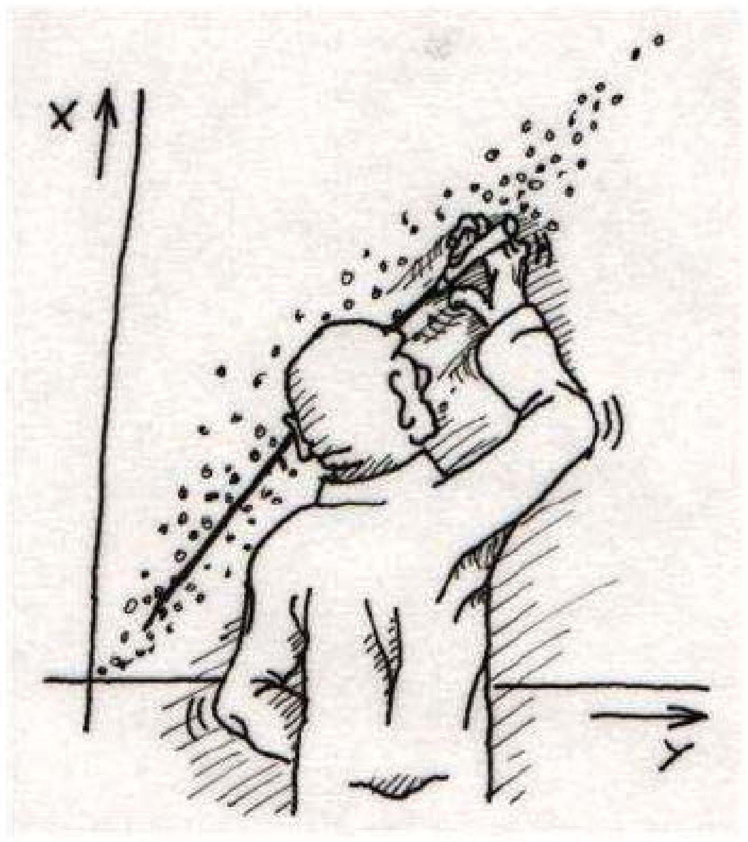
Model Fitting: The Hough transform I

Guido Gerig, CS6640 Image Processing, Utah



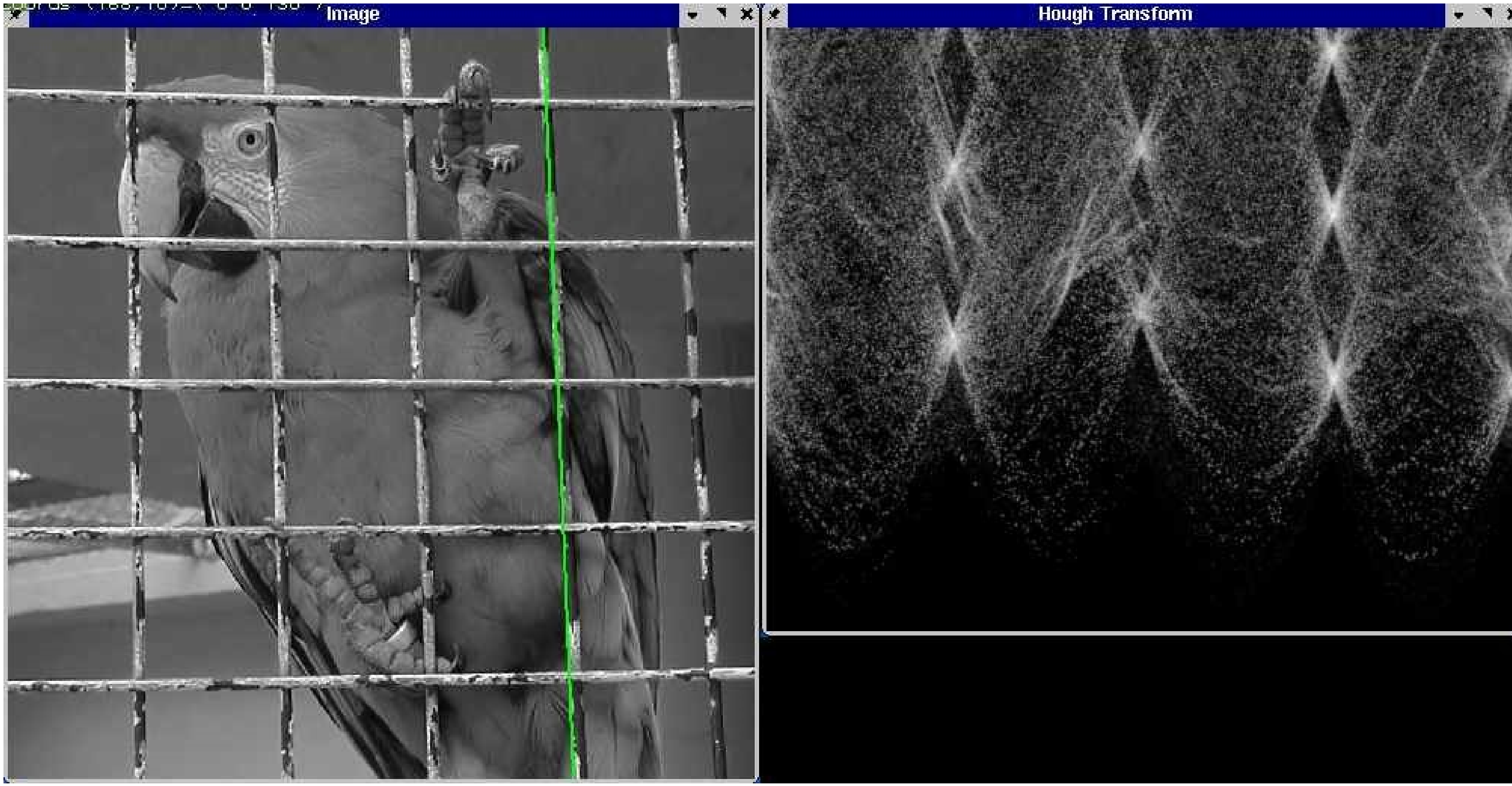
Materials:

DIP book: 10.2, pages 733-738

Handouts G. Gerig

Credit to most slides: Svetlana Lazebnik, University of Illinois at Urbana-Champaign (slides)

Fitting: The Hough transform



http://web.engr.illinois.edu/~slazebni/spring14/lec11_hough.pptx

Voting schemes

- Let each feature vote for all the models that are compatible with it
- Hopefully the noise features will not vote consistently for any single model
- Missing data doesn’t matter as long as there are enough features remaining to agree on a good model

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

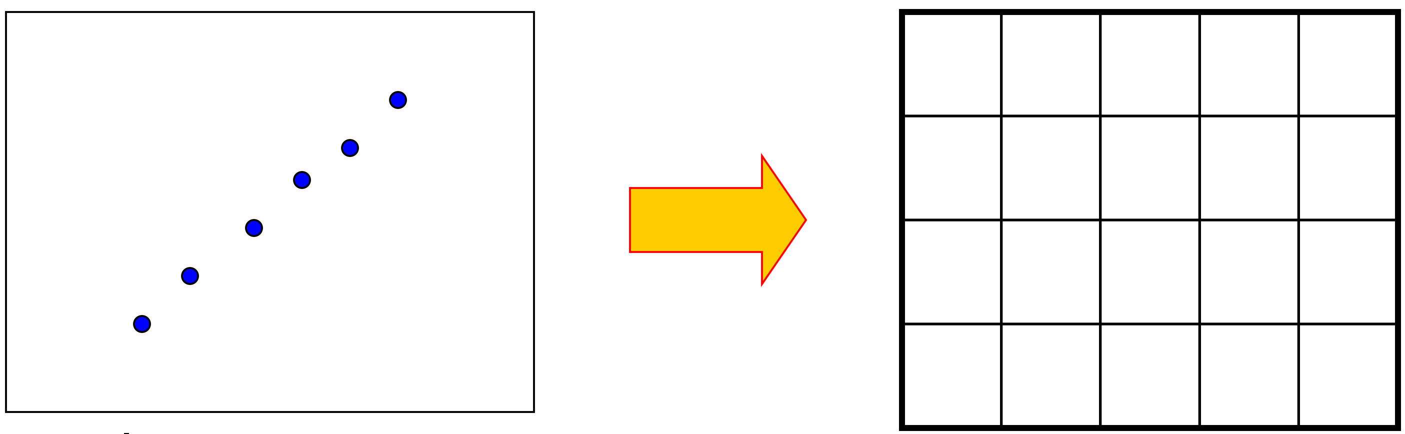


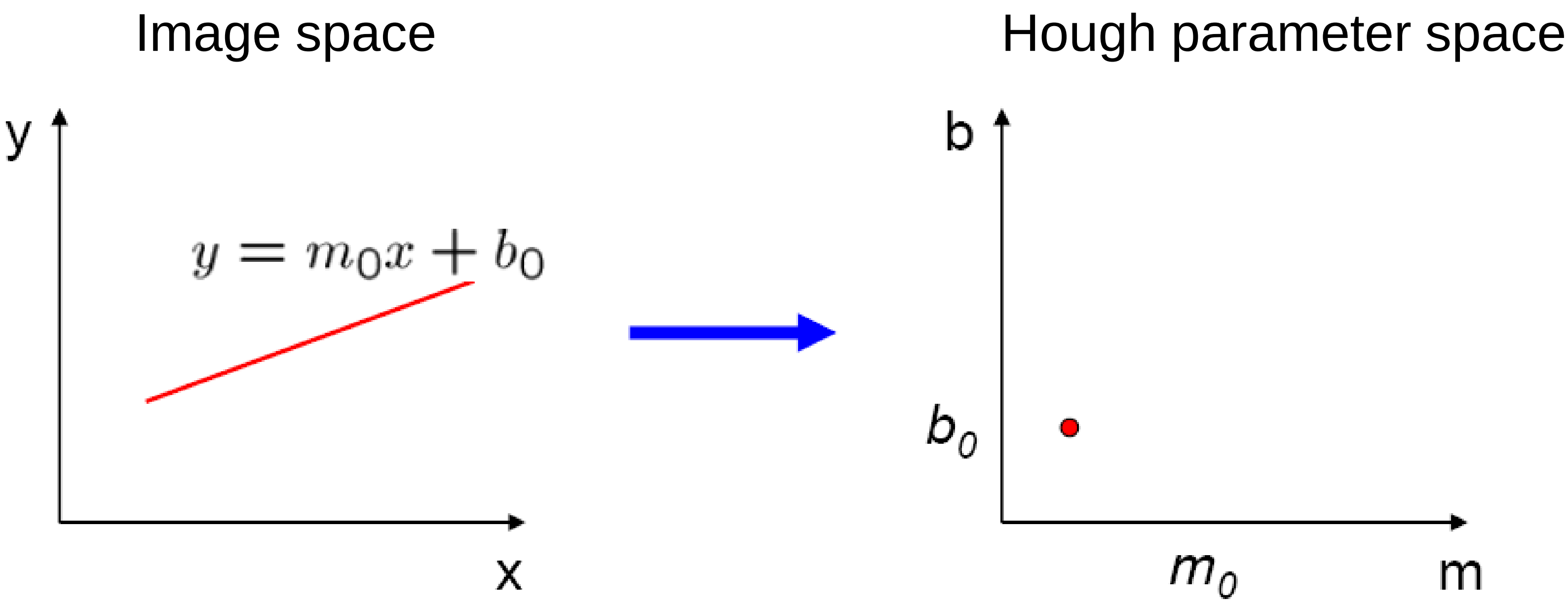
Image space

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



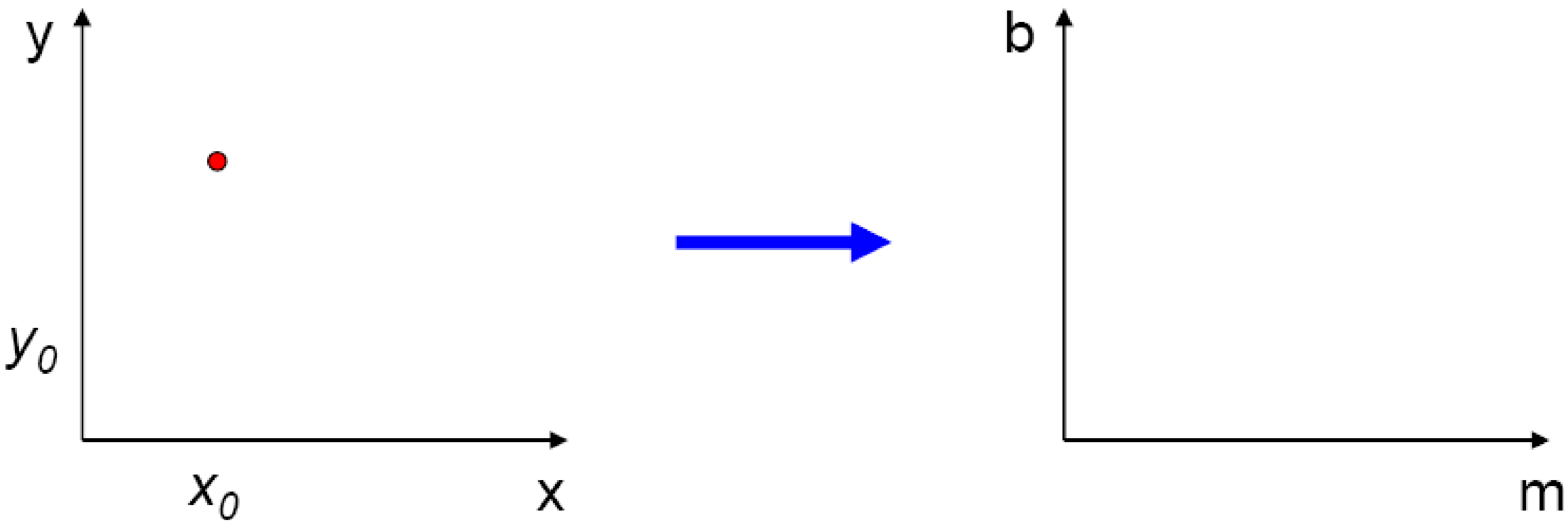
Source: S. Seitz

Parameter space representation

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

Image space

Hough parameter space

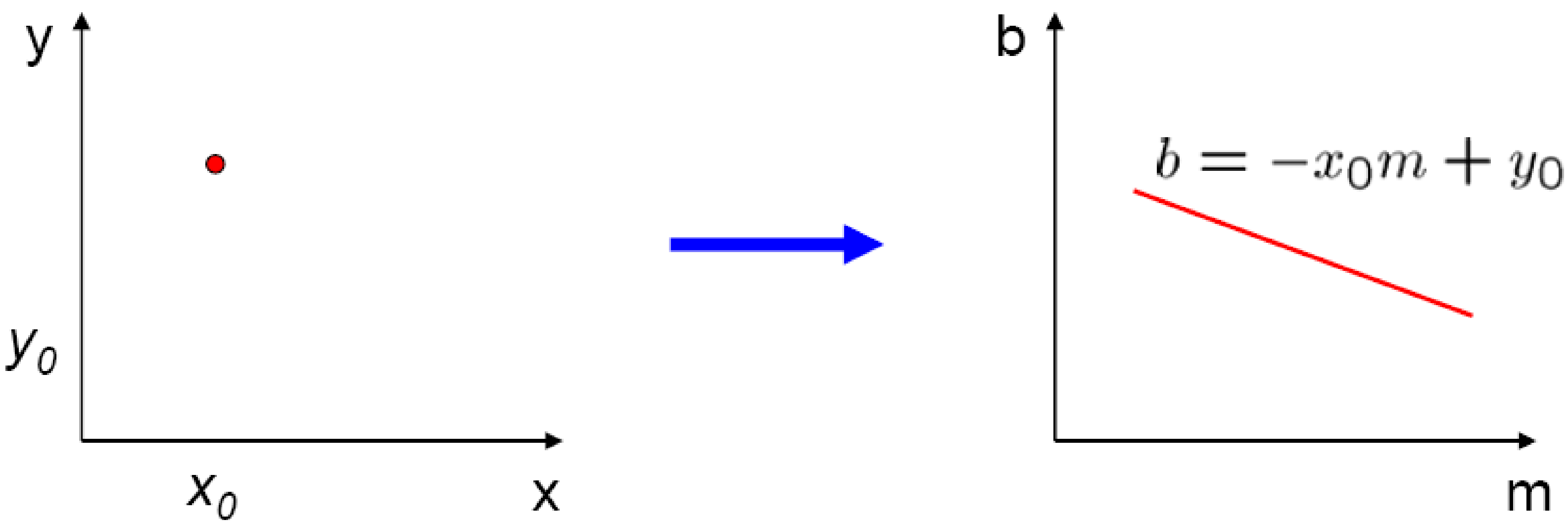


Parameter space representation

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

Image space

Hough parameter space

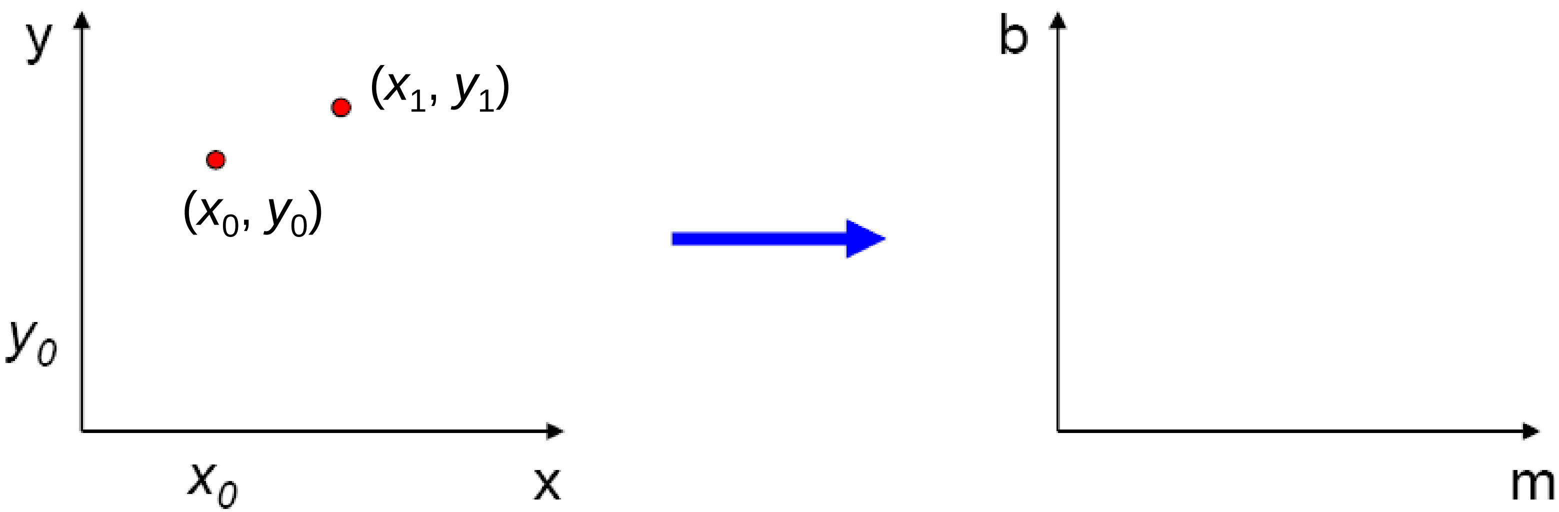


Parameter space representation

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

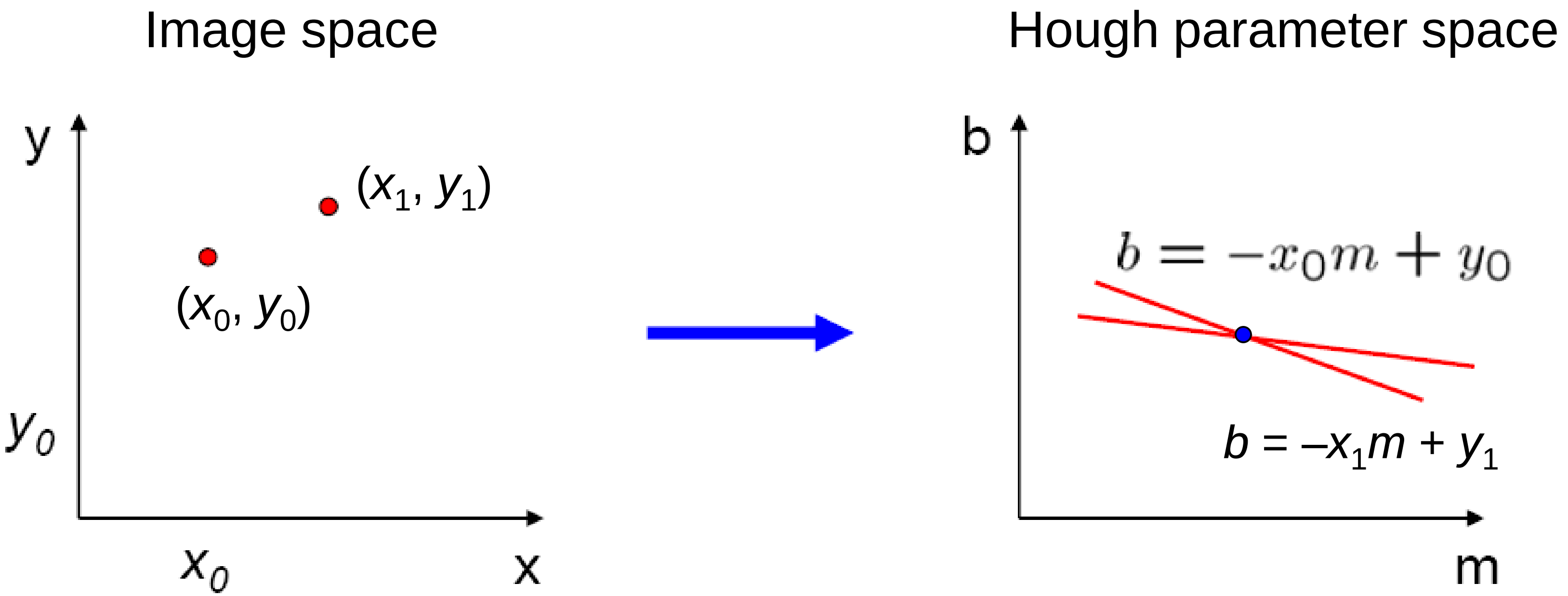
Image space

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 $b = -x_0m + y_0$ and $b = -x_1m + y_1$

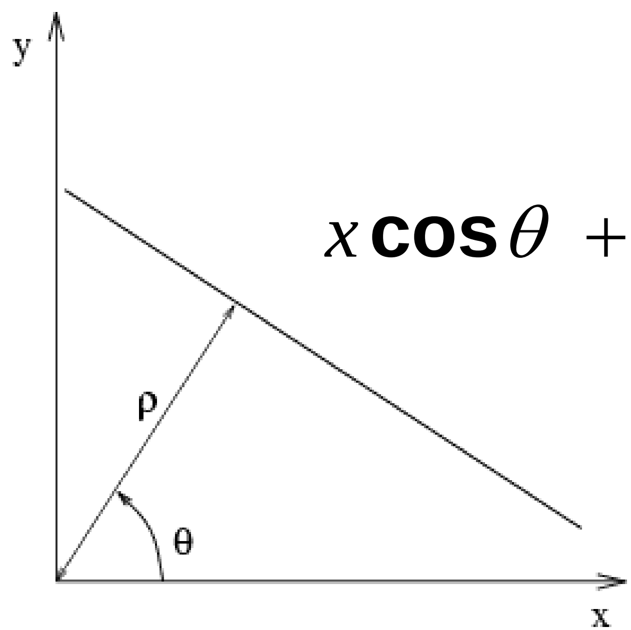


Parameter space representation

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

Parameter space representation

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



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

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

Properties of polar representation

See handwritten notes G. Gerig

Web-based demonstrations:

- Duality of image space and parameter space: Interactive demo: http://users.cs.cf.ac.uk/Paul.Rosin/CM3102/LABS/dual2/hough.html
- Similar to above: http://www.dis.uniroma1.it/~iocchi/slides/icra2001/java/hough.html
- Select among images, then show results: http://users.ecs.soton.ac.uk/msn/book/new_demo/hough/
- Load own images, then run: http://peaks.informatik.uni-erlangen.de/peaks/cv/Hough.html

Algorithm outline

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

  For θ = 0 to 180

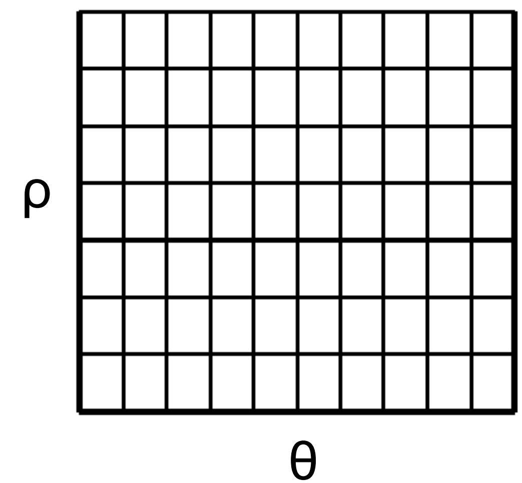
  $\rho = x \cos \theta + y \sin \theta$

  $H(\theta, \rho) = H(\theta, \rho) + 1$

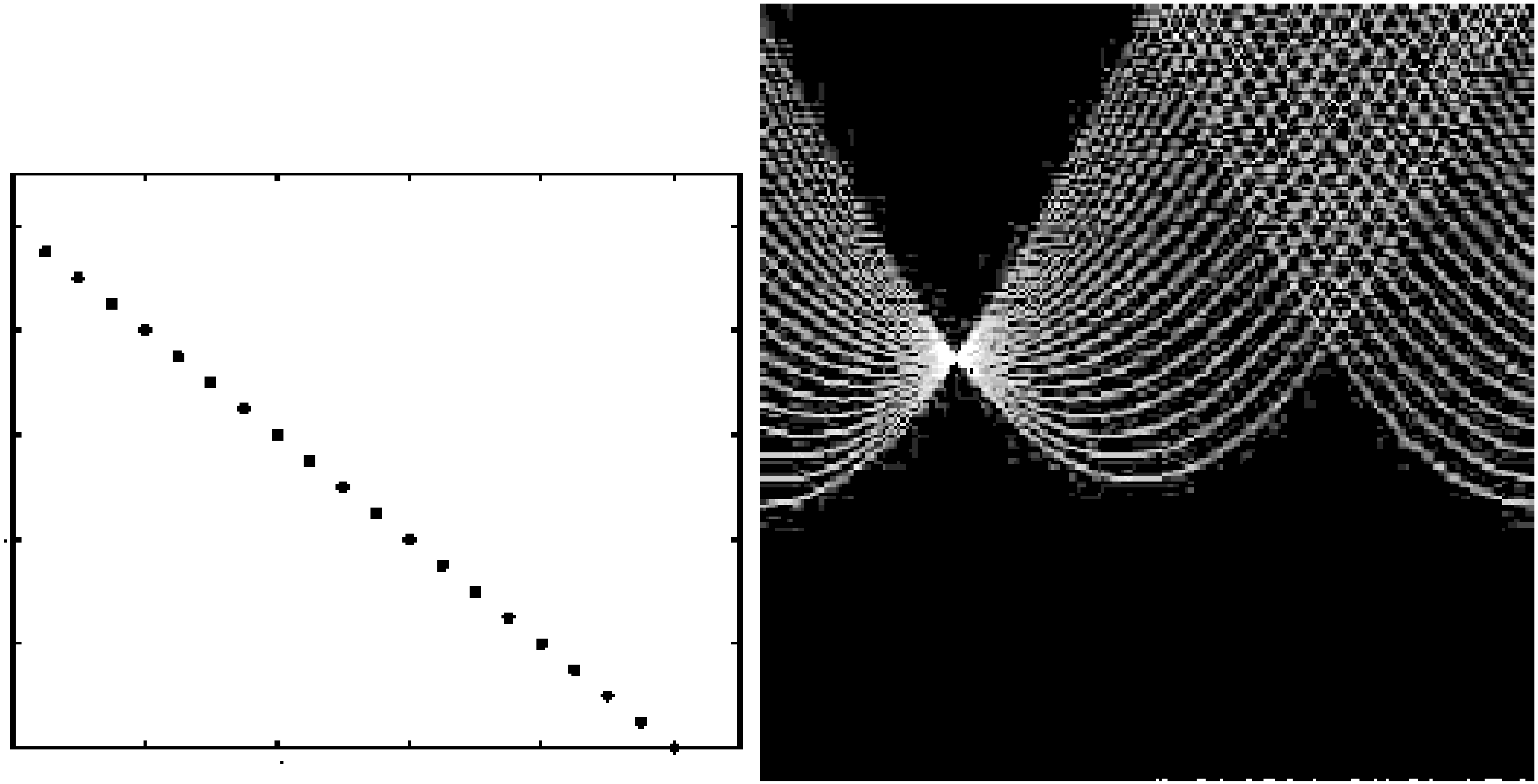
  end

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

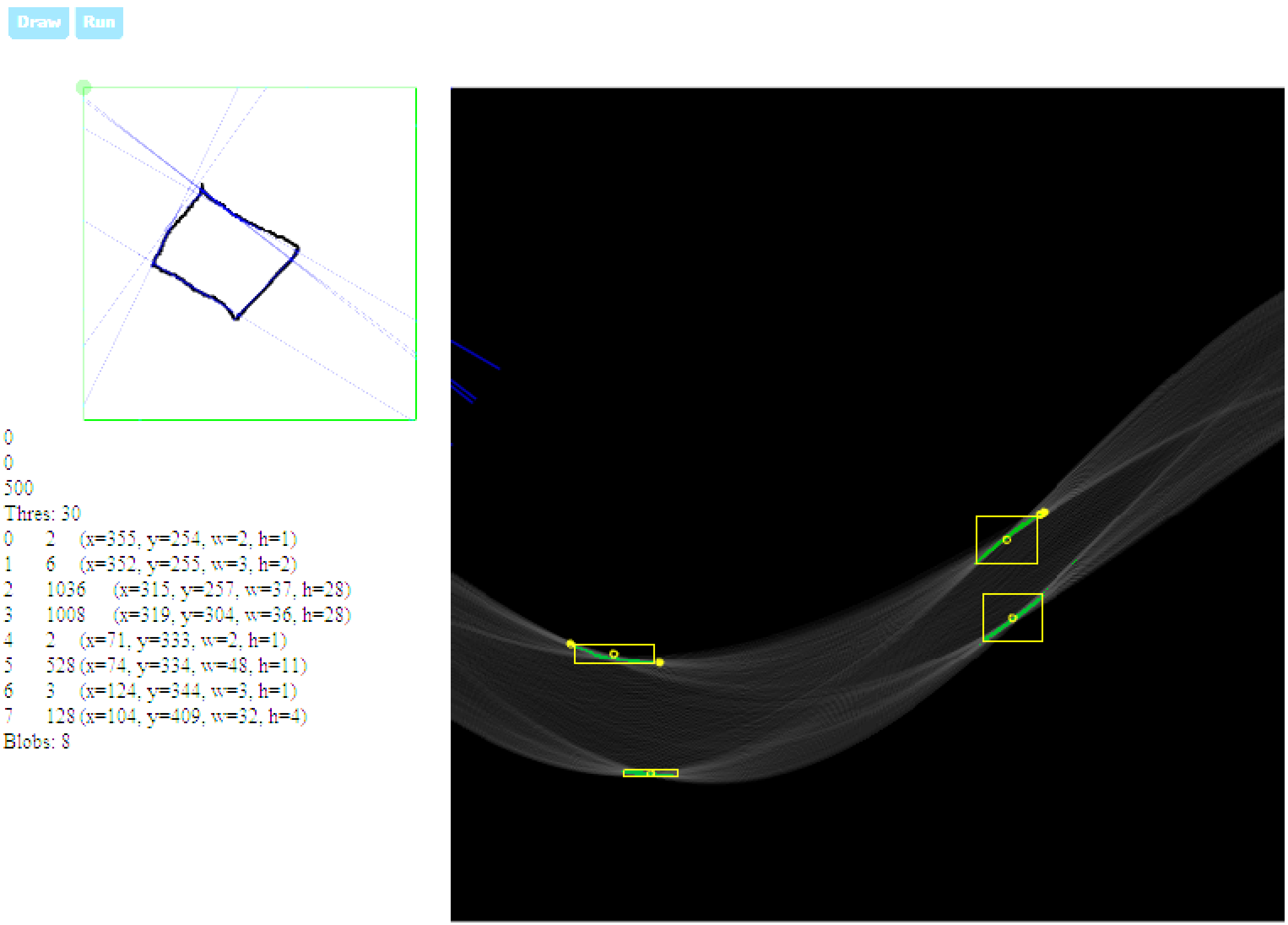
H: accumulator array (votes)



Basic illustration



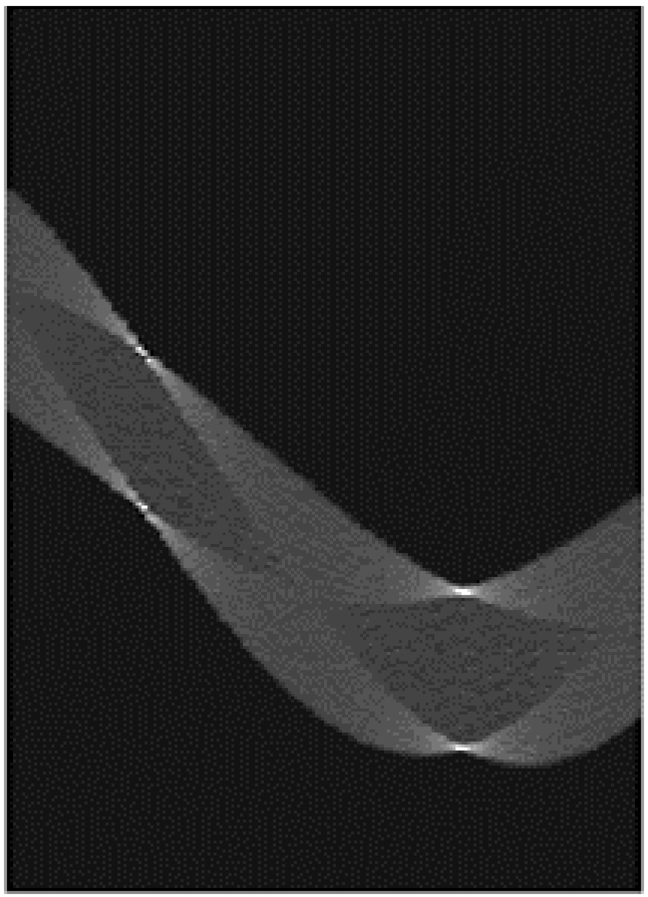
Detection of Clusters



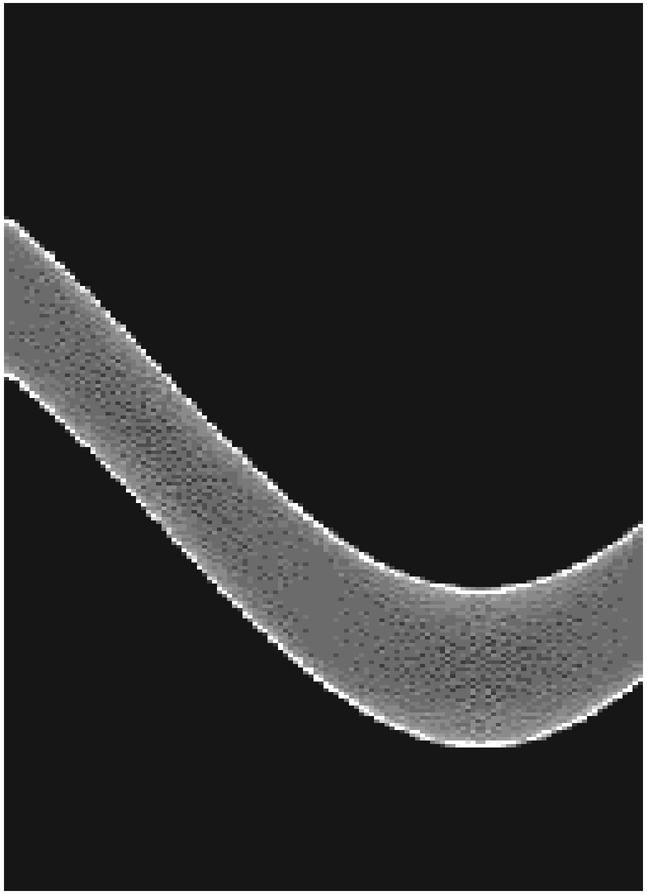
Hough transform demo:
http://liquify.eu/swf/HoughTransform.swf

Other shapes

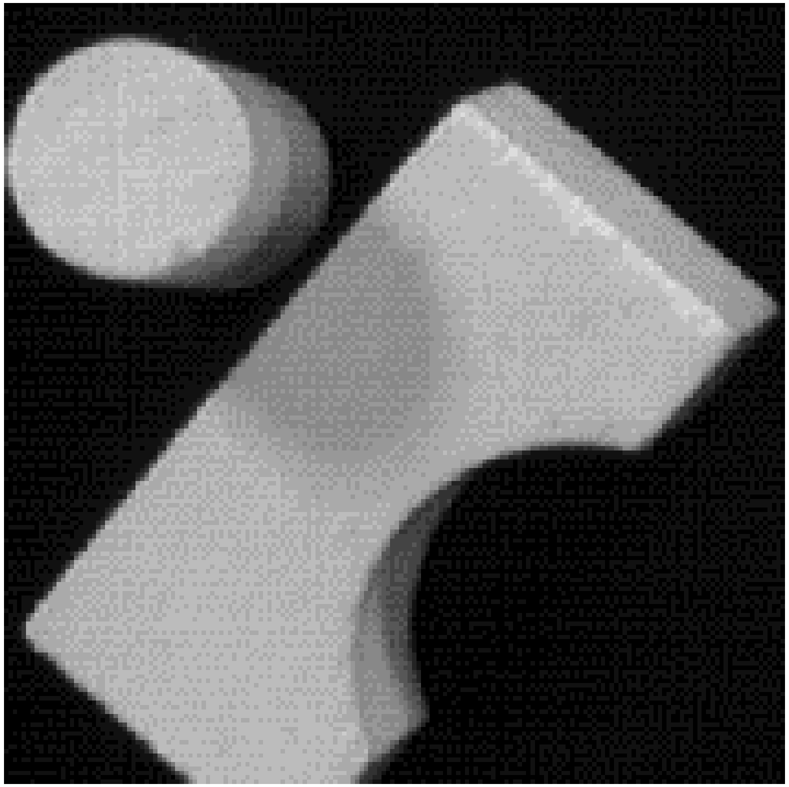
Square

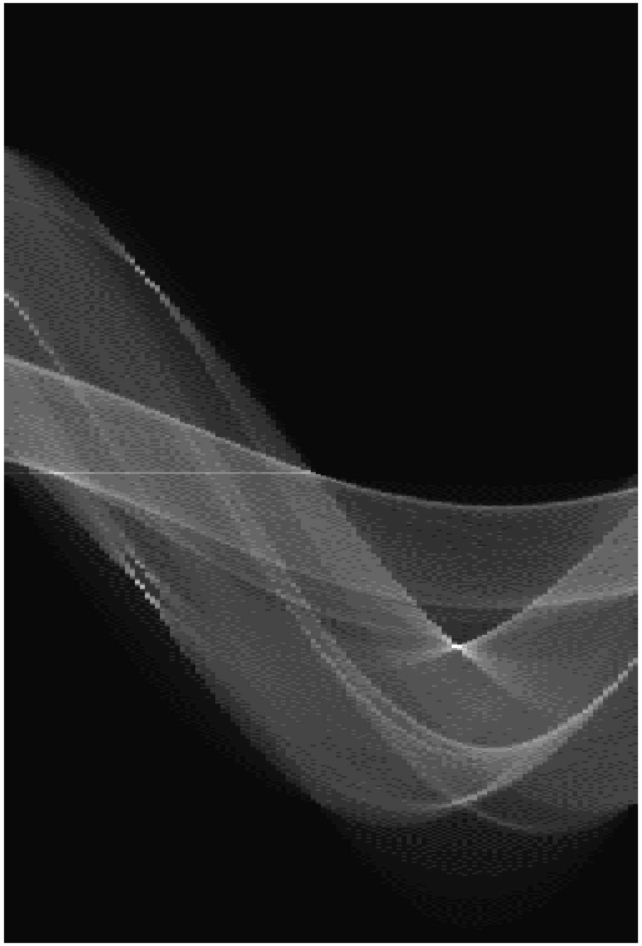


Circle

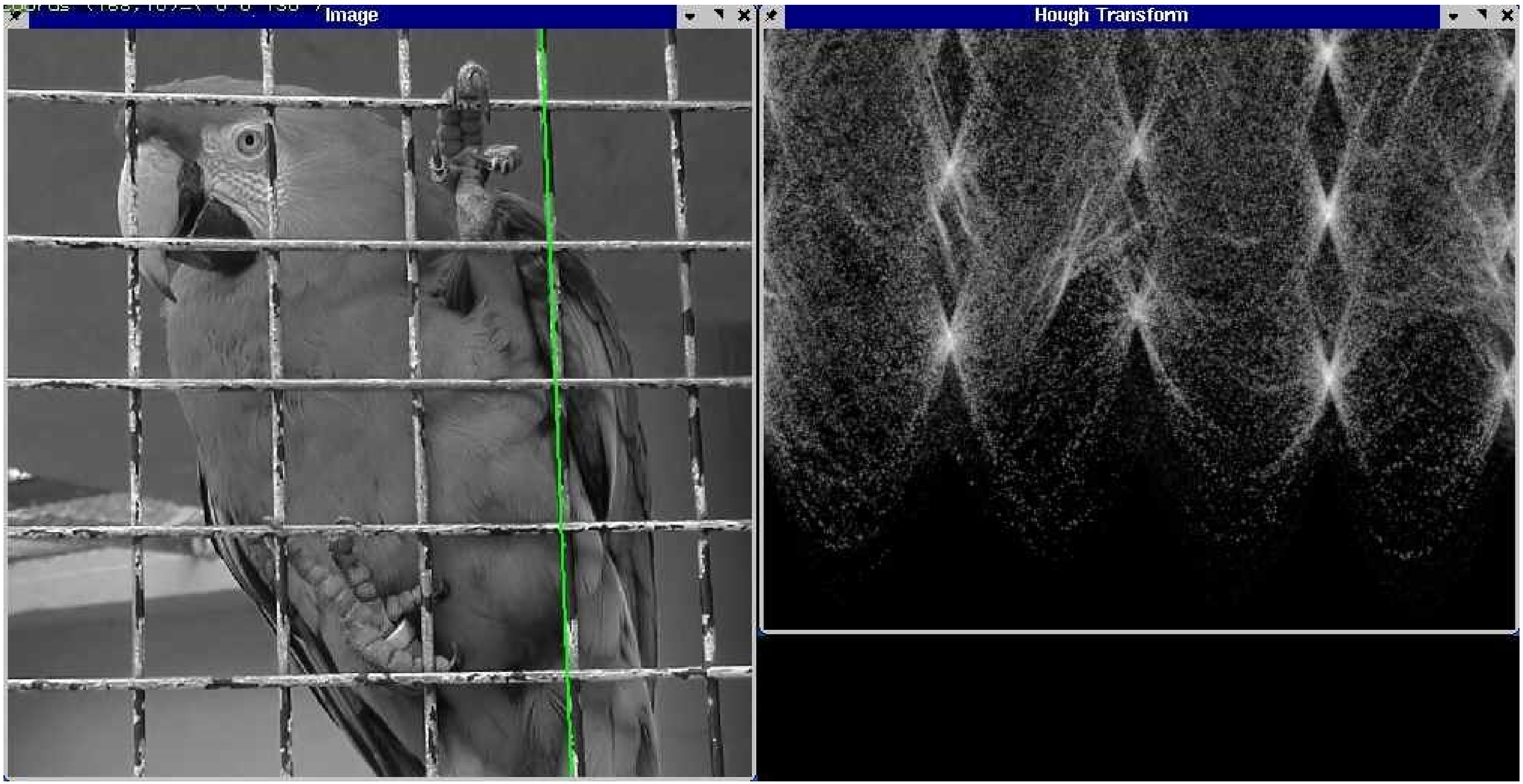


Several lines



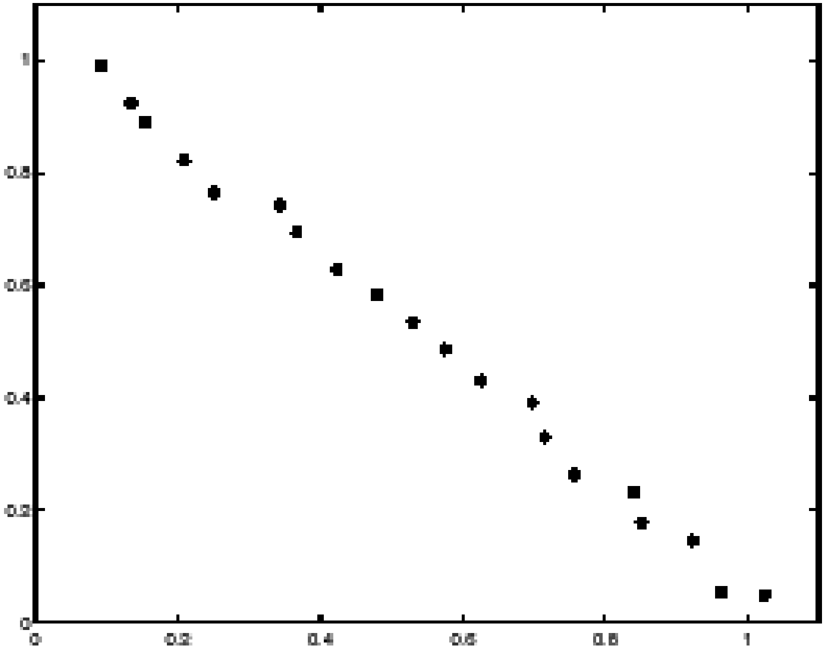


A more complicated image



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

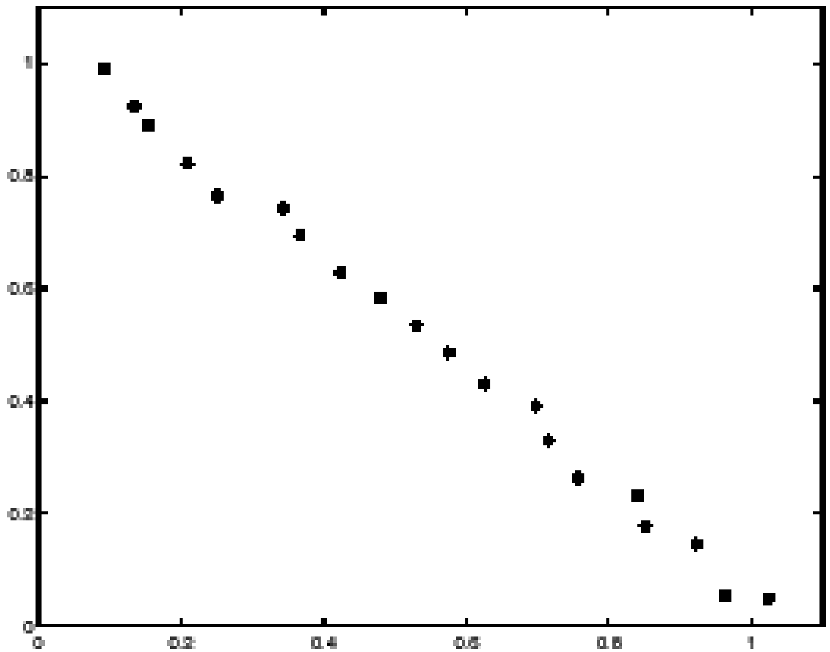
Effect of noise



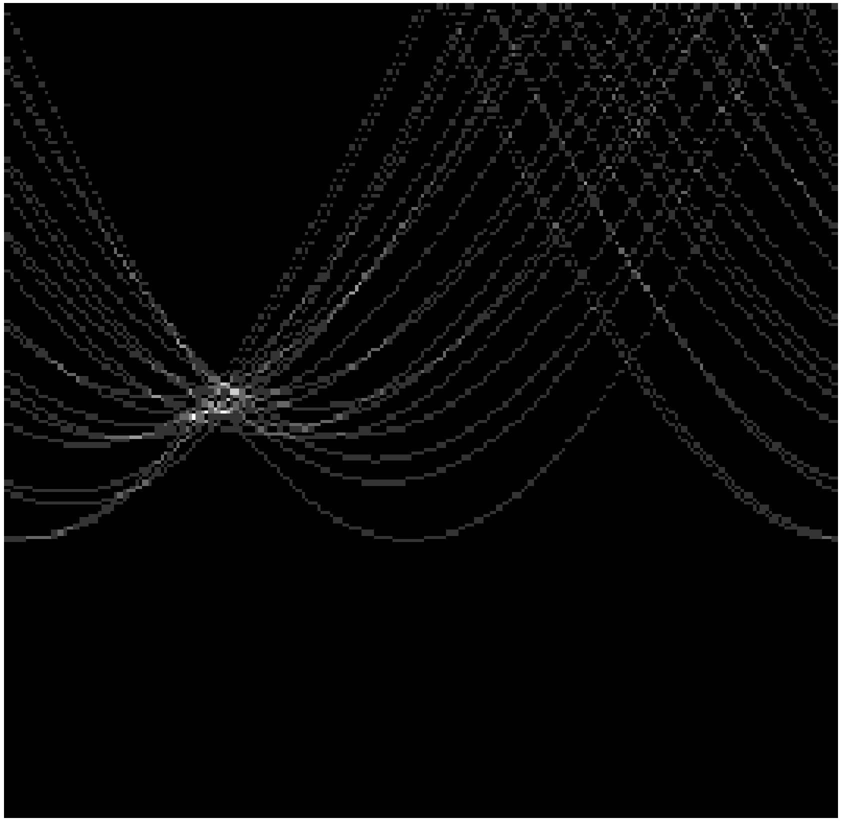
features

Show with demo:
http://liquify.eu/swf/HoughTransform.swf

Effect of noise



features

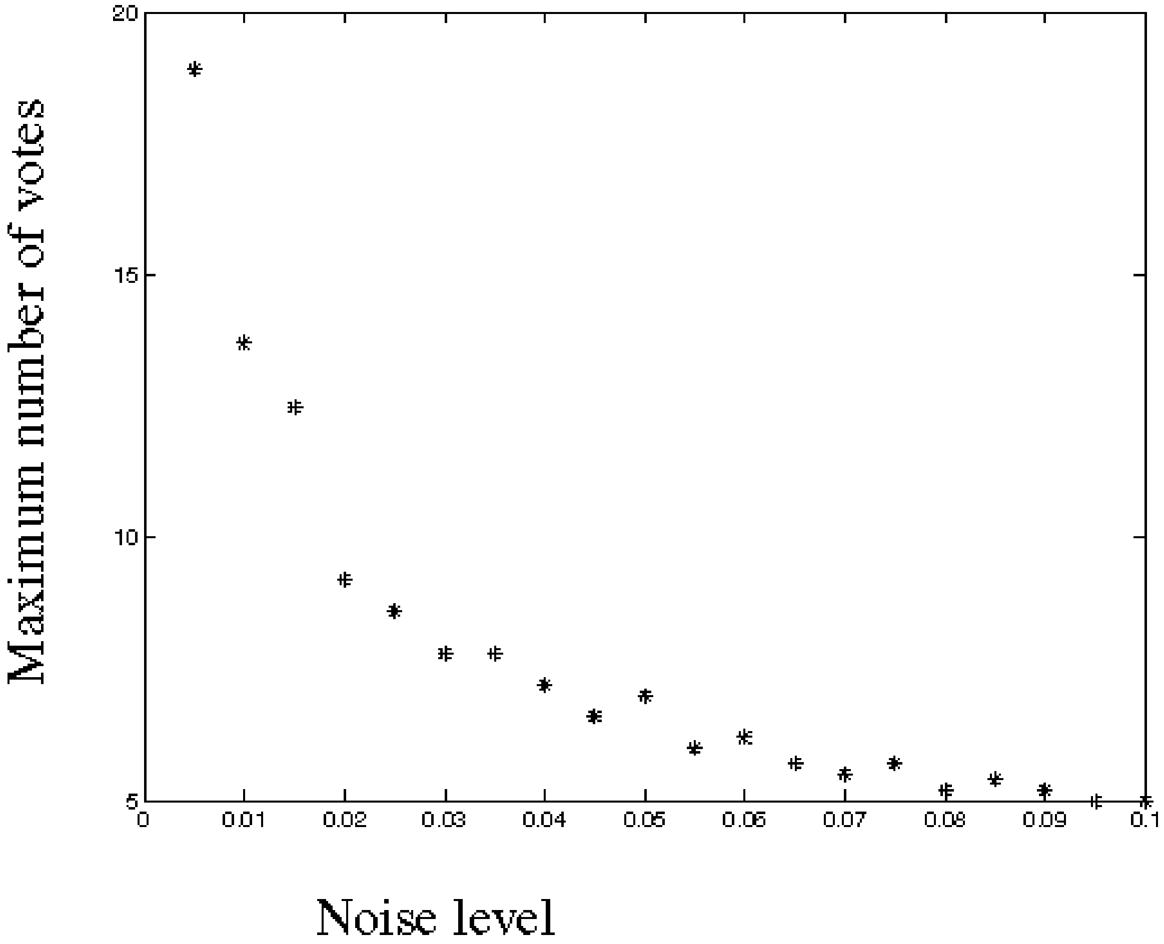


votes

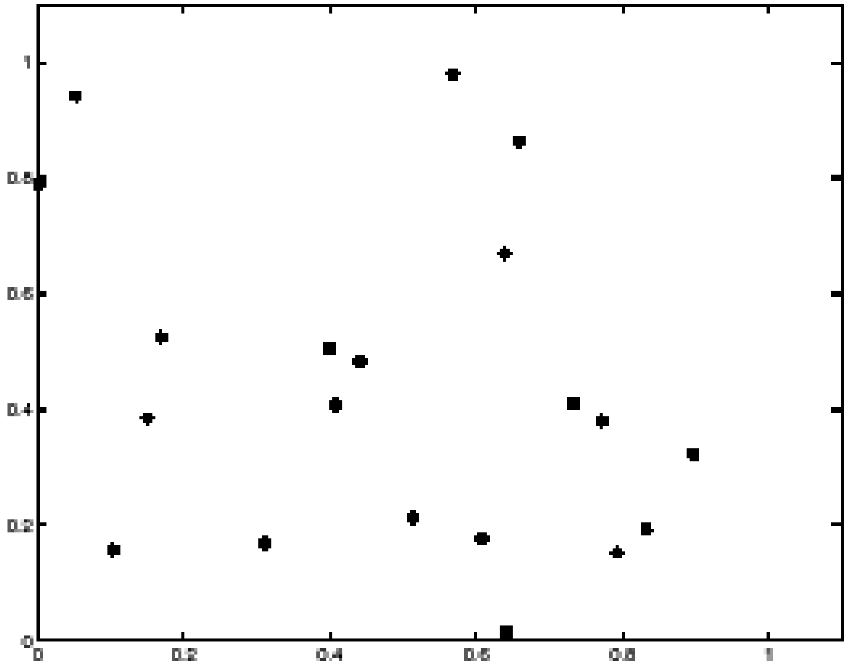
Peak gets fuzzy and hard to locate

Effect of noise

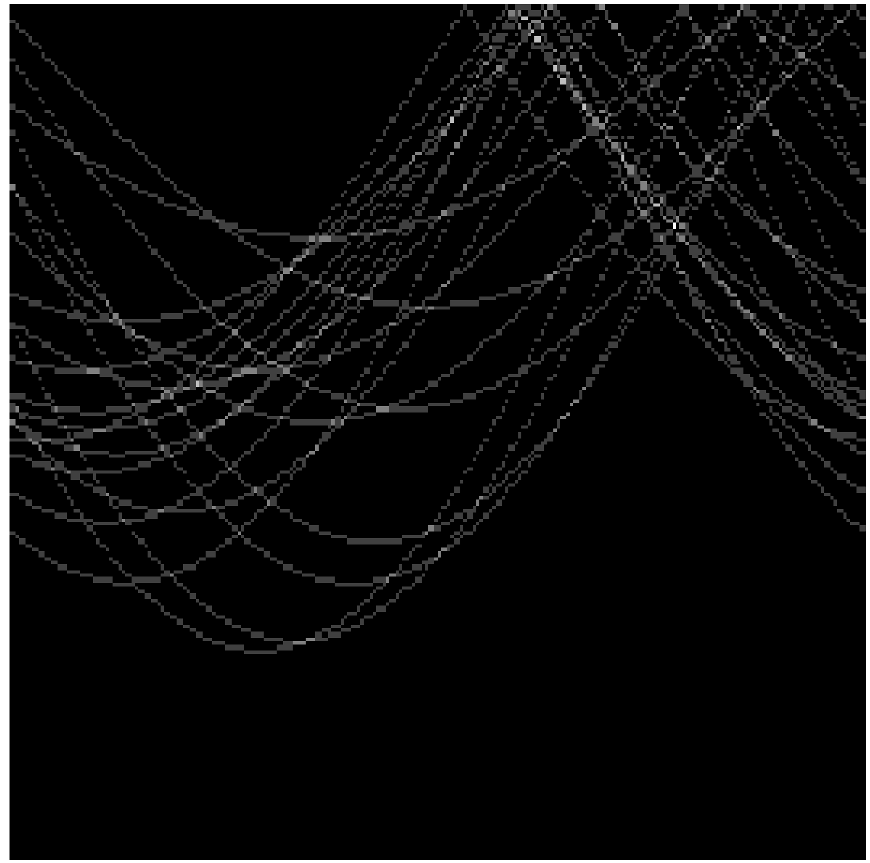
- Number of votes for a line of 20 points with increasing noise:



Random points



features

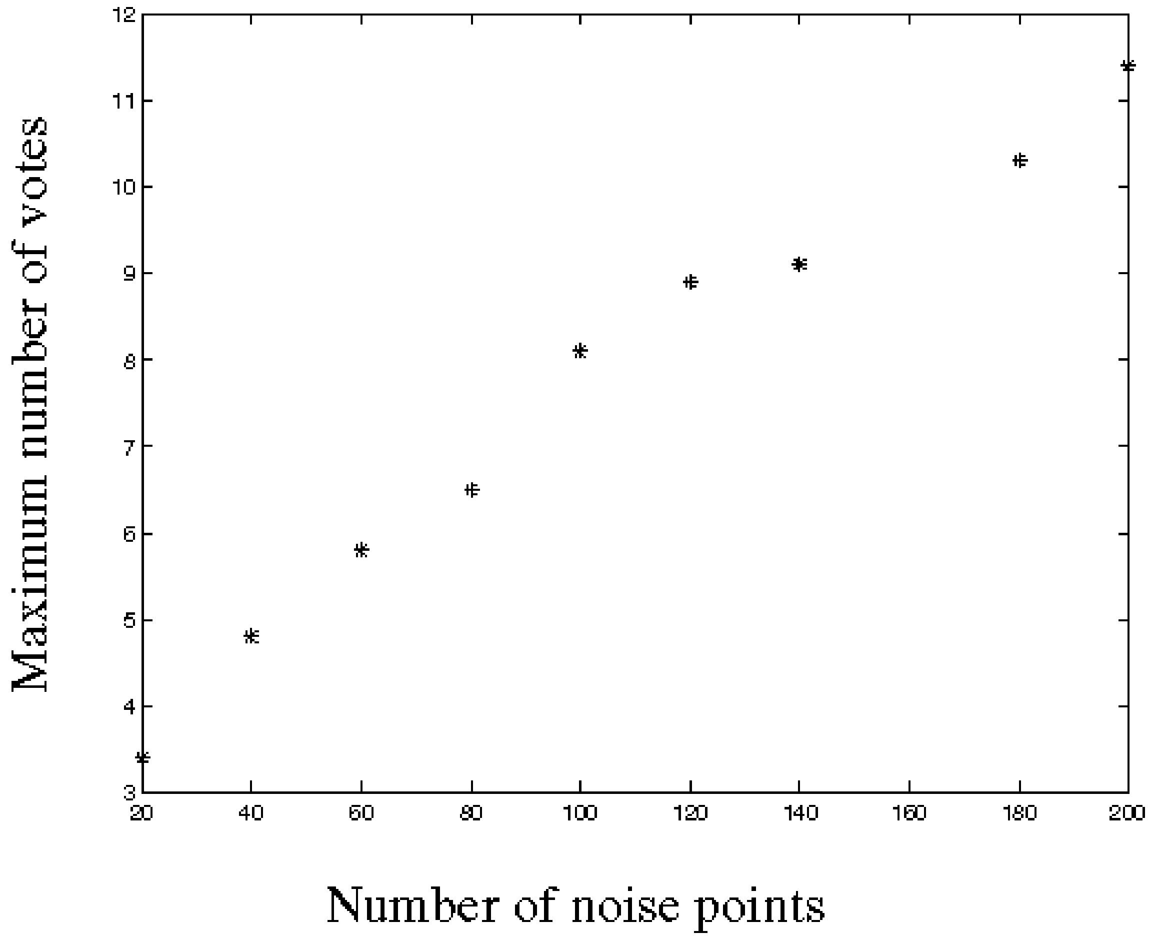


votes

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:

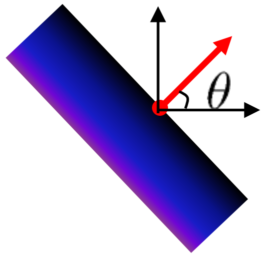


Dealing with noise

- 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
  - Show discretization with following demo: http://www.dis.uniroma1.it/~iocchi/slides/icra2001/java/hough.html
- Increment neighboring bins (smoothing in accumulator array)
- Try to get rid of irrelevant features
  - E.g., 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!



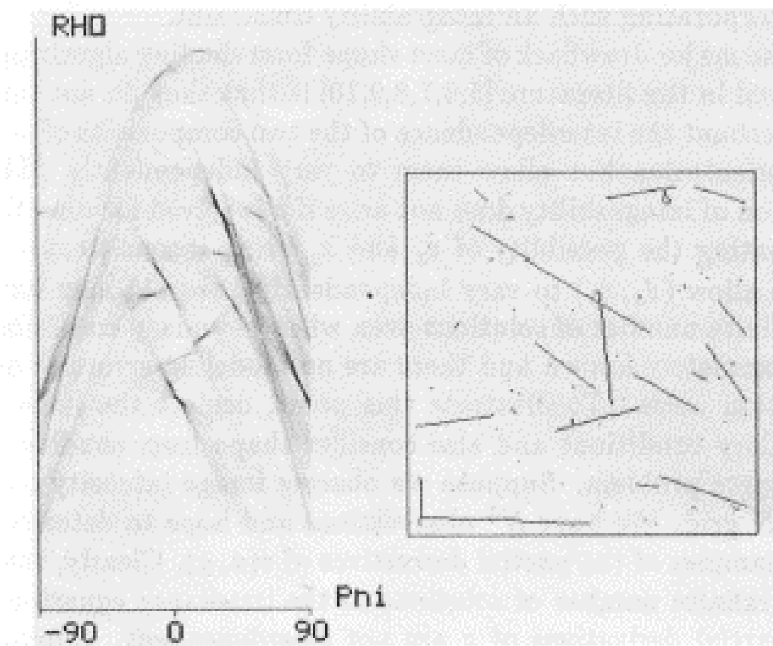
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

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

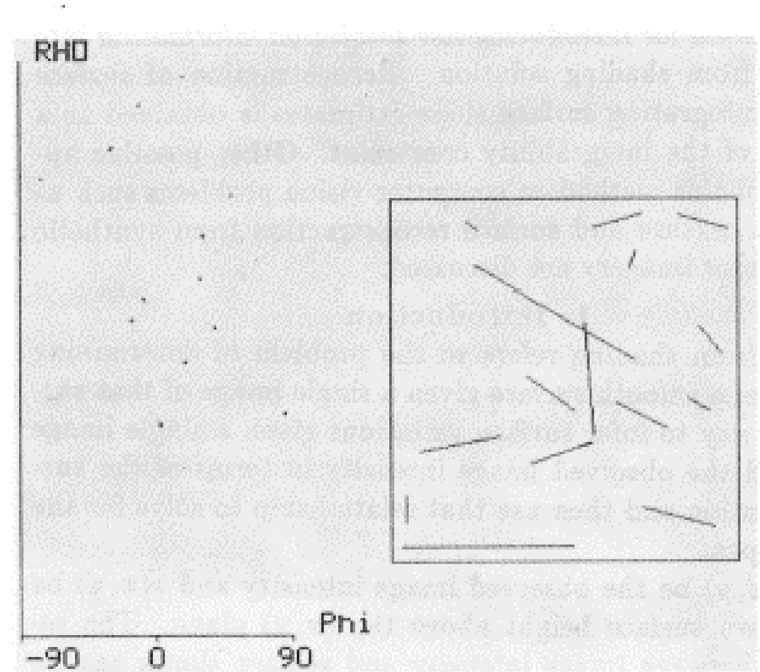
- Modified Hough transform:

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

Comparison with/without edge orientation



Full parameter space



Using edge orientation

Challenge: Presence of noise creates scattering around expected strong peaks.
Solution: Accumulate subregions rather than points.

Hough transform for circles

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