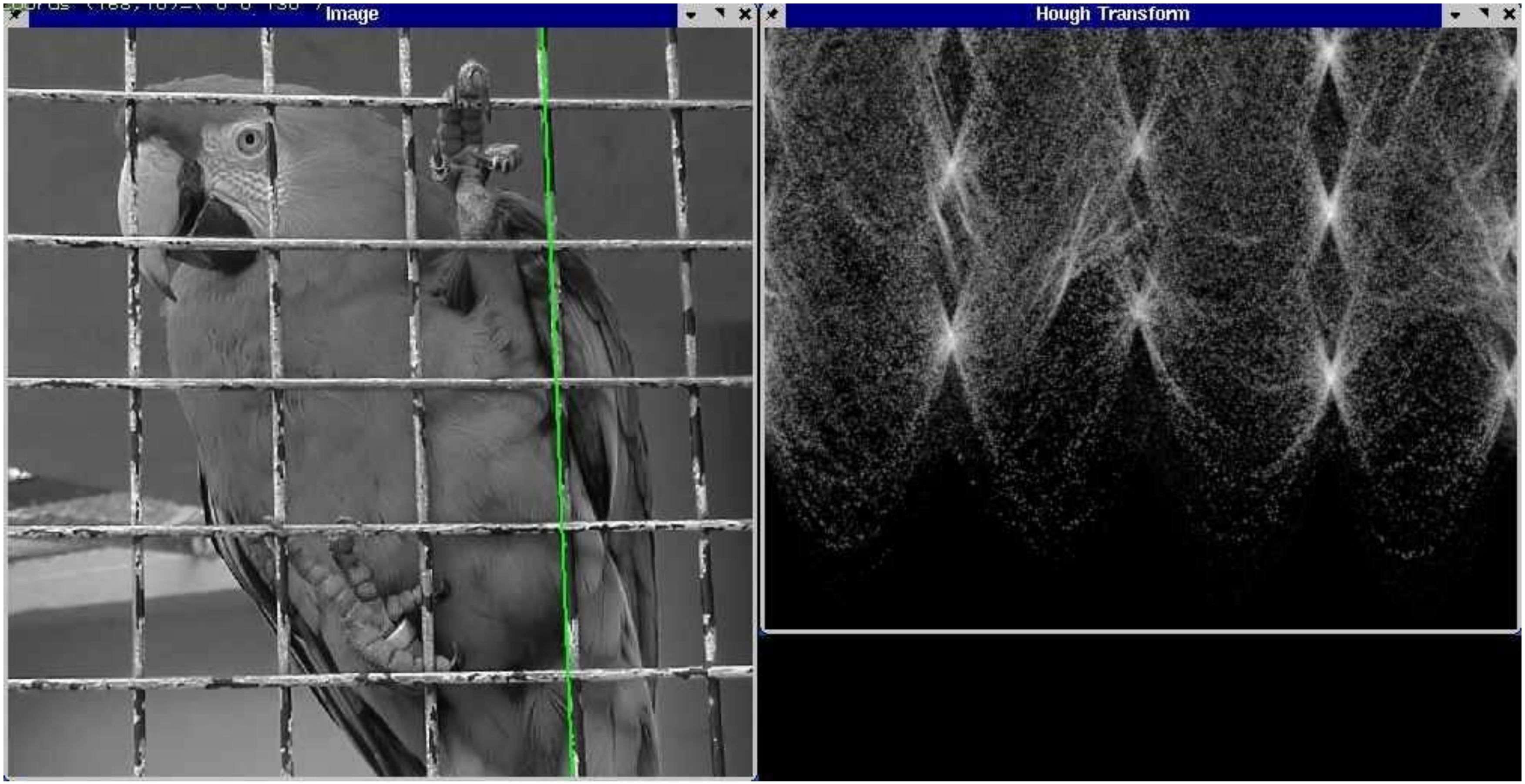
Fitting: The Hough transform

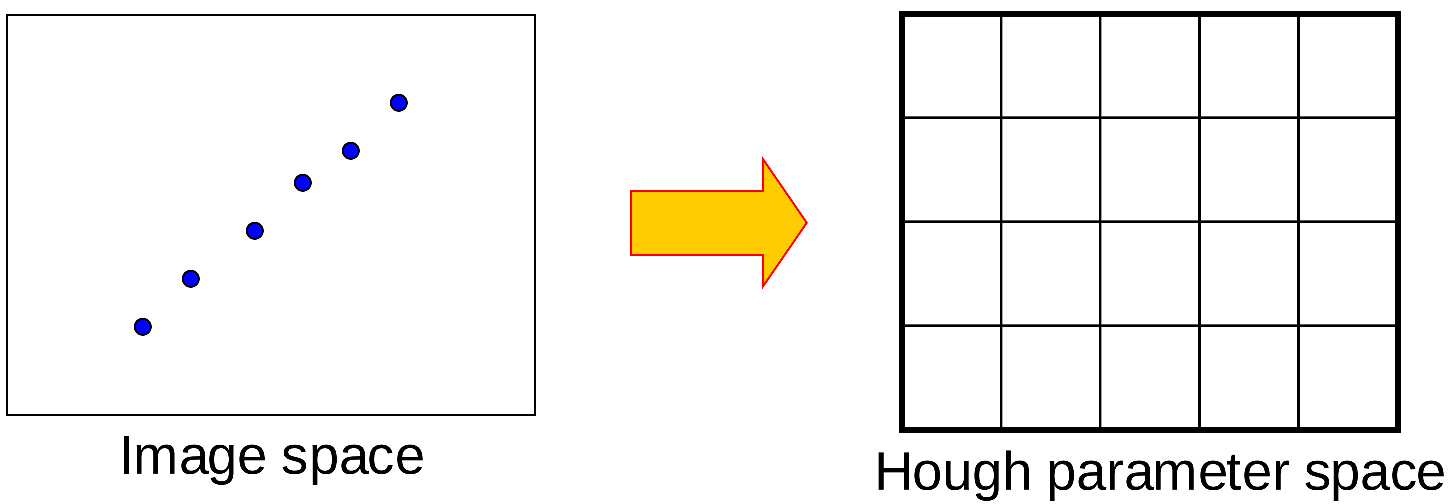


Voting schemes

- 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

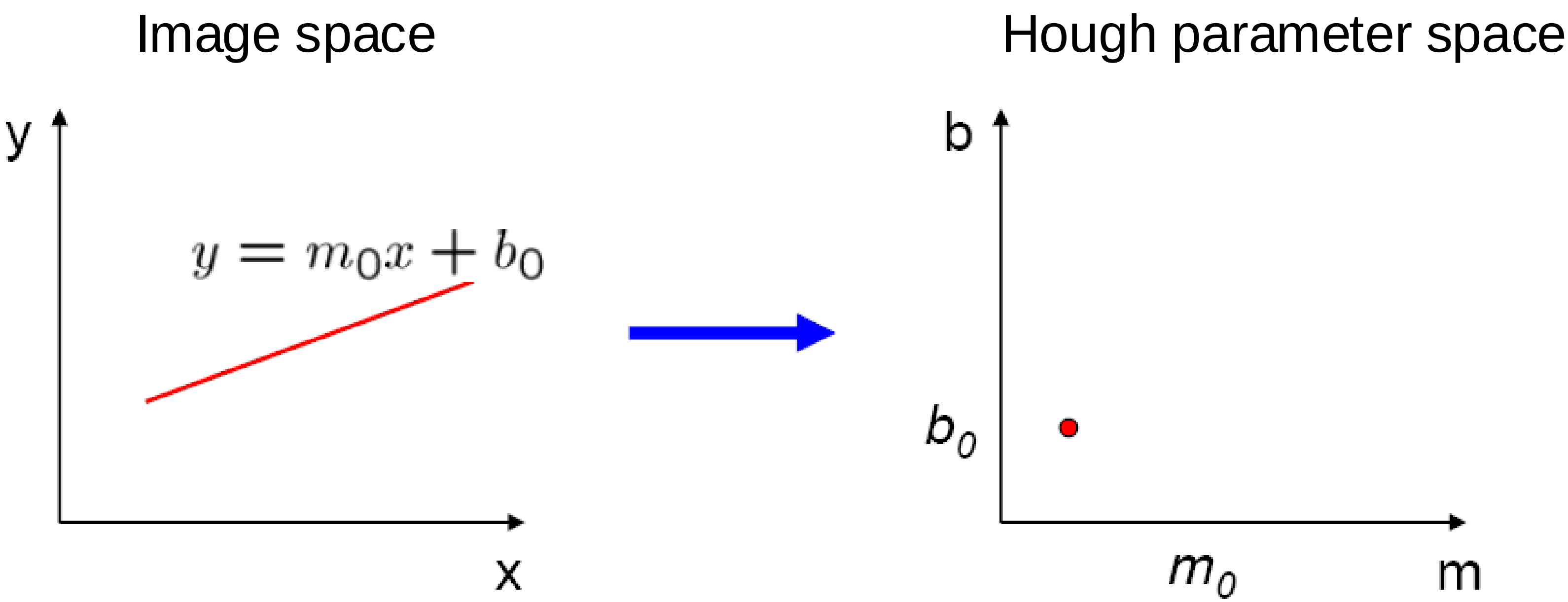
- An early type of voting scheme
- General outline:
  - Discretize *parameter space* into bins
  - For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  - Find bins that have the most votes



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

Parameter space representation

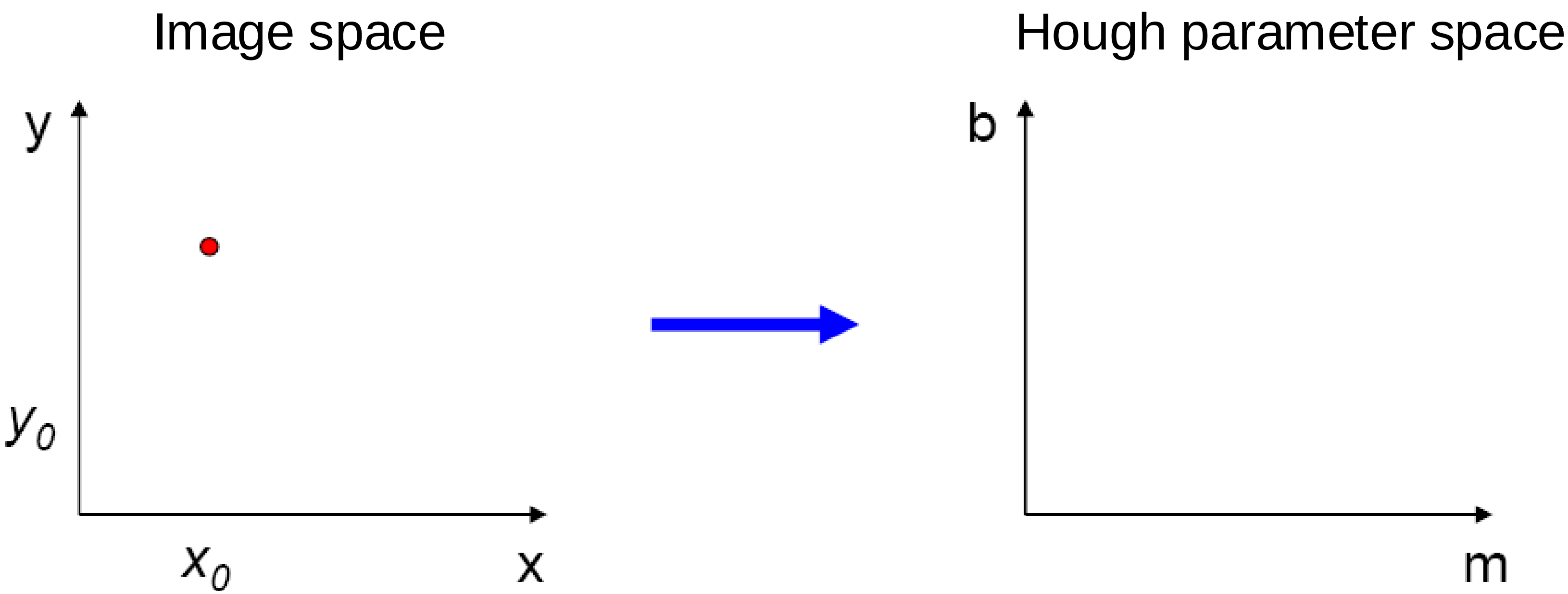
- A line in the image corresponds to a point in Hough space



Source: S. Seitz

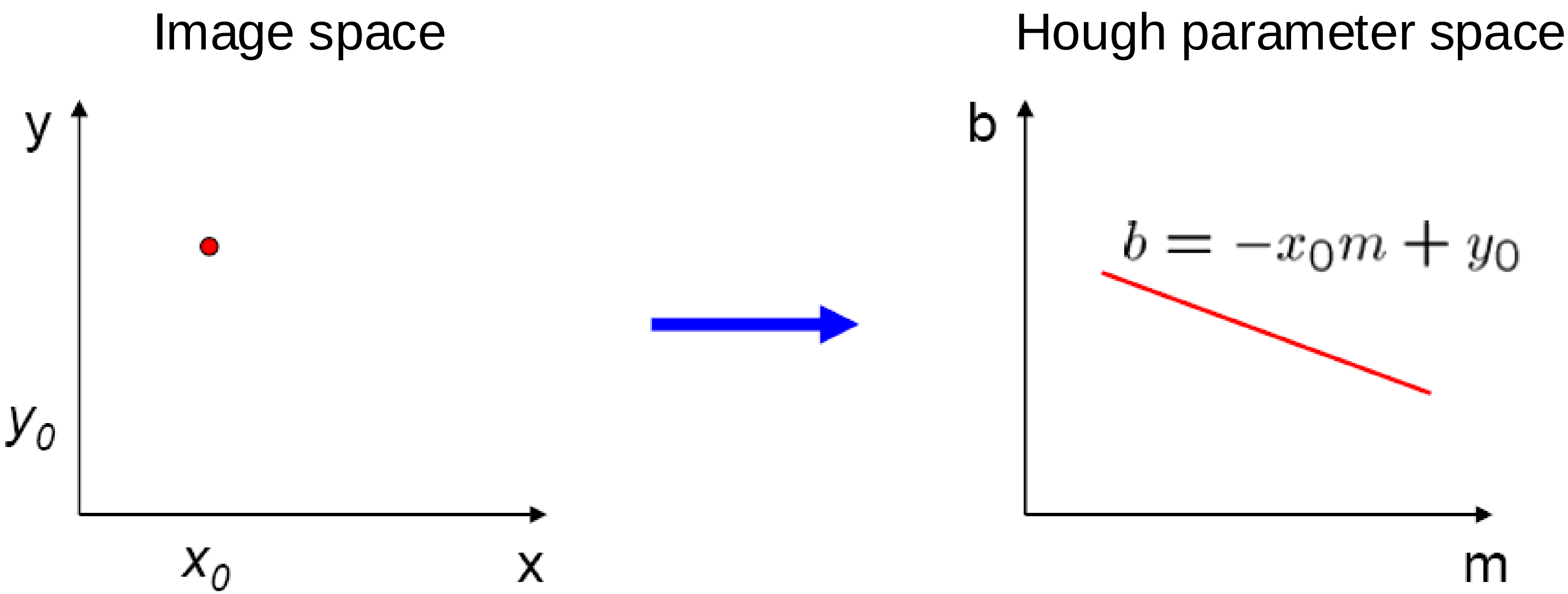
Parameter space representation

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



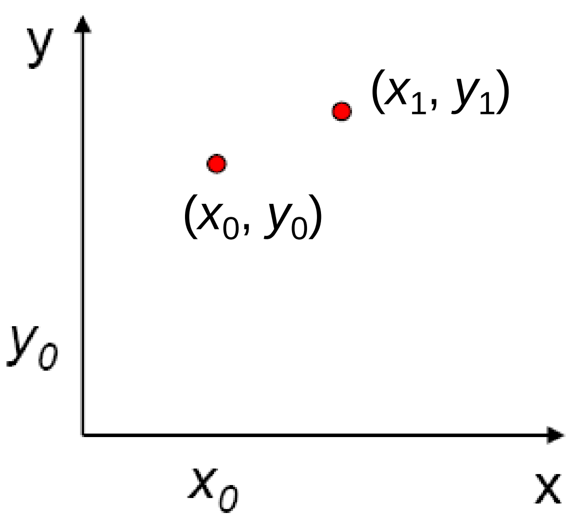
Parameter space representation

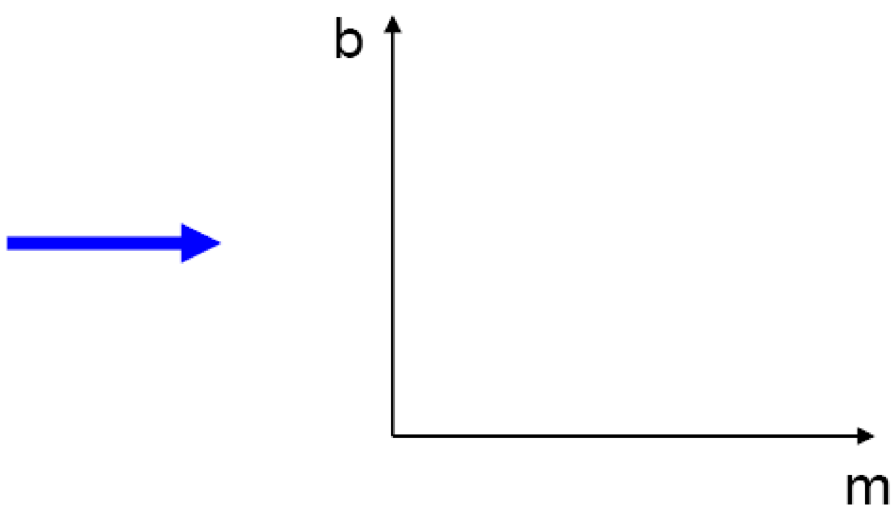
- What does a point (x_0, y_0) in the image space map to in the Hough space?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - This is a line in Hough space



Parameter space representation

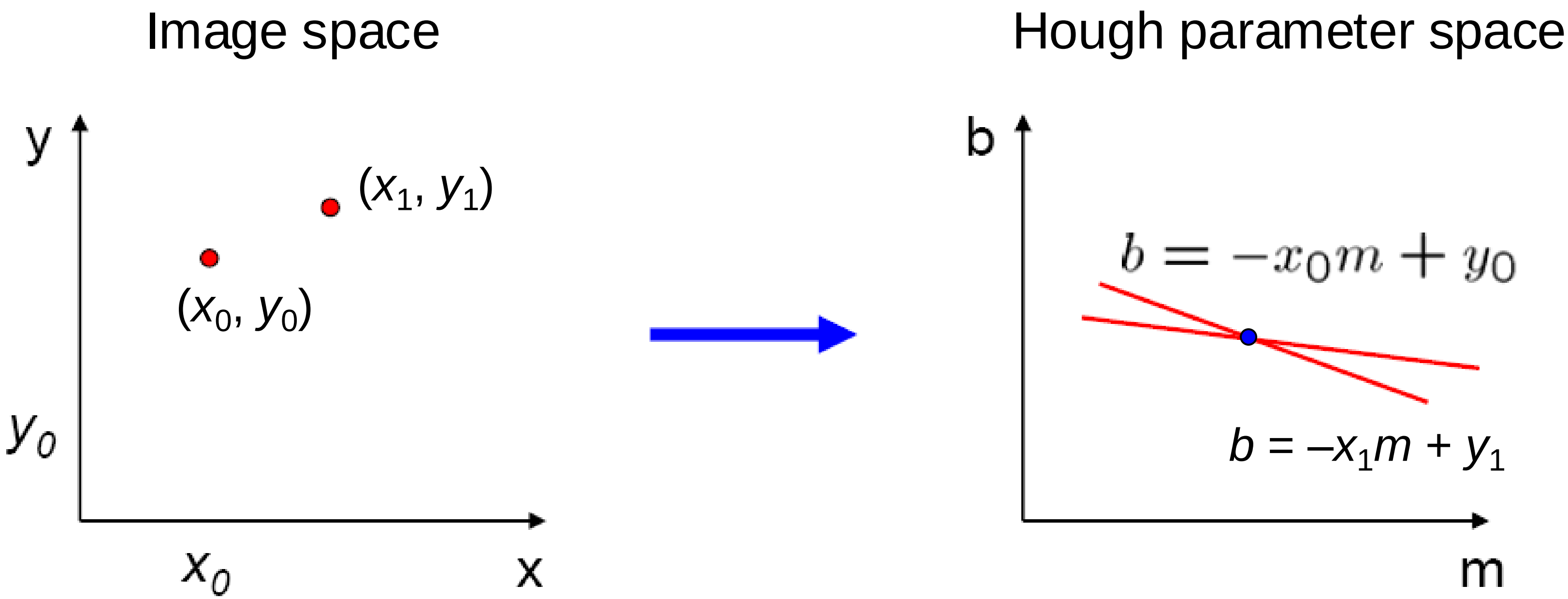
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?





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 $b = -x_1m + y_1$



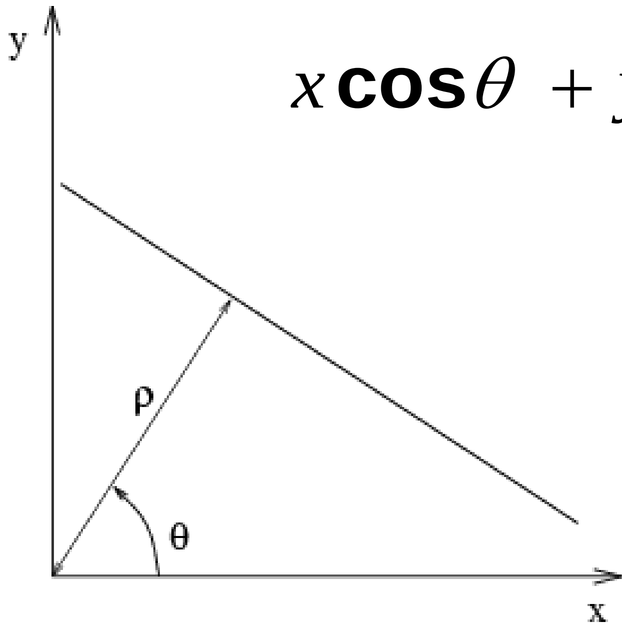
Parameter space representation

- 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*

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



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

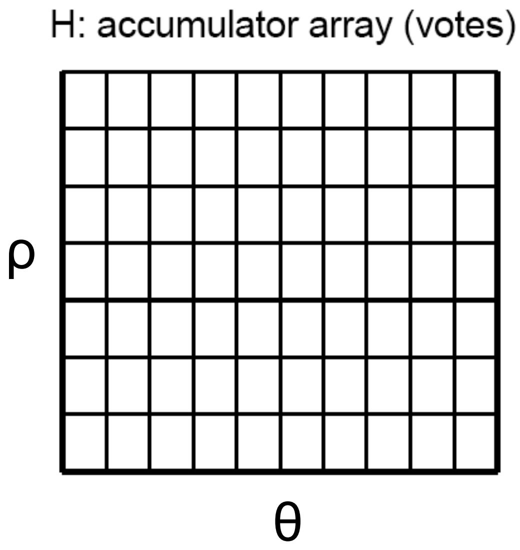
    For θ = 0 to 180

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

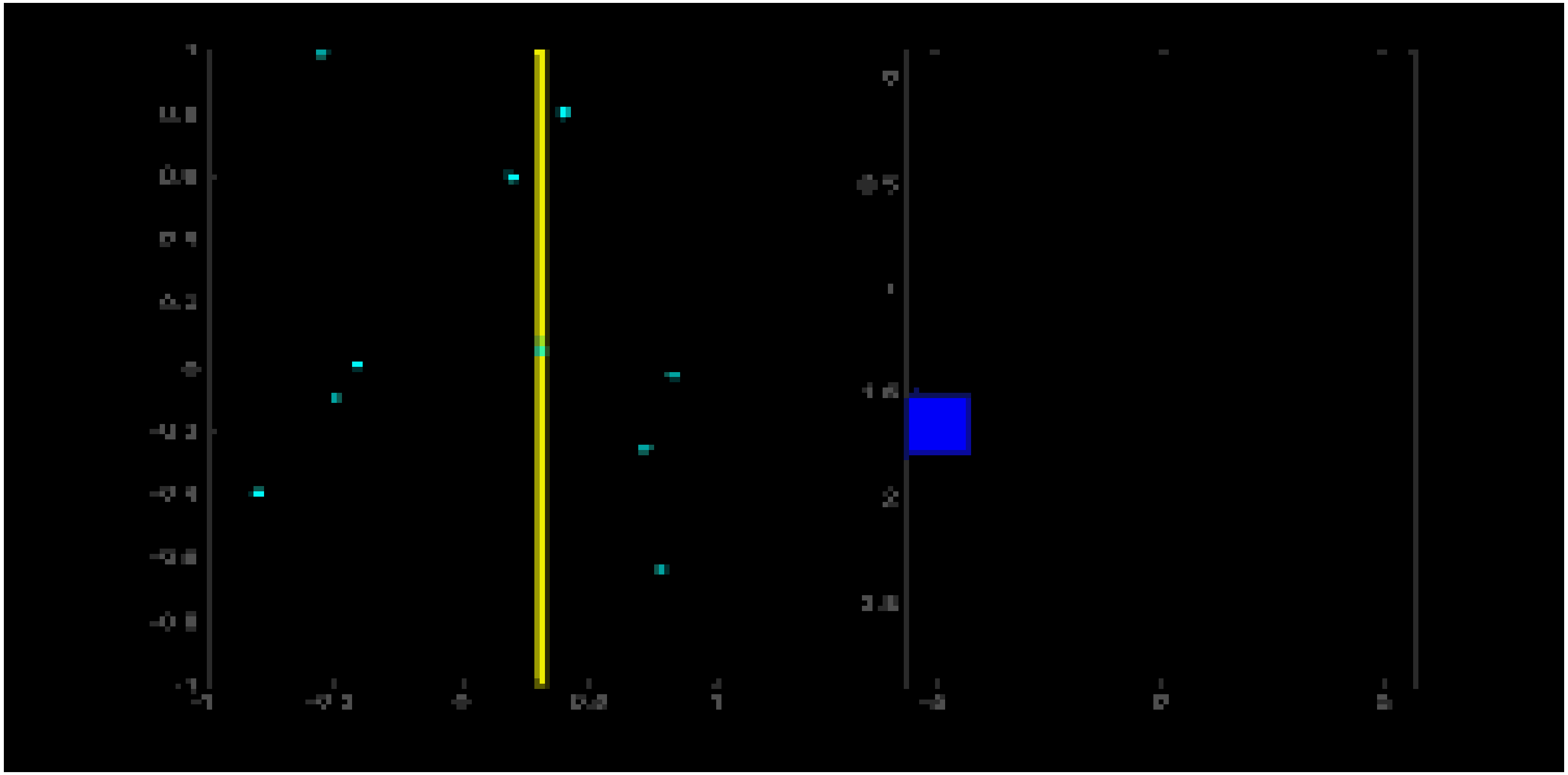
        $H(\theta, \rho) \leftarrow H(\theta, \rho) + 1$

    end

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

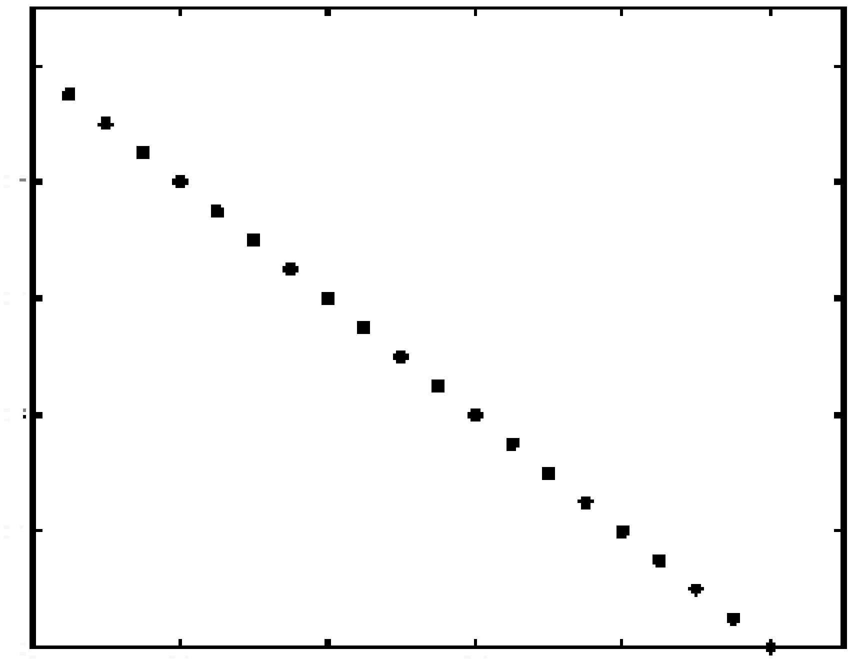


Animation Illustration of Algorithm

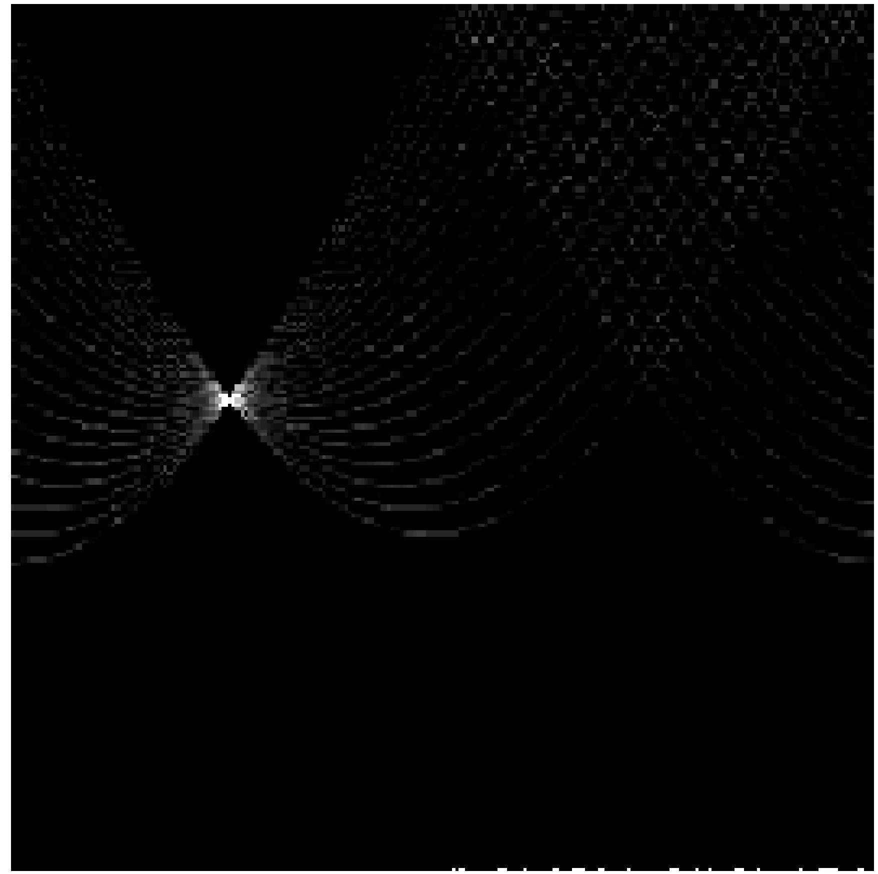


https://docs.opencv.org/4.x/d3/de6/tutorial_js_houghlines.html

Basic illustration



features



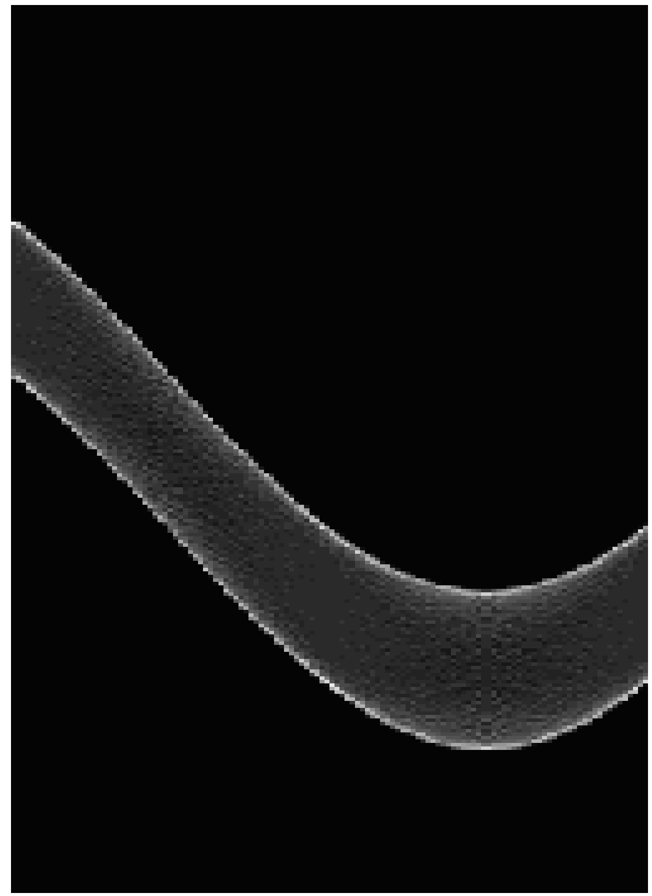
votes

Hough transform demo

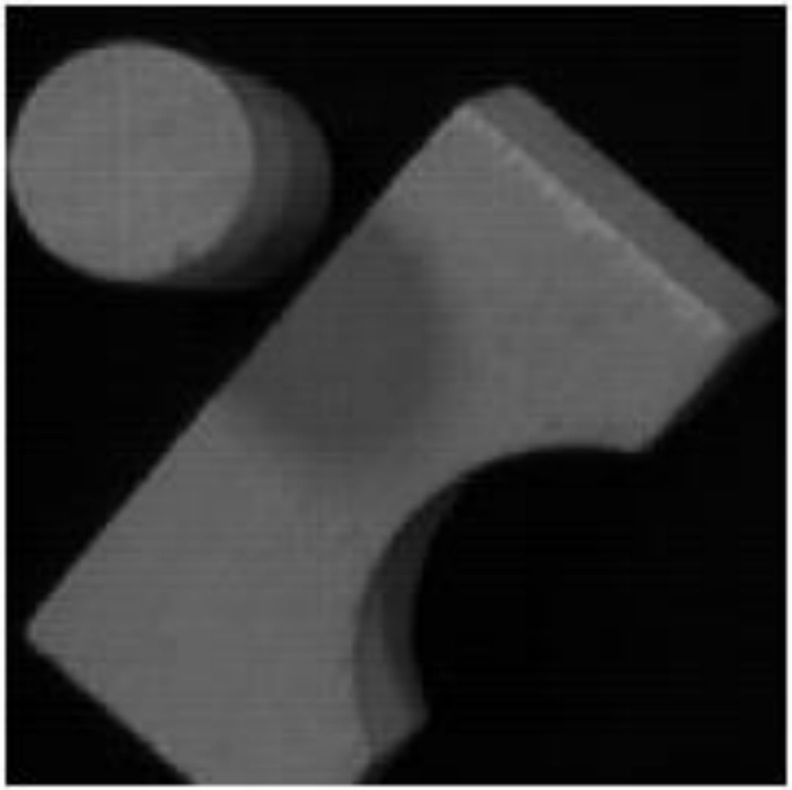
Other shapes

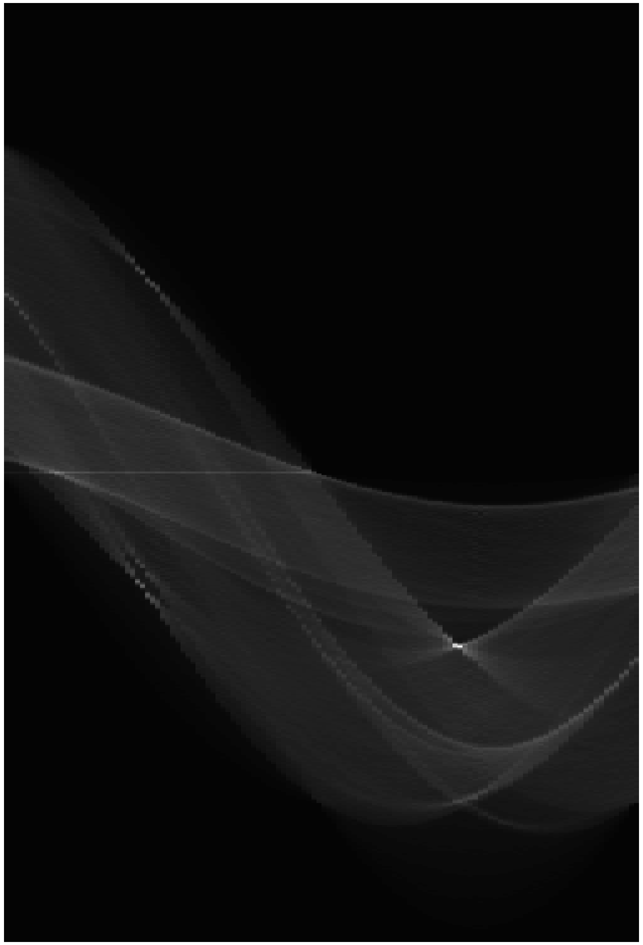
Square

Circle

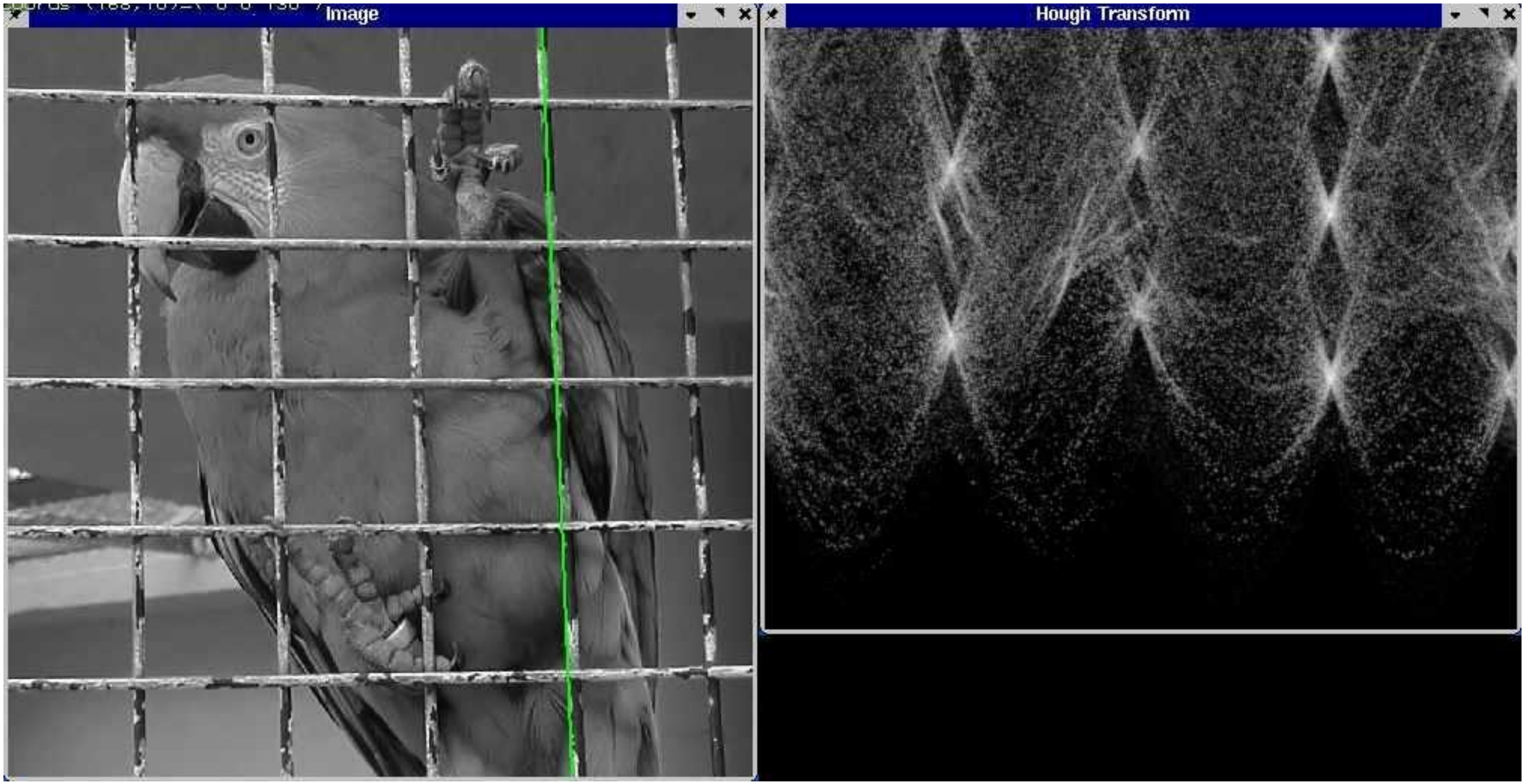


Several lines



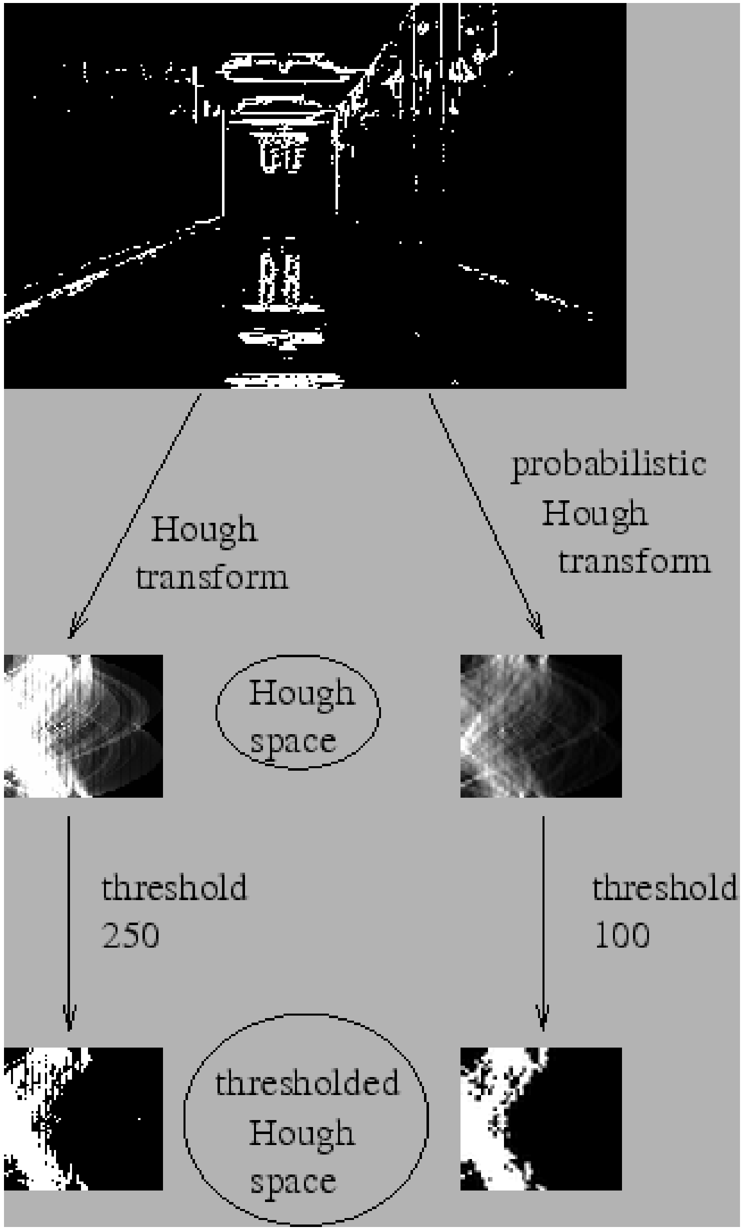


A more complicated image



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

OpenCV: Hough Line Transform

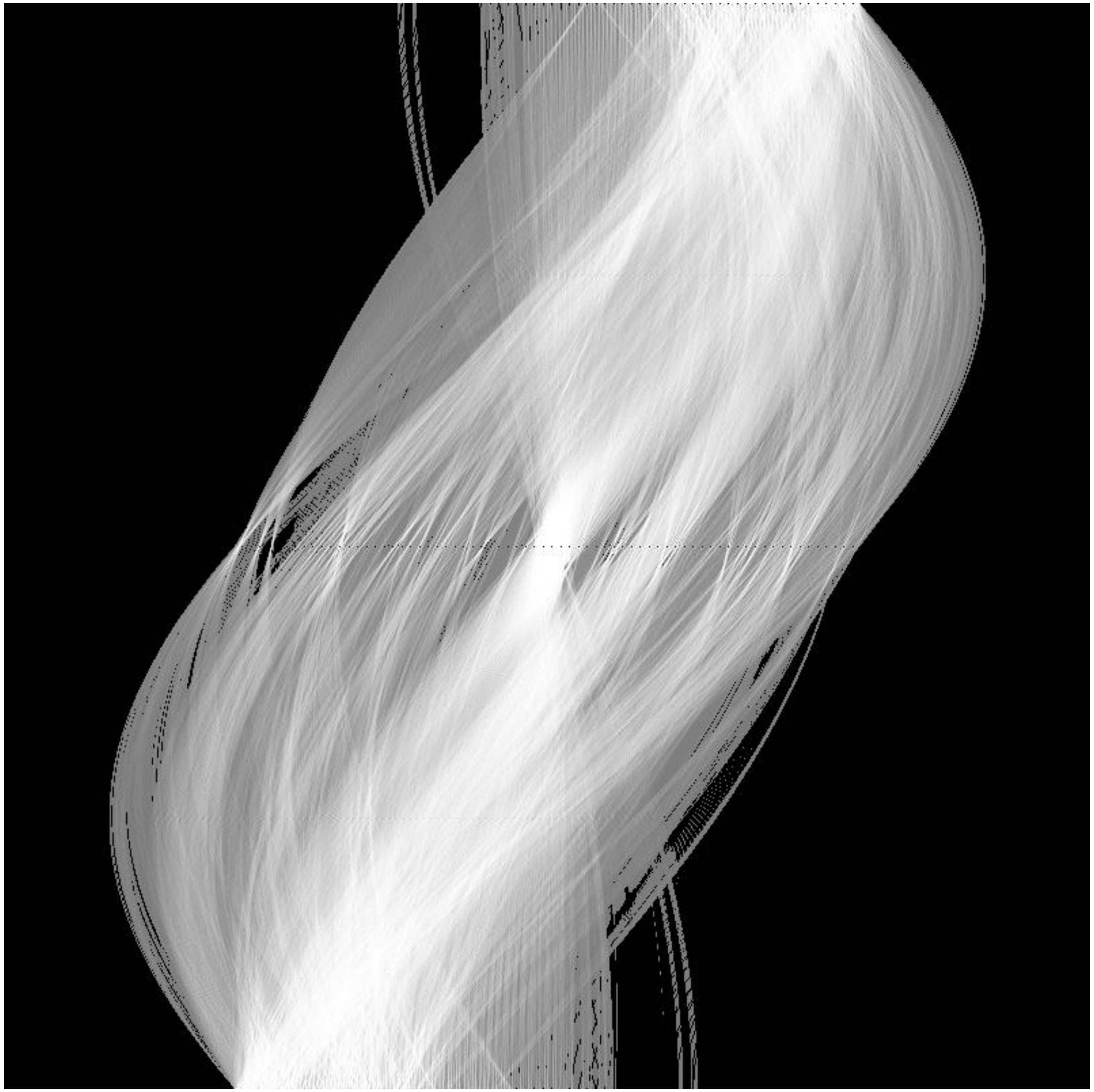


Franck Bettinger's home page

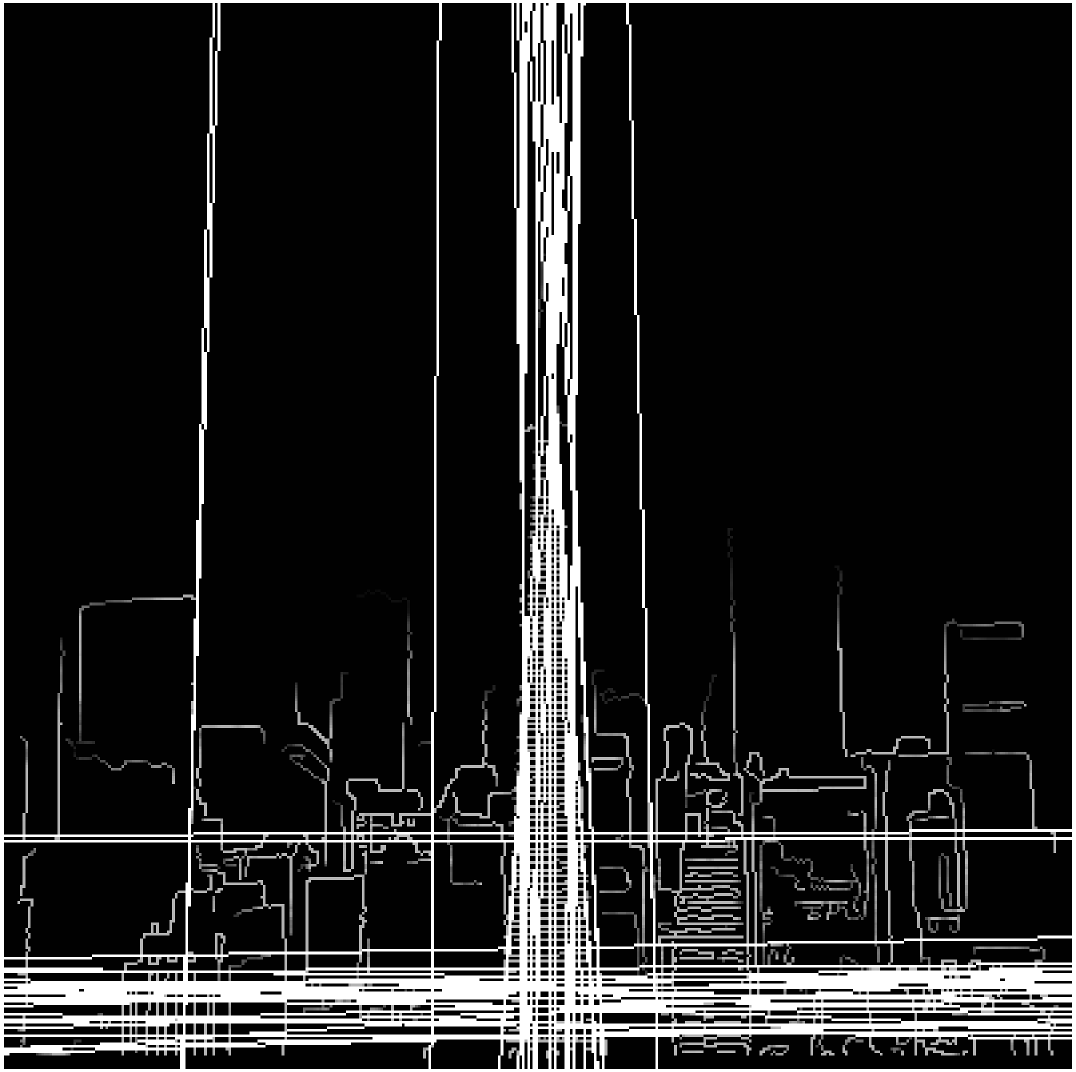
Example: Detect Lines in Foggy Photo

Apply Canny Edge Detector

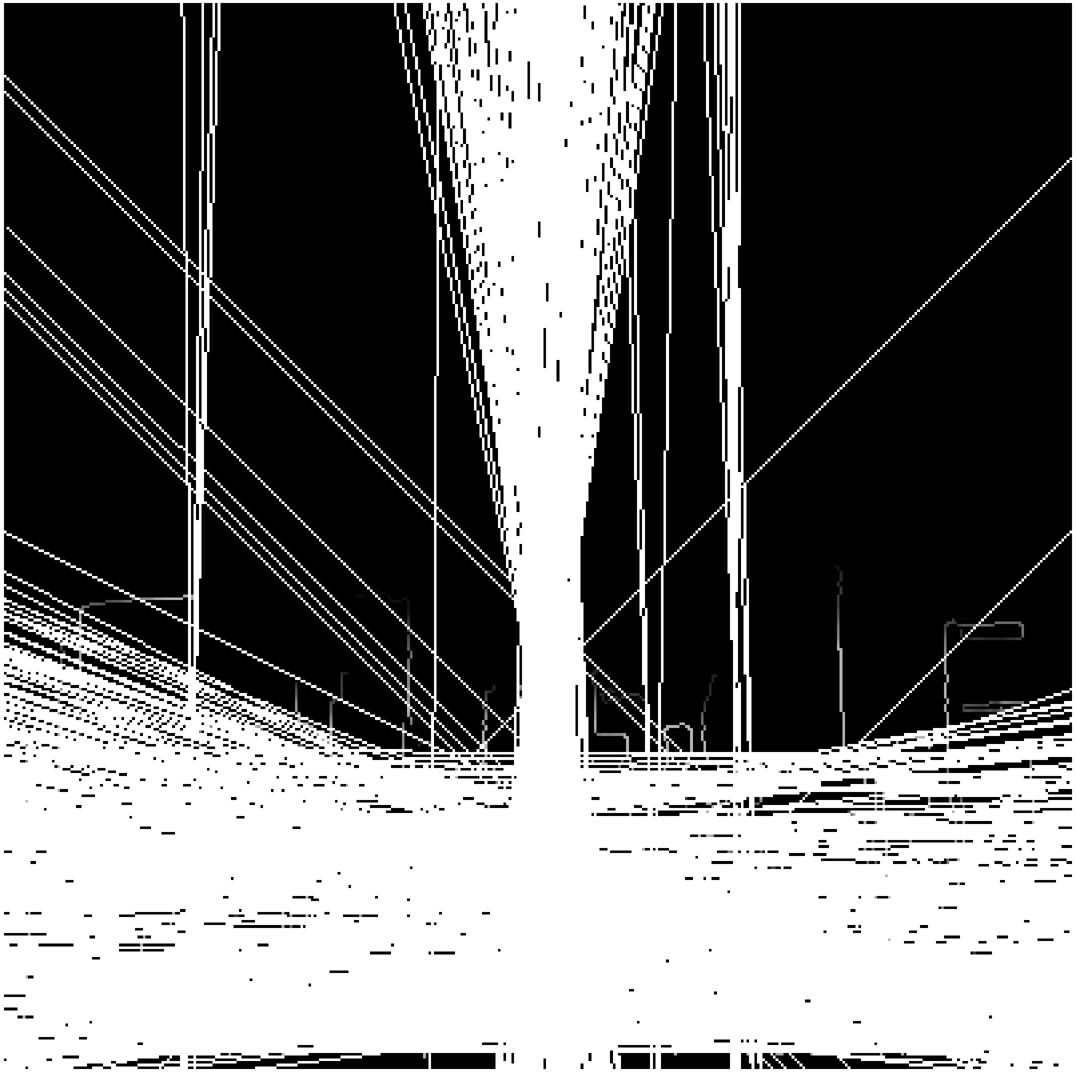
Hough Transform of Edge



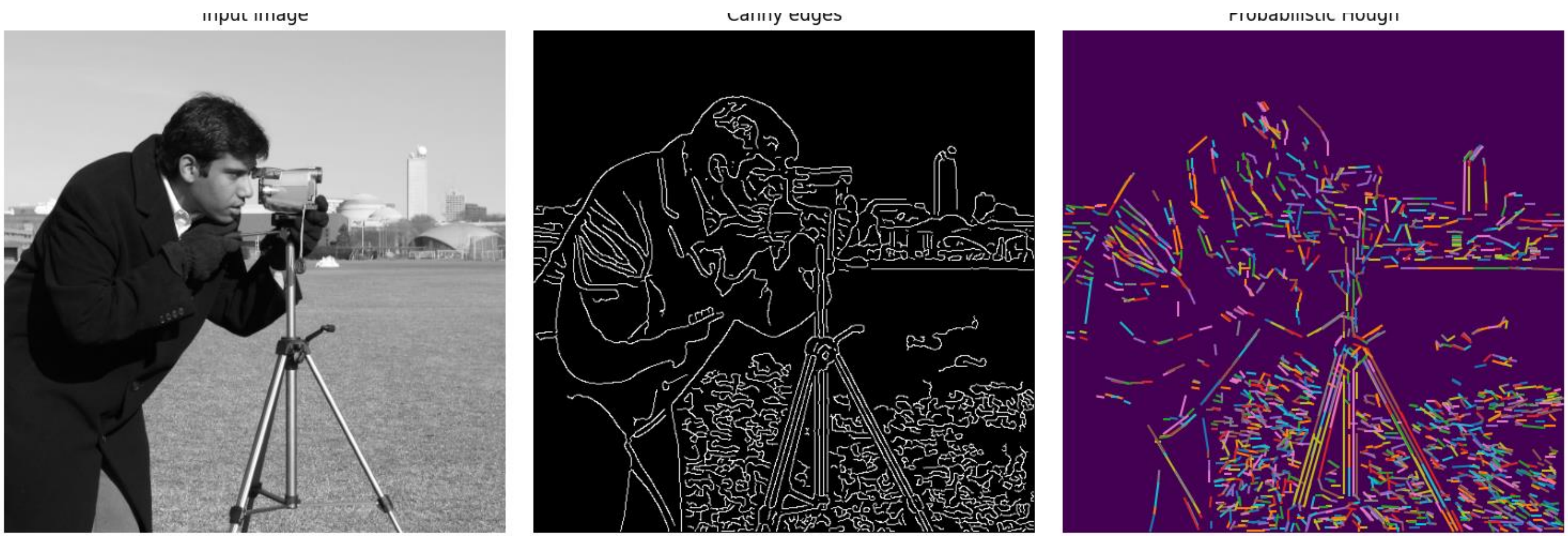
Thresholding



Thresholding



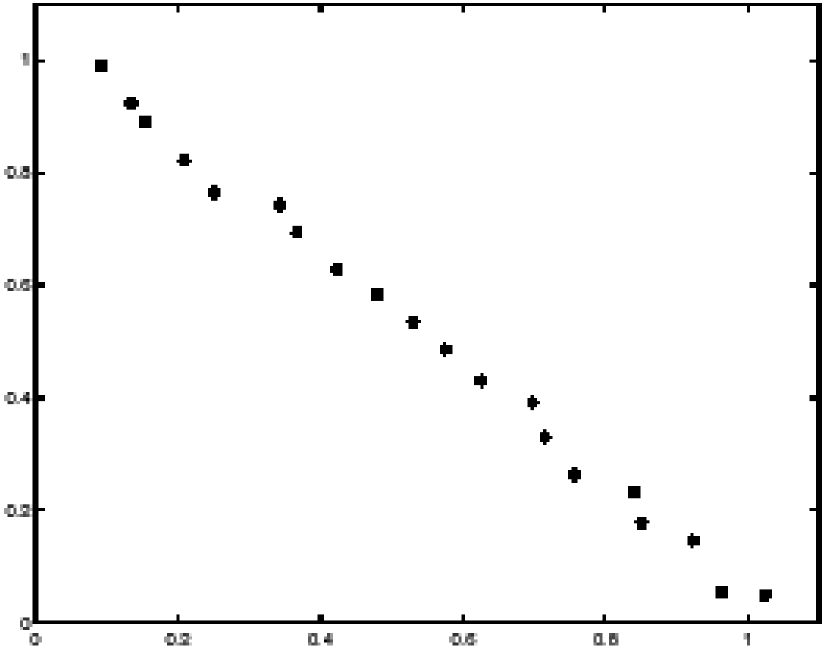
Probabilistic Hough Transform



Progressive probabilistic Hough transform for line detection

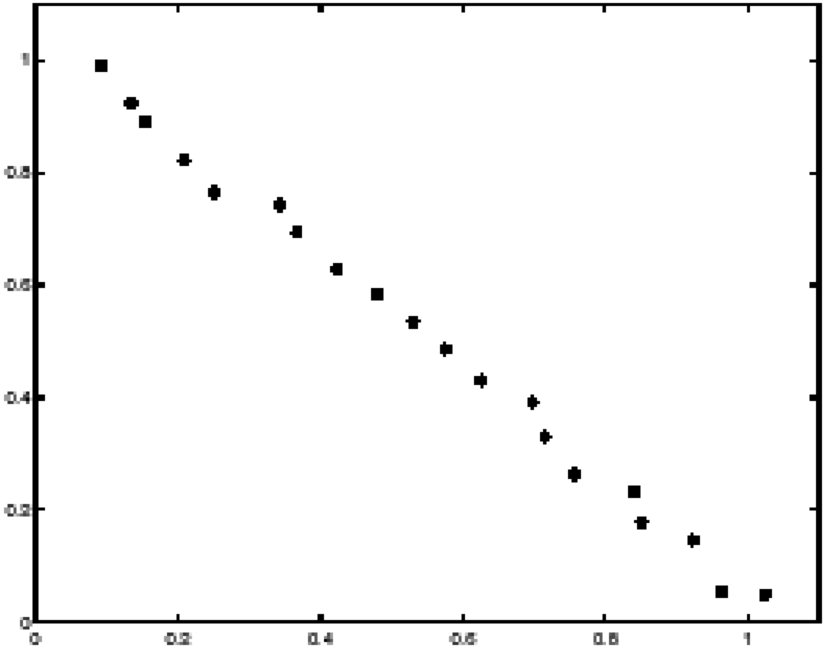
https://scikit-image.org/docs/stable/auto_examples/edges/plot_line_hough_transform.html

Effect of noise

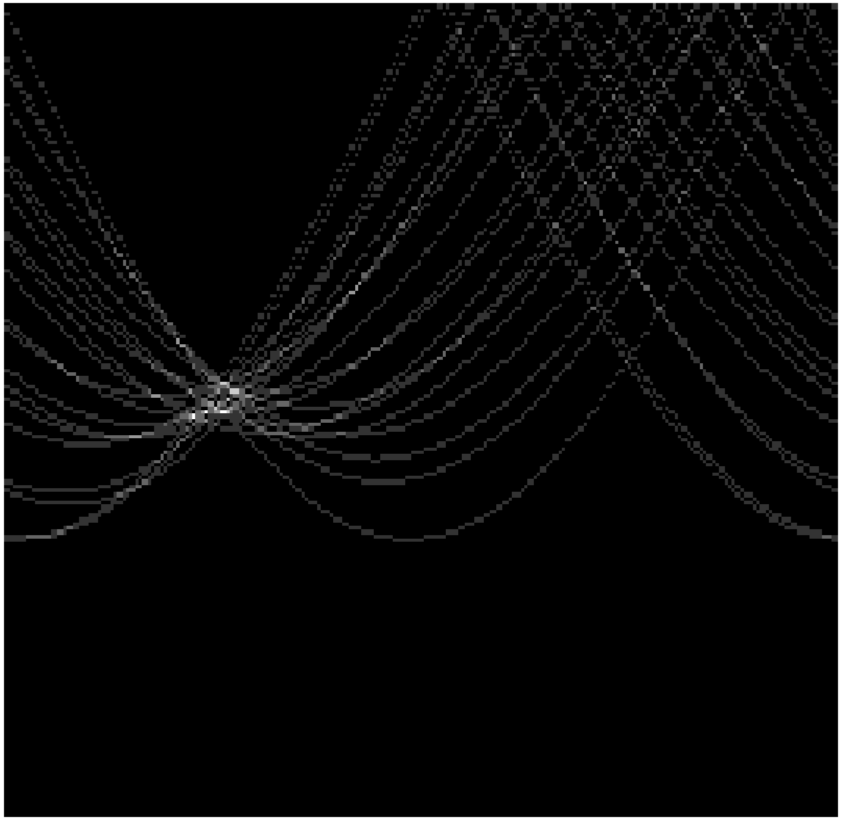


features

Effect of noise



features

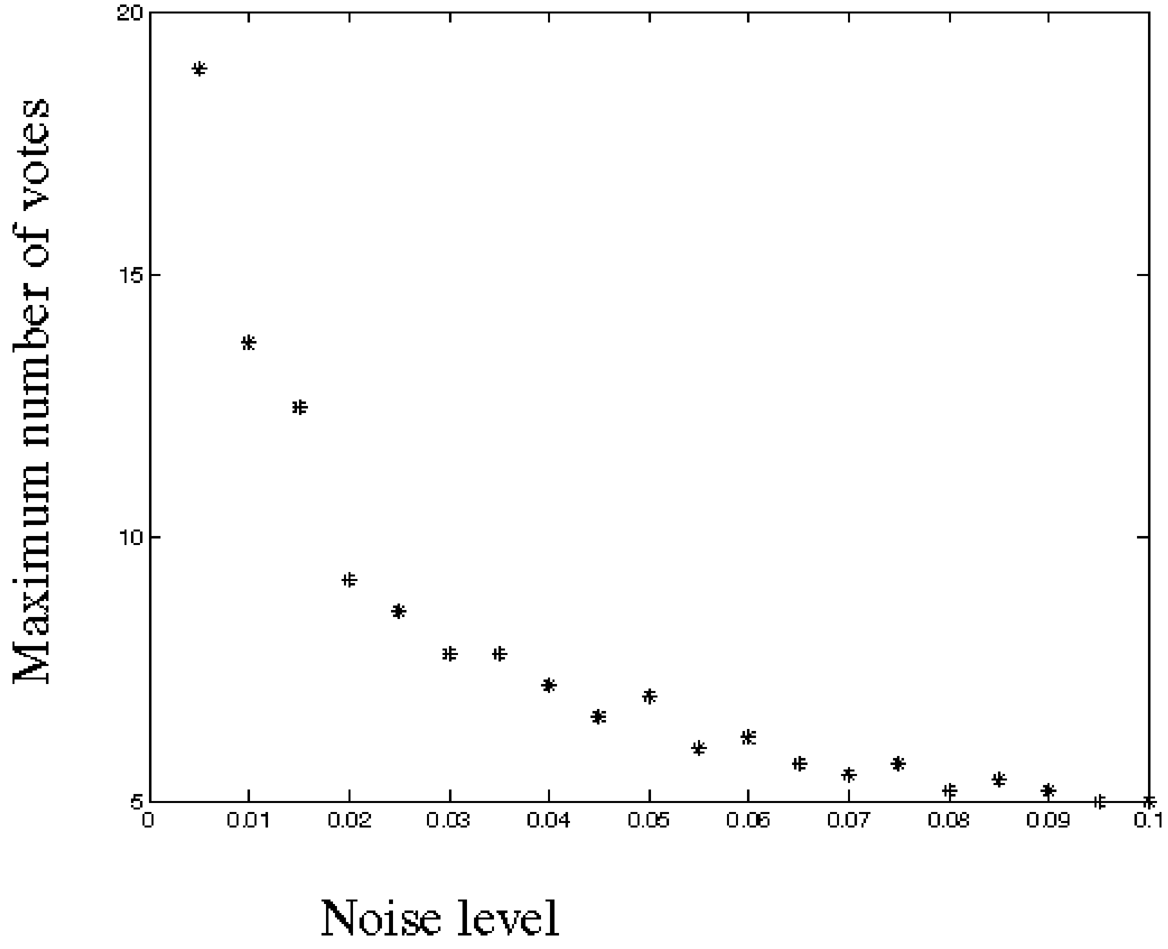


votes

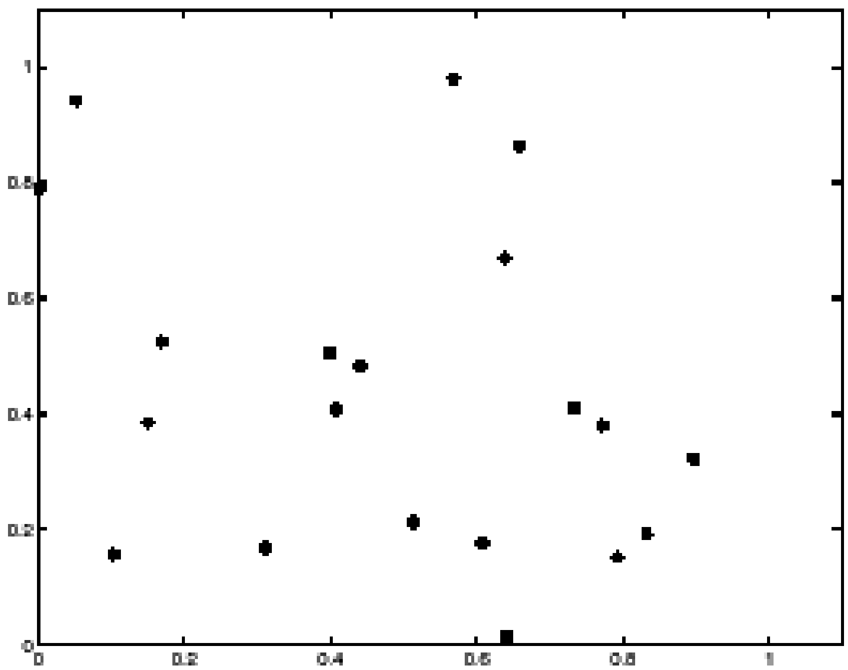
Peak gets fuzzy and hard to locate

Effect of noise

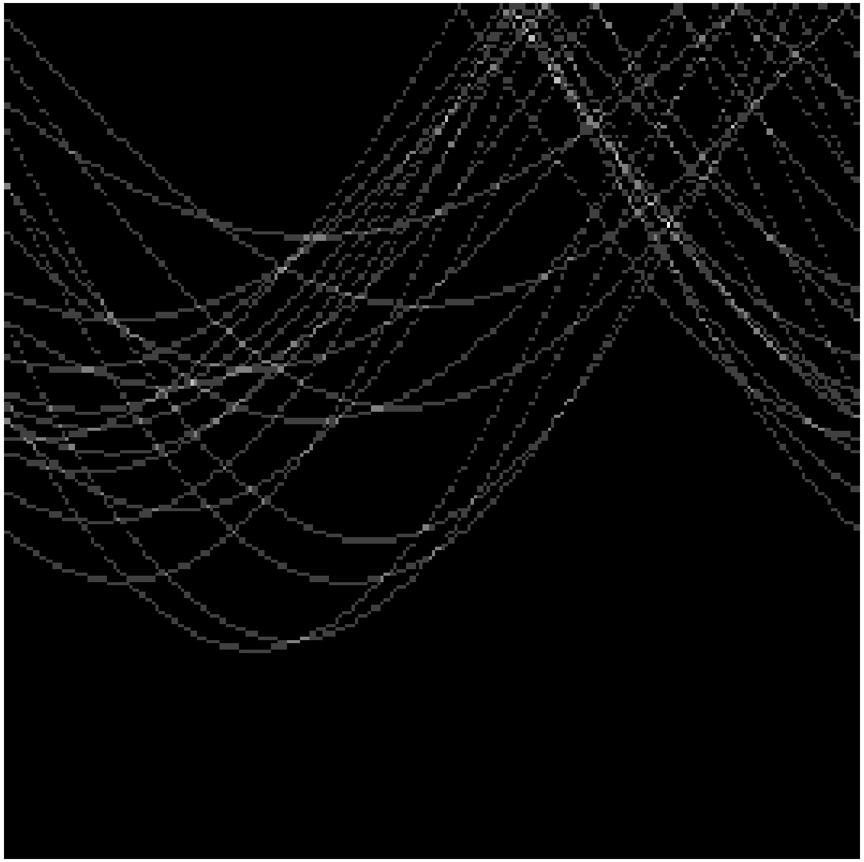
- Number of votes for a line of 20 points with increasing noise:



Random points



features

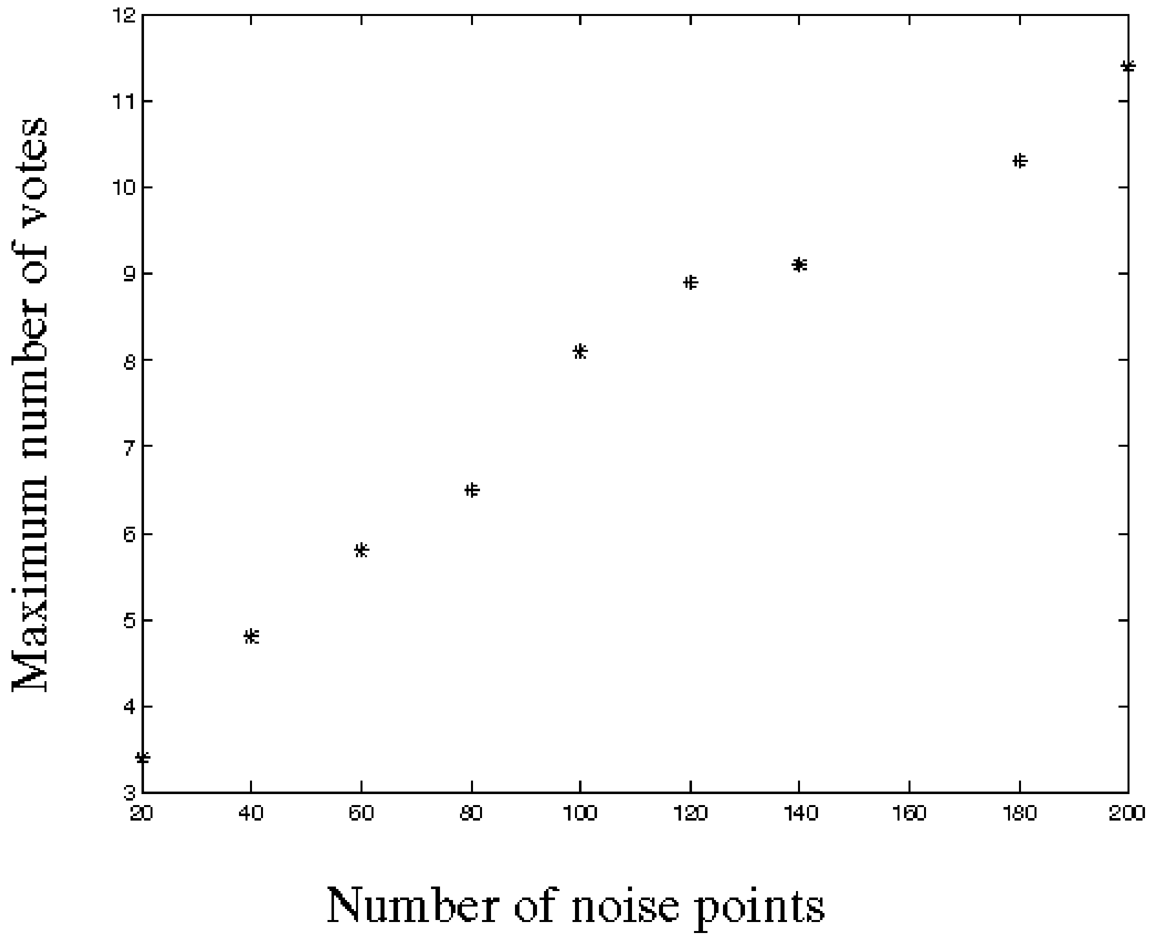


votes

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

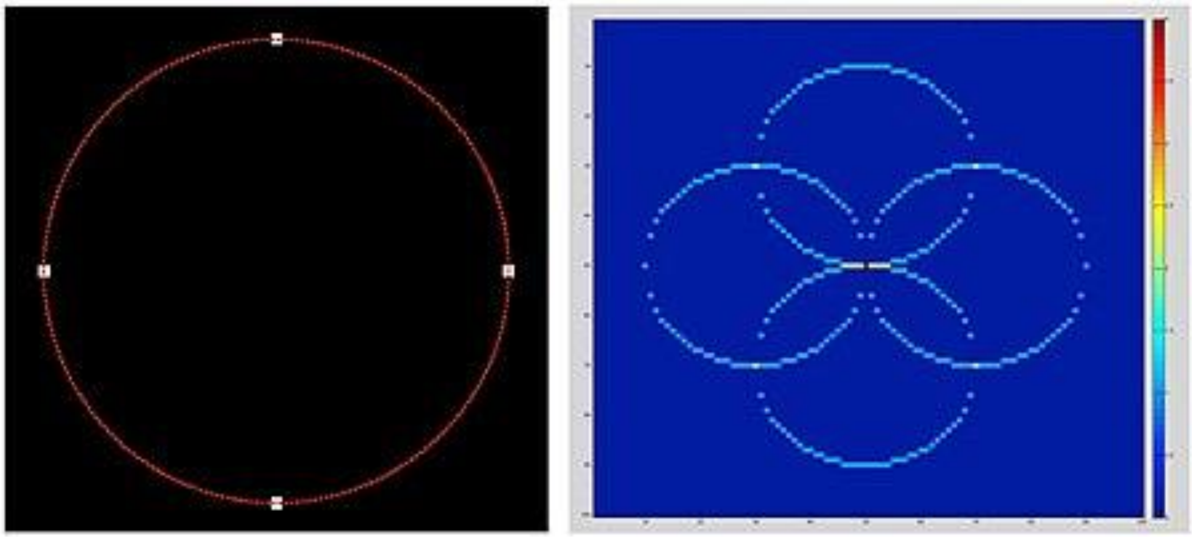
- Choose a good grid / discretization
  - **Too coarse:** large votes obtained when too many different lines correspond to a single bucket
  - **Too fine:** miss lines because some points that are not exactly collinear cast votes for different buckets
- Increment neighboring bins (smoothing in accumulator array)

Circle Hough Transform

- How can we find circles using Hough Transform?
- How to define the parameter space for circles?
- (a, b, r) in a 3D space:
  - $(x-a)^2 + (y-b)^2 = r^2$

Circle Hough Transform

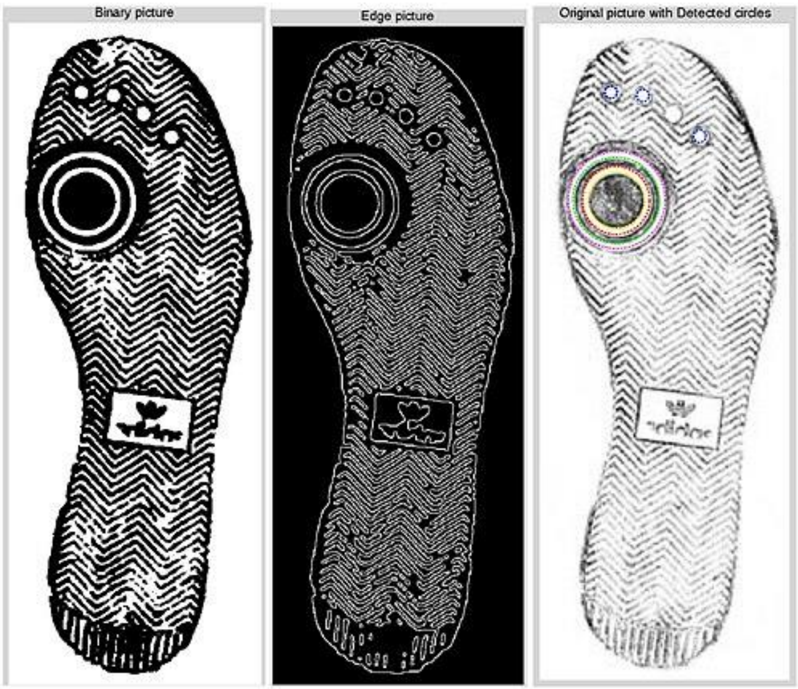
- 3D parameter space (a, b, r):
  - $(x-a)^2 + (y-b)^2 = r^2$
- Find parameters with known radius R

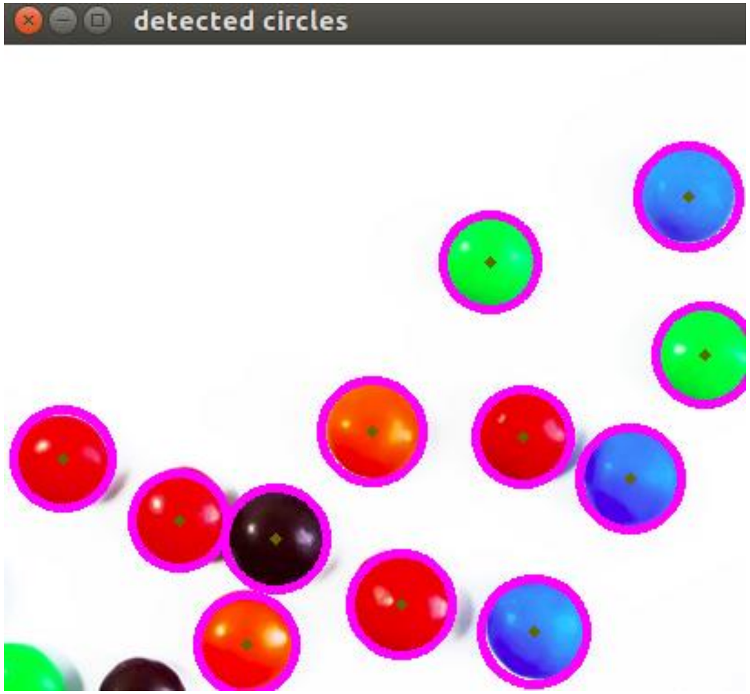


From wiki

Pseudo Algorithm

```
For each A[a,b,r] = 0; // fill with zeroes initially, instantiate 3D matrix
 For each cell(x,y)
     For each theta t = 0 to 360  // the possible  theta 0 to 360
        b = y – r * sin(t * PI / 180);  //polar coordinate for center (convert to radians)
        a = x – r * cos(t * PI / 180); //polar coordinate for center (convert to radians)
        A[a,b,r] +=1; //voting
     end
 end
```



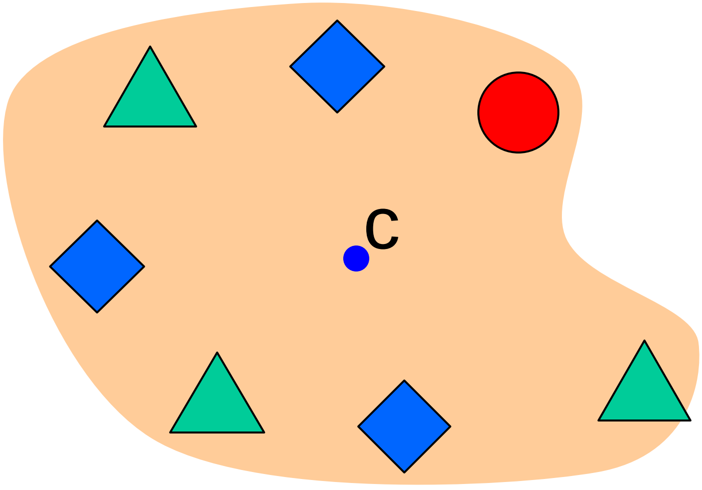


From wiki and opencv tutorial

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

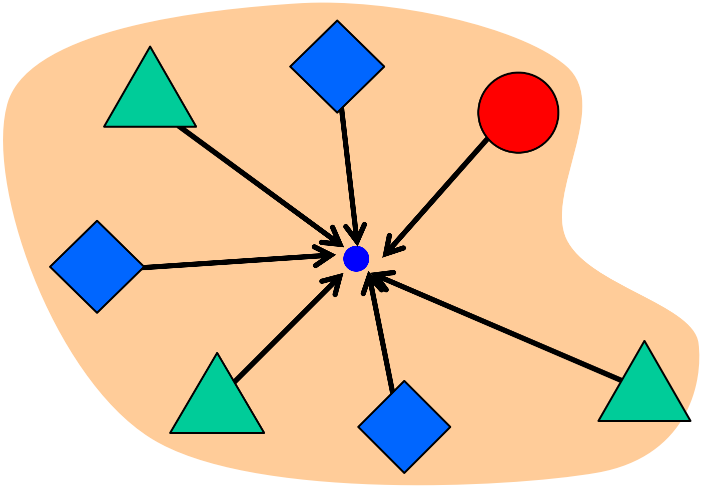
Template



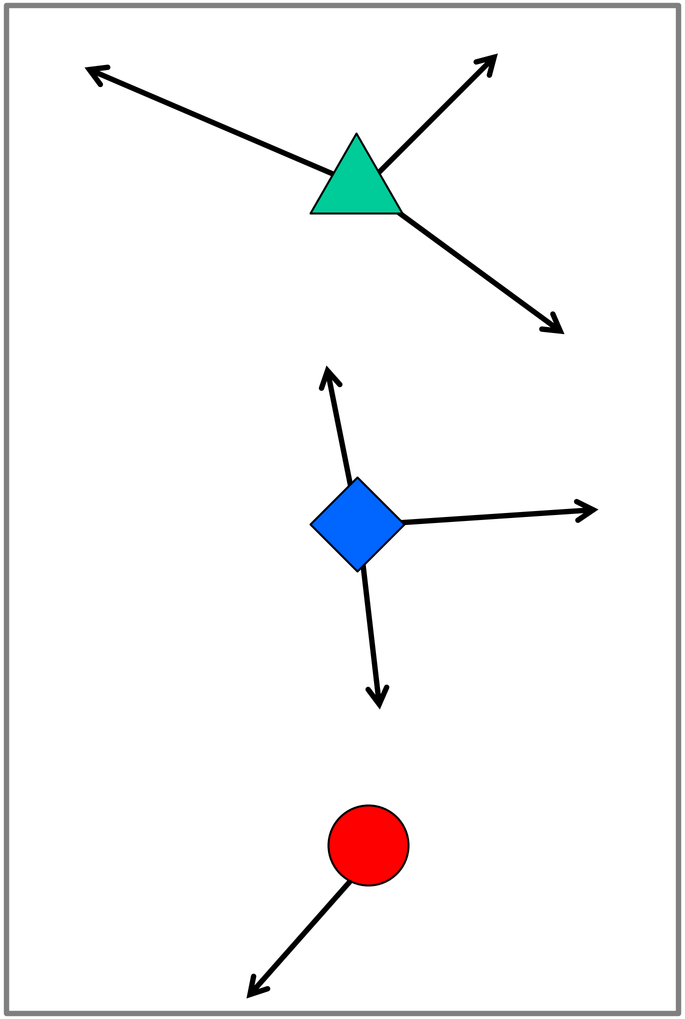
Generalized Hough transform

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

Template



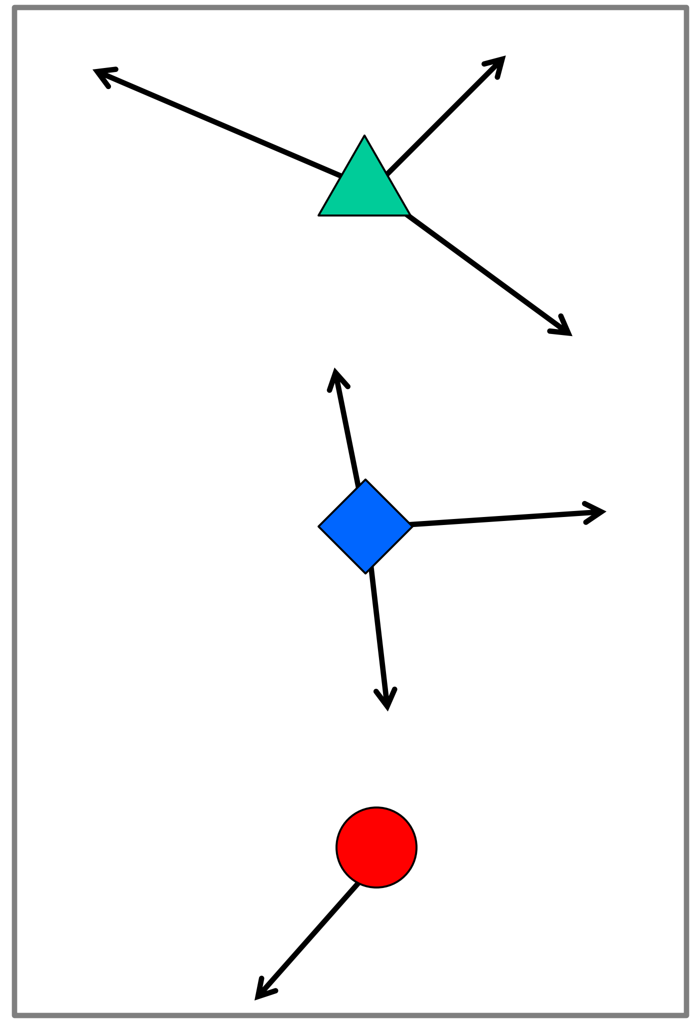
Model



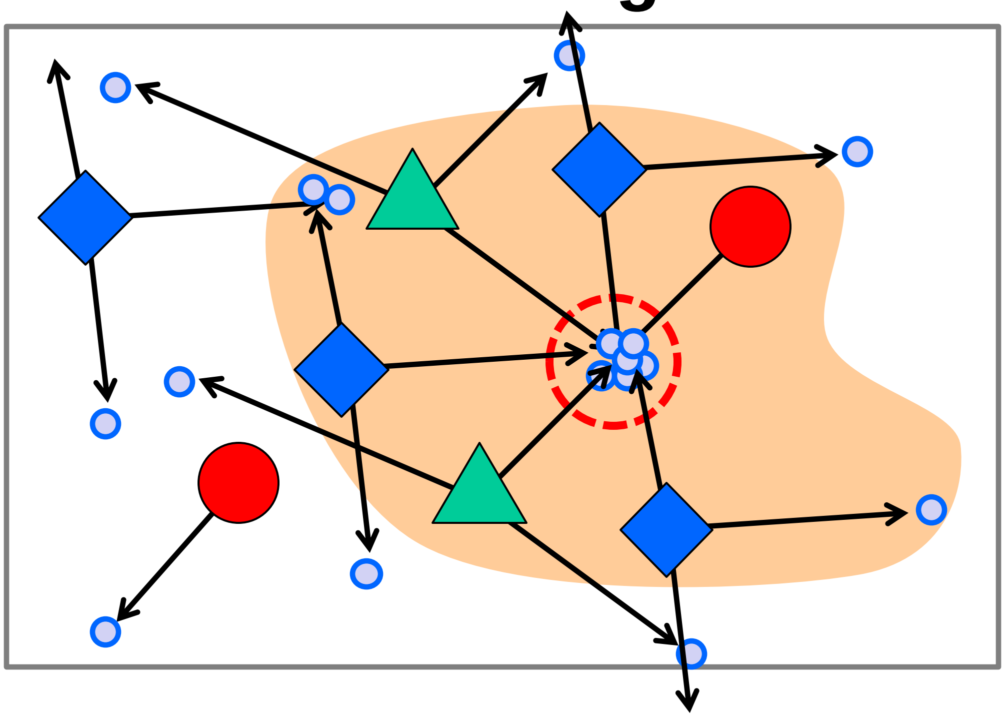
Generalized Hough transform

- 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

Model

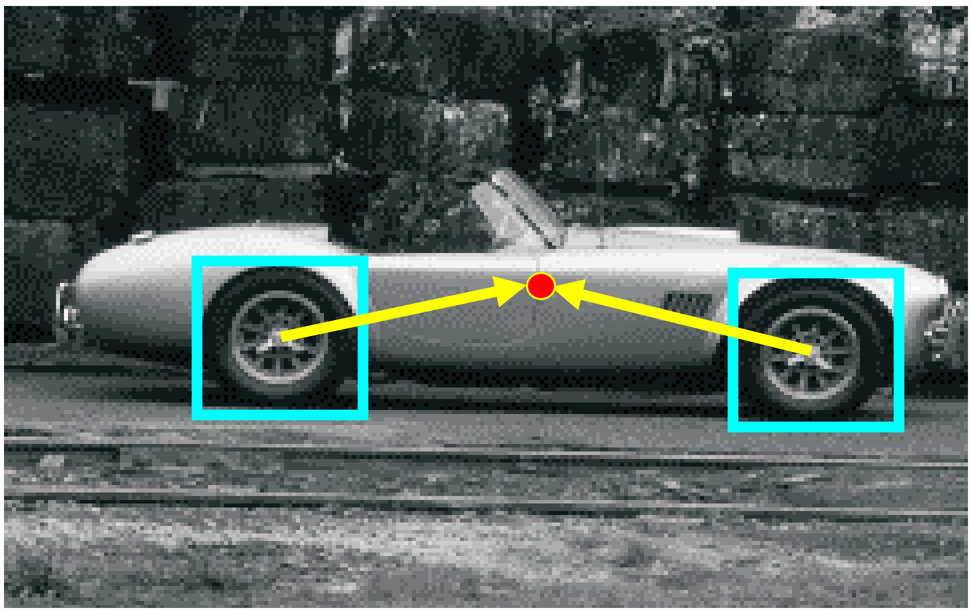


Test image

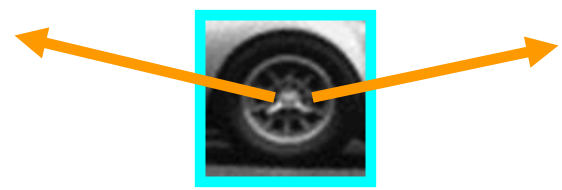


Application in recognition

- 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

K-means

A clustering algorithm (unsupervised learning method)

Group similar things together.

K-means clustering

- Want to minimize sum of squared Euclidean distances between points $\mathbf{x}_i$ and their nearest cluster centers $\mathbf{m}_k$

$$D(X,M) = \sum_{\text{cluster } k} \sum_{\substack{\text{point } i \text{ in} \\ \text{cluster } k}} (\mathbf{x}_i - \mathbf{m}_k)^2$$

Algorithm:

- Randomly initialize K cluster centers
- Iterate until convergence:
  - Assign each data point to the nearest center
  - Recompute each cluster center as the mean of all points assigned to it

K-means demo

Source: http://shabal.in/visuals/kmeans/1.html

Another demo: http://www.kovan.ceng.metu.edu.tr/~maya/kmeans/

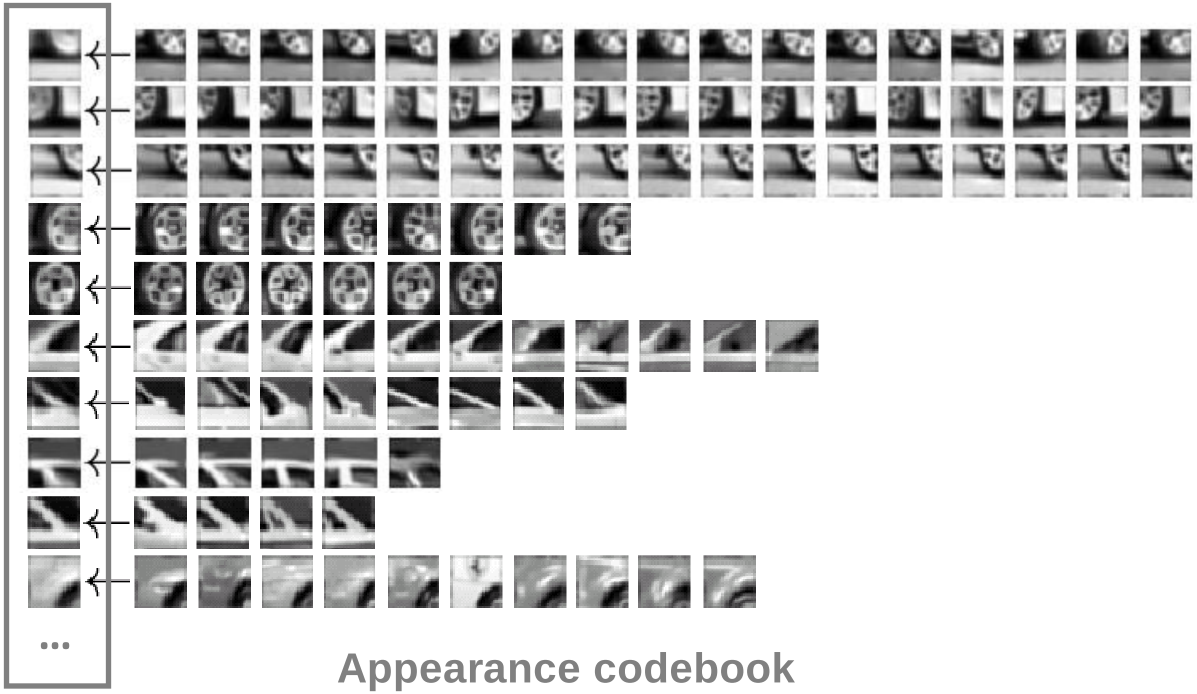
K-means demo: one initialization

Local Minimum

K-means demo: another initialization

Local Minimum

Visual codebook for generalized Hough transform

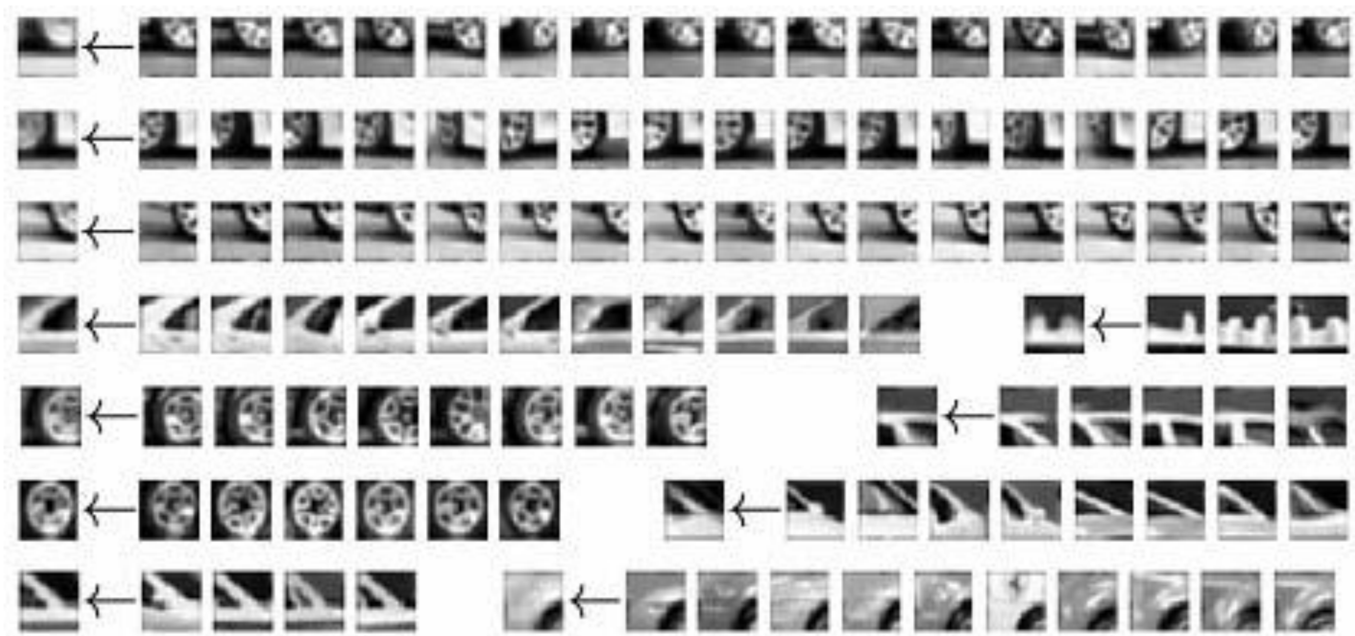


Appearance codebook

Source: B. Leibe

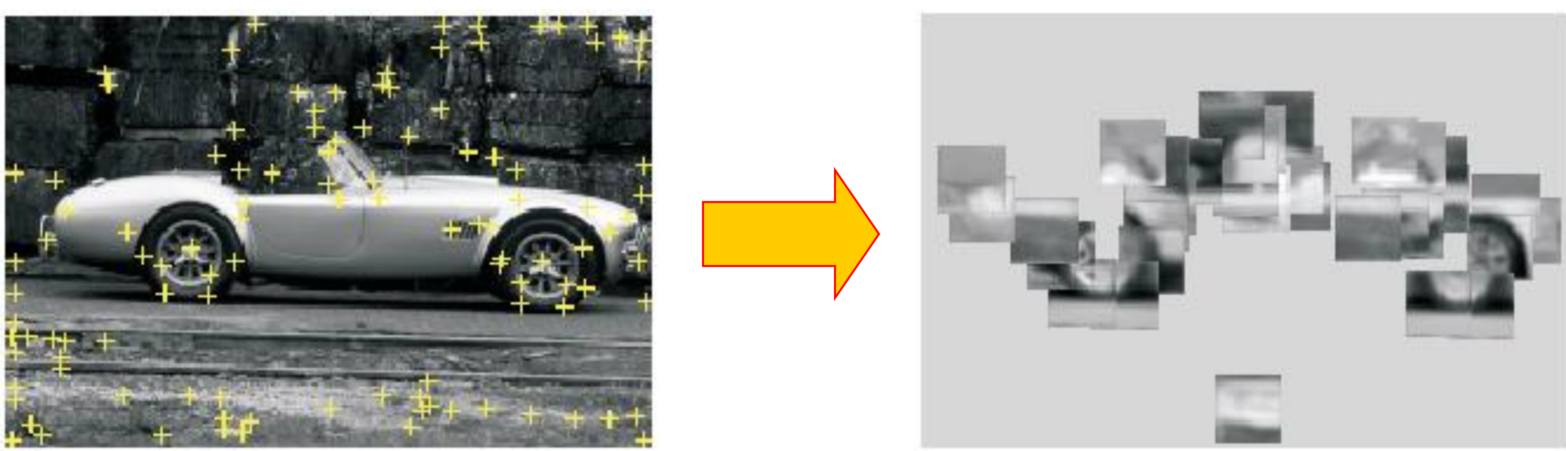
Implicit shape models: Training

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



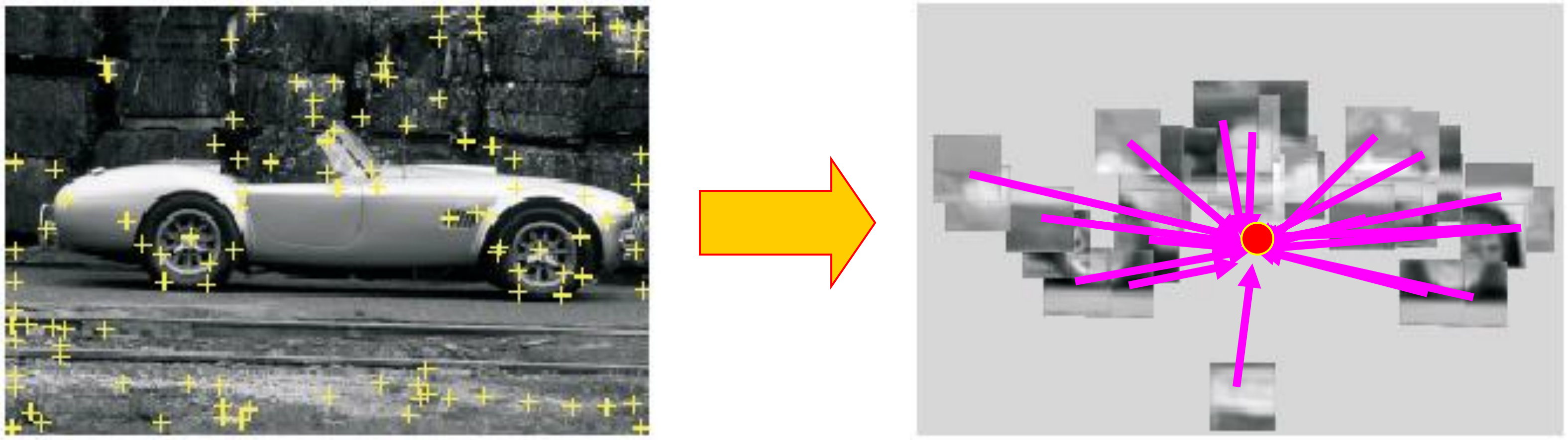
Implicit shape models: Training

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



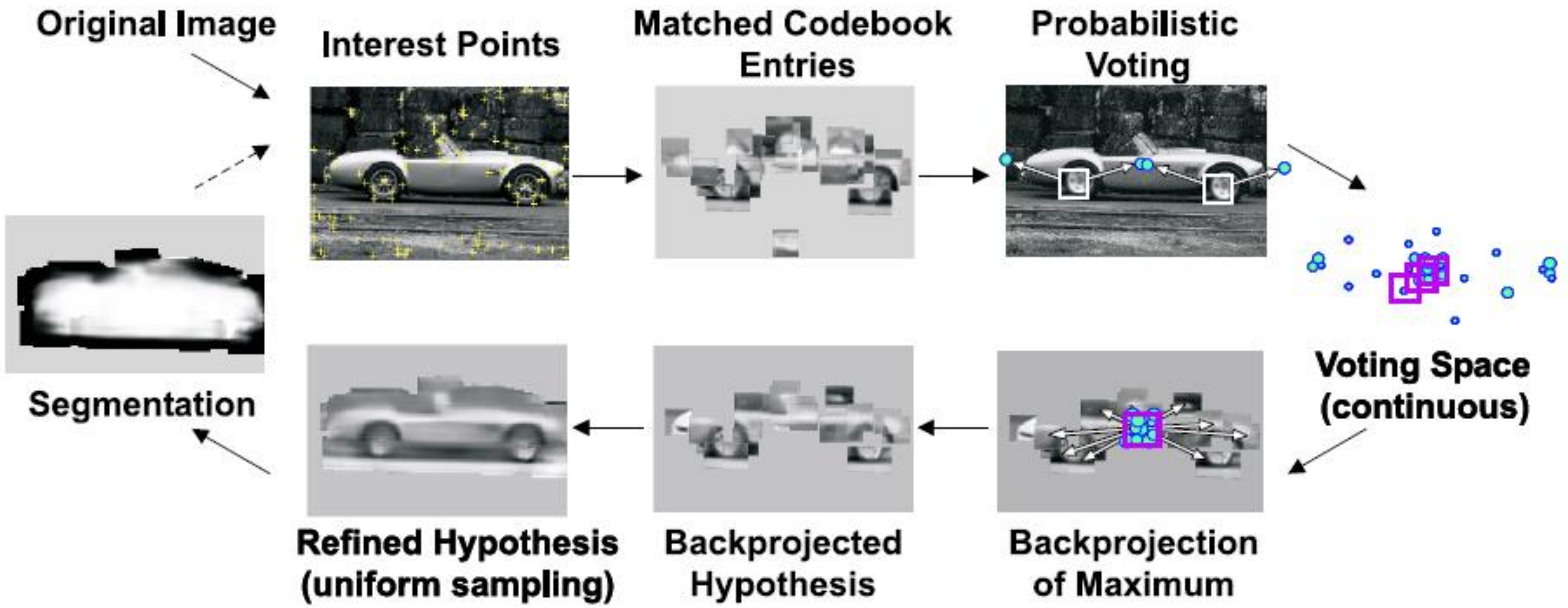
Implicit shape models: Training

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

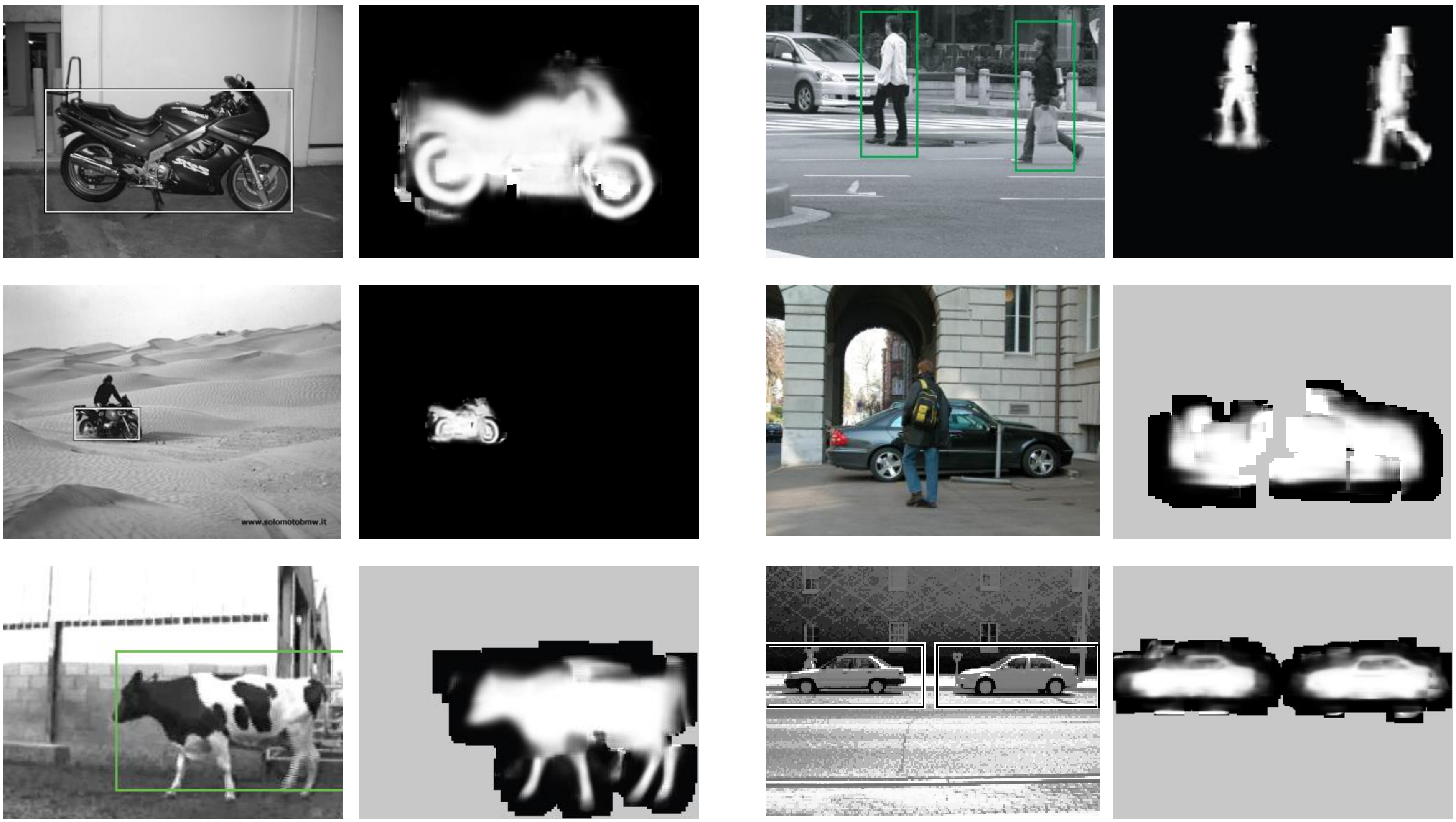


Implicit shape models: Testing

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.

Implicit shape models: Details

- Supervised training
  - Need reference location and segmentation mask for each training car
- Voting space is continuous, not discrete
  - Clustering algorithm needed to find maxima
- How about dealing with scale changes?
  - Option 1: search a range of scales, as in Hough transform for circles
  - Option 2: use interest points with characteristic scale
- Verification stage is very important
  - Once we have a location hypothesis, we can overlay a more detailed template over the image and compare pixel-by-pixel, transfer segmentation masks, etc.

Review: Hough transform

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

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