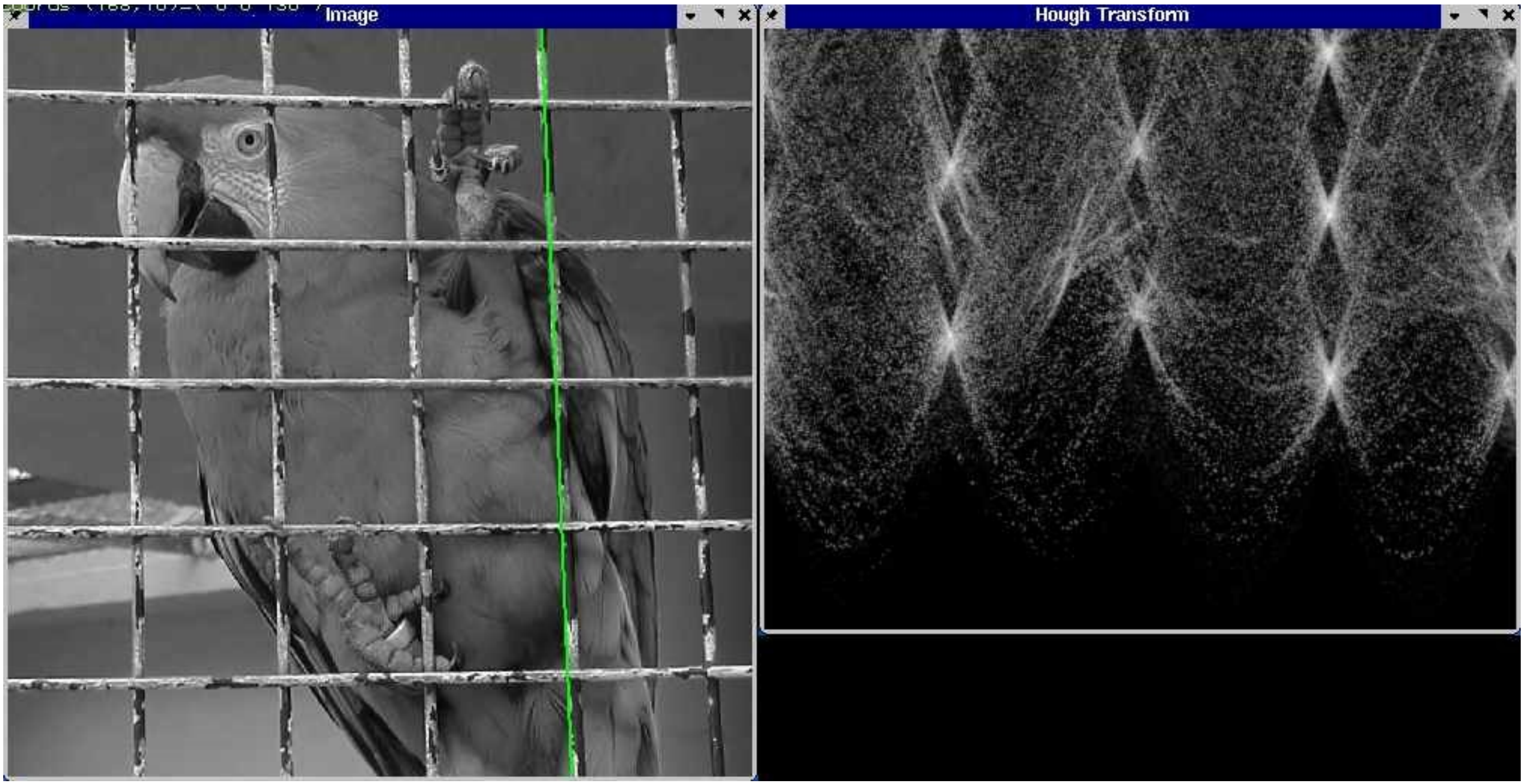
Fitting: The Hough transform



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
  - 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

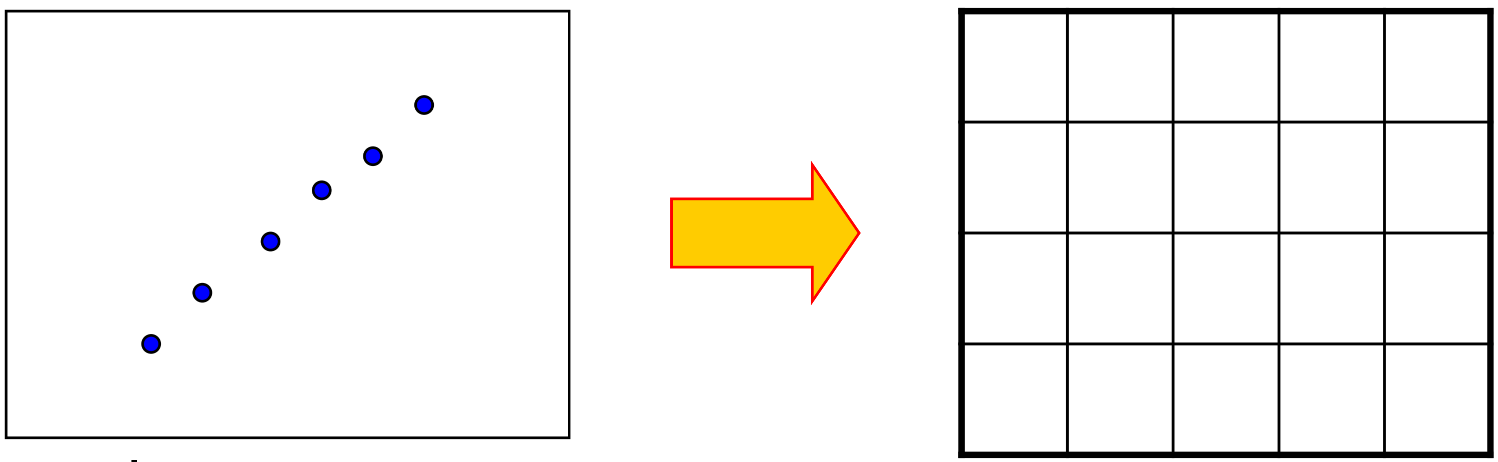


Image space

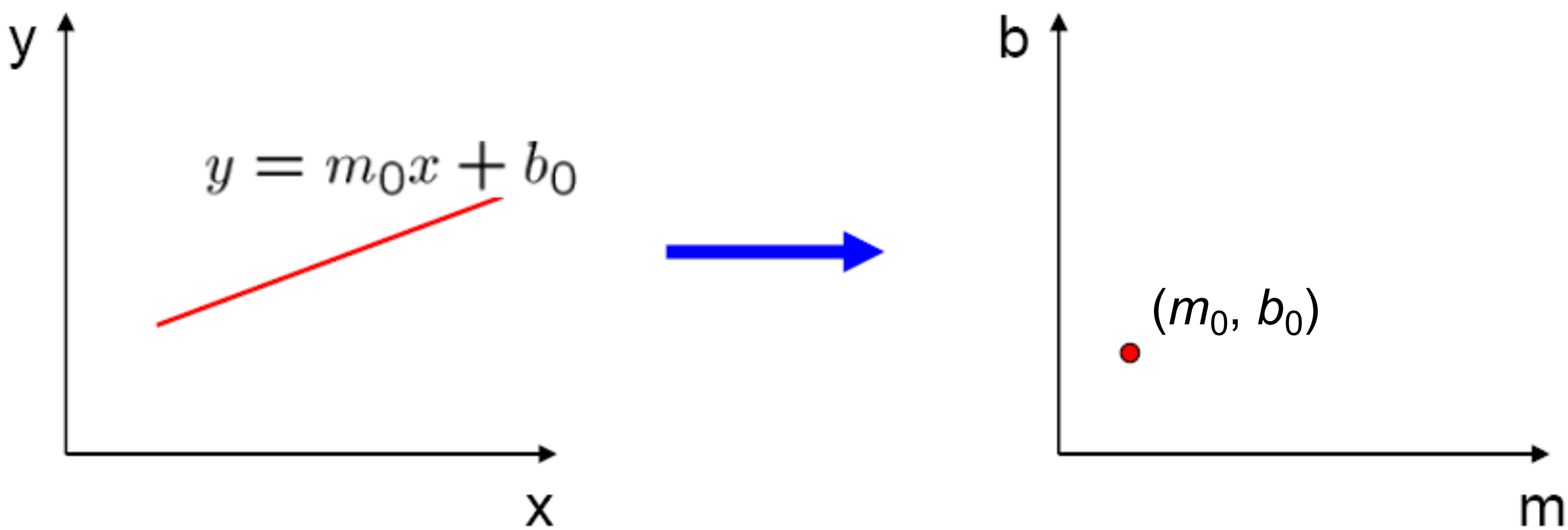
Hough parameter space

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

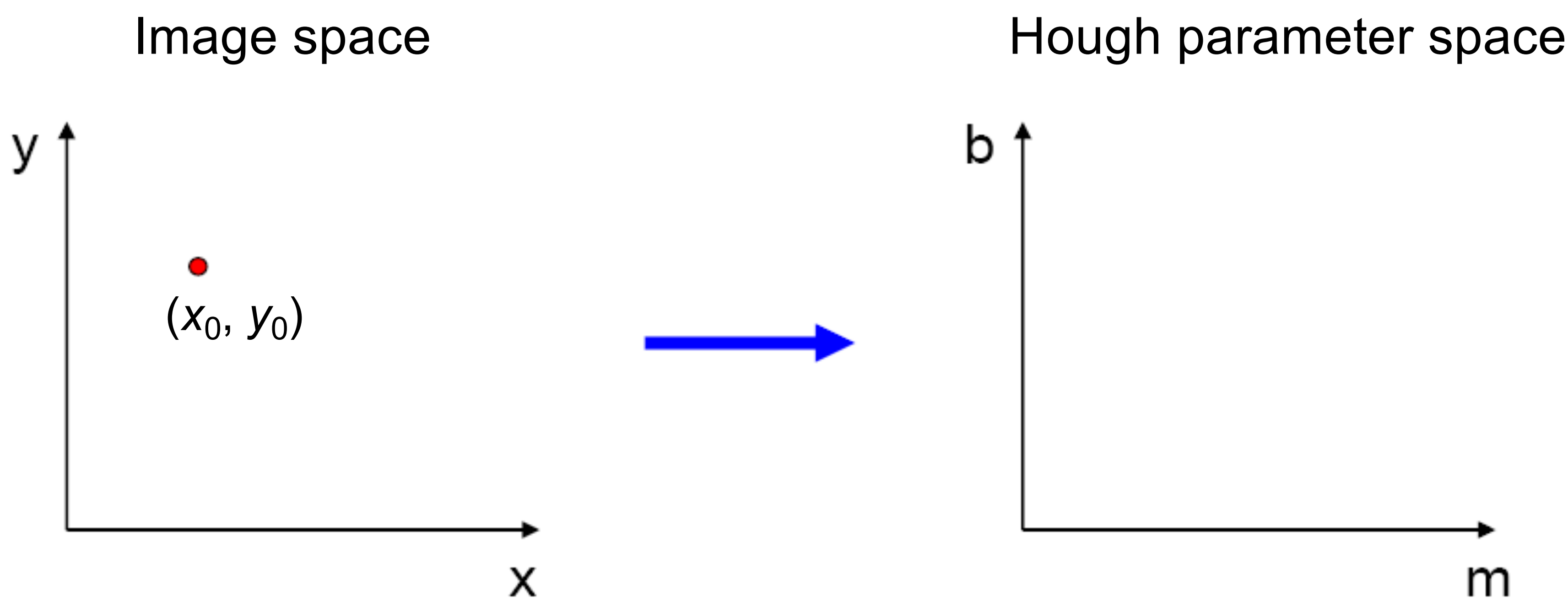
Image space   Hough parameter space



Source: S. Seitz

Parameter space representation

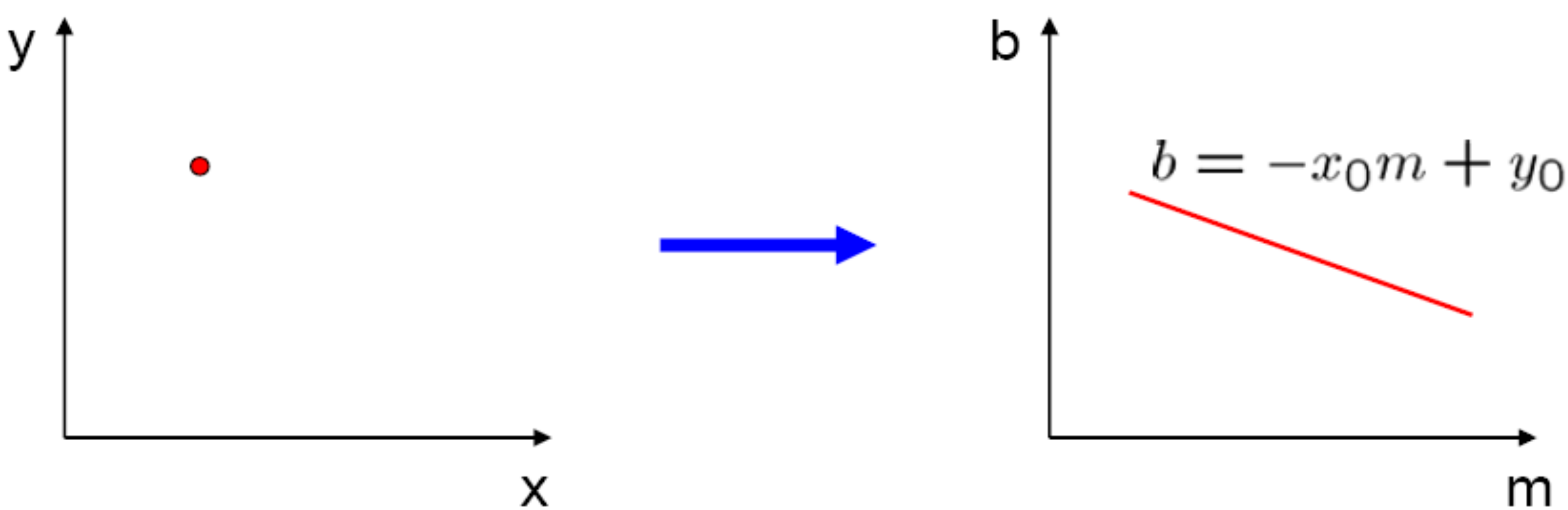
- What does a **point** (x_0, y_0) in the image space map to in the Hough space?



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 $y_0 = m\,x_0 + b$
  - This is a **line** in Hough space

Image space | Hough parameter space

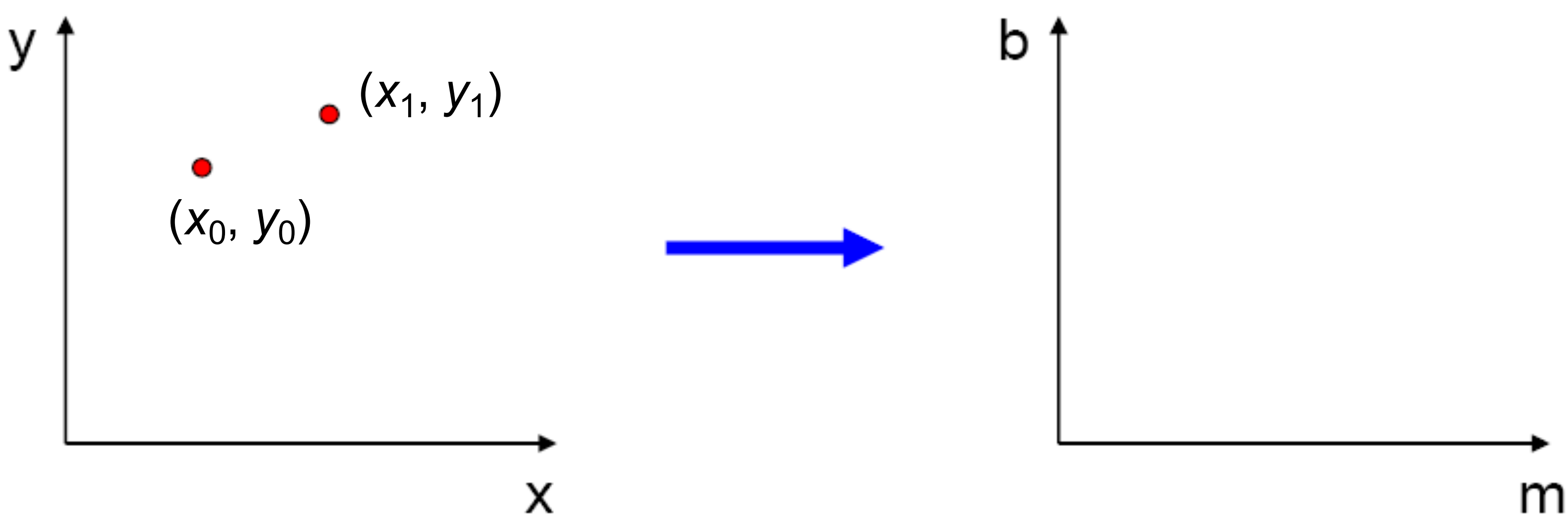


Parameter space representation

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

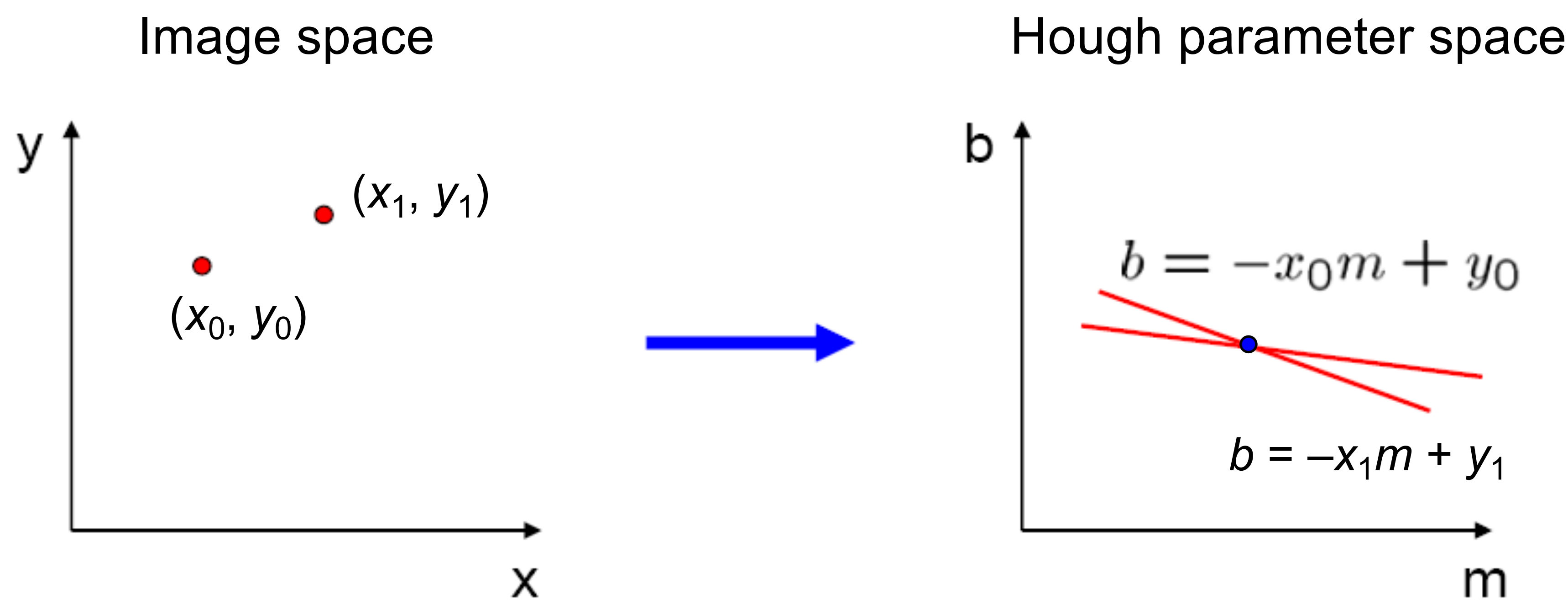
Image space

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 $b = -x_1m + y_1$

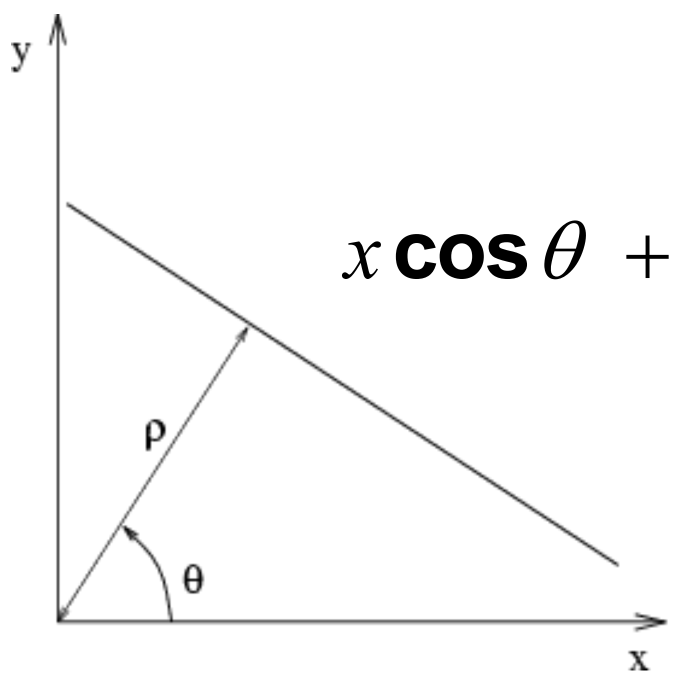


Parameter space representation

- 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*

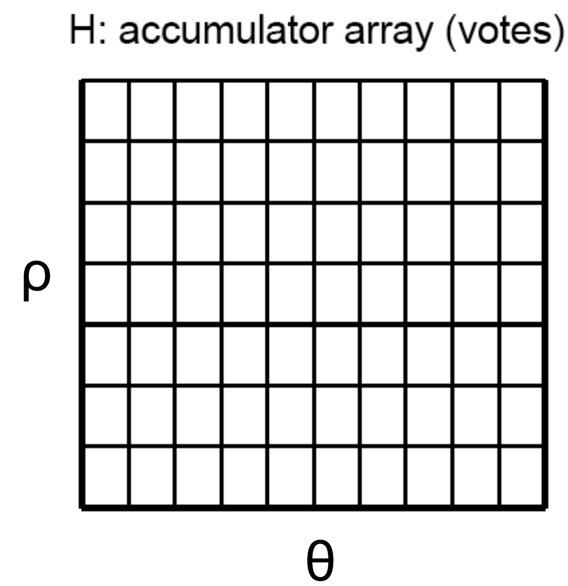


$$x\,\mathbf{cos}\,\theta + y\,\mathbf{sin}\,\theta = \rho$$

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

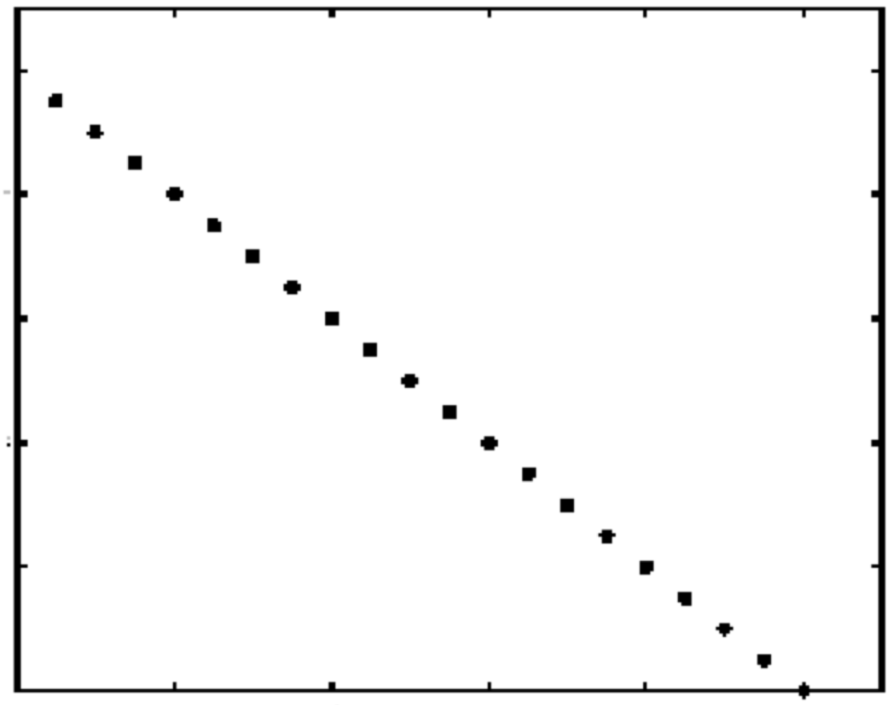
Algorithm outline

- 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

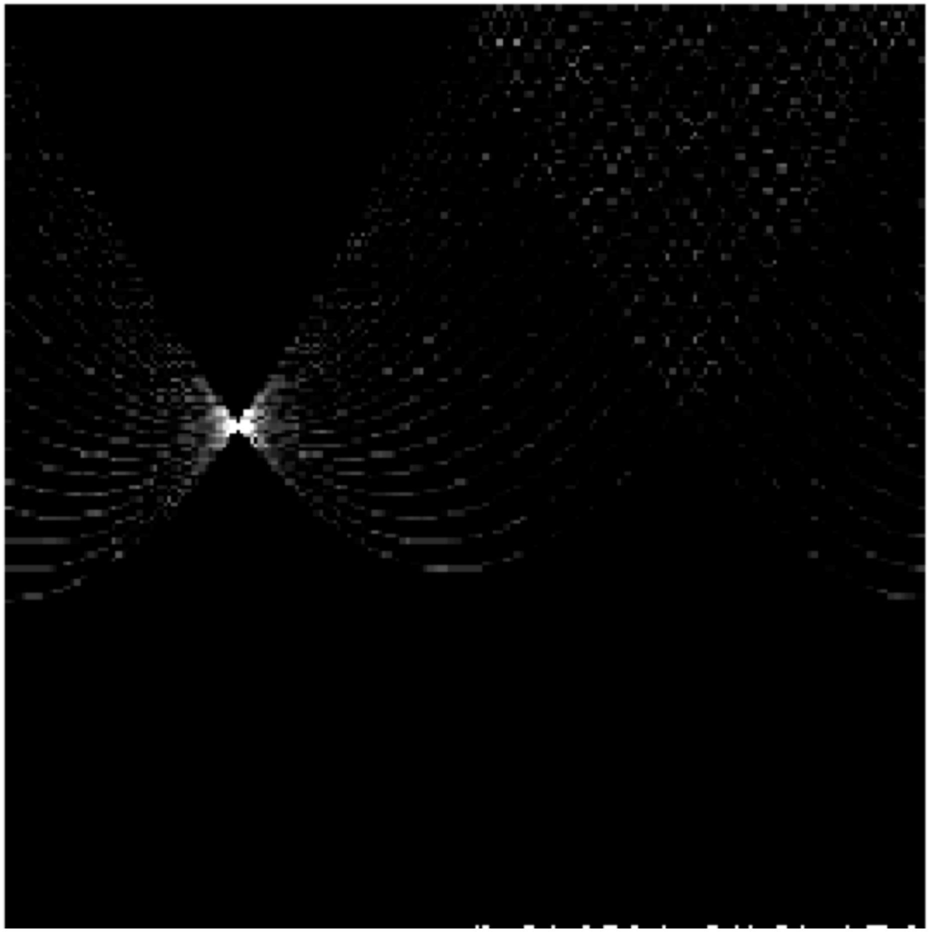


- Find the value(s) of (θ, ρ) where $H(\theta, \rho)$ is a local maximum
  - The detected line in the image is given by $\rho = x \cos \theta + y \sin \theta$

Basic illustration



features



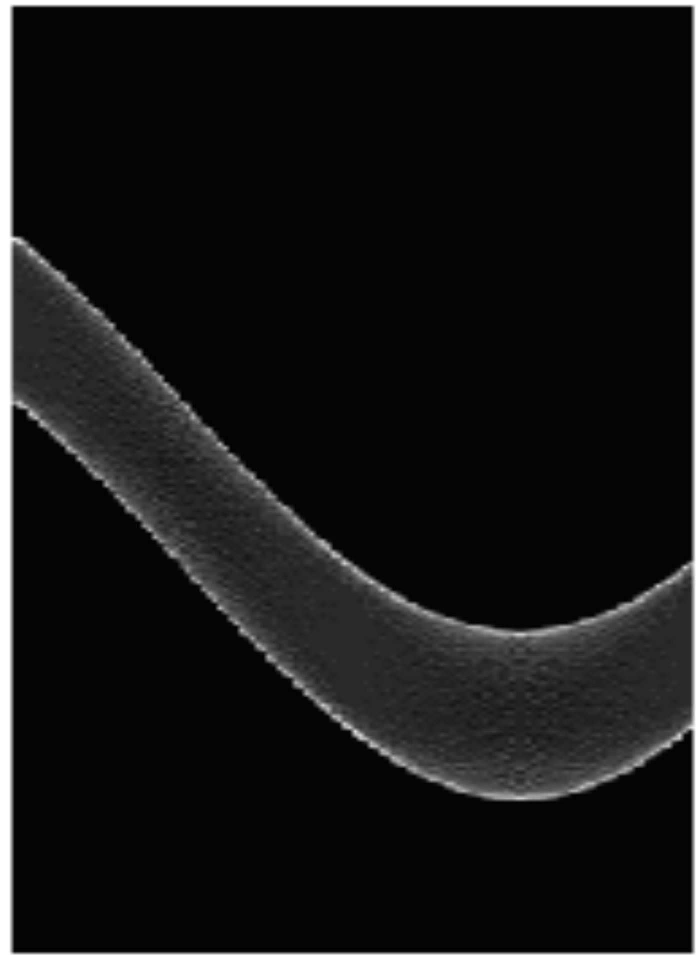
votes

[Hough transform demo]()

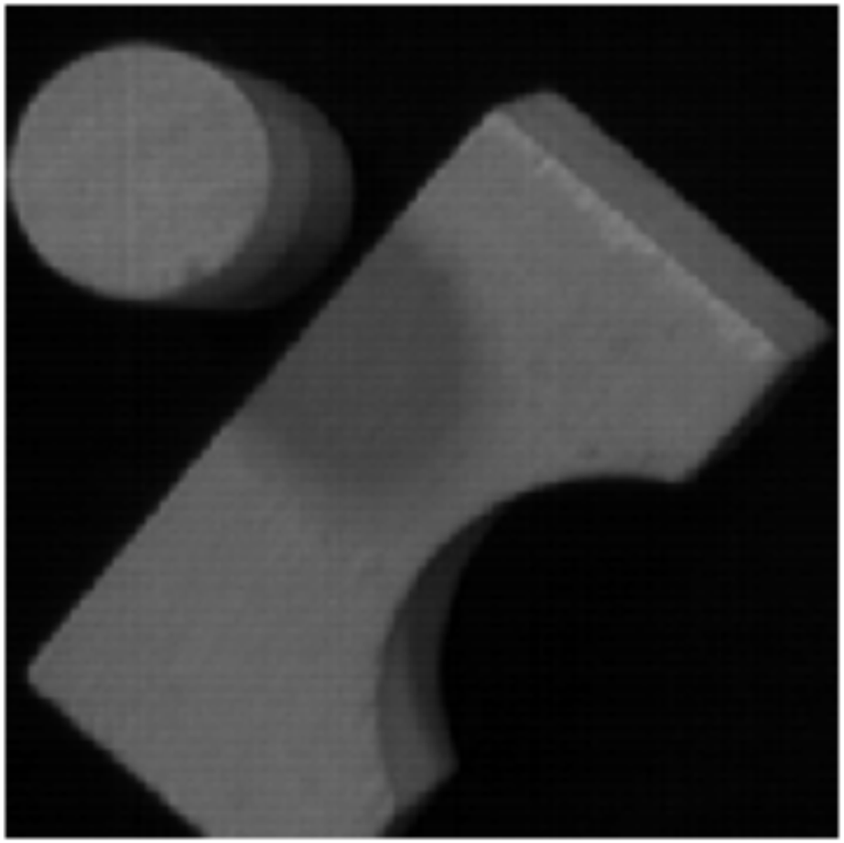
Other shapes

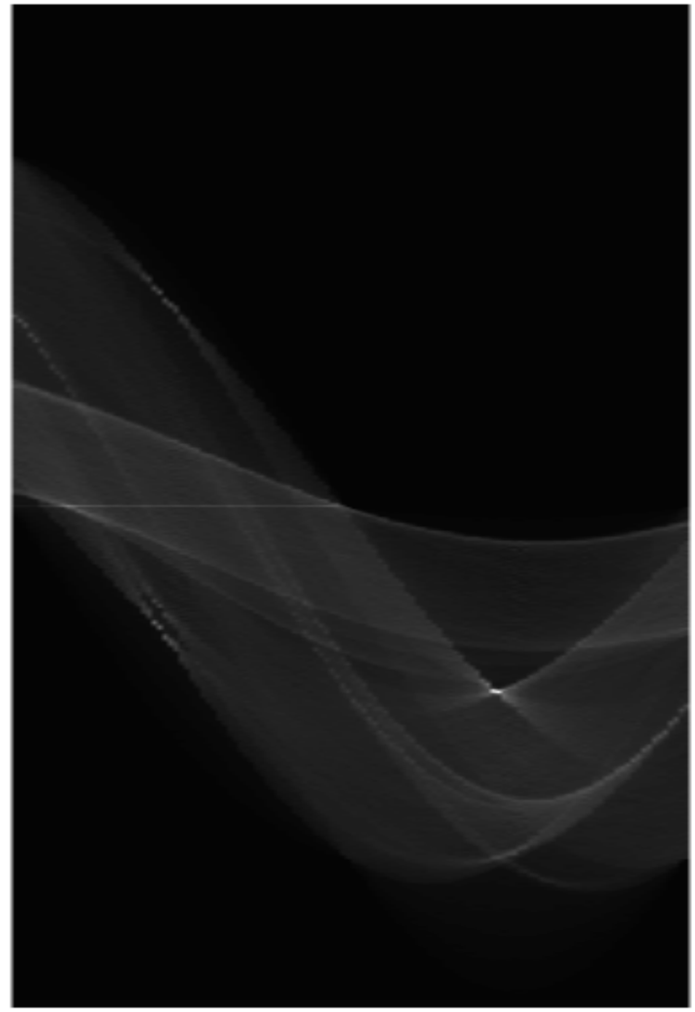
Square

Circle

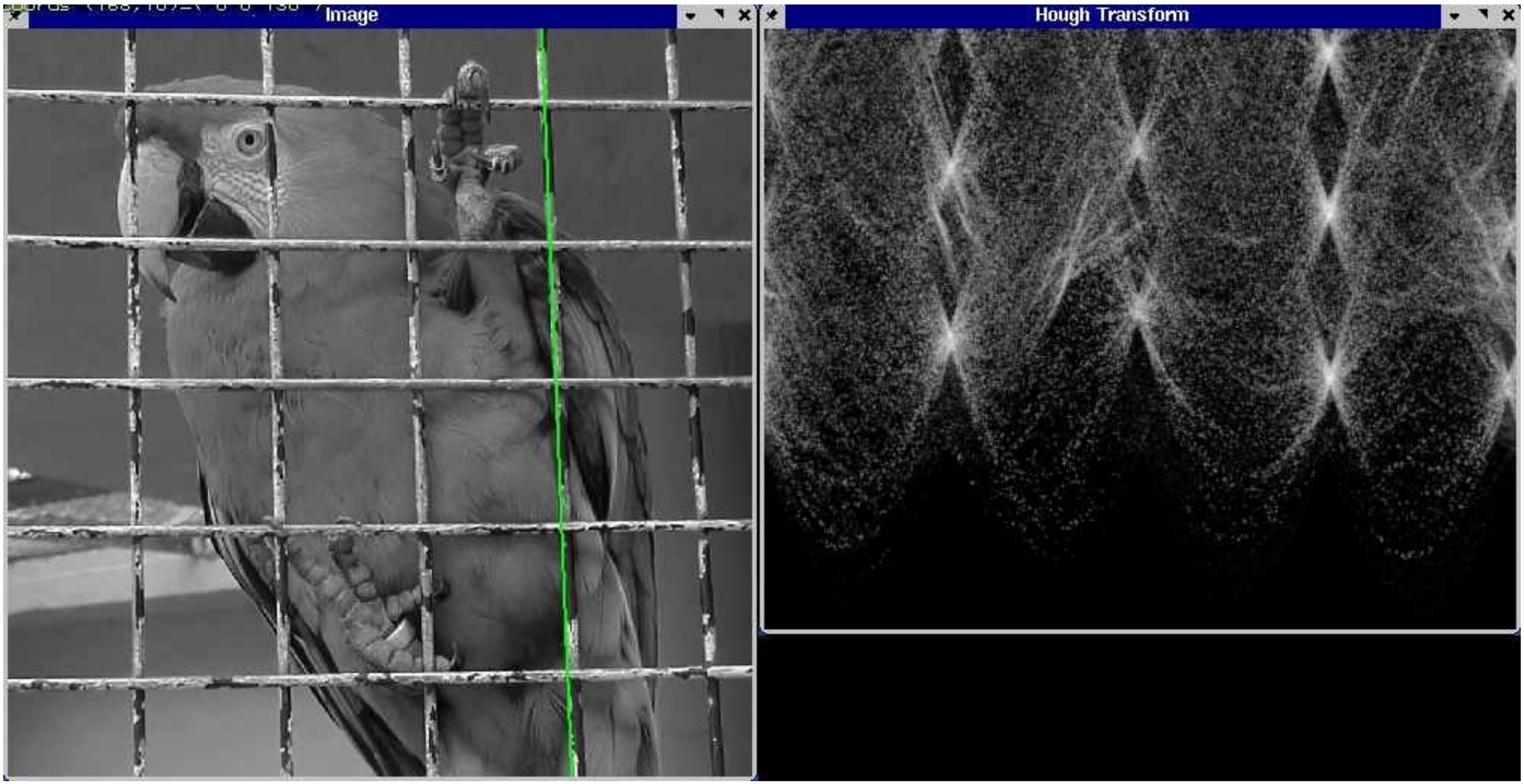


Several lines



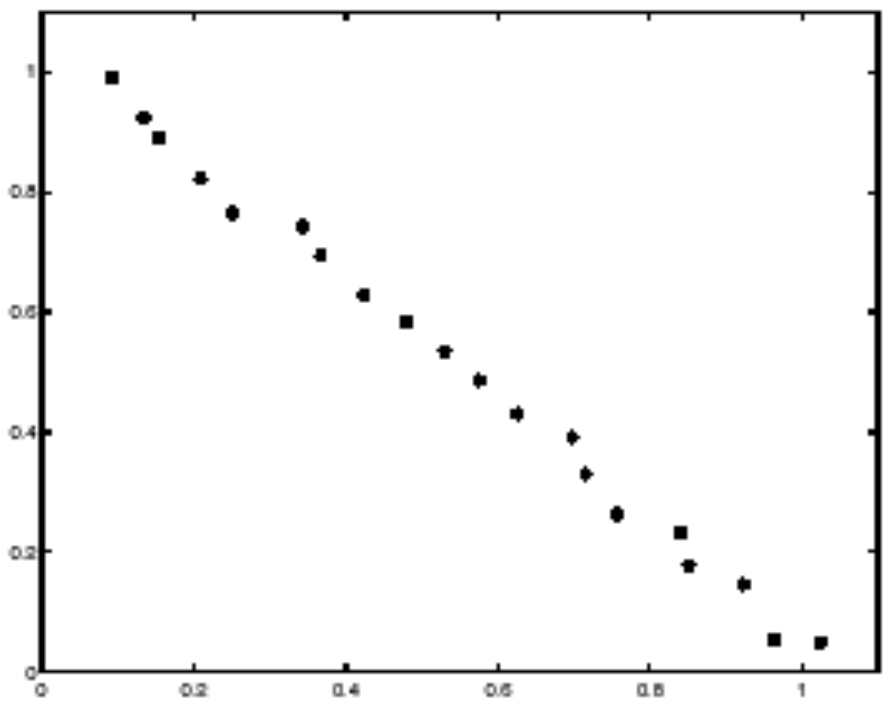


A more complicated image



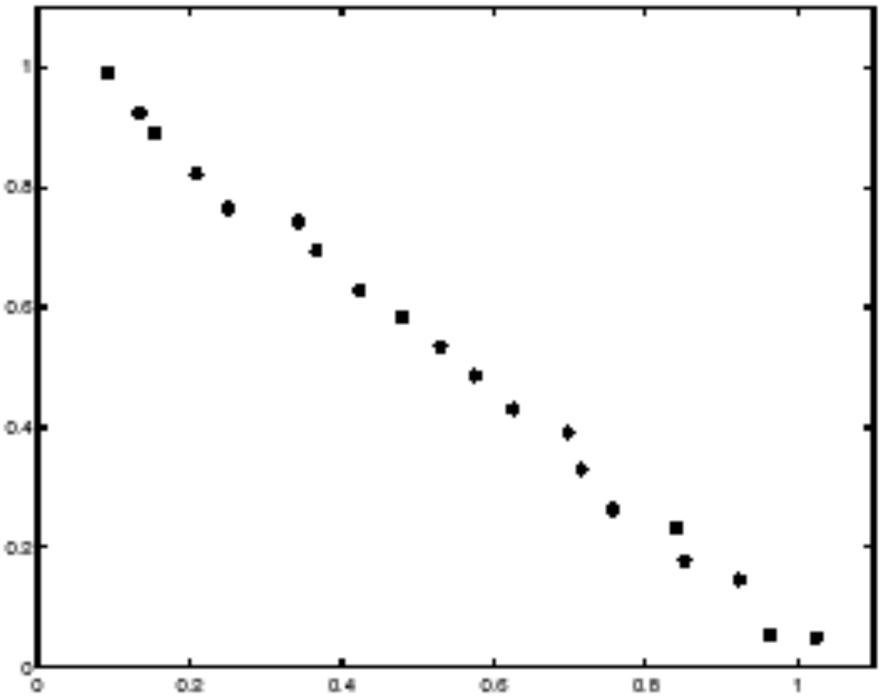
http://ostatic.com/files/images/ss_hough.jpg

Effect of noise

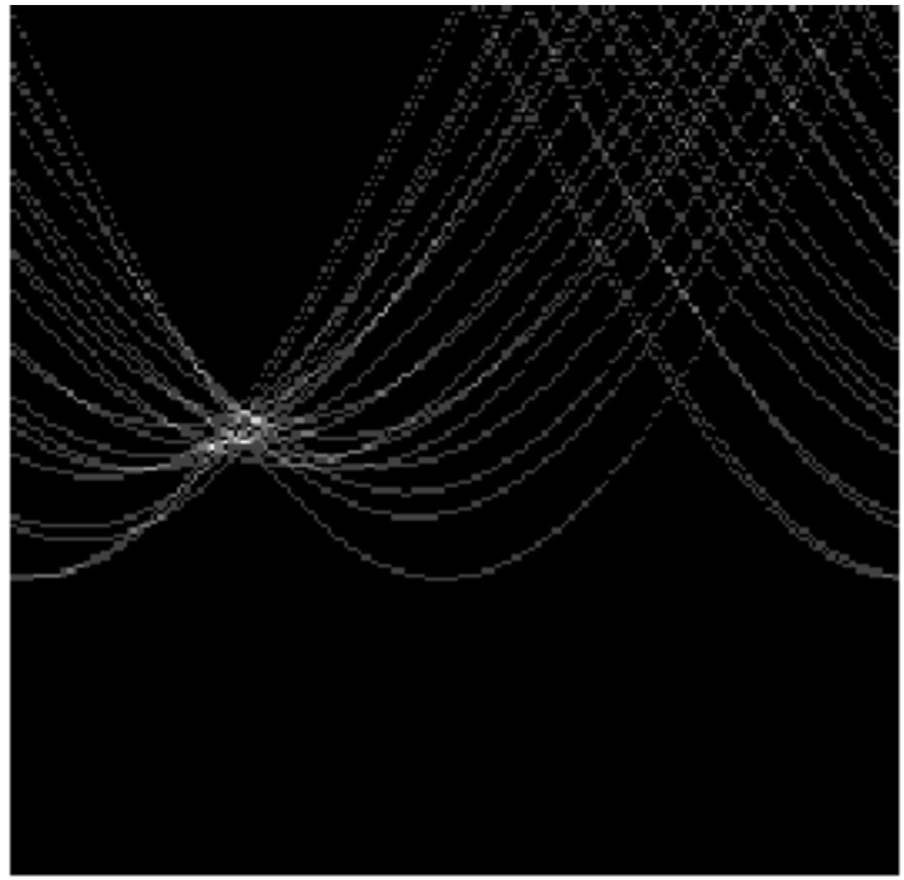


features

Effect of noise



features

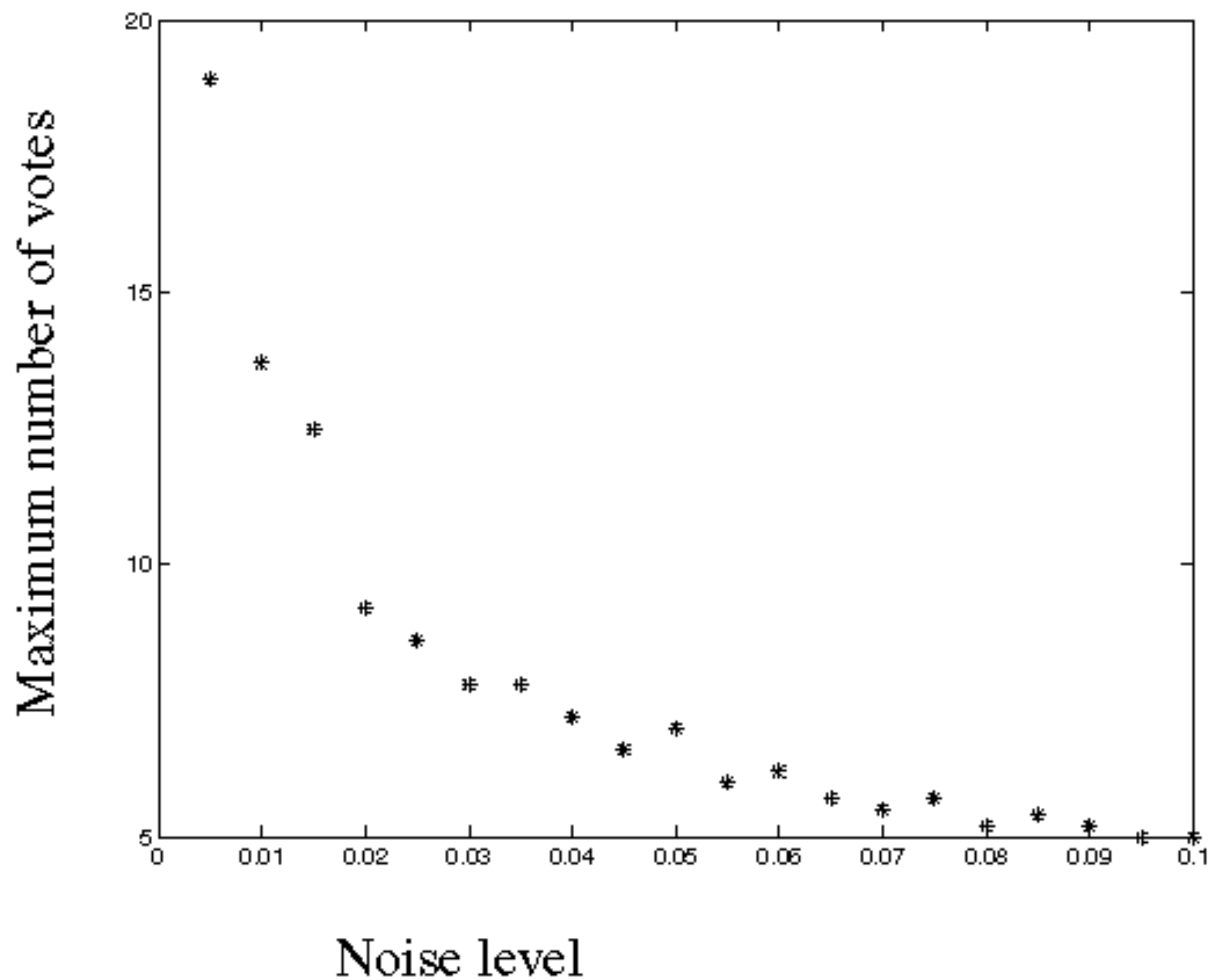


votes

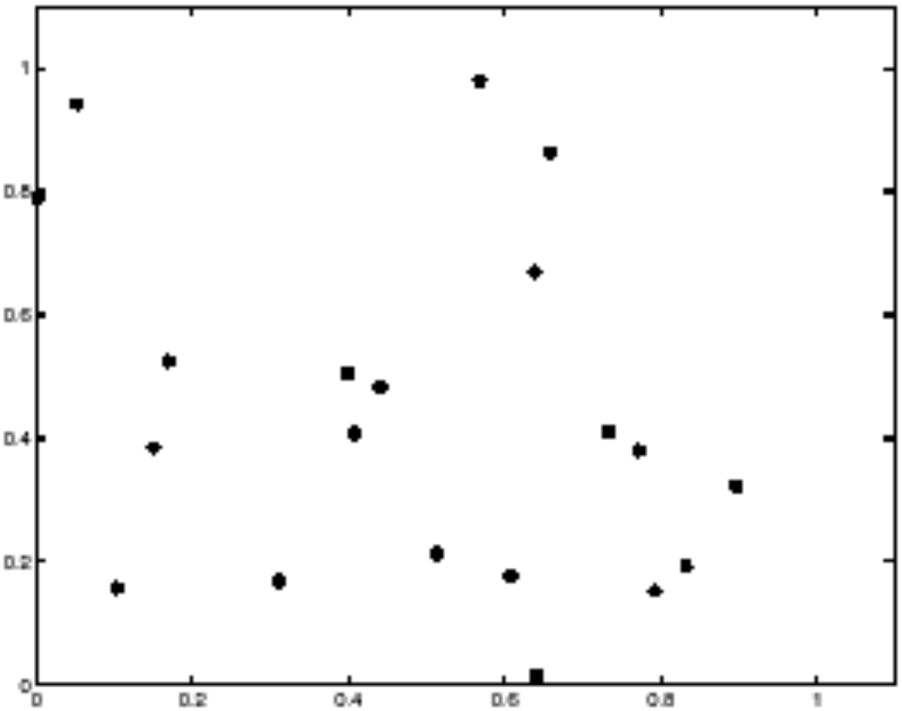
Peak gets fuzzy and hard to locate

Effect of noise

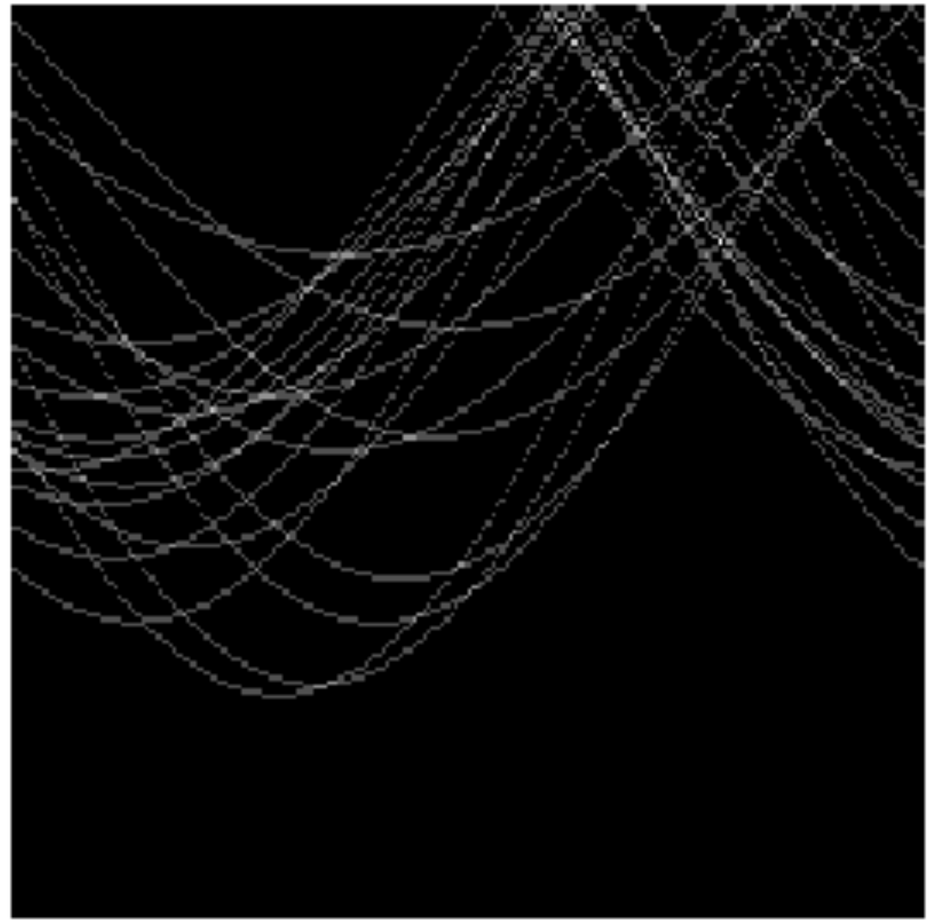
- Number of votes for a line of 20 points with increasing noise:



Random points



features

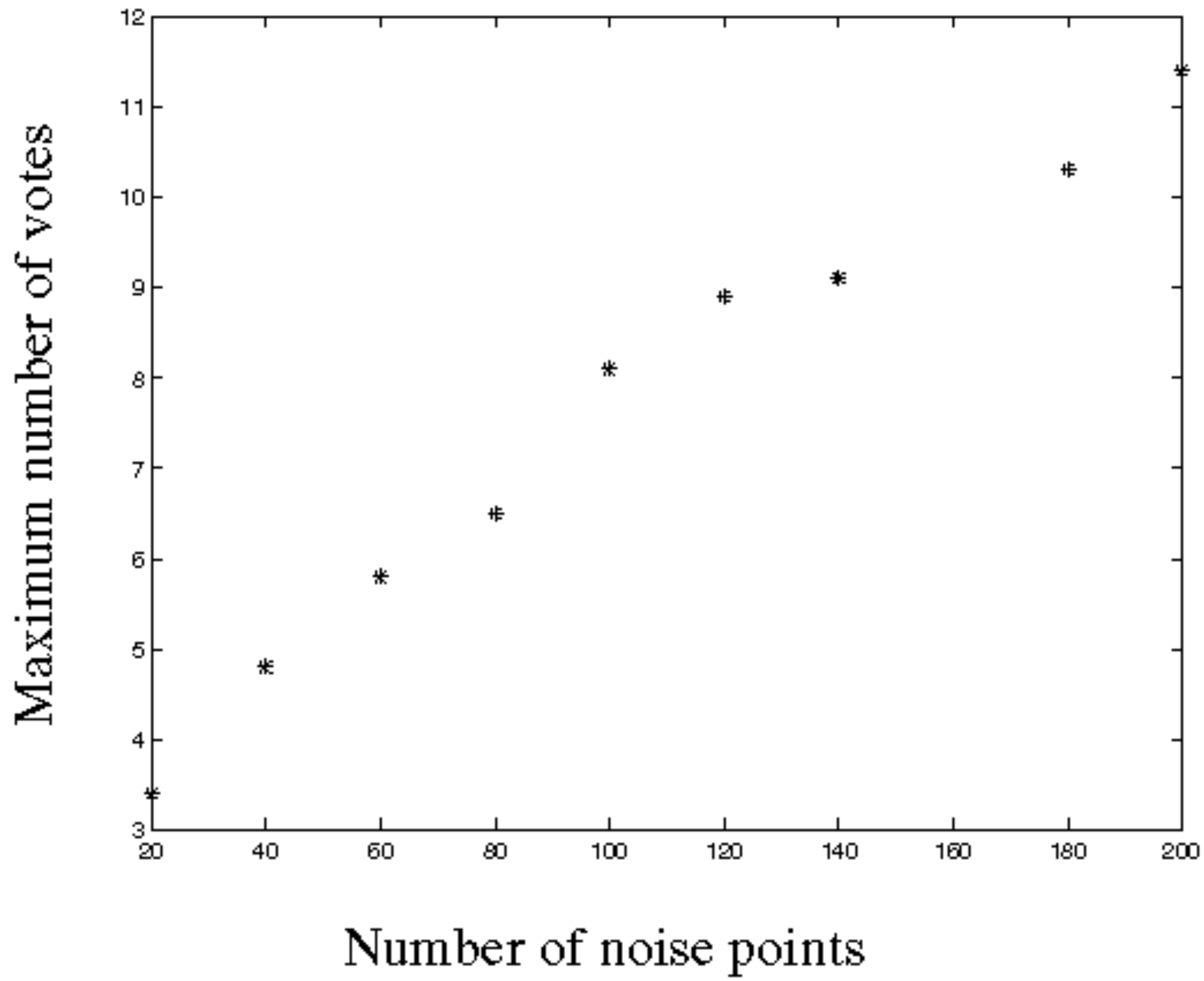


votes

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:

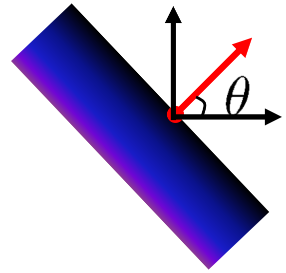


Dealing with noise

- 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
  - E.g., take only edge points with significant gradient magnitude

Incorporating image gradients

- When we detect an edge point, we also know its gradient orientation
- How does this constrain possible lines passing through the point?



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

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

- Modified Hough transform:

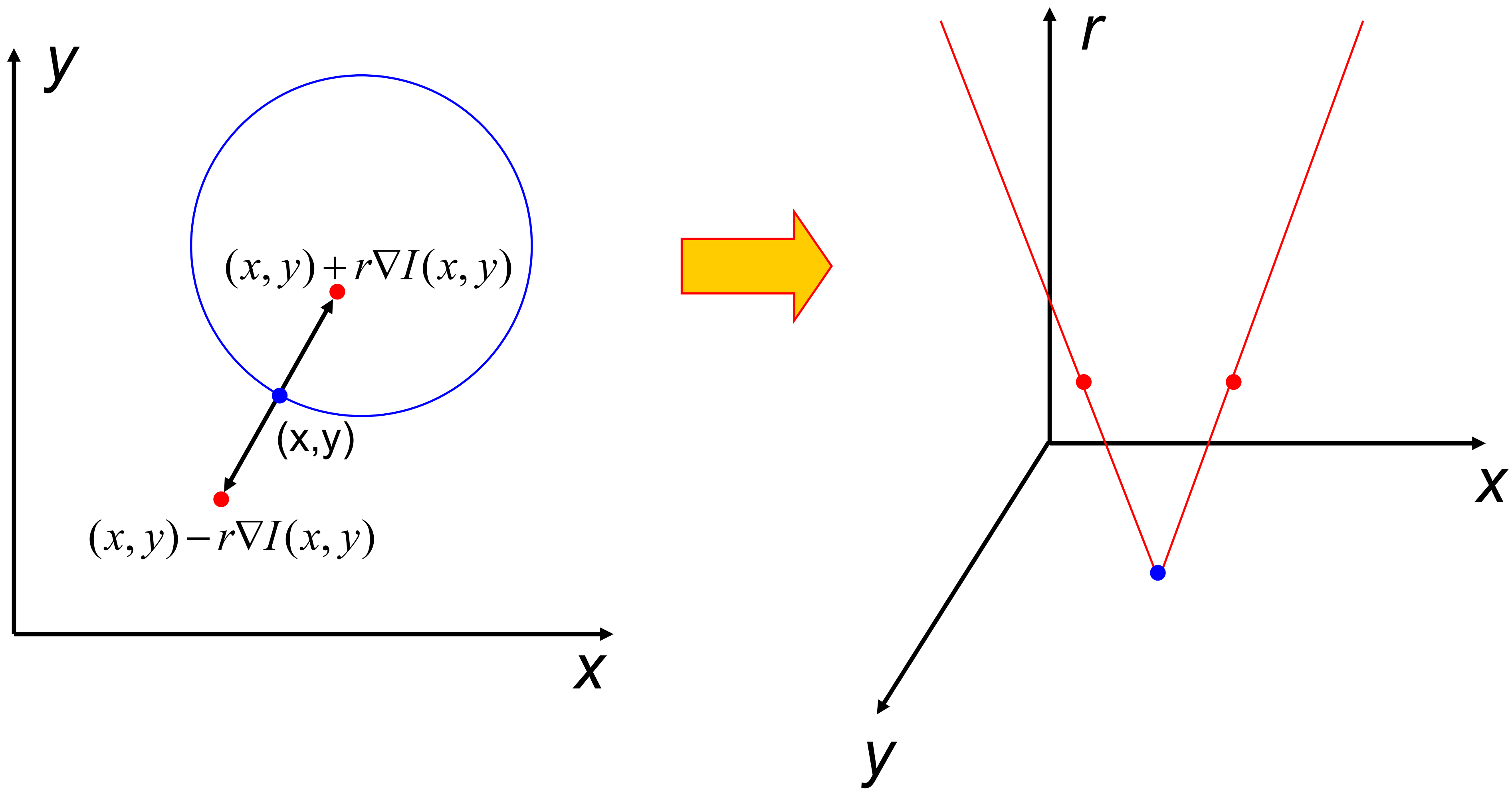
For each edge point (x,y)
  θ = gradient orientation at (x,y)
  ρ = x cos θ + y sin θ
  H(θ, ρ) = H(θ, ρ) + 1
end

Hough transform for circles

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

Hough transform for circles

image space Hough parameter space



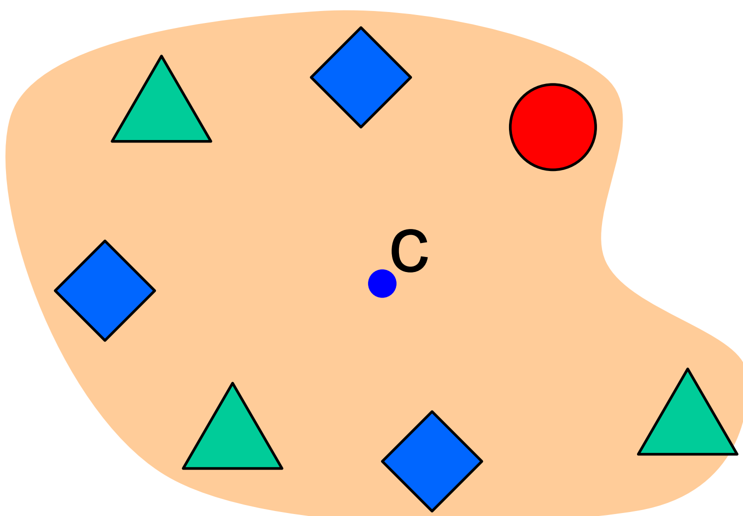
Hough transform: Pros and cons

- Pros
  - Can deal with non-locality and occlusion
  - Can detect multiple instances of a model
  - Some robustness to noise: noise points unlikely to contribute consistently to any single bin
- Cons
  - Complexity of search time increases exponentially with the number of model parameters
  - Non-target shapes can produce spurious peaks in parameter space
  - It’s hard to pick a good grid size

Generalized Hough transform

- We want to find a template defined by its reference point (center) and several distinct types of landmark points in stable spatial configuration

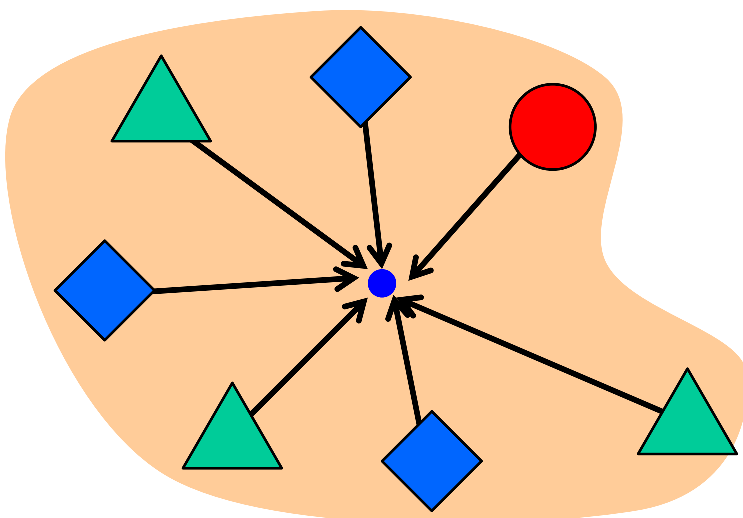
Template



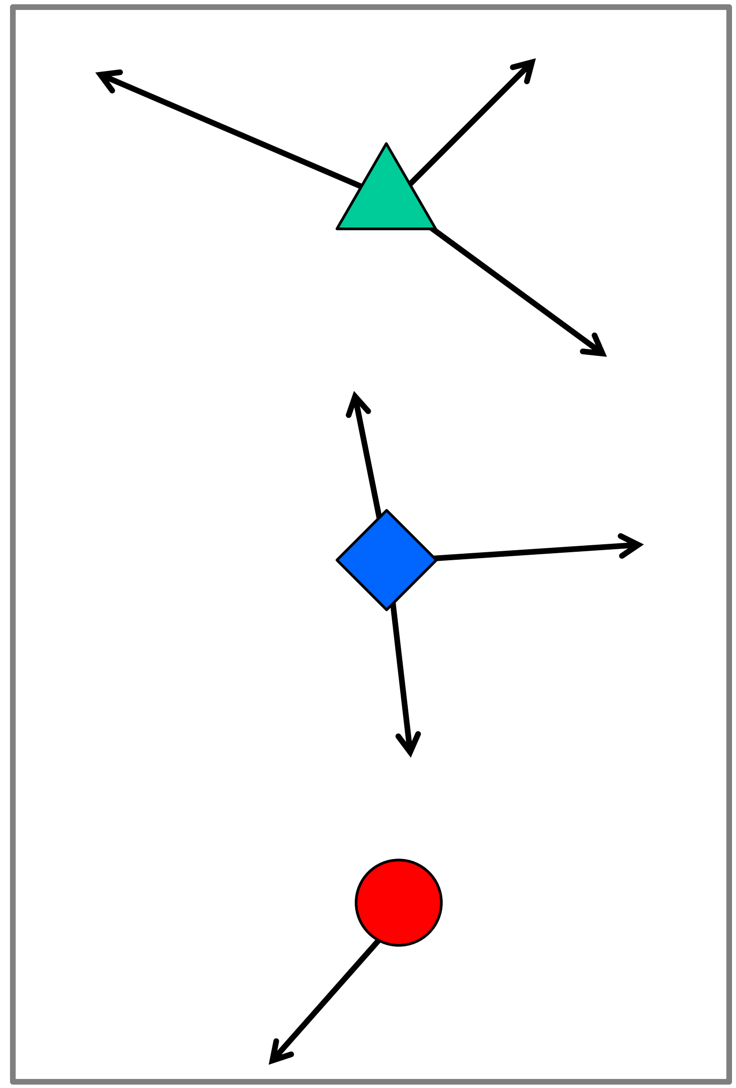
Generalized Hough transform

- Template representation: for each type of landmark point, store all possible displacement vectors towards the center

Template



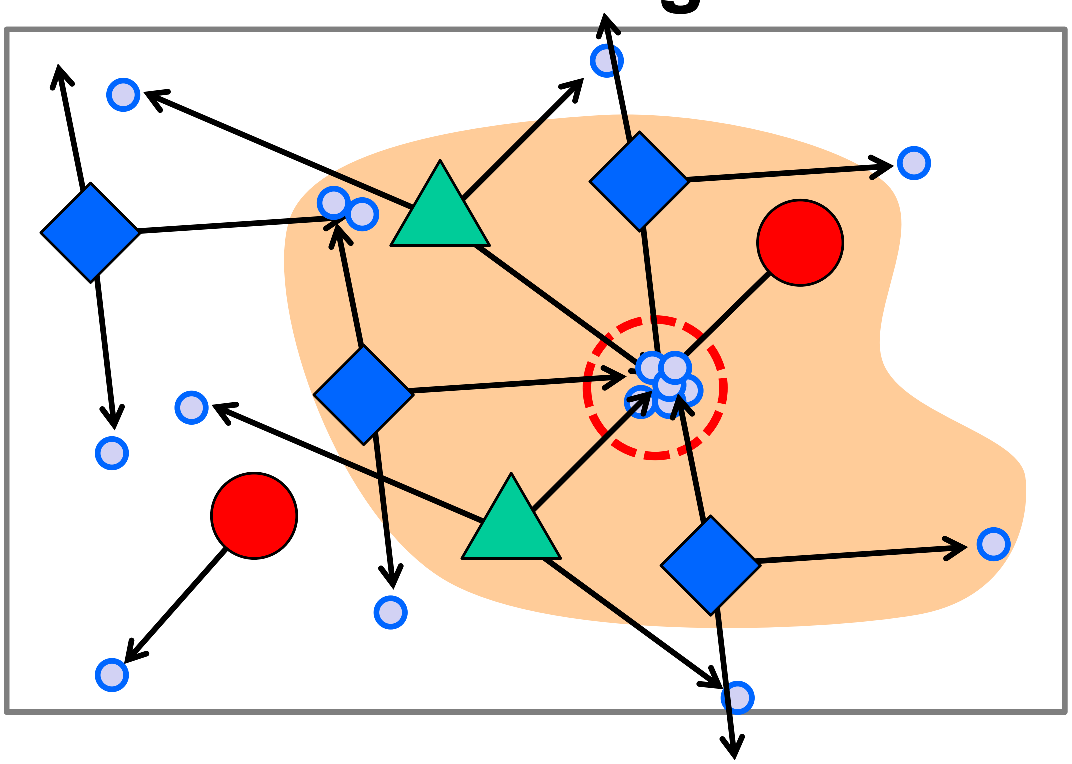
Model



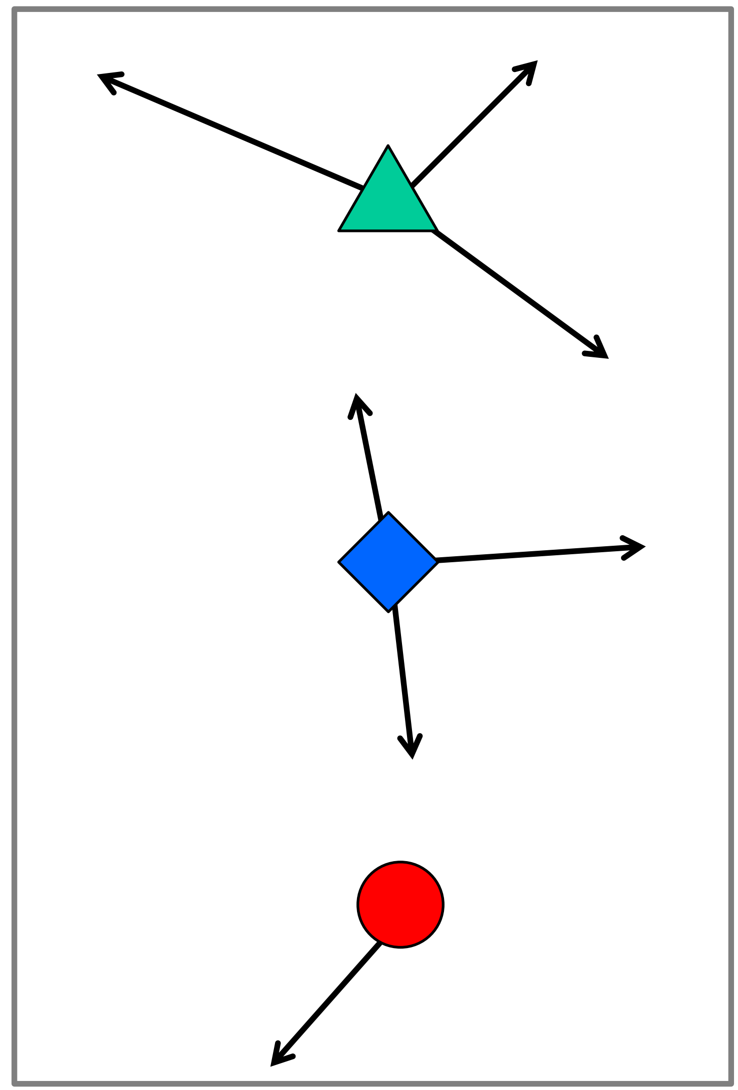
Generalized Hough transform

- Detecting the template:
  - For each feature in a new image, look up that feature type in the model and vote for the possible center locations associated with that type in the model

Test image

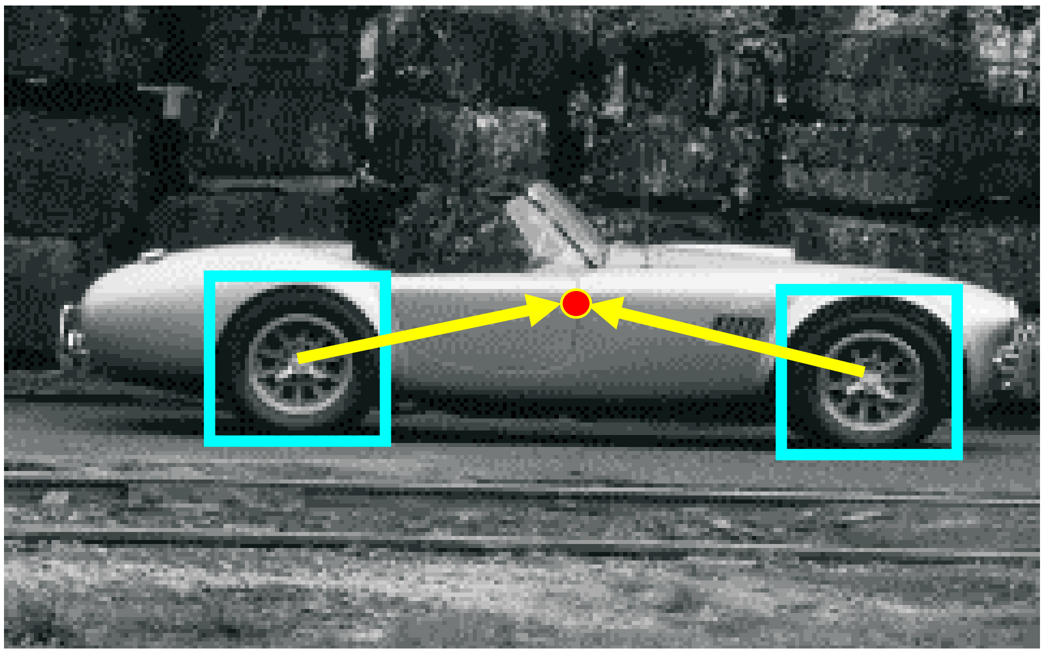


Model

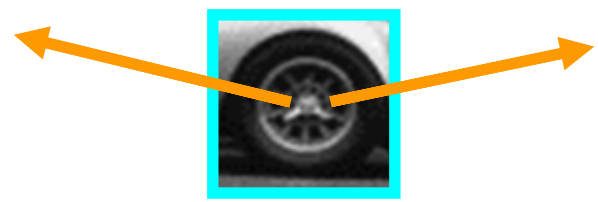


Application in recognition

- Index displacements by "visual codeword"



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

Application in recognition

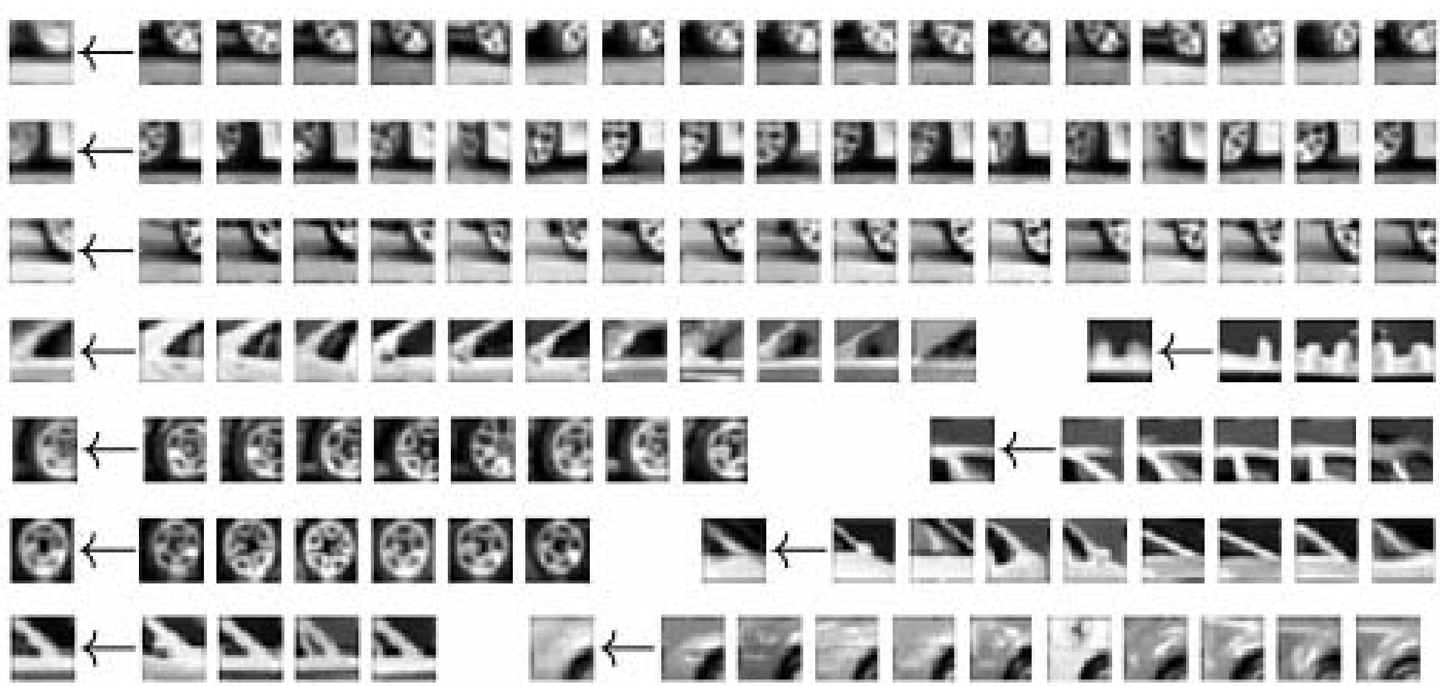
- Index displacements by "visual codeword"

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

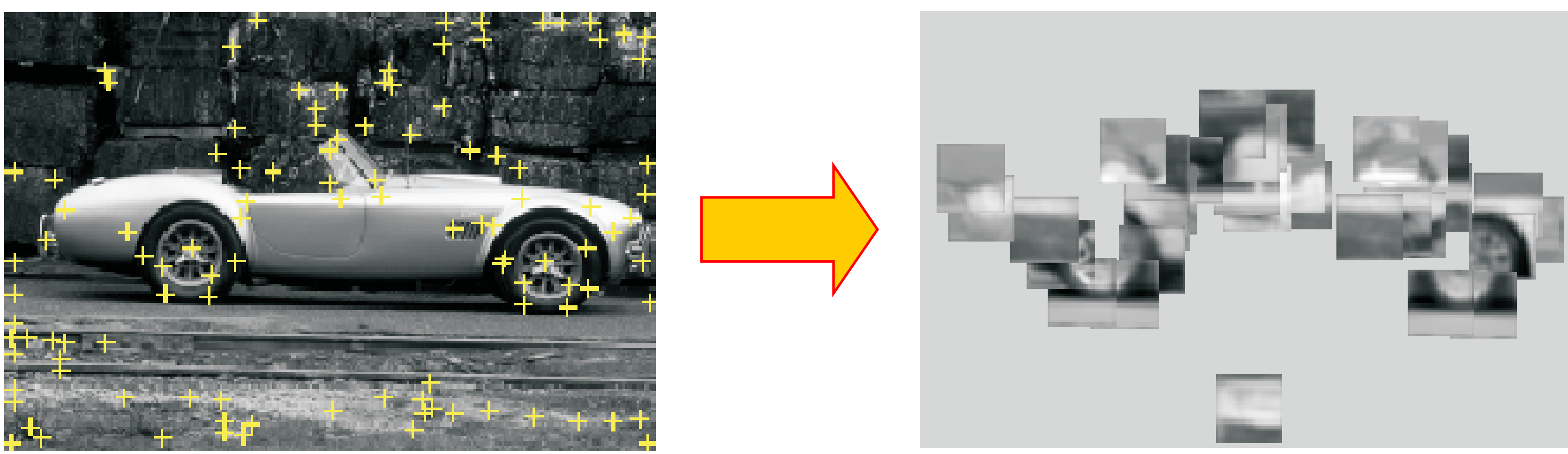
Implicit shape models: Training

1. Build *codebook* of patches around extracted interest points using clustering (more on this later in the course)



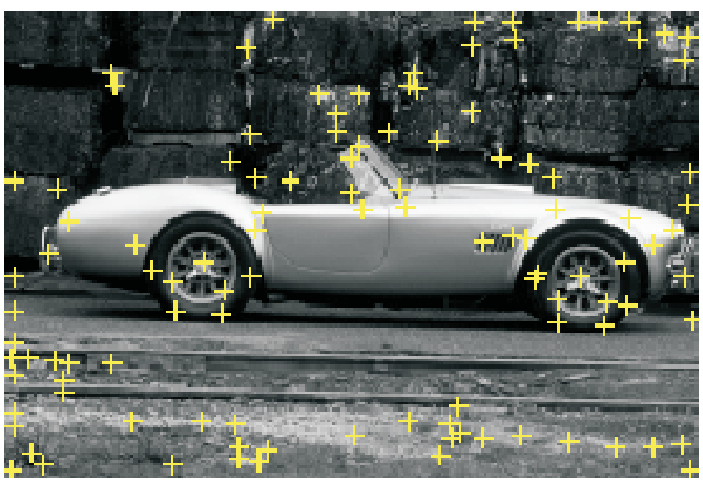
Implicit shape models: Training

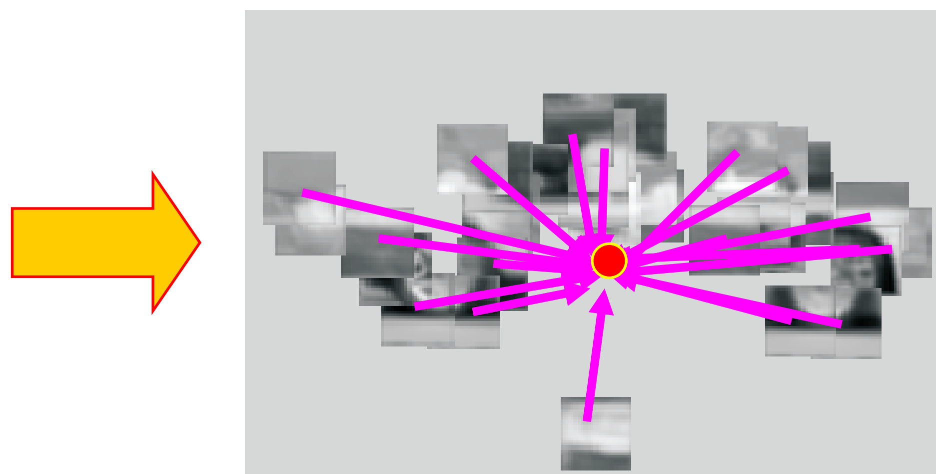
1. Build *codebook* of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest codebook entry



Implicit shape models: Training

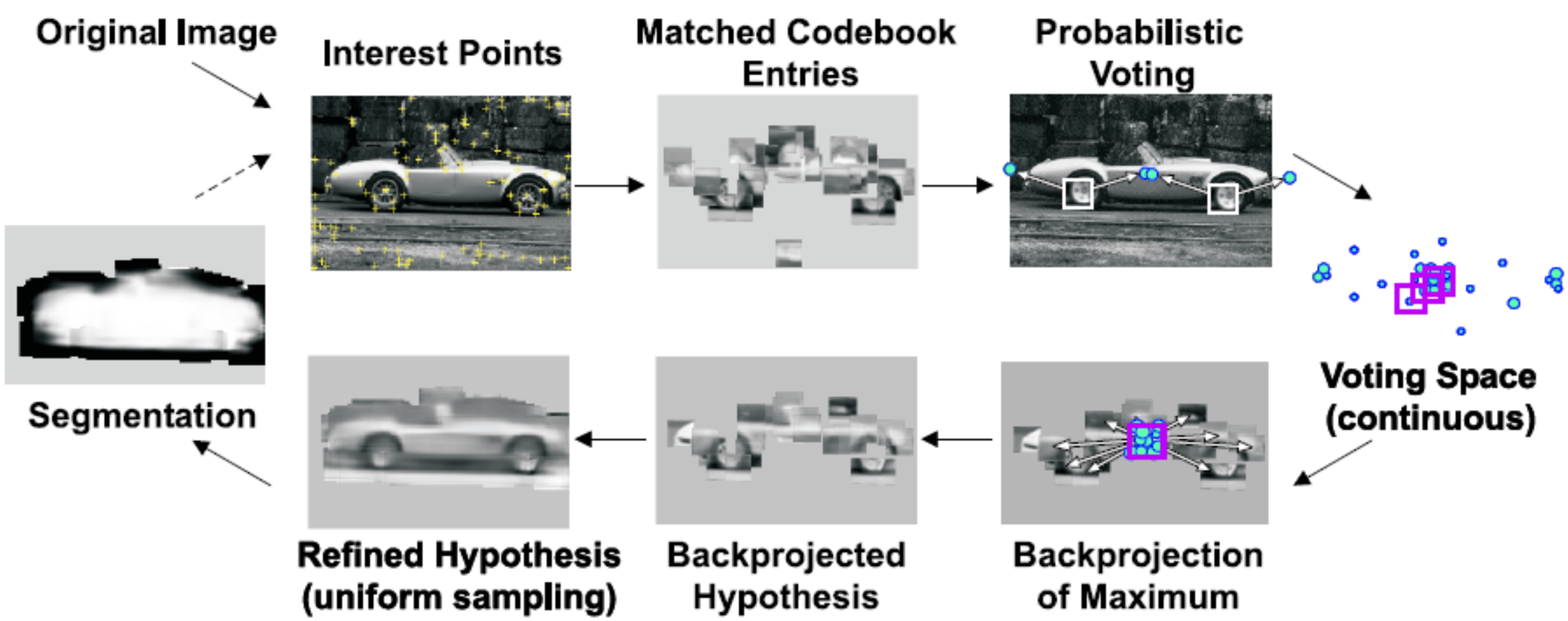
1. Build *codebook* of patches around extracted interest points using clustering
2. Map the patch around each interest point to closest codebook entry
3. For each codebook entry, store all positions it was found, relative to object center



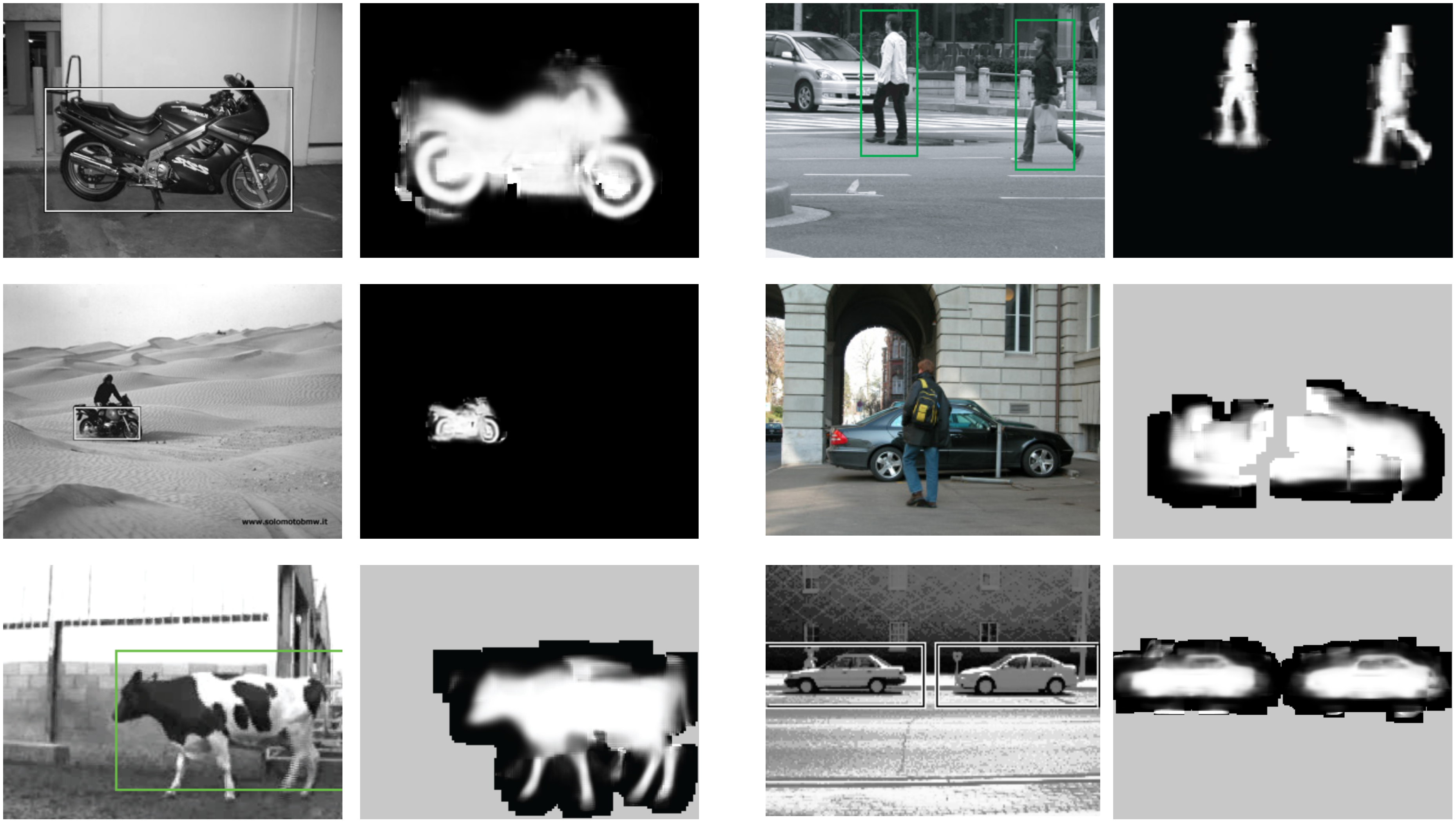


Implicit shape models: Testing

1. Given test image, extract patches, match to codebook entry
2. Cast votes for possible positions of object center
3. Search for maxima in voting space
4. Extract weighted segmentation mask based on stored masks for the codebook occurrences

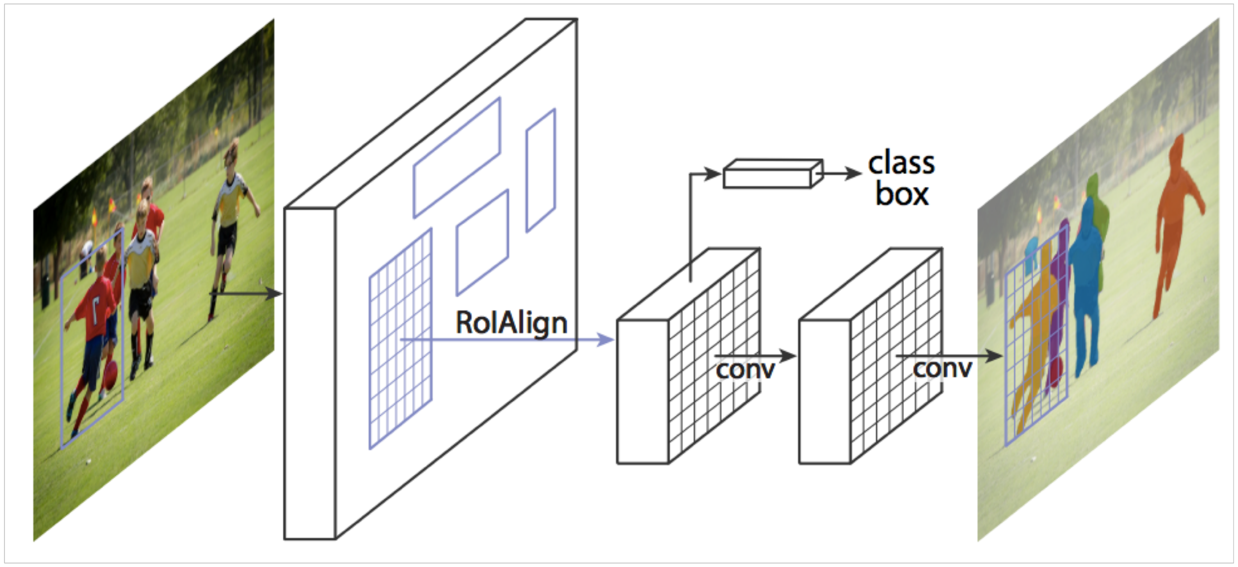


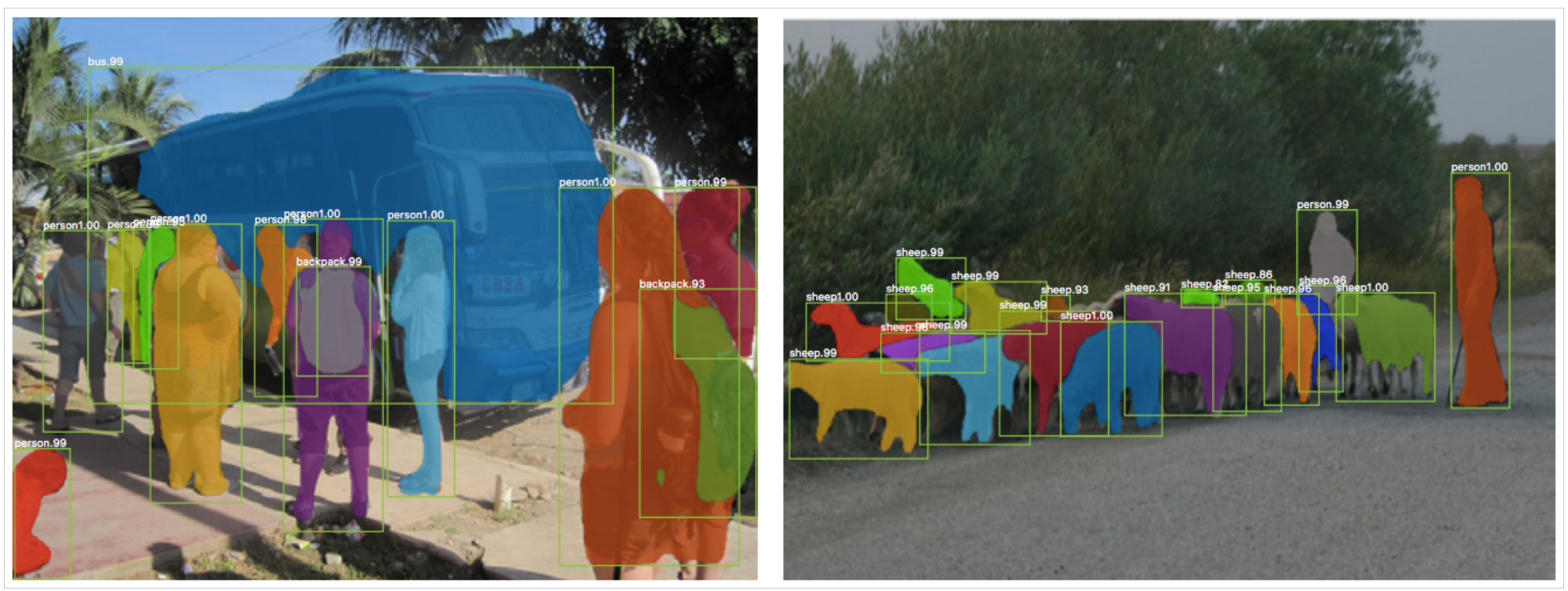
Additional examples



B. Leibe, A. Leonardis, and B. Schiele, [Robust Object Detection with Interleaved Categorization and Segmentation](),
IJCV 77 (1-3), pp. 259-289, 2008.

These days: Mask R-CNN





K. He, G. Gkioxari, P. Dollar, and R. Girshick, [Mask R-CNN](), ICCV 2017

Review: Hough transform

- Hough transform for lines
- Hough transform for circles
- Generalized Hough transform for template detection
- Hough transform pros and cons