate

Effect of noise

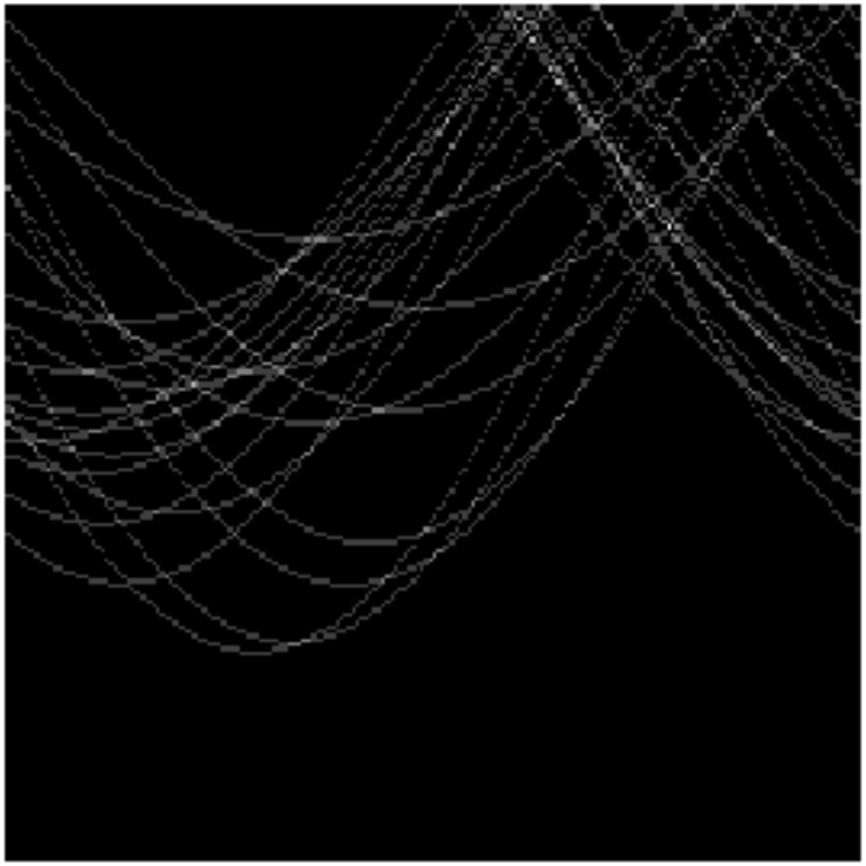
- Number of votes for a line of 20 points with increasing noise:



Random points



features

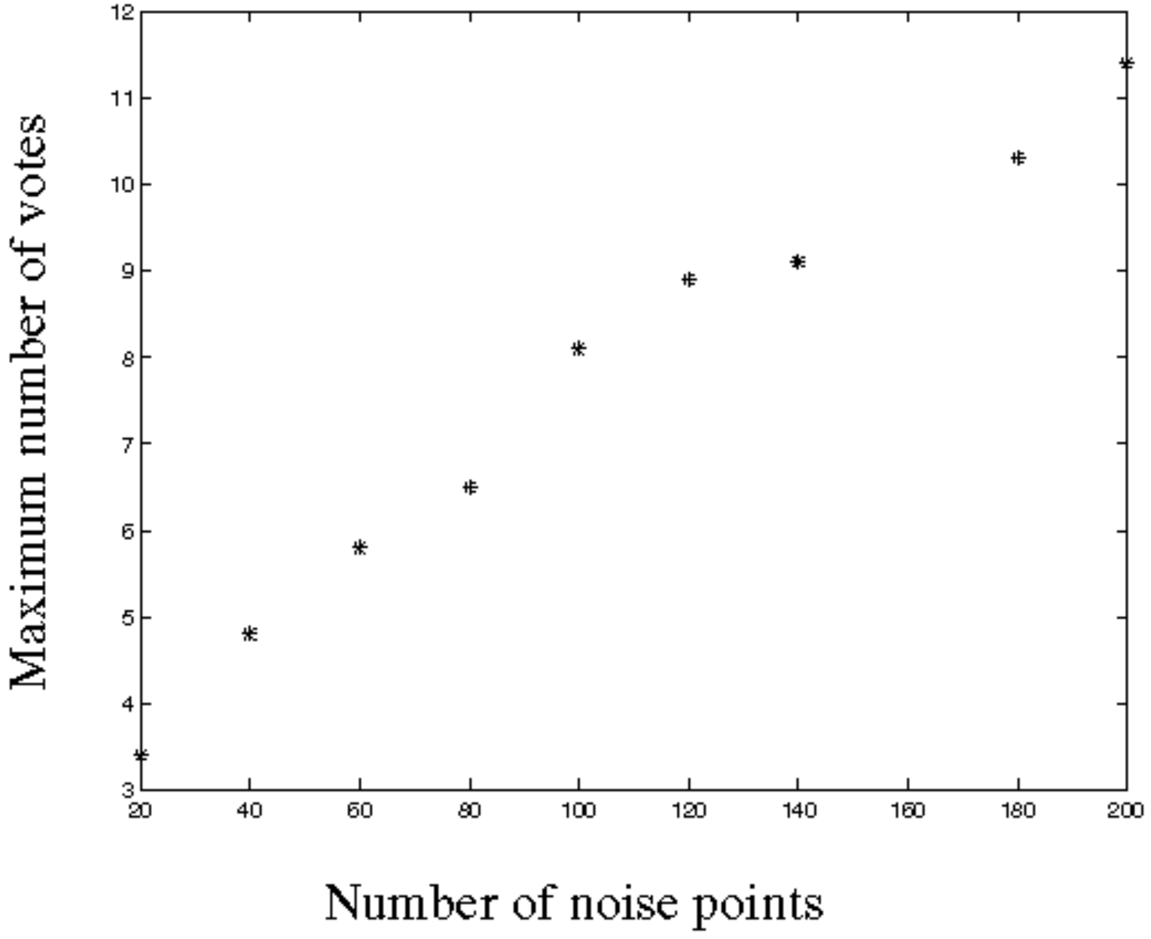


votes

Uniform noise can lead to spurious peaks in the array

Random points

- As the level of uniform noise increases, the maximum number of votes increases too:

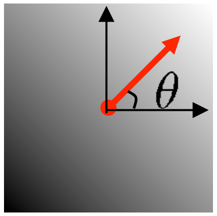


Dealing with noise

- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets
- Increment neighboring bins (smoothing in accumulator array)
- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude

Incorporating image gradients

- Recall: when we detect an edge point, we also know its gradient direction
- But this means that the line is uniquely determined!



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

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

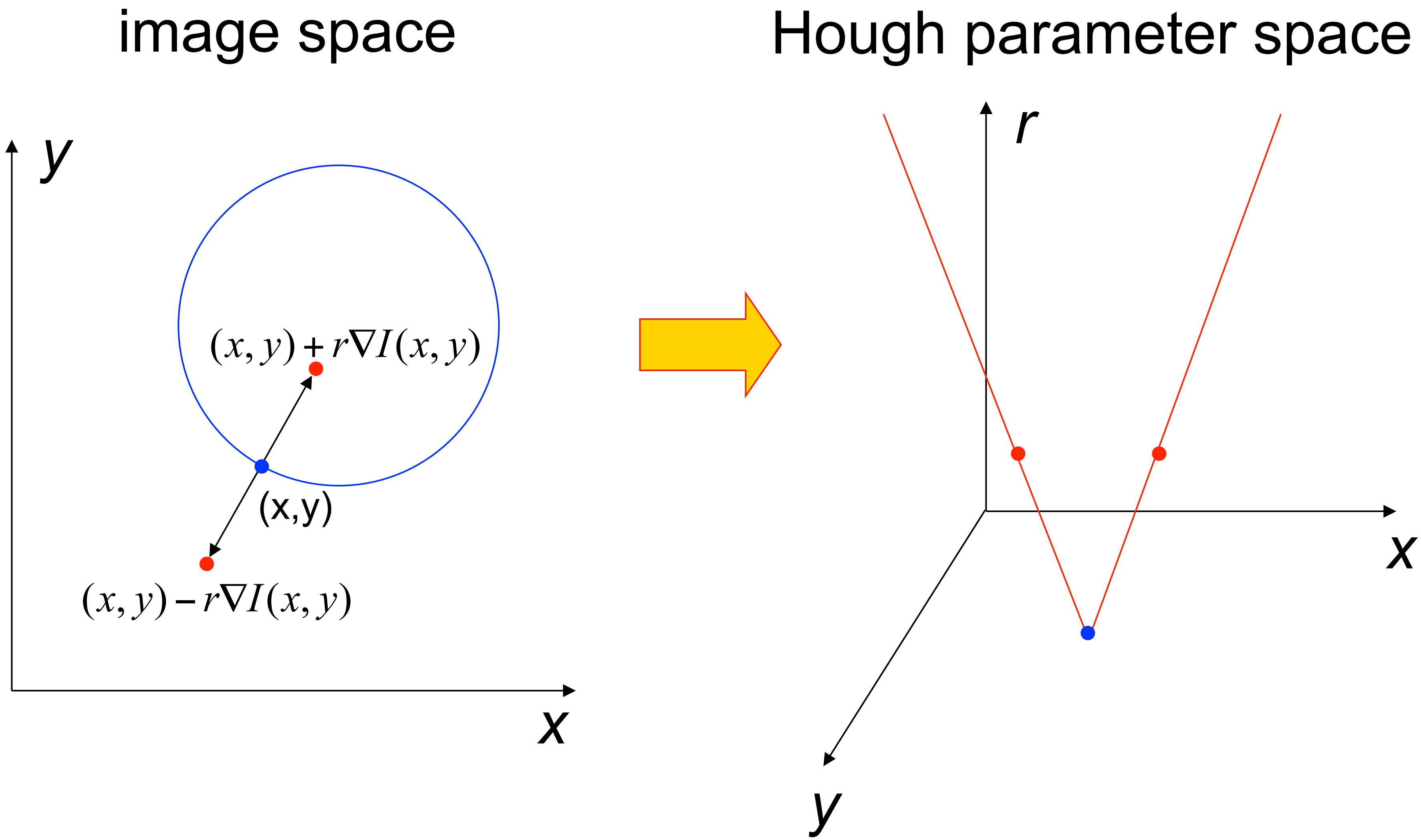
- Modified Hough transform:

```
For each edge point (x,y)
    θ = gradient orientation at (x,y)
    ρ = x cos θ + y sin θ
    H(θ, ρ) = H(θ, ρ) + 1
end
```

Hough transform for circles

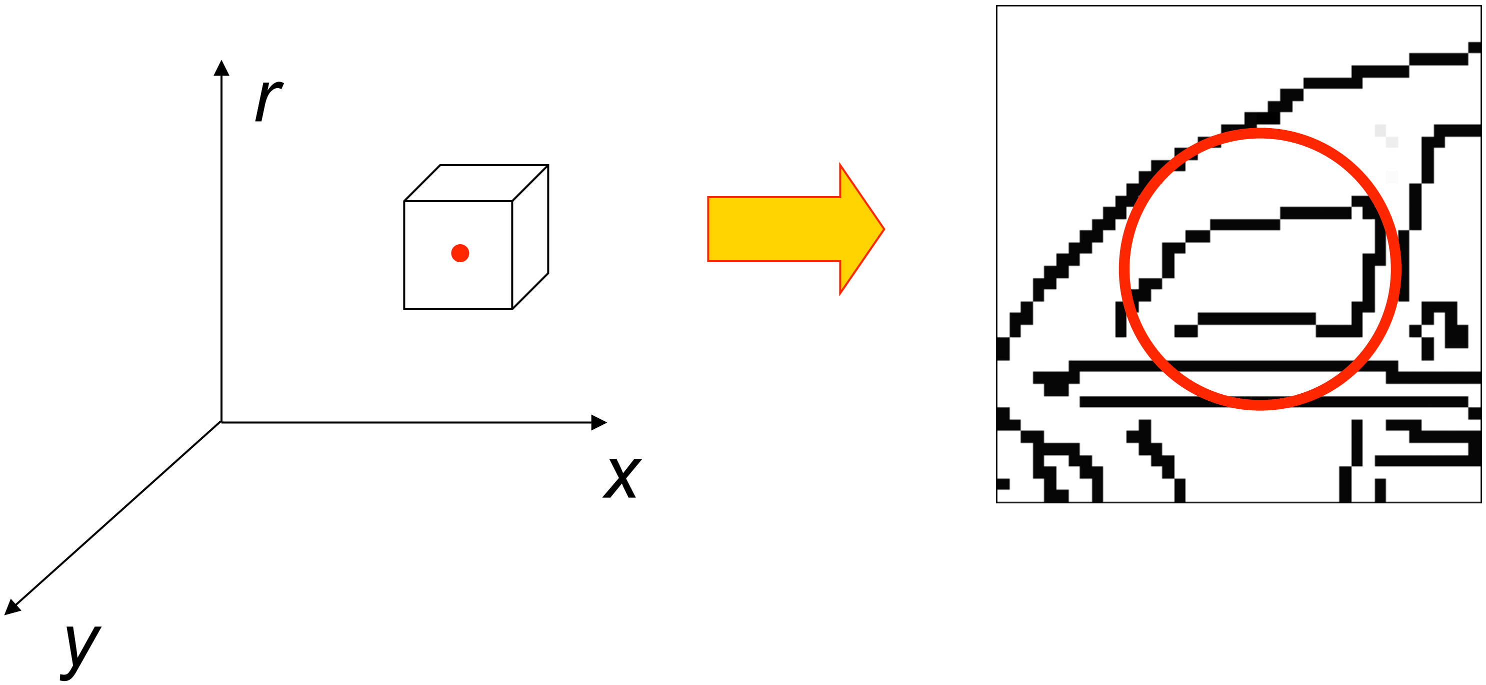
- How many dimensions will the parameter space have?
- Given an oriented edge point, what are all possible bins that it can vote for?

Hough transform for circles



Hough transform for circles

- Conceptually equivalent procedure: for each (x,y,r), draw the corresponding circle in the image and compute its “support”



Is this more or less efficient than voting with features?

Finding straight lines

- Another solution: get connected components of pixels and check for straightness

Finding line segments using connected components

1. Compute canny edges
   - Compute: gx, gy (DoG in x,y directions)
   - Compute: theta = atan(gy / gx)
2. Assign each edge to one of 8 directions
3. For each direction d, get edgelets:
   - find connected components for edge pixels with directions in {d-1, d, d+1}
4. Compute straightness and theta of edgelets using eig of x,y 2nd moment matrix of their points

$$\mathbf{M} = \begin{bmatrix} \sum (x-\mu_x)^2 & \sum (x-\mu_x)(y-\mu_y) \\ \sum (x-\mu_x)(y-\mu_y) & \sum (y-\mu_y)^2 \end{bmatrix} \qquad [v,\lambda] = \mathrm{eig}(\mathbf{M})$$

Larger eigenvector

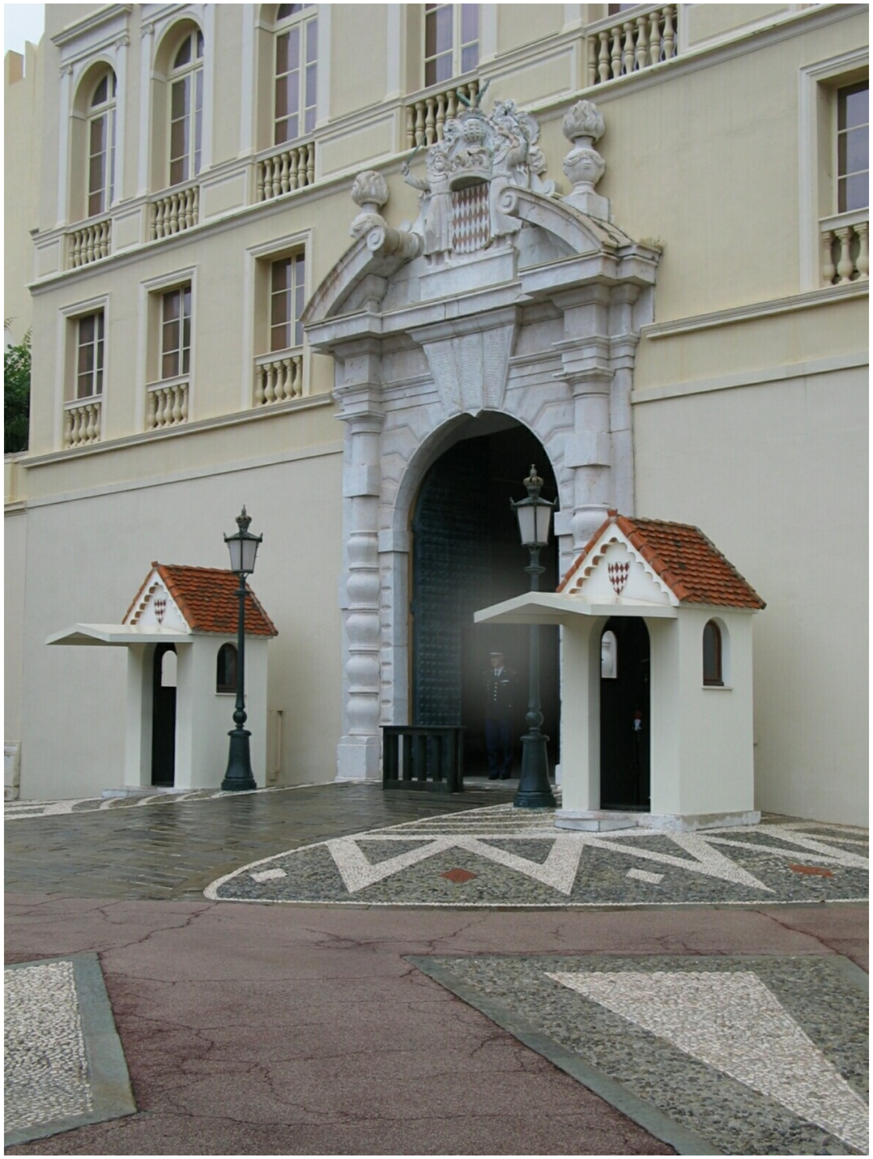
$$\theta = \operatorname{atan} 2(v(2,2), v(1,2))$$

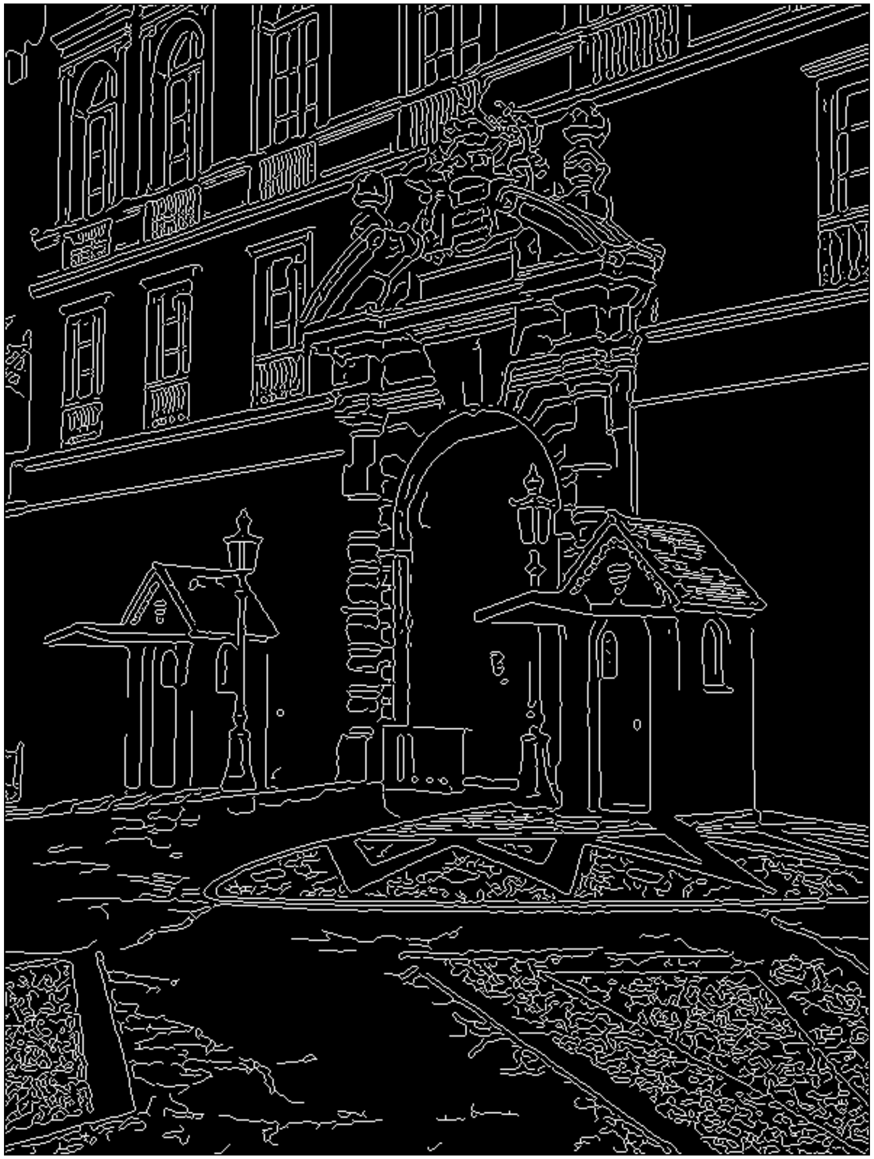
$$conf = \lambda_2 / \lambda_1$$

5. Threshold on straightness, store segment

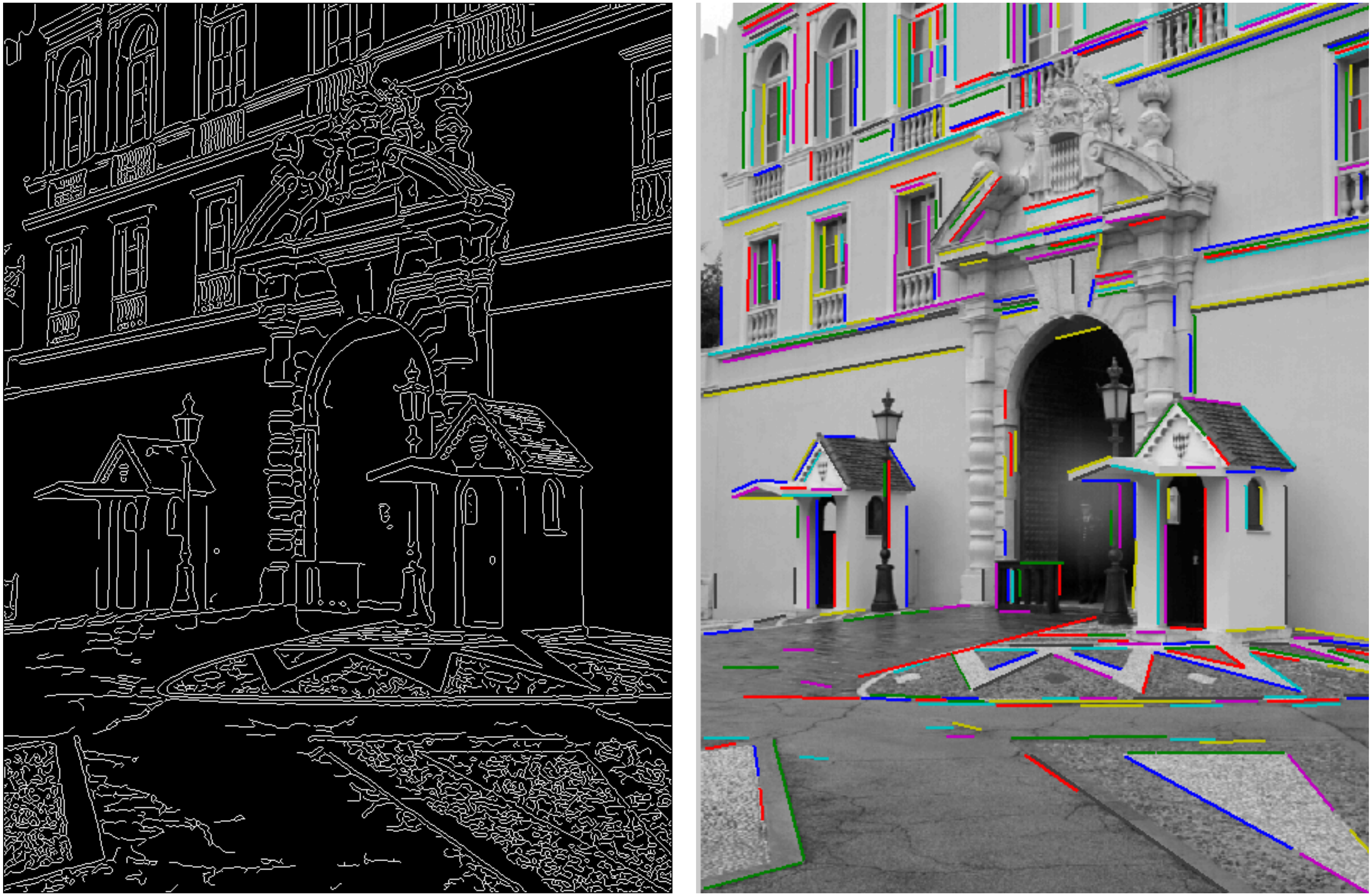
Slides from Derek Hoiem

1. Image → Canny

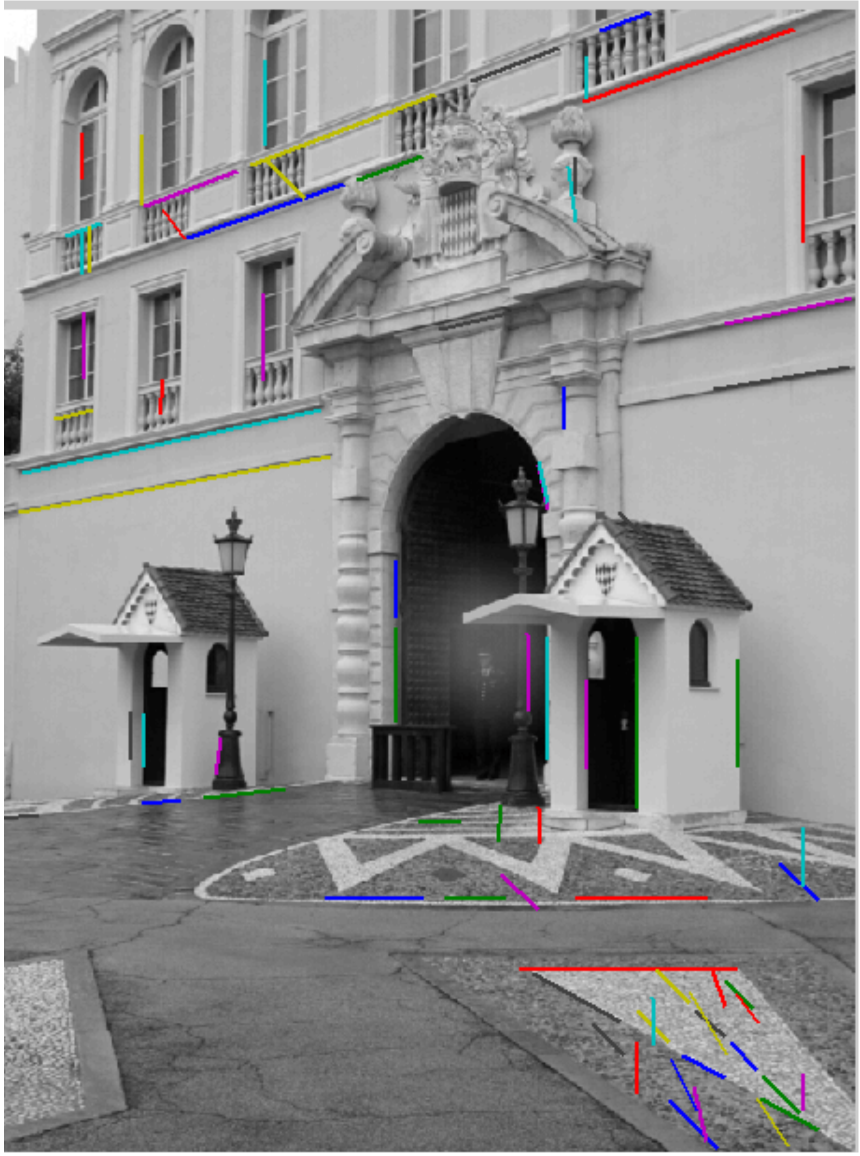




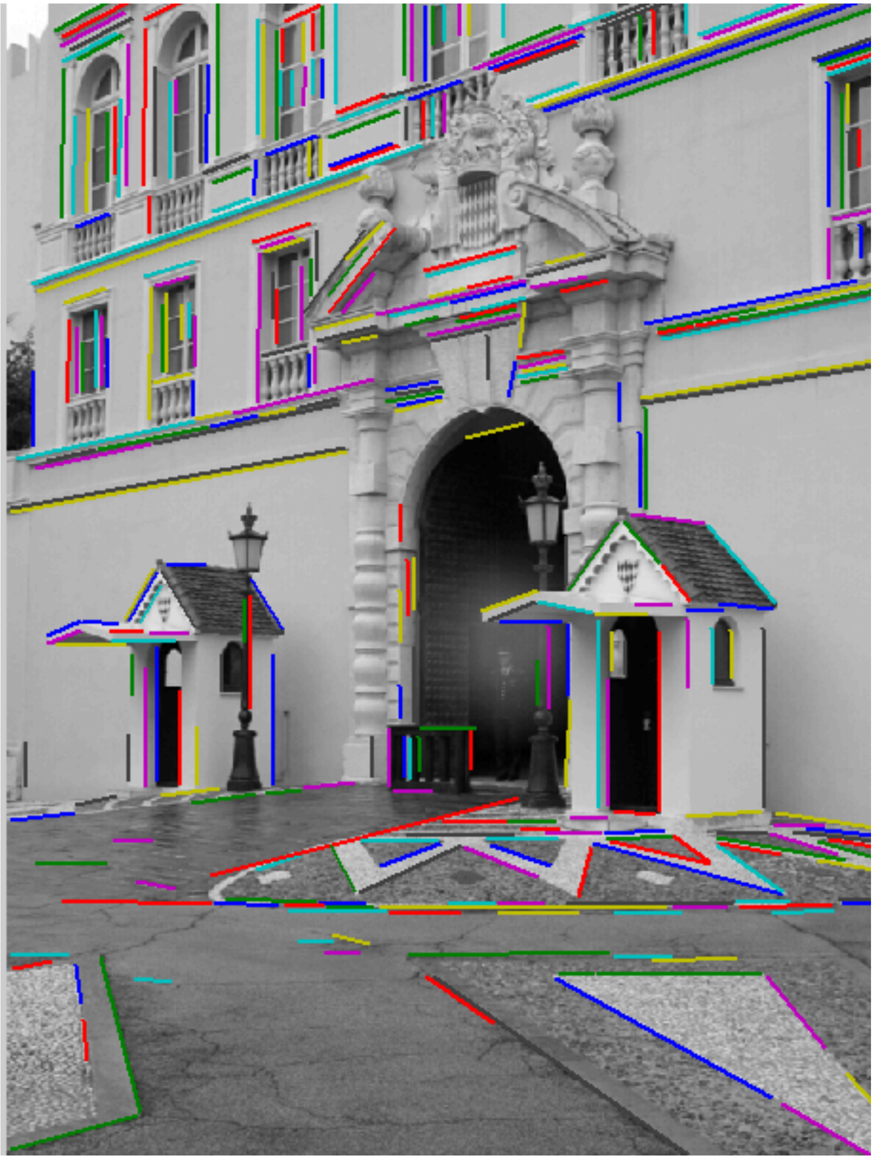
2. Canny lines → … →straight edges



Comparison



Hough Transform Method

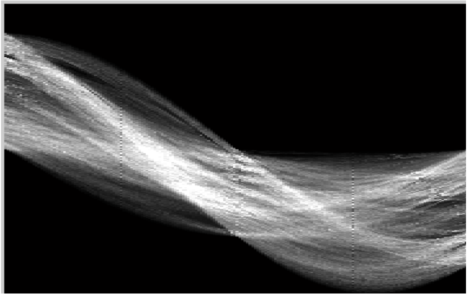


Connected Components Method

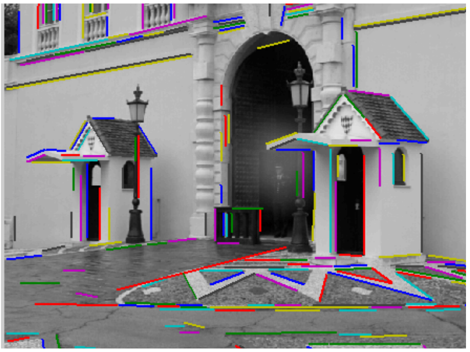
Things to remember

- Canny edge detector = smooth → derivative → thin → threshold → link

- Generalized Hough transform = points vote for shape parameters



- Straight line detector = canny + gradient orientations → orientation binning → linking → check for straightness



Next classes

- Generalized Hough Transform
- Distance Transform
- Chamfer Matching

Questions