CS4501: Introduction to Computer Vision

RANSAC

Various slides from previous courses by:

D.A. Forsyth (Berkeley / UIUC), I. Kokkinos (Ecole Centrale / UCL). S. Lazebnik (UNC / UIUC), S. Seitz (MSR / Facebook), J. Hays (Brown / Georgia Tech), A. Berg (Stony Brook / UNC), D. Samaras (Stony Brook) . J. M. Frahm (UNC), V. Ordonez (UVA).

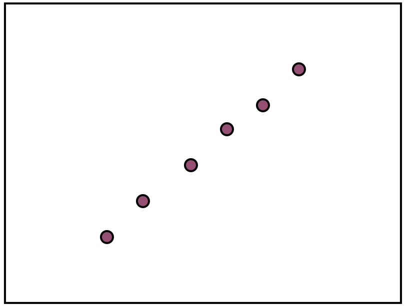
Last Class

- Interest Points (DoG extrema operator)
- SIFT Feature descriptor
- Feature matching

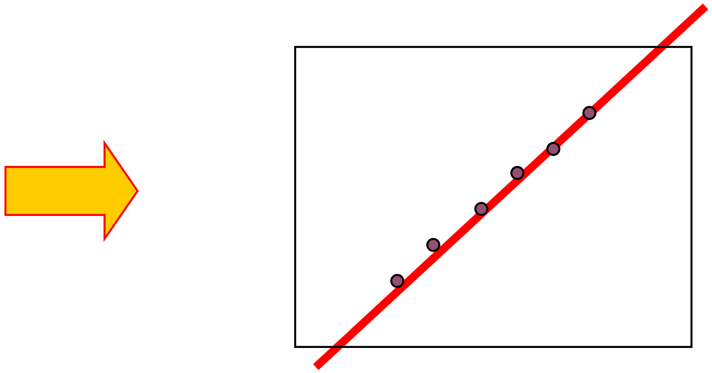
Today’s Class

- Line Detection using the Hough Transform
- Least Squares / Hough Transform / RANSAC

Line Detection



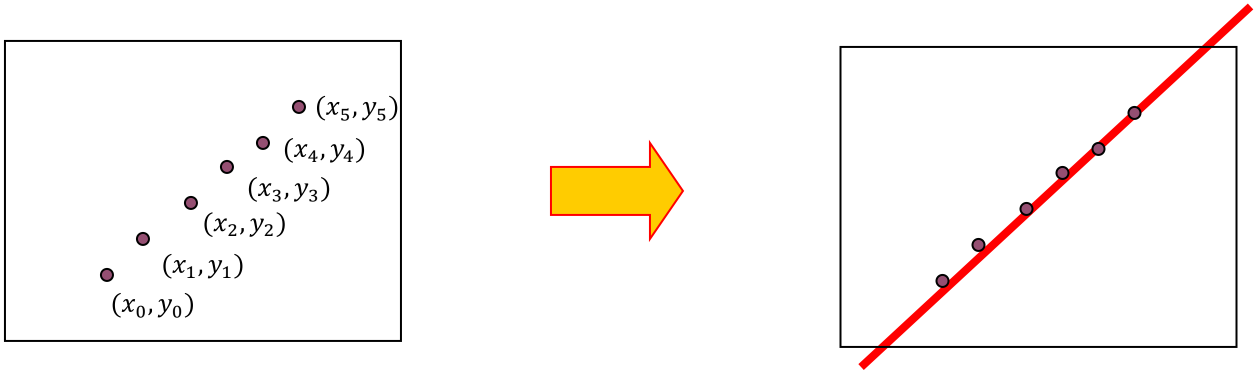
Pixels in input image



$$y = \beta_1 x + \beta_0$$

- Have you encountered this problem before?

Line Detection – Least Squares Regression



Pixels in input image

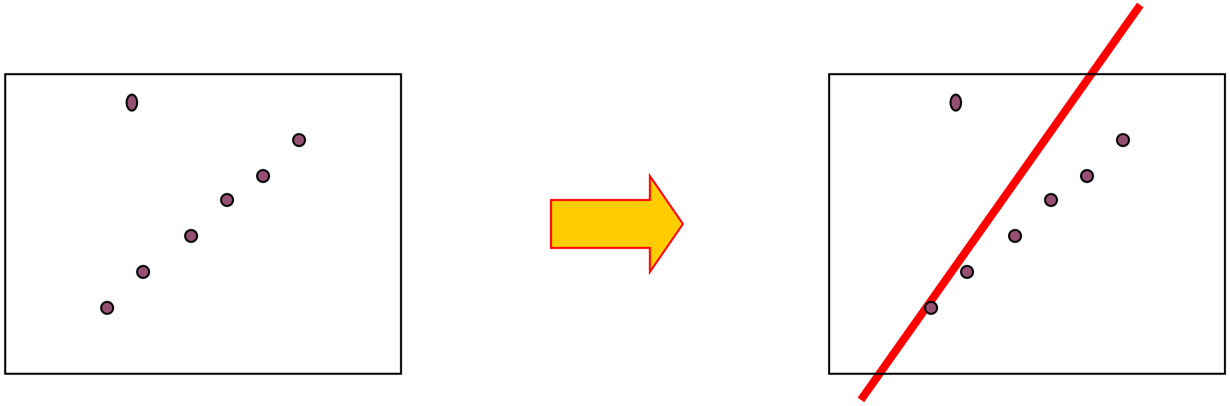
$$y = \beta_1 x + \beta_0$$

- Have you encountered this problem before?

Find betas that minimize: $$\sum_i (y_i - \beta_1 x_i - \beta_0)^2 = \|\boldsymbol{y} - \boldsymbol{X\beta}\|^2$$

Solution: $$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

However Least Squares is not Ideal under Outliers



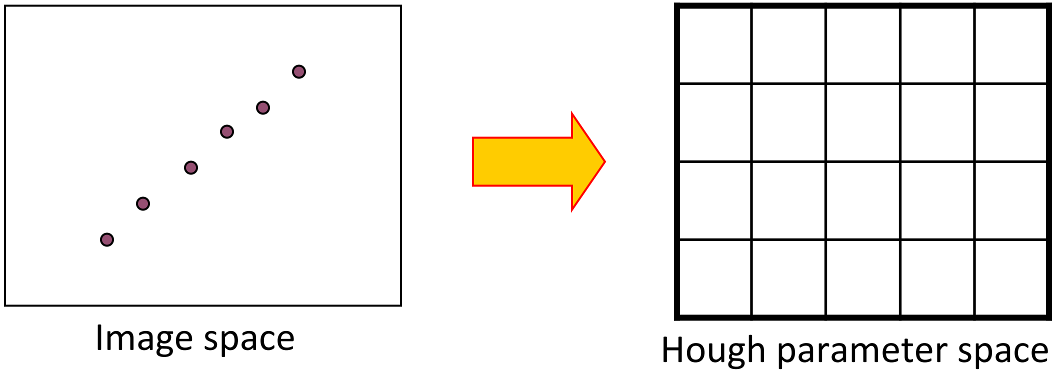
Pixels in input image

Solution: 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

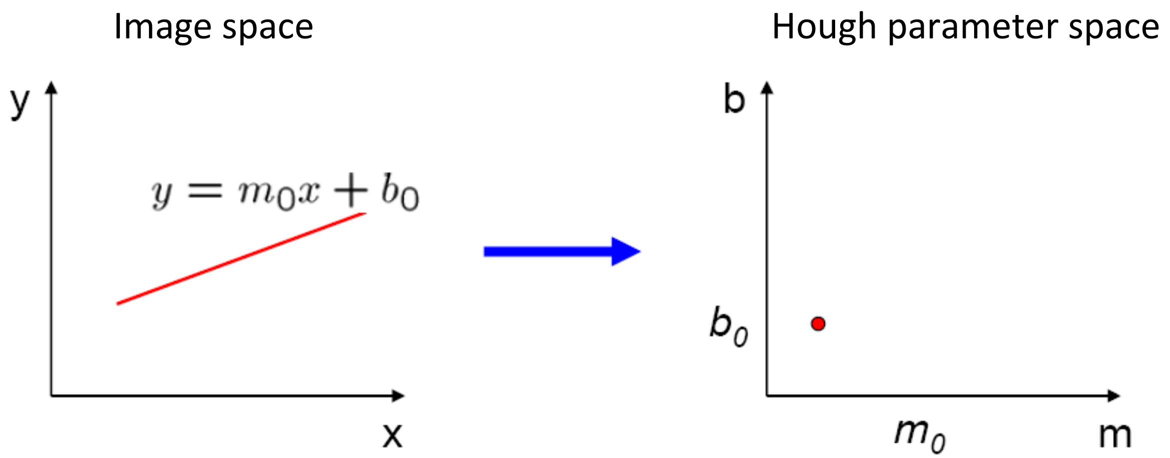
- An early type of voting scheme
- General outline:
  - Discretize *parameter space* into bins
  - For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  - Find bins that have the most votes



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

Parameter space representation

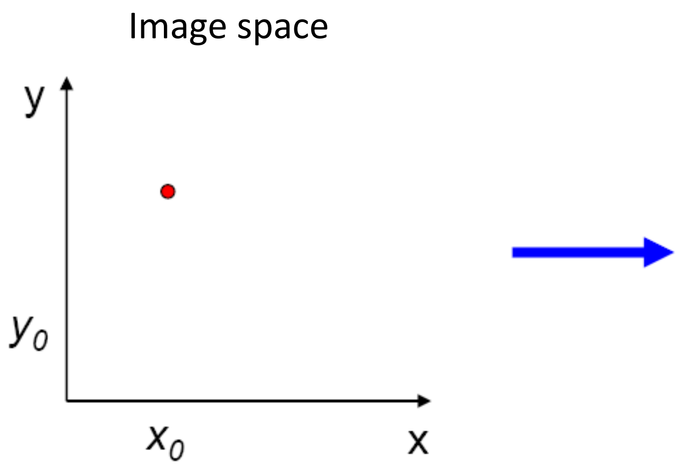
- A line in the image corresponds to a point in Hough space

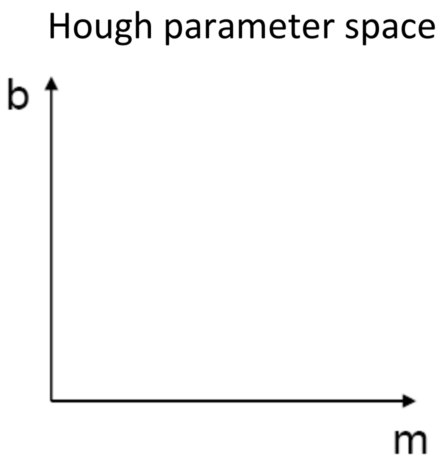


Source: S. Seitz

Parameter space representation

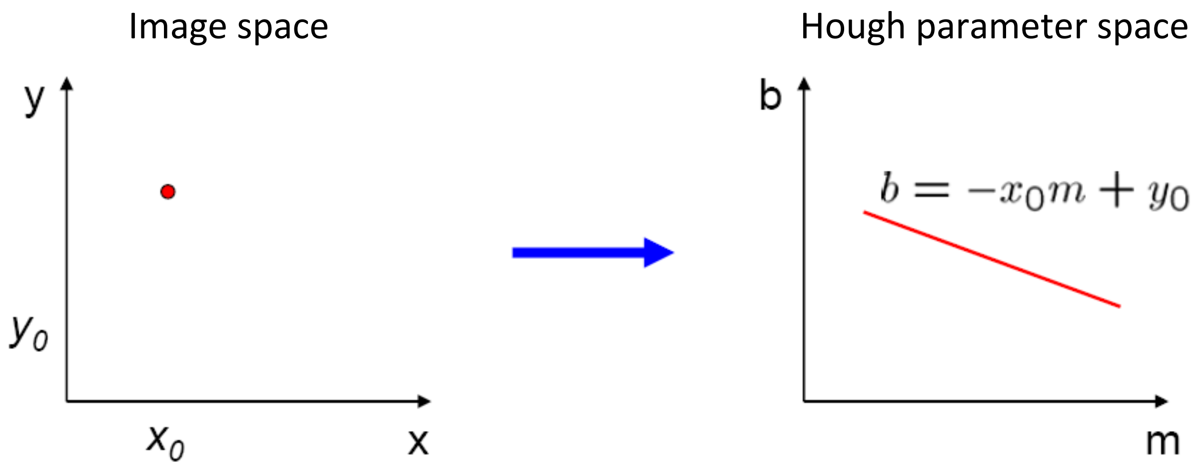
- What does a point (x_0, y_0) in the image space map to in the Hough space?





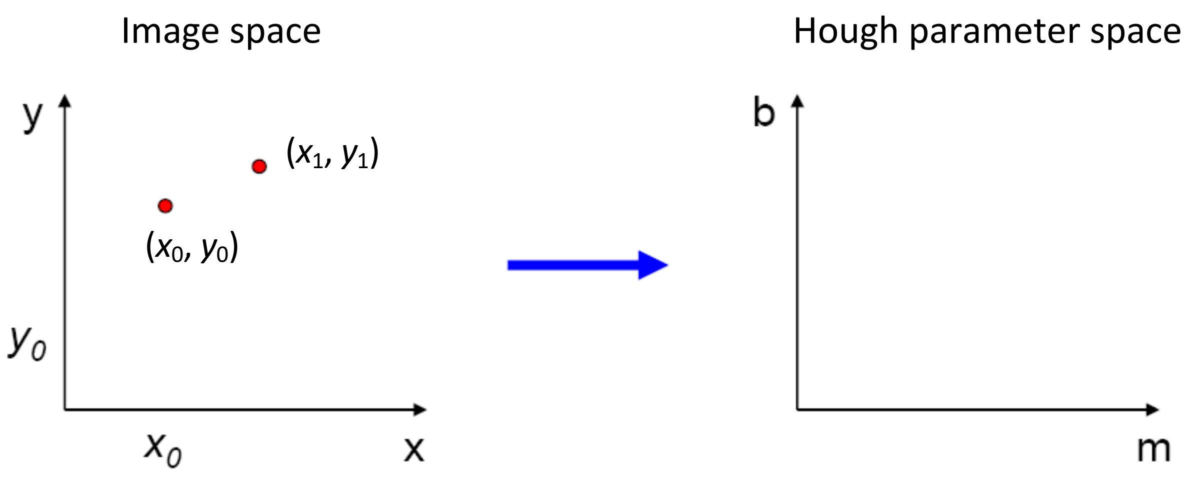
Parameter space representation

- 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)?

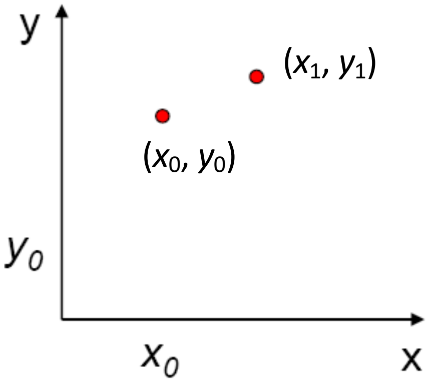


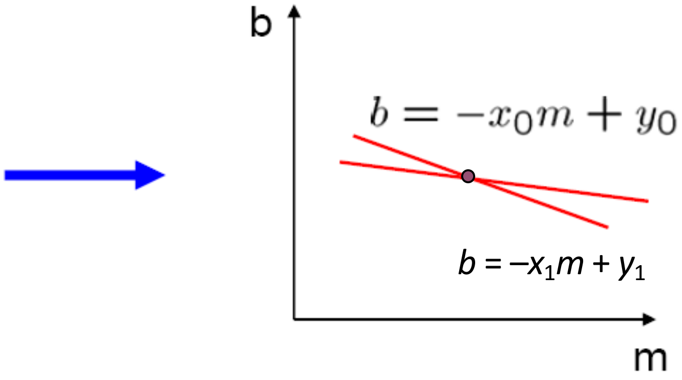
Parameter space representation

- 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 $b = –x_1m + y_1$

Image space

Hough parameter space



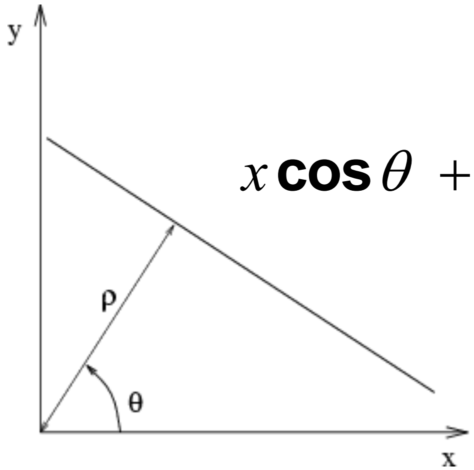


Parameter space representation

- 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*



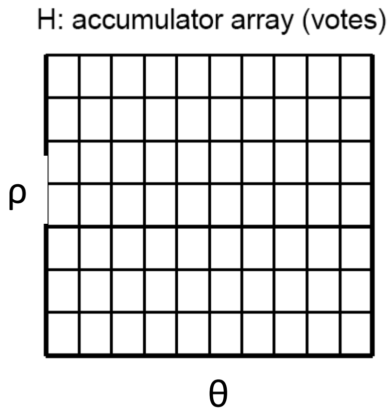
$$x\cos\theta + y\sin\theta = \rho$$

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

Algorithm outline

- Initialize accumulator H to all zeros

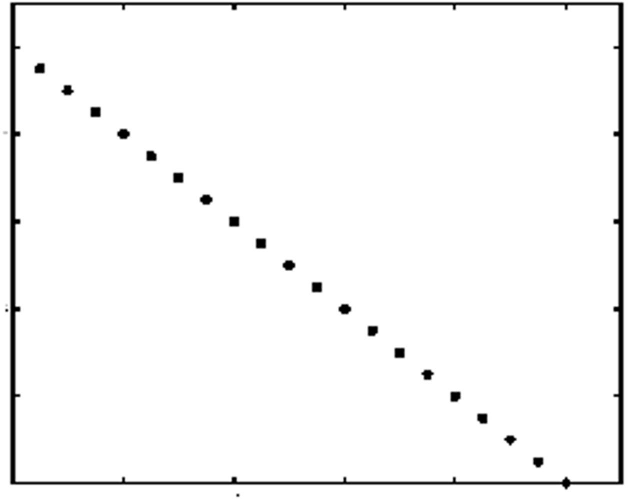
- For each feature point (x,y)
  in the image
  For θ = 0 to 180
  $\rho = x \cos \theta + y \sin \theta$
  $H(\theta, \rho) = H(\theta, \rho) + 1$
  end
  end



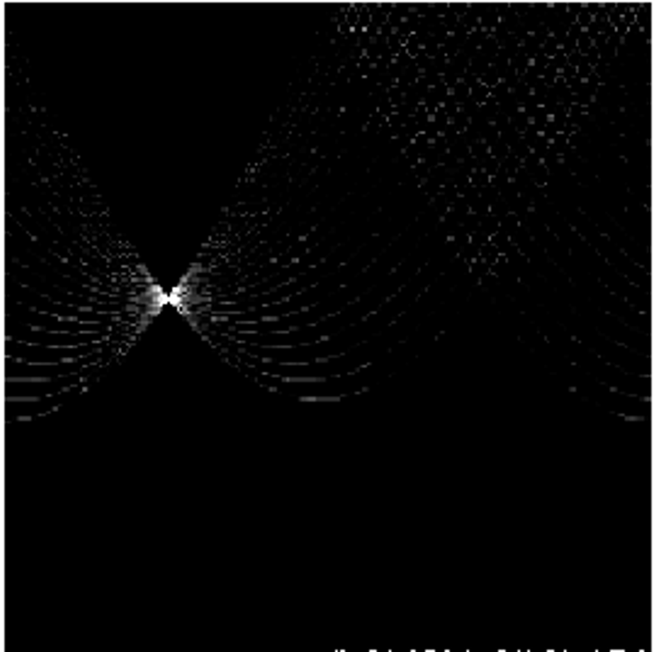
- Find the value(s) of (θ, ρ) where $H(\theta, \rho)$ is a local maximum
  - The detected line in the image is given by
    $\rho = x \cos \theta + y \sin \theta$

Slide by Svetlana Lazebnik

Basic illustration



features



votes

Hough Transform for an Actual Image

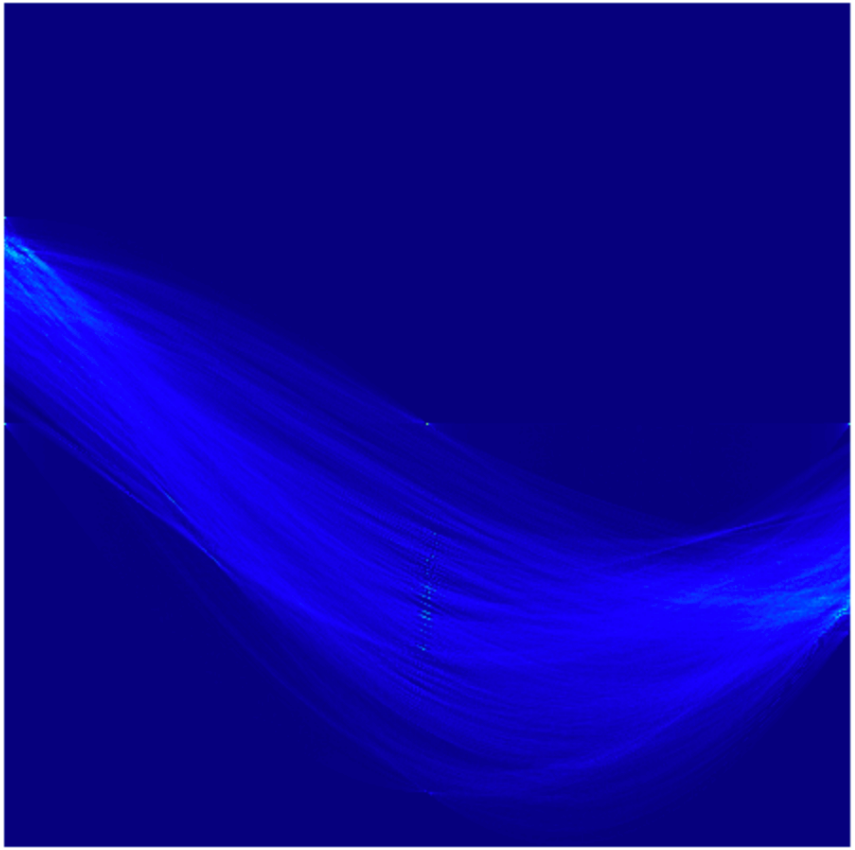
Edges using threshold on Sobel's magnitude

Hough Transform (High Resolution)

$\rho = -\sqrt{h^2 + w^2}$

$\rho = 0$

$\rho = \sqrt{h^2 + w^2}$



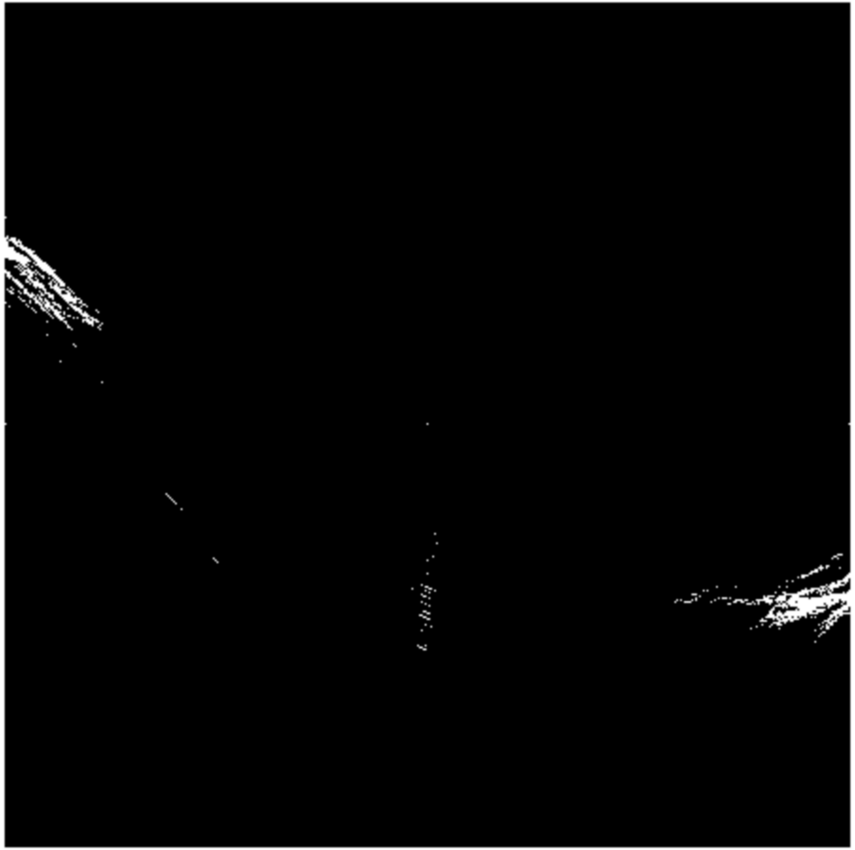
$\theta = -90^0$ $\theta = 0$ $\theta = 90^0$

Hough Transform (After threshold)

$\rho = -\sqrt{h^2 + w^2}$

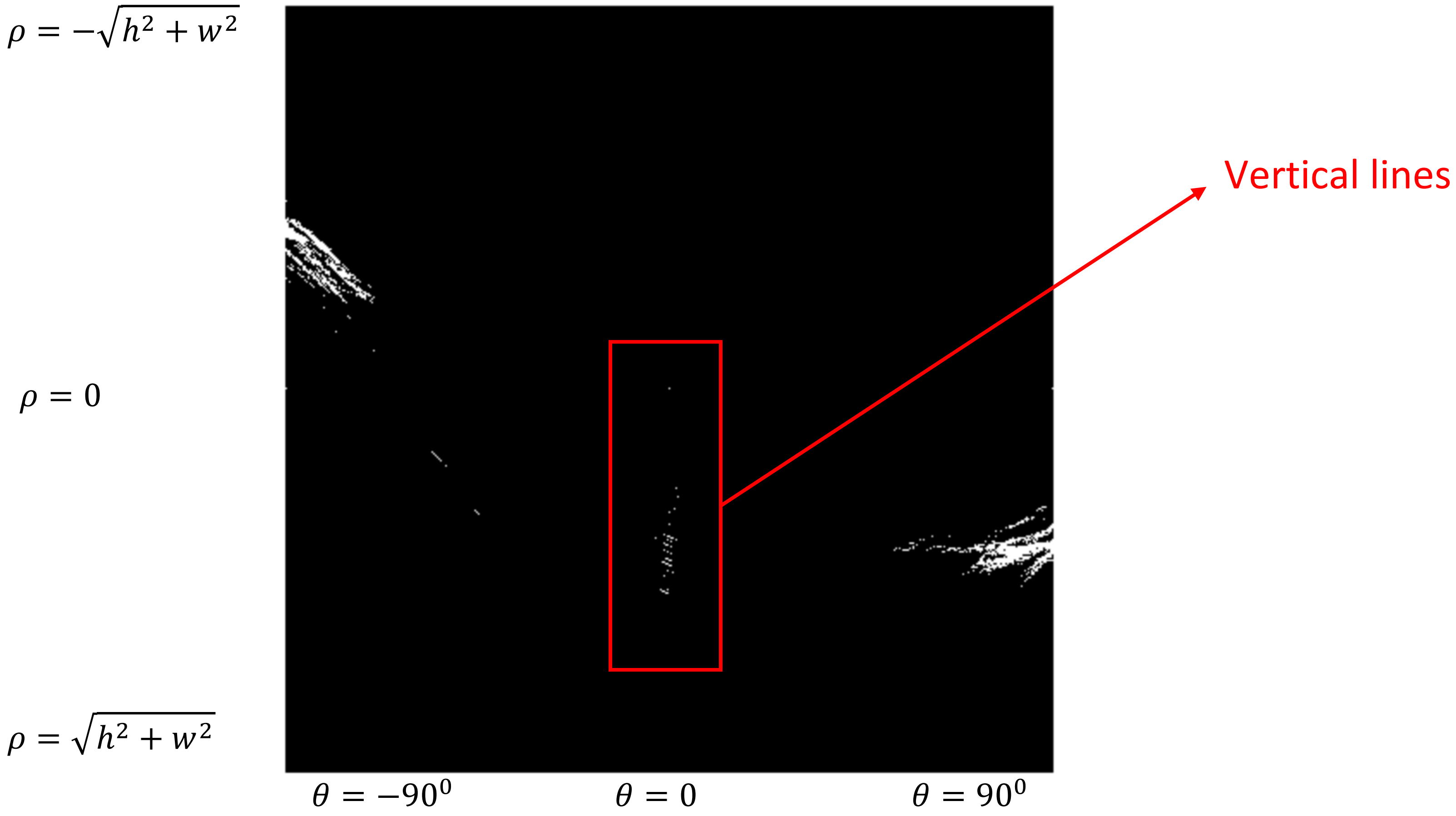
$\rho = 0$

$\rho = \sqrt{h^2 + w^2}$

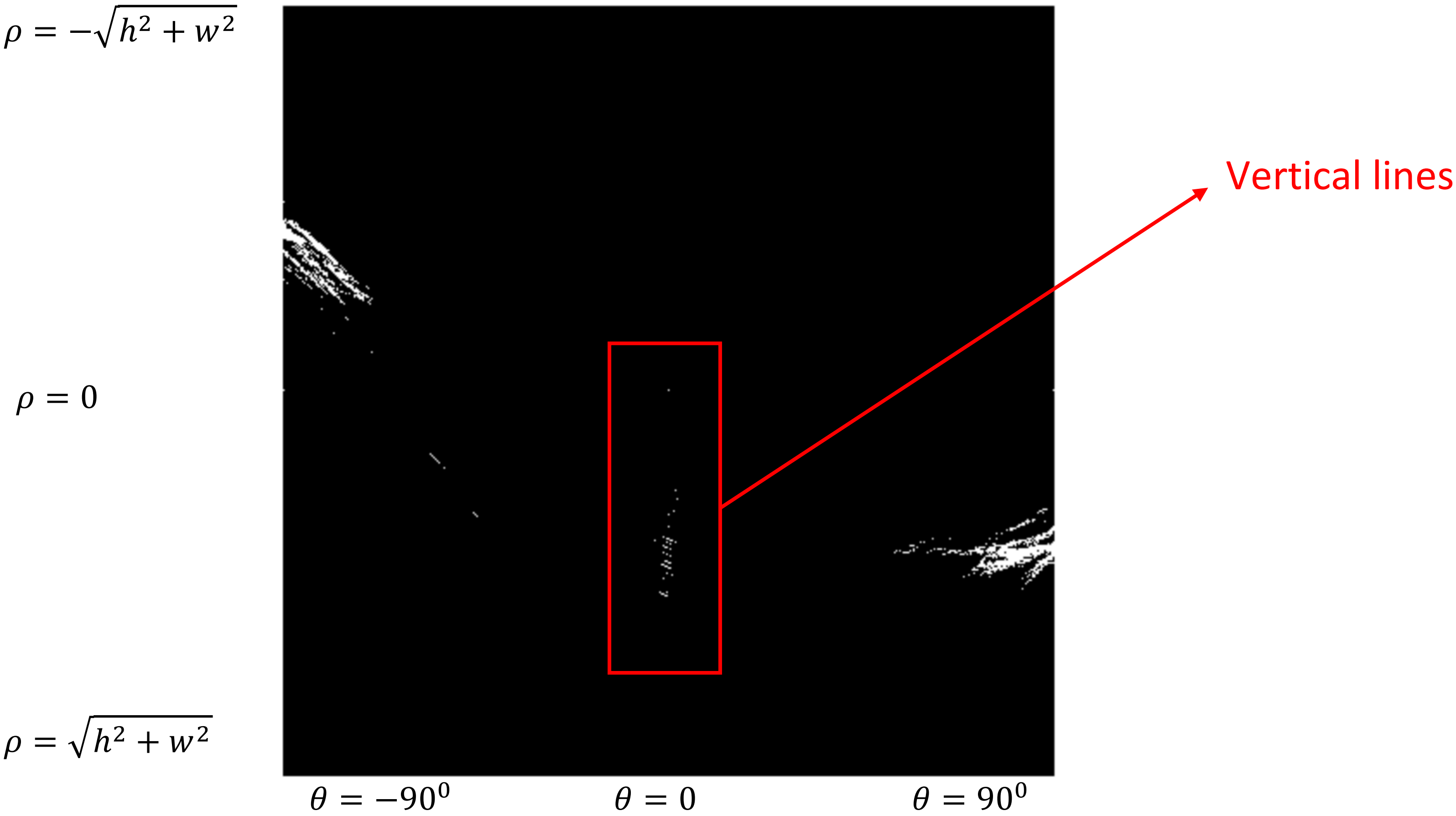


$\theta = -90^0$ $\theta = 0$ $\theta = 90^0$

Hough Transform (After threshold)



Hough Transform (After threshold)

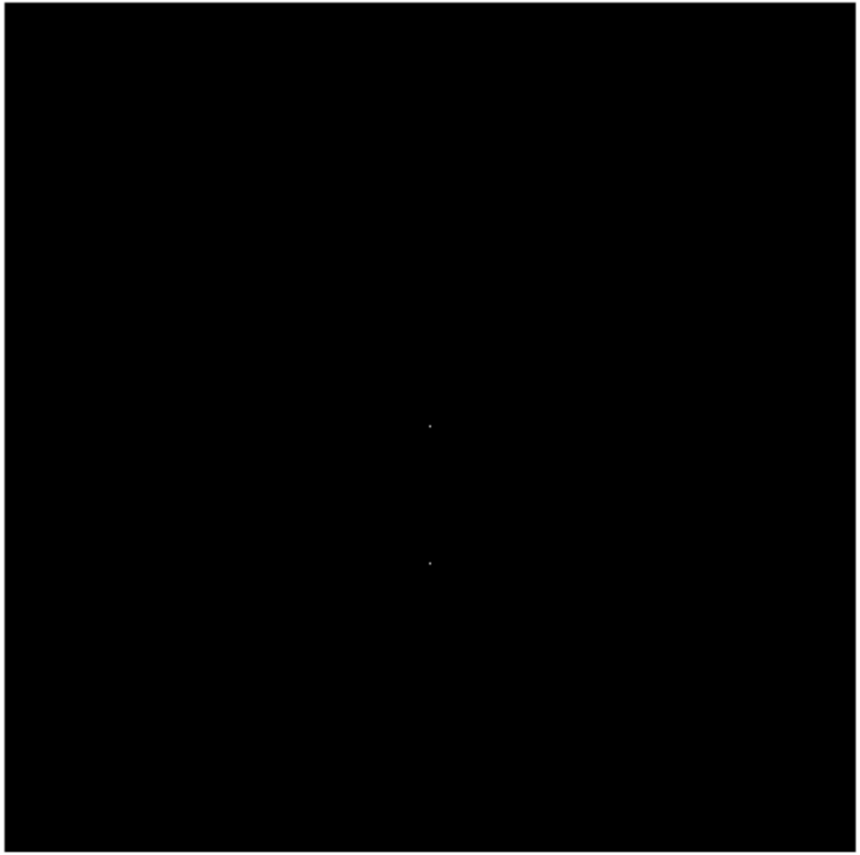


Hough Transform with Non-max Suppression

$\rho = -\sqrt{h^2 + w^2}$

$\rho = 0$

$\rho = \sqrt{h^2 + w^2}$

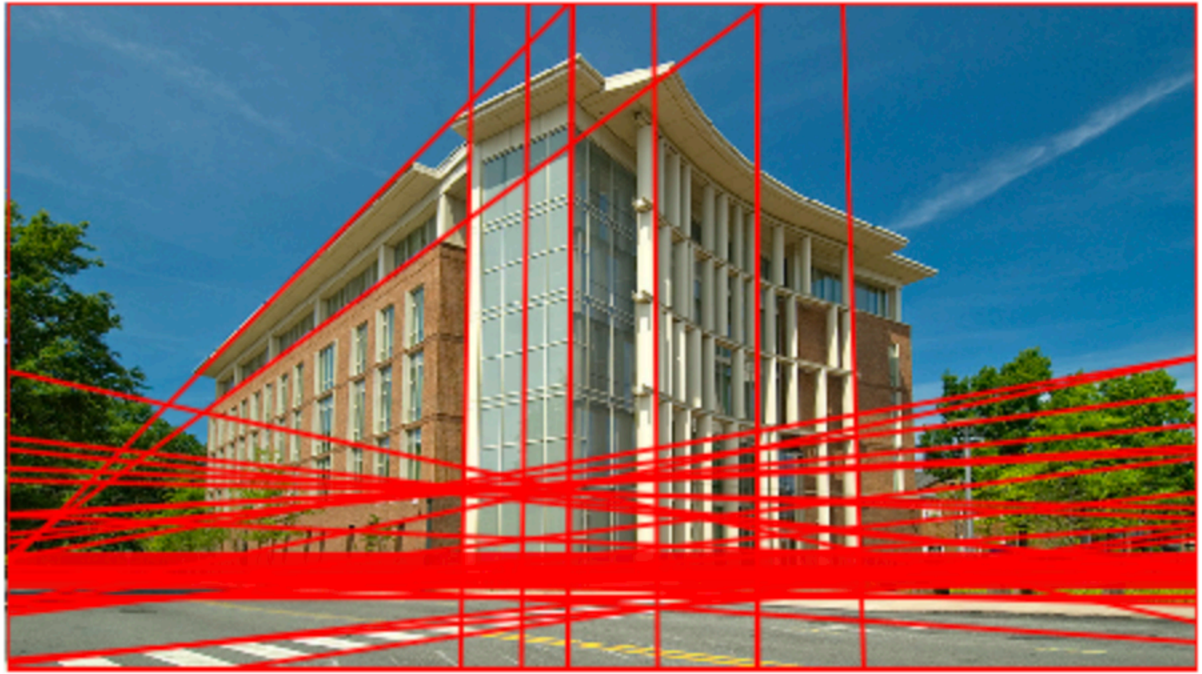


$\theta = -90^0$ $\theta = 0$ $\theta = 90^0$

Back to Image Space – with lines detected

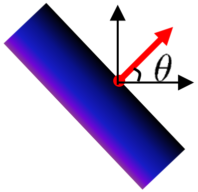
$$y = -\frac{cos\theta}{sin\theta}x + \frac{\rho}{sin\theta}$$

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



Hough transform demo

Incorporating image gradients



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

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

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

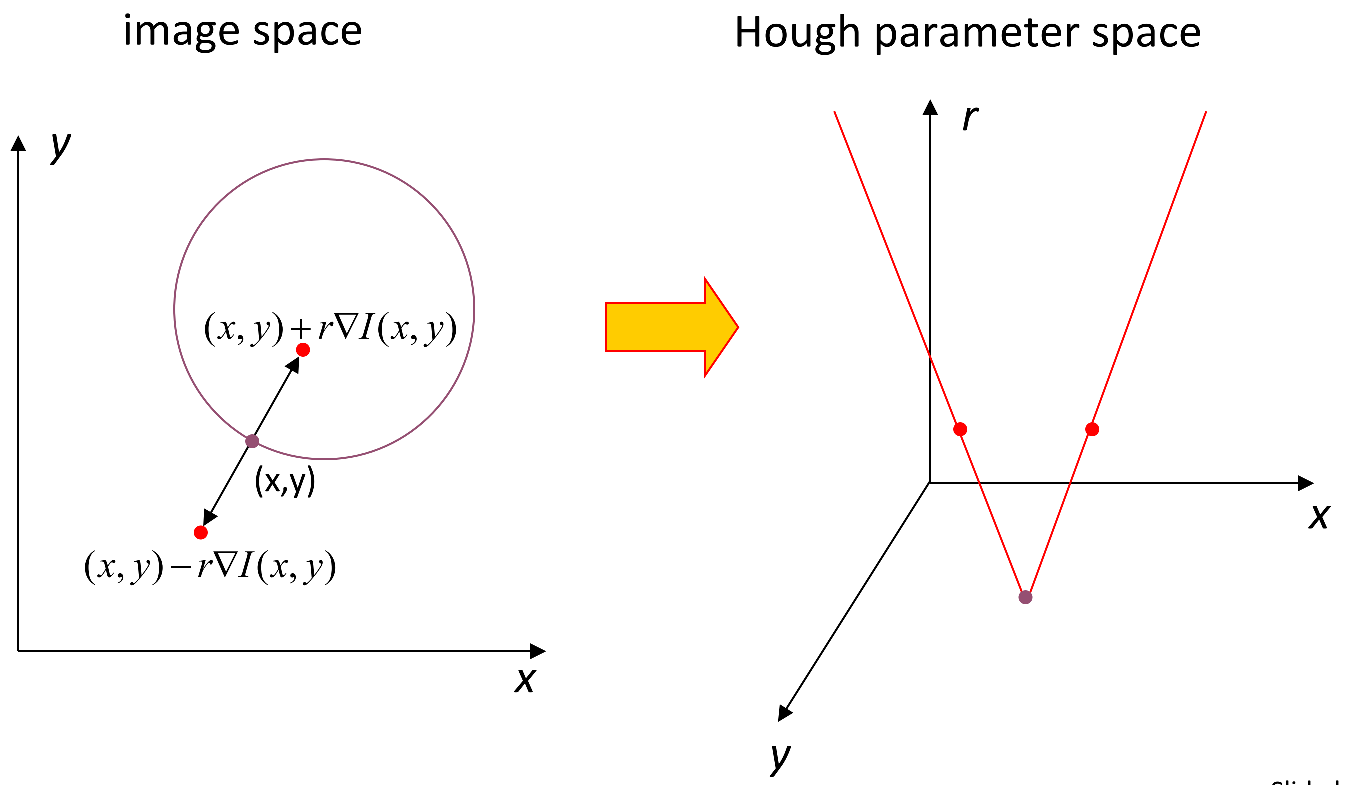
- Modified Hough transform:

- 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

Slide by Svetlana Lazebnik

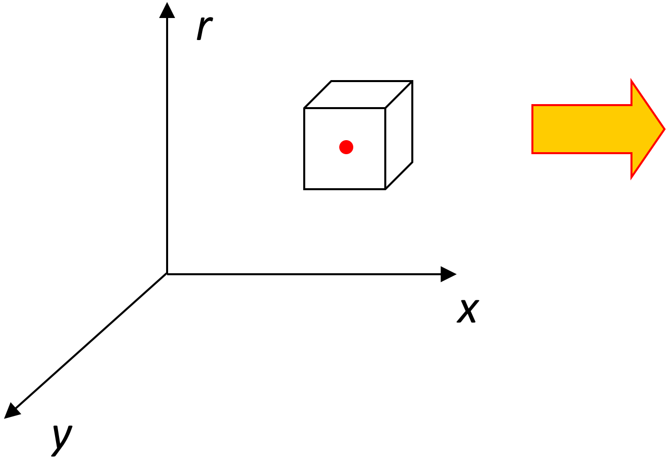
Hough transform for circles

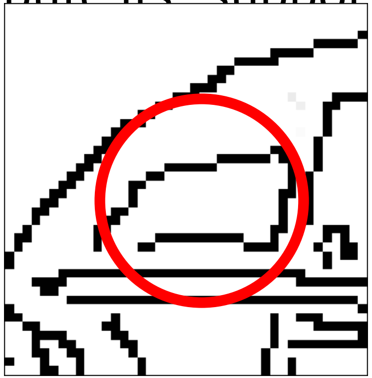


Slide by Svetlana Lazebnik

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?

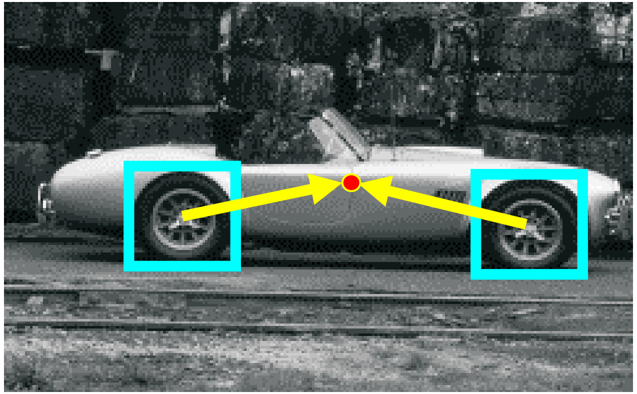
Slide by Svetlana Lazebnik

RANSAC – Random Sample Consensus

- Another Voting Scheme
- Idea: Maybe you do not need to have all samples have a vote.
  - Only a random subset of samples (points) vote.

Generalized Hough Transform

- You can make voting work for any type of shape / geometrical configuration. Even irregular ones.



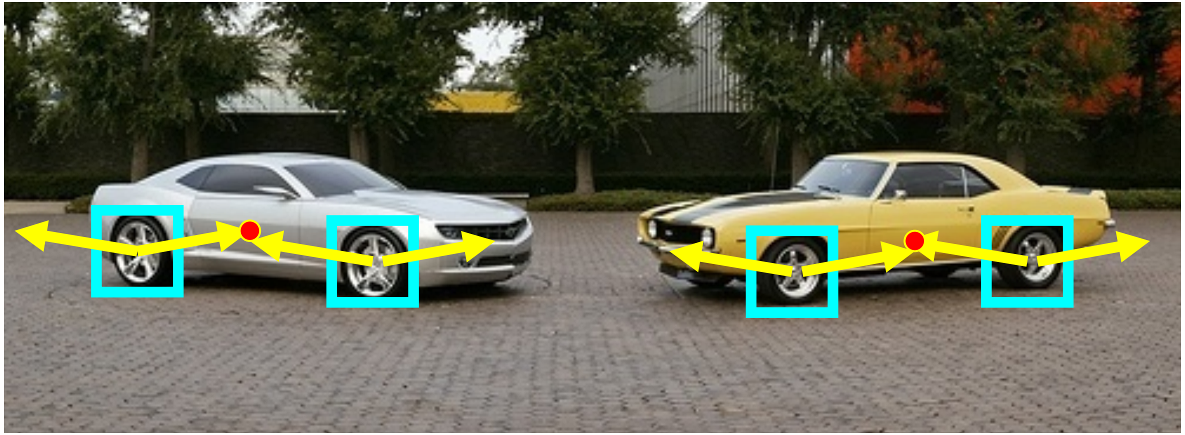
training image

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

Generalized Hough Transform

- You can make voting work for any type of shape / geometrical configuration. Even irregular ones.



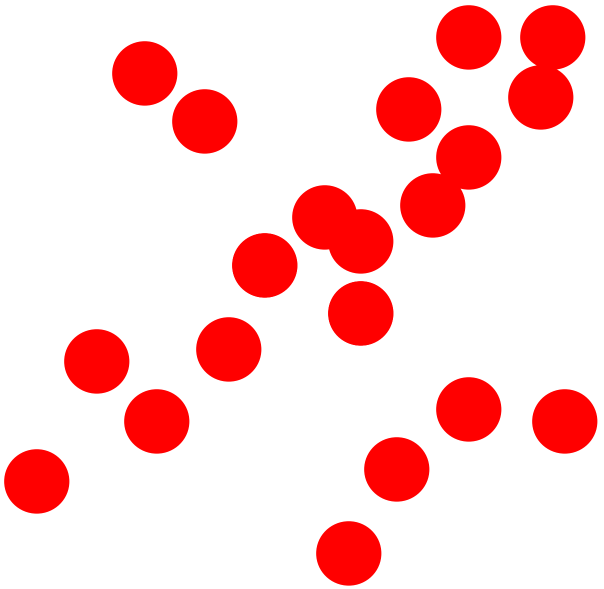
test image

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

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.



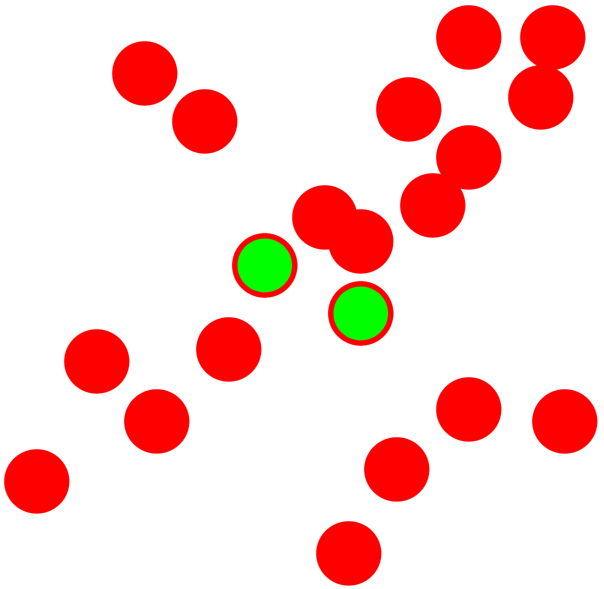
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example



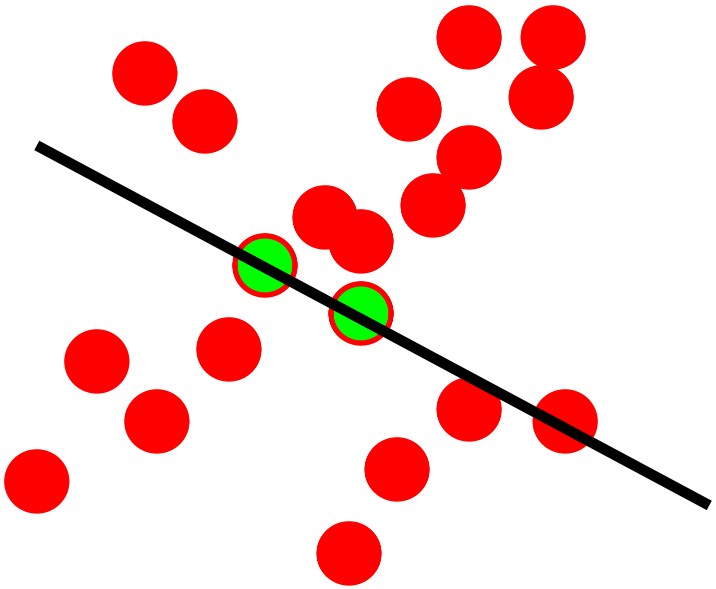
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Illustration by Savarese

RANSAC



Line fitting example

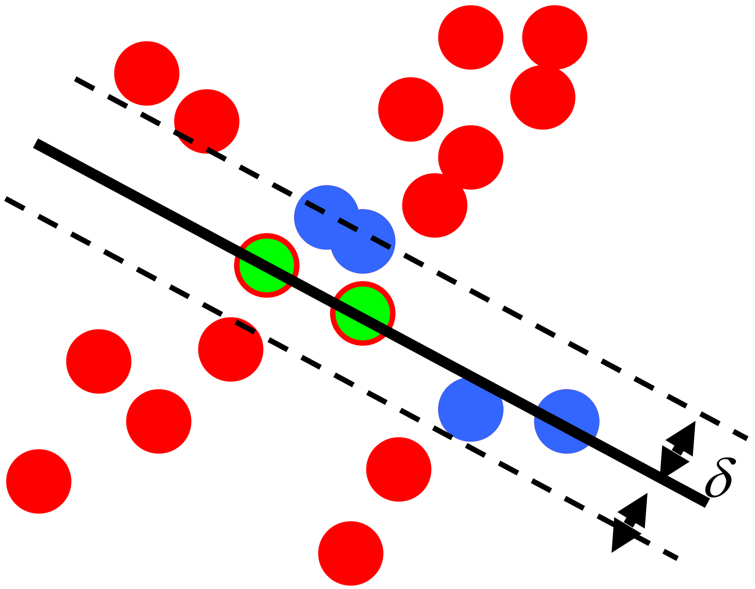
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example



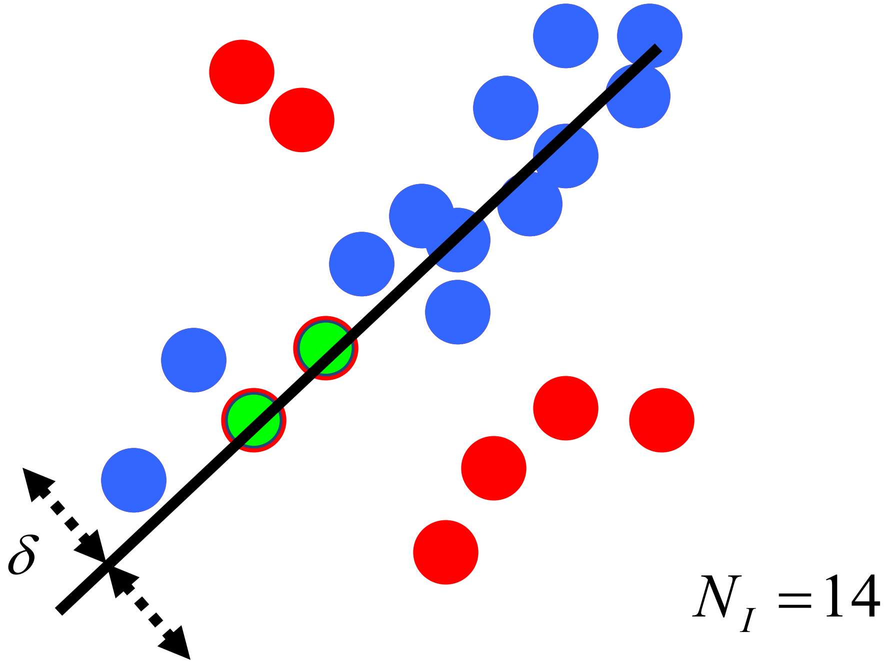
$$N_I = 6$$

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC



Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (#=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

How to choose parameters?

- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)
- Number of sampled points *s*
  - Minimum number needed to fit the model
- Distance threshold δ
  - Choose δ so that a good point with noise is likely (e.g., prob=0.95) within threshold
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$

$$\boxed{N = \log(1-p)/\log\left(1-(1-e)^s\right)}$$

	proportion of outliers e						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

For p = 0.99

modified from M. Pollefeys

RANSAC conclusions

Good

- Robust to outliers
- Applicable for larger number of model parameters than Hough transform
- Optimization parameters are easier to choose than Hough transform

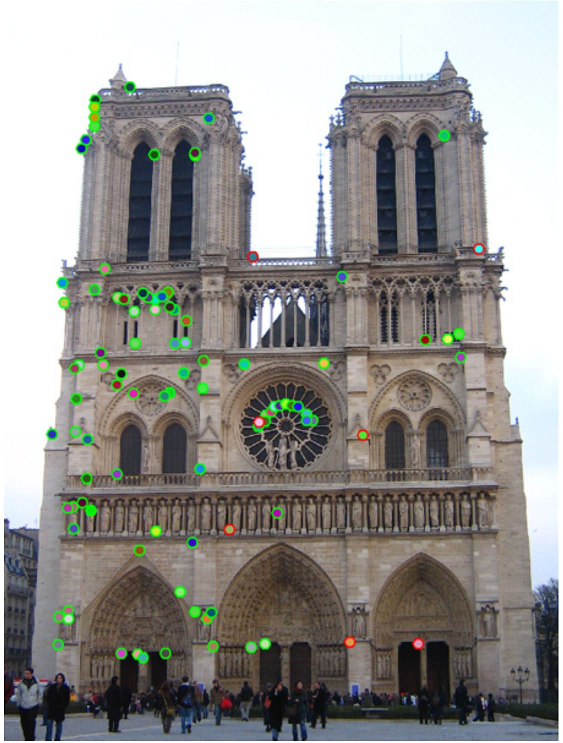
Bad

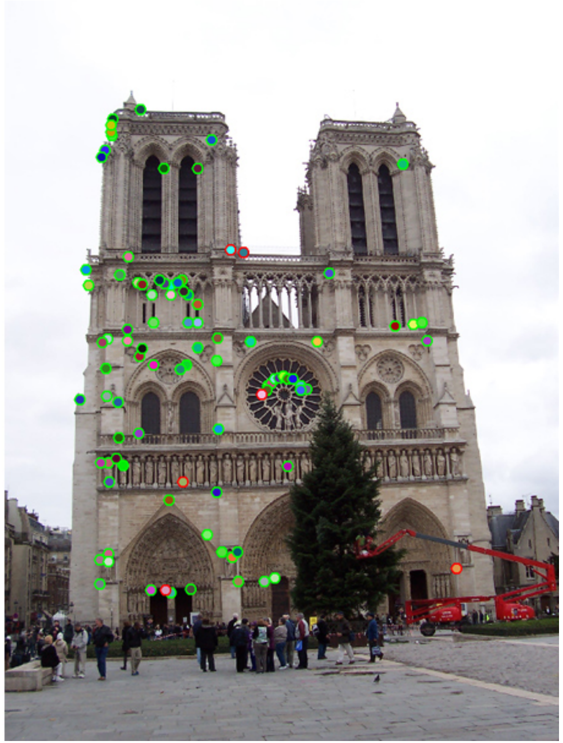
- Computational time grows quickly with fraction of outliers and number of parameters
- Not good for getting multiple fits

Common applications

- Computing a homography (e.g., image stitching)
- Estimating fundamental matrix (relating two views)

How do we fit the best alignment?





How many points do you need?

Questions?