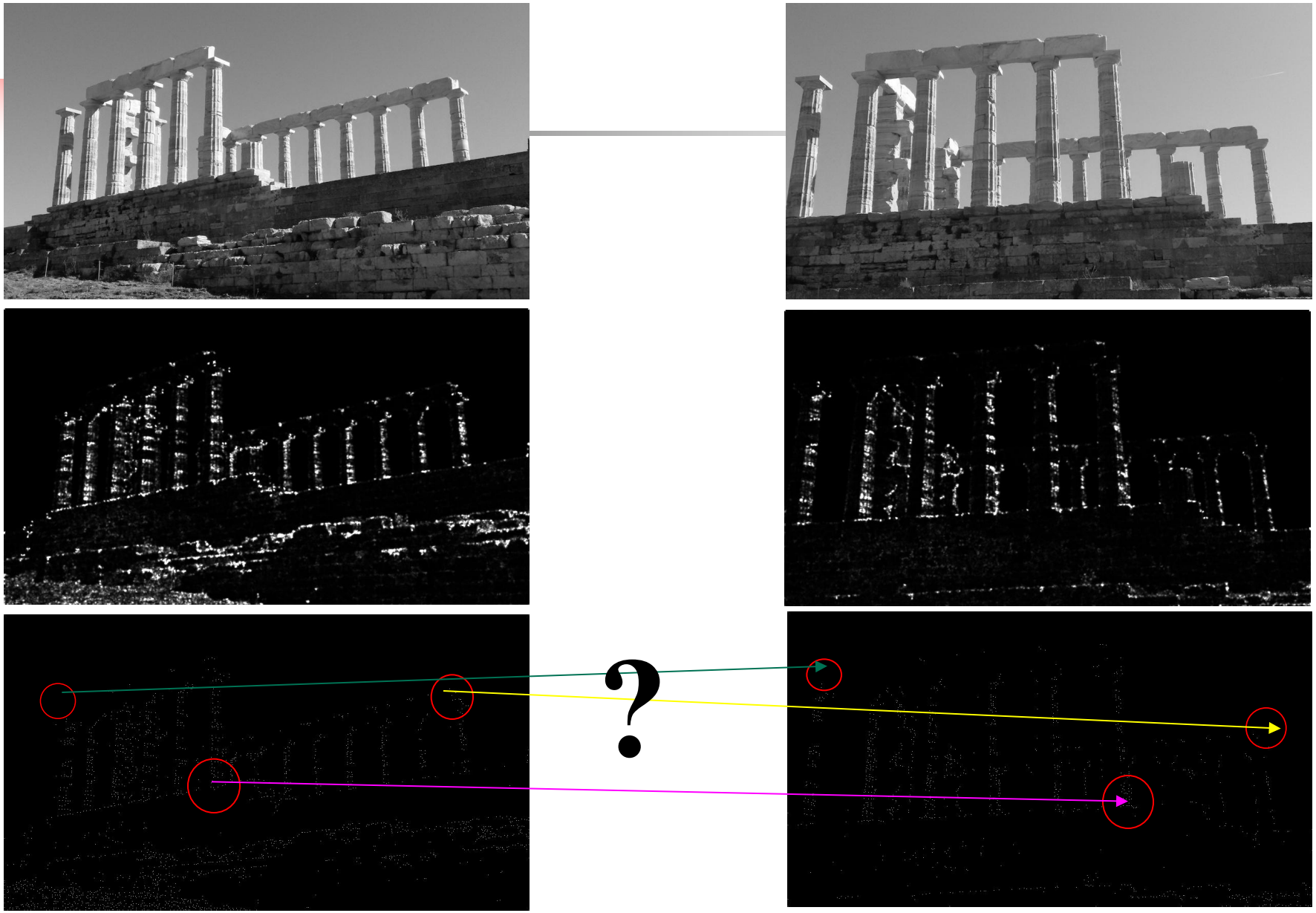
Feature matching and model fitting (Week-07 Lectures 29-33)

Jayanta Mukhopadhyay
Dept. of Computer Science and Engg.

Can you get the 3-D structure of the scene?

- Get a set of pairs of corresponding points.
- Compute fundamental matrix.
- Derive camera matrices.
- Solve for 3-D coordinates of scene points for each pair of corresponding points.

Feature matching



Matching with Features

- Detect feature points in both images.
- Describe them by local statistics.
- Find corresponding pairs (Matching).

Slide adapted from Darya Frolova, Denis Simakov, Weizmann Institute.

Matching key points

- Representation of a key-point by a feature vector.
  - e.g. $[f_0 f_1 \ldots f_n]^T$
- Use distance functions / similarity measures.
  - L_1 norm $$L_1(\vec{f},\vec{g}) = \sum_{i=0}^{n} |f_i - g_i|$$
  - L_2 norm $$L_2(\vec{f},\vec{g}) = \left(\sum_{i=0}^{n} |f_i - g_i|^2 \right)^{\frac{1}{2}}$$
  - L_p norm $$L_p(\vec{f},\vec{g}) = \left(\sum_{i=0}^{n} |f_i - g_i|^p \right)^{\frac{1}{p}}$$

Weighted Distance function

- More weights to reliable components.

$$d_w(\boldsymbol{f}, \boldsymbol{g}) = \sqrt{(\boldsymbol{f} - \boldsymbol{g})^T A (\boldsymbol{f} - \boldsymbol{g})}$$

$\boldsymbol{f}, \boldsymbol{g}$: Column vectors of dimension n.

A: A +ve semidefinite matrix

- Symmetric.
- $v^T A v >= 0$ for all v.
- Typical example: Diag($w_0, w_2, \ldots w_{n-1}$), $w_i >= 0$

$$d_w(\vec{f}, \vec{g}) = \sqrt{\sum_{i=0}^{n-1} w_i (f_i - g_i)^2}$$

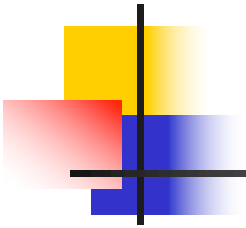
A few other similarity measures

- Normalized Cross Correlation.

$$\rho(\vec{f},\vec{g}) = \frac{cov(f,g)}{s.d.(f)\times s.d.(g)}$$

$$= \frac{\frac{1}{n}\sum_{i=0}^{n-1}(f_i-\bar{f})(g_i-\bar{g})}{\sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(f_i-\bar{f})^2}\sqrt{\frac{1}{n}\sum_{i=0}^{n-1}(g_i-\bar{g})^2}}$$

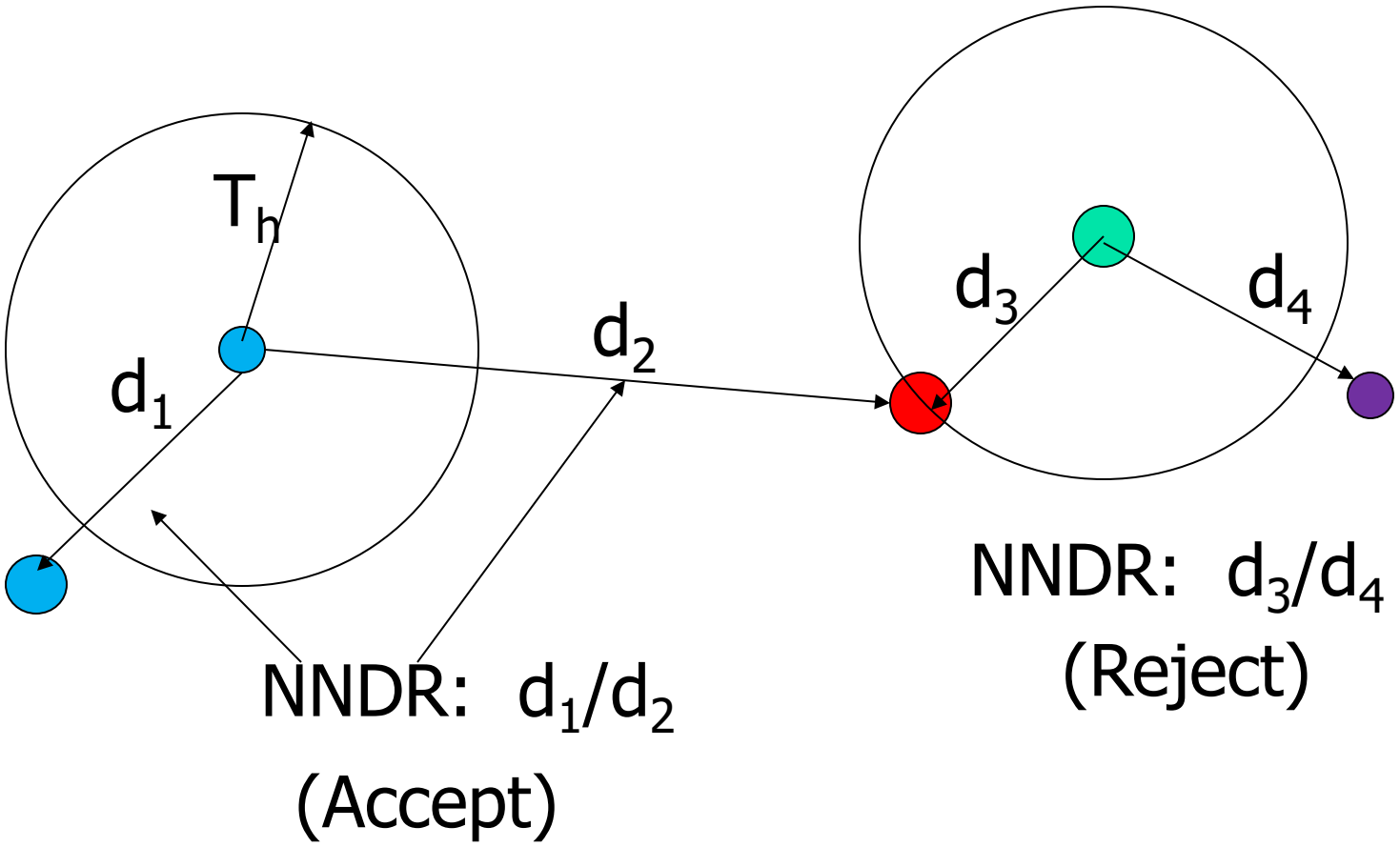
- Cosine similarity. $\frac{\vec{f}.\vec{g}}{\|\vec{f}\|\|\vec{g}\|}$



Matching criteria

- Distance based
  - Fixed threshold (FT):
    - Report all matches within the threshold value.
  - Nearest neighbor (NN):
    - Report the nearest neighbor.
  - Nearest Neighbor Distance Ratio (NNDR):
    - Report the NN if ratio of distances between the NN and 2^{nd} NN is small.

Matching criteria



Matching histograms

- L_p norm of bins (representing same intervals of covariates).
  - Usually L_1 used.

- Kullback-Leiber divergence (of pdf).

$$D_{KL}(P||Q) = -\sum_{x} P(x) \ln \frac{P(x)}{Q(x)}$$

- Earth Mover's Distance (EMD)

Earth Mover's Distance (EMD)

- EMD of two histograms P and Q measures the minimum accumulated cost of transferring masses from any bin of P to any bin of Q so that the histogram P gets transformed into Q.
  - Cost defined by the product of transferred mass and distance between bin.
    - m x $|i-j|$ for transferring mass m from ith bin of P to jth bin of Q.
    - This would make $P[i]=P[i]-m$ and $Q[j]=Q[j]+m$.
  - Total mass of P and Q should be same.
  - Accumulated cost is normalized w.r.t. to the total transfer of mass.

H. Ling and K. Okada, An Efficient Earth Mover's Distance Algorithm for Robust Histogram Comparison, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (5), pp. 840-853, 2007.

Computing EMD

- Two normalized histograms:
  - $P=\{p_i\}$ and $Q=\{q_i\}$, for $i=0,1,\ldots,N\text{-}1$.
  - m_{ij} transferred from i th bin of P to j th bin of Q.
  - d_{ij} distance between i th and j th bin.
- EMD is the minimum *normalized* work (transfer of masses) required for transforming P into Q.

$$EMD(P,Q) = \min_{M=\{m_{ij}\}} \left(\frac{\sum_{i,j} m_{ij} d_{ij}}{\sum_{i,j} m_{ij}} \right) \quad m_{ij} \geq 0$$

$$\sum_j m_{ij} \leq p_i \qquad \sum_i m_{ij} \leq q_j \qquad \sum_i \sum_j m_{ij} \leq min\left(\sum_i p_i , \sum_j q_j \right)$$

H. Ling and K. Okada, An Efficient Earth Mover's Distance Algorithm for Robust Histogram Comparison, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (5), pp. 840-853, 2007.

Computing EMD

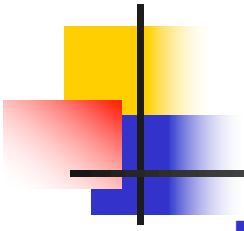
- EMD is the minimum *normalized* work (transfer of masses) required for transforming P into Q.
- Before initiating transfer operation, bins of Q is intialized to zero. The input specificaiton q_j's are used for specifying constraints of the optimization equations.

$$EMD(P,Q) = \min_{M=\{m_{ij}\}} \left(\frac{\sum_{i,j} m_{ij} d_{ij}}{\sum_{i,j} m_{ij}} \right) \quad m_{ij} \geq 0$$

$$\sum_j m_{ij} \leq p_i \qquad \sum_i m_{ij} \leq q_j \qquad \sum_i \sum_j m_{ij} \leq min\left(\sum_i p_i , \sum_j q_j \right)$$

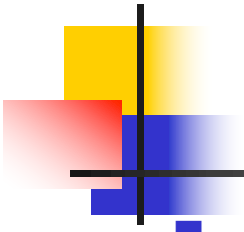
Efficient computation of feature matching

- Range query
  - Nearest Neighbor of the query.
  - Report all the features within a distance from the query.
  - Ordinarily of linear time complexity O(N),
    - N is the number of features in the target image / database.
- Use of indexing and hashing.
  - K-D Tree
  - Locality Sensitive Hashing.



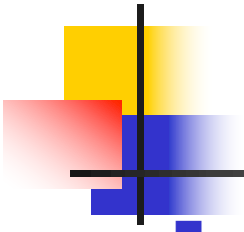
K-D tree

- A binary tree.
- Each node contains a (key) feature vector.
- Each node behaves like a node of binary search tree (BST) for values of a particular dimension
  - Called cut-dimension.
  - Alternate periodically on nodes along any path.
  - Places a key feature vector following the rules of BST comparing on the value of the cut-dimension.



NN Search using K-D tree

- As you walk through the tree nodes
  - Store the current smallest distance and the respective key-feature.
  - Prune sub-trees by comparing the minimum distance with the corner nodes of the hypervolume (bounding boxes) represented by the sub-tree.
  - Search the sub-tree to maximize the chance of pruning.
    - Go to the sub-tree closer to the query.



NN Search using K-D tree

- May require to traverse all the nodes of the tree.
  - Worst case: O(N)
  - In practice close to: $O(\log(N)+2^d)$, where d is the dimension of the feature space.
    - log(N) to find the cell near the query point.
    - 2^d search around cells in the neighborhood.

Locality sensitive hashing (LSH)

- Locality preserving hashing function (h(.)) w.r.t a distance function d(.).
  - Prob.(h(**x**)=h(**y**)) should be high, if d(**x**,**y**) is small,
    - else the probability should be small.

LSH: A typical example

- Choose a random unit vector **r** of dimension n (e.g. following Normal distribution (N(0,1)) independently in each dimension).
- For any vector **x** of dimension n define a hash function as follows:

$$h_{\mathbf{r}}(\mathbf{x}) = \begin{cases} 1 & if\ \mathbf{x}.\mathbf{r} > 0 \\ 0 & otherwise \end{cases}$$

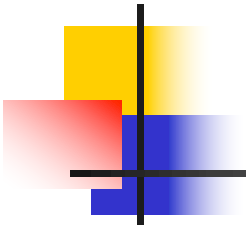
It can be shown, for any **x** and **y**

$$\Pr[h_{\mathbf{r}}(\mathbf{x}) = h_{\mathbf{r}}(\mathbf{y})] = 1 - \frac{\theta(\mathbf{x},\mathbf{y})}{\pi}$$

Angle between **x** and **y**.

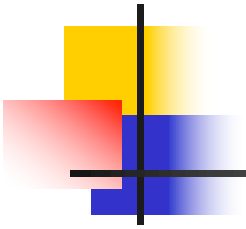
LSH multidimenstional bucketing

- You may also have a set of k random vectors independently generating k hash values, and define the multidimensional bucket for hashing to place the input vector.
  - $H(\mathbf{x})=[h_{\mathbf{r1}}(\mathbf{x})\ h_{\mathbf{r2}}(\mathbf{x})\ \ldots.\ h_{\mathbf{rk}}(\mathbf{x})]$
- You may have L multiple buckets (corresponding to L sets of k random vectors) for the same input.
  - $H_1(\mathbf{x}), H_2(\mathbf{x}), \ldots, H_L(\mathbf{x})$
- Given a query **y** compute L buckets and search for NN in all of them.



Model fitting

- Given a set of data points fit a model to establish relationship among the data points.
  - Homography matrix between pixels in two images of the same planar scene.
  - Fundamental matrix between corresponding points in two stereo images.
  - Given a set of 2-D points fit a straight line / parabolic curve / circle / a high degree polynomial curve passing through them.
- Obtain data points and fit appropriate model.
  - Preprocessing, feature detection-description-matching.



Knowledge of models

- Mathematical form of the class / family of Models.
  - 3x3 non-singular homography matrix for projective transformation.
  - 3x3 singular fundamental matrix for stereo geometry.
  - 3x4 Projection matrix.

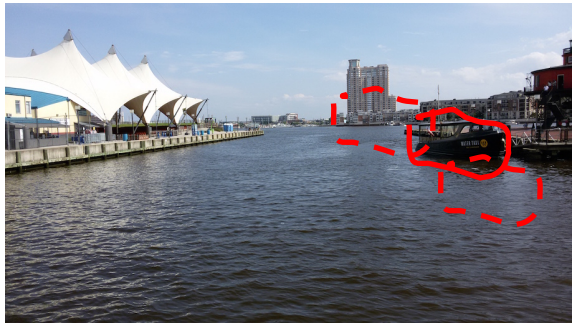
Choice of a model

- Error of fitting
  - Mean square error
- Likelihood
  - Pr (Data | Model)
- Size of a model
  - Number of independent parameters
- Training and Test Error
    - Large training error: Under-fitting.
    - Low training error, but large test error: Over-fitting
    - Low training and test error: A good fit.

Fitting curves and lines

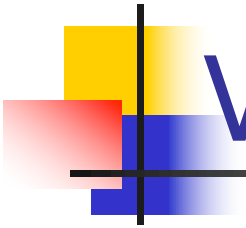
simple model: lines

simple model: circles



To decide about appropriate parametric models.

complicated model: Steam-boat

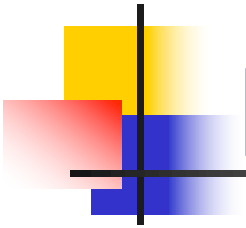


Various Issues

- **Noise / Error**
  - in estimation of feature locations.
- **Extraneous data**
  - clutter (outliers), multiple lines.
- **Missing data**
  - occlusions

Adapted from slides by S. Lazebnik.

Fitting a straight line over 2D points



- Fitting techniques
  - Least Squares
  - Total Least Squares
  - Random Sample Consensus (RANSAC)
  - Hough Voting

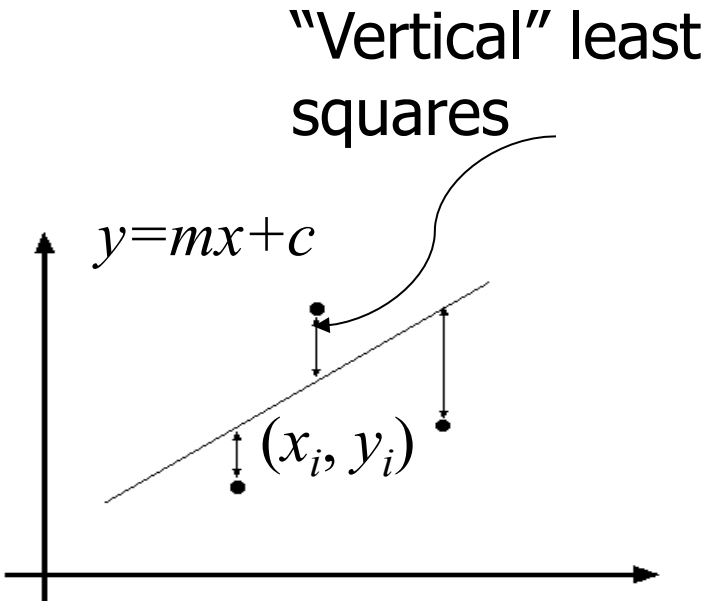
Line fitting in varying contexts

- Given points belonging to a line, to find the "optimal" line parameters.
  - Least squares
- To handle outliers.
  - RANSAC
- Presence of many lines.
  - Voting methods: Hough transform

Adapted from slides by S. Lazebnik.

Least squares line fitting

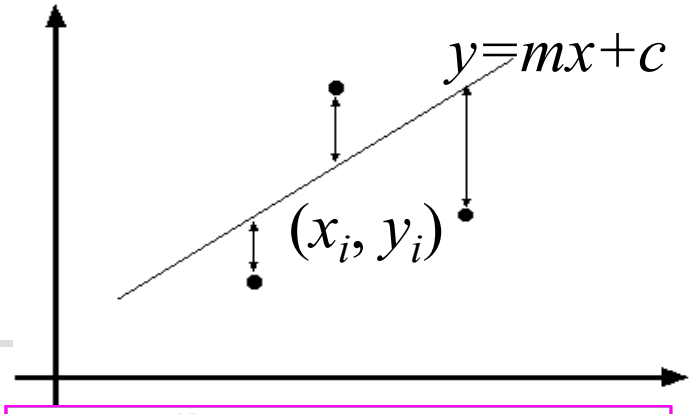
$$E = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$



- Data:
  - $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation:
  - $y_i = m\, x_i + c$
- Find (m, c) to minimize

Adapted from slides by S. Lazebnik.

Least squares line fitting



- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + c$
- Find (m, c) to minimize

$$E = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

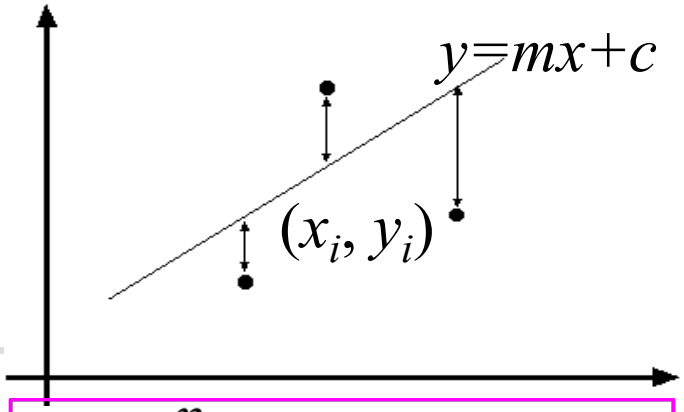
$$E = \sum_{i=1}^{n} \left(y_i - \begin{bmatrix} x_i & 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} \right)^2 = \left\| \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} - \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} m \\ c \end{bmatrix} \right\|^2 = \|Y - XC\|^2$$

$$E = (Y - XC)^T (Y - XC) = Y^T Y - 2(XC)^T Y + (XC)^T (XC)$$

$$\frac{dE}{dC} = 2X^T XC - 2X^T Y = 0 \quad \Rightarrow \quad C = (X^T X)^{-1} X^T Y$$

Adapted from slides by S. Lazebnik.

Least squares line fitting



$$C = (X^T X)^{-1} X^T Y$$

$$E = \sum_{i=1}^{n} (y_i - mx_i - c)^2$$

$$m = \frac{cov(X,Y)}{var(X)} = \frac{\frac{1}{n}\sum_i x_i y_i - \bar{x}\bar{y}}{\frac{1}{n}\sum_i x_i^2 - \bar{x}^2}$$

$$E = n(var(y) - m^2 var(x))$$

$$c = \bar{y} - m\bar{x}$$

The R^2 goodness-of-fit criterion compares the variability in the measurements not explained by the model to the total variability in the measurements.

Goodness of fit: $R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \Rightarrow R^2 = 1 - \frac{E}{n.var(y)}$

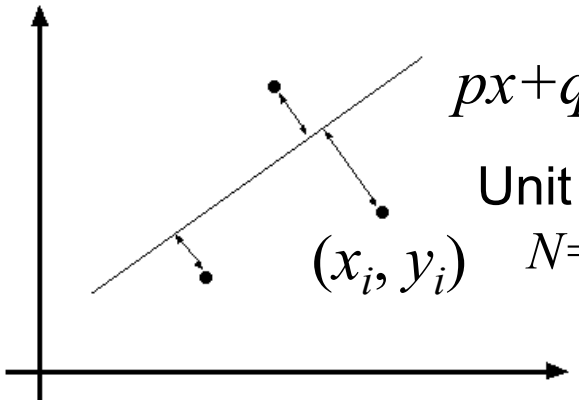
- Not rotation-invariant.
- Fails completely for vertical lines.

Adapted from slides by S. Lazebnik.

Total least squares

- Distance between a point (x_i, y_i) and the line $px+qy=d$: $|px_i + qy_i - d|$ given $p^2+q^2=1$.

$$E = \sum_{i=1}^{n} (px_i + qy_i - d)^2$$



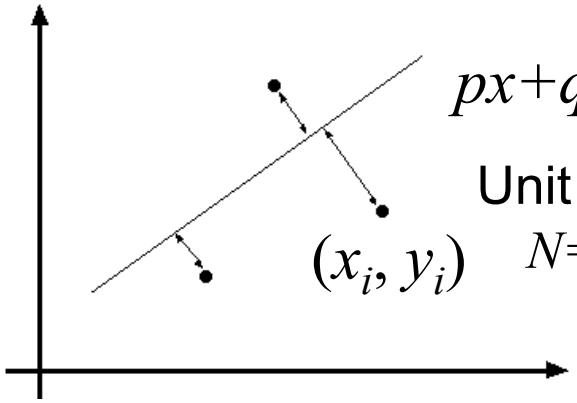
$$\frac{\partial E}{\partial d} = \sum_{i=1}^{n} -2(px_i + qy_i - d) = 0$$

$$\Rightarrow \quad d = p\bar{x} + q\bar{y} \quad \Rightarrow \quad E = \sum_{i=1}^{n} (p(x_i - \bar{x}) + q(y_i - \bar{y}))^2$$

Adapted from slides by S. Lazebnik.

Total least squares

$$E = \sum_{i=1}^{n} (p(x_i - \bar{x}) + q(y_i - \bar{y}))^2$$



$$E = \left\| \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} \right\|^2 = (UN)^T (UN)$$

(U and N labeled above the matrix and vector, respectively)

$$\frac{dE}{dN} = 2(U^T U)N = 0$$

Subject to $\|N\|=1$.

Eigen vector of $U^T U$ corresponding to the smallest eigen value.

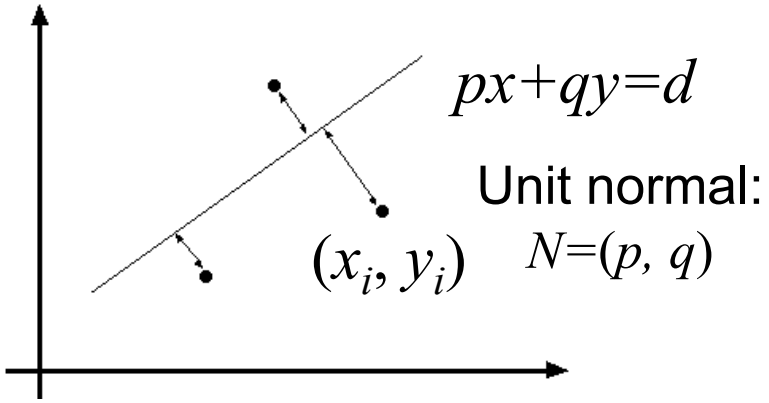
Adapted from slides by S. Lazebnik.

Total least squares

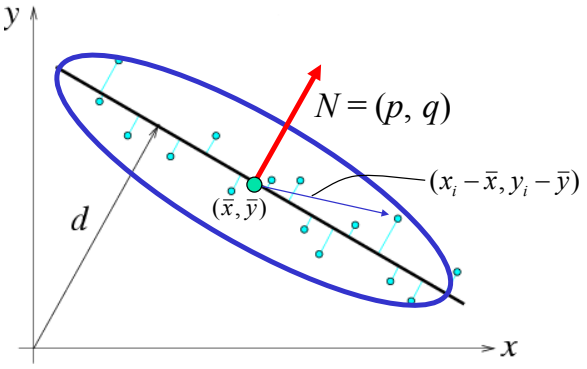
$$E = \left\| \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \begin{bmatrix} p \\ q \end{bmatrix} \right\|^2 = (UN)^T (UN)$$

$$\frac{dE}{dN} = 2(U^T U)N = 0$$

Subject to $\|N\|=1$.



$$U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$



Eigen vector of $U^T U$ corresponding to the smallest eigen value.

Adapted from slides by S. Lazebnik.

Random sample consensus (RANSAC)

- A general framework for model fitting in the presence of outliers.
- Outline
  - Choose a small subset of points uniformly at random.
  - Fit a model to that subset.
  - Find all remaining points that are "close" to the model and reject the rest as outliers.
  - Do this many times and choose the best model.

M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.

Adapted from slides by S. Lazebnik.

RANSAC for line fitting

- Perform the following ***N*** times:
- Select ***s*** points uniformly at random.
- Fit a line using the LSE method.
- Find inliers to this line
  - points whose distance from the line is less than ***t***.
- For sufficient number of inliers (say > ***d***), accept the line and refit using all inliers.

Adapted from slides by M. Pollefeys

Choice of parameters

- Initial number of points ***s***
  - Typically minimum number needed to fit the model.
- Distance threshold ***t***
  - Choose ***t*** so probability for inlier is *p* (e.g. 0.95).
  - Zero-mean Gaussian noise with std. dev. σ: t=1.96σ.
- Number of trials ***N***
  - Choose ***N*** so that, with probability *p*, at least in one trial random selections are free from any outlier. (e.g. *p*=0.99) (outlier ratio: *e*).
- Consensus set size ***d***
  - Should match expected inlier ratio: (1-e) x n.

n=No. of data points

Estimating number of trials N

Outlier ratio: e ← Prob. that a sample is outlier.

Prob. that all s samples are inliers: $(1- e)^s$

Prob. that at least one sample is an outlier in a trial: ⇨ $(1- (1- e)^s)$

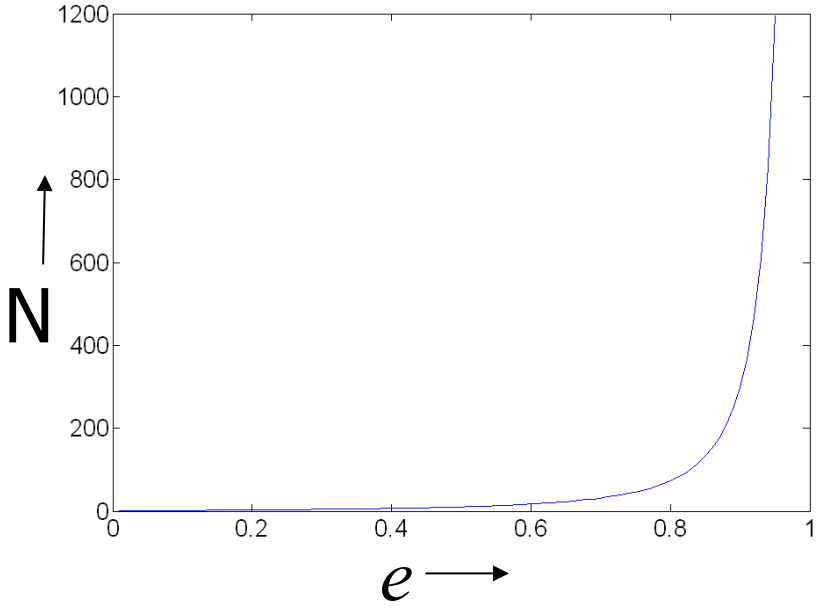
Prob. that all N trials have an outlier:$(1- (1- e)^s)^N$

Given probability p, so that at least one random sample is free from outliers. ⇨ $(1- (1- e)^s)^N=1-p$

$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

Adapted from slides by M. Pollefeys

Choosing N



$$N = \frac{\log(1-p)}{\log(1-(1-e)^s)}$$

p=0.99

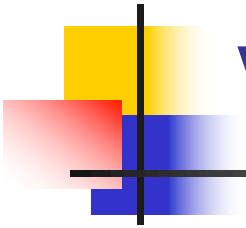
	proportion of outliers e						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Adapted from slides by M. Pollefeys

RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Many parameters to tune
  - Can’t always get a good initialization of the model based on the minimum number of samples
  - Sometimes too many iterations are required
  - Not appropriate for low inlier ratio.

Adapted from slides by M. Pollefeys

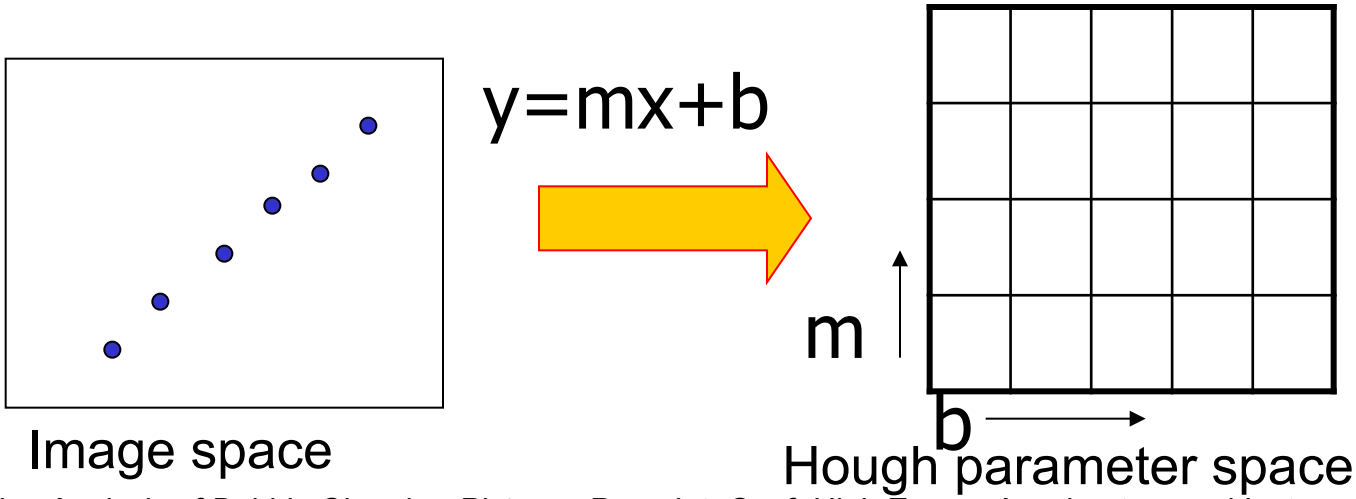


Voting schemes

- 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

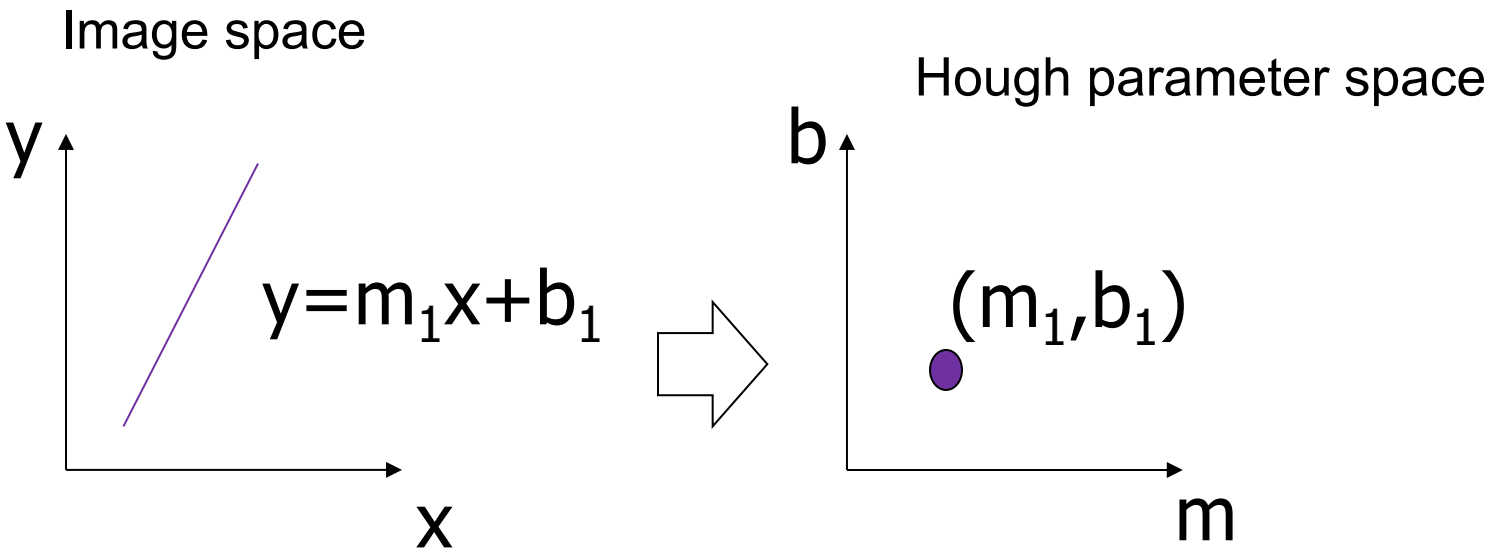
- Discretize parameter space into bins
- For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point.
- Find bins that have the most votes.



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

Parameter space representation

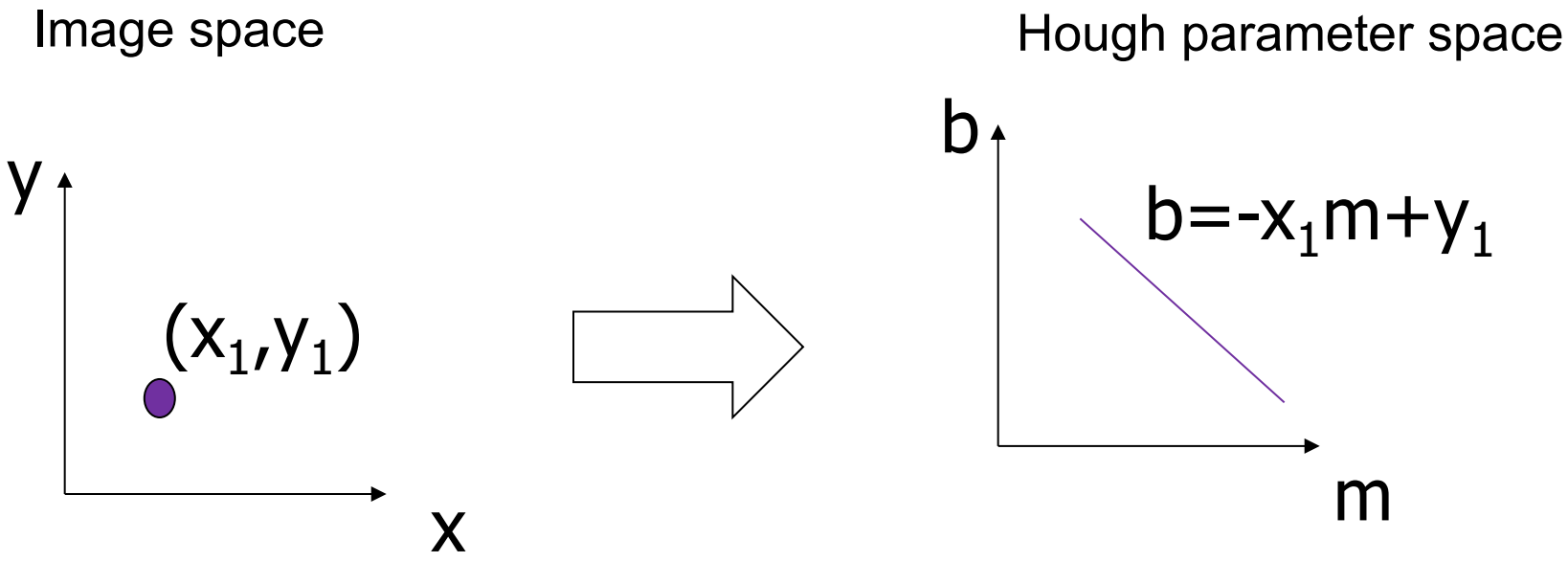
- A straight line in the image corresponds to a point in Hough space.



Adapted from slides by S. Seitz

Parameter space representation

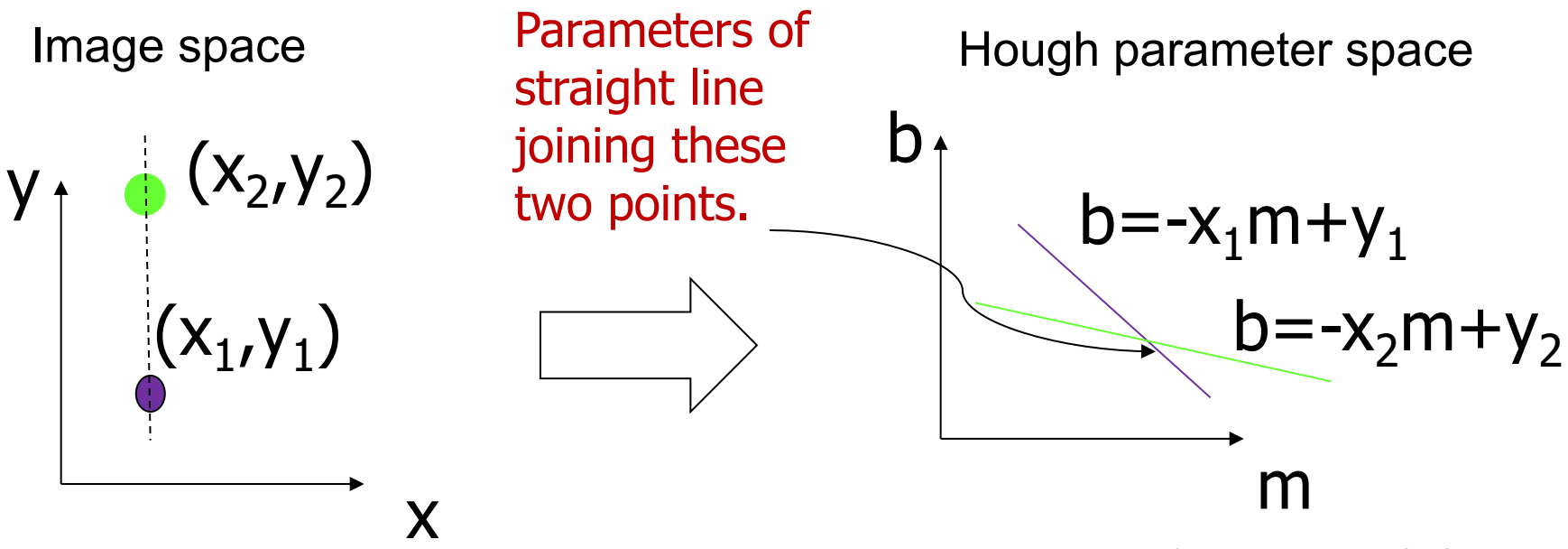
- A point in the image corresponds to a line in the Hough space.



Adapted from slides by S. Seitz

Parameter space representation

- Two points in the image correspond to two lines in the Hough space.



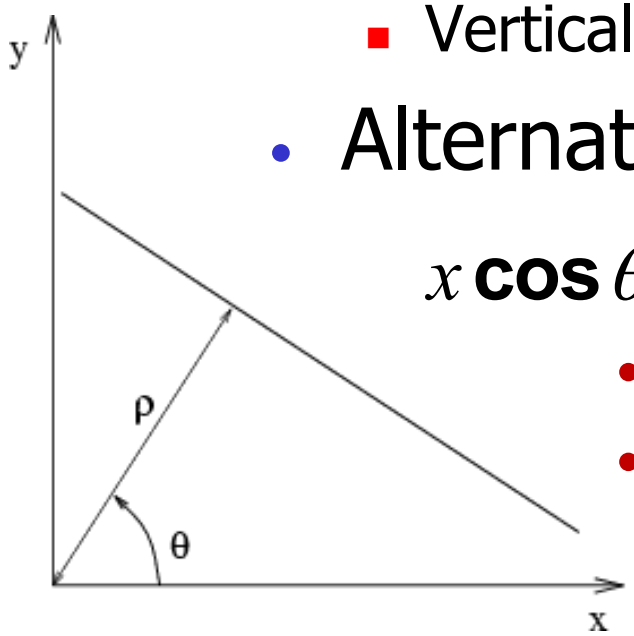
Adapted from slides by S. Seitz

Parameter space representation

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

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

  - θ varies from 0 to 180°
  - ρ varies from 0 to the length of diagonal of the image grid.



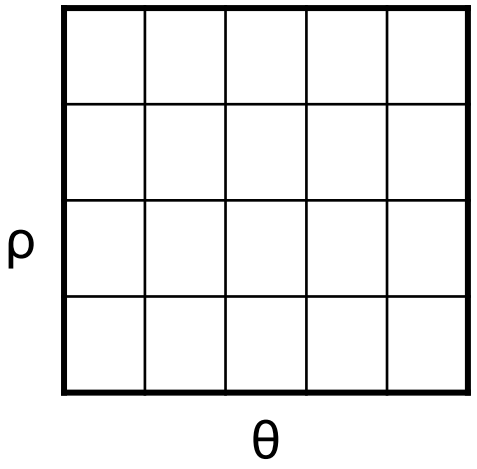
Each point will add a sinusoid in the (θ,ρ) parameter space

Adapted from slides by S. Seitz

Algorithm

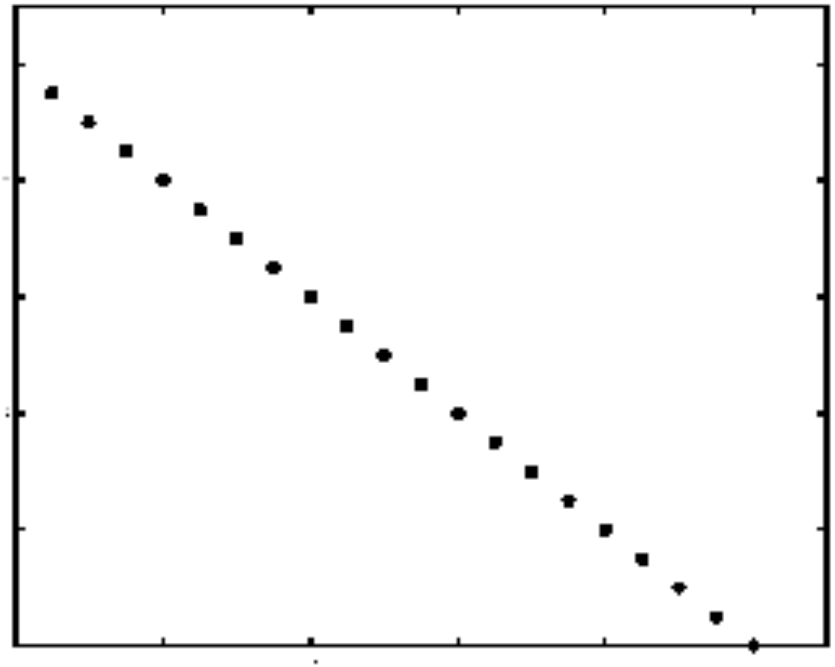
- Initialize accumulator A to all zeros
- For each edge point (x,y) in the image
- {
  For θ = 0 to 180
- {
  ρ = x cos θ + y sin θ
  A(θ, ρ) = A(θ, ρ) + 1
  }
  }
- Find the value(s) of (θ, ρ) where A(θ, ρ) is a local maximum
  - The detected line in the image is given by
    ρ = x cos θ + y sin θ

A: Accumulator array

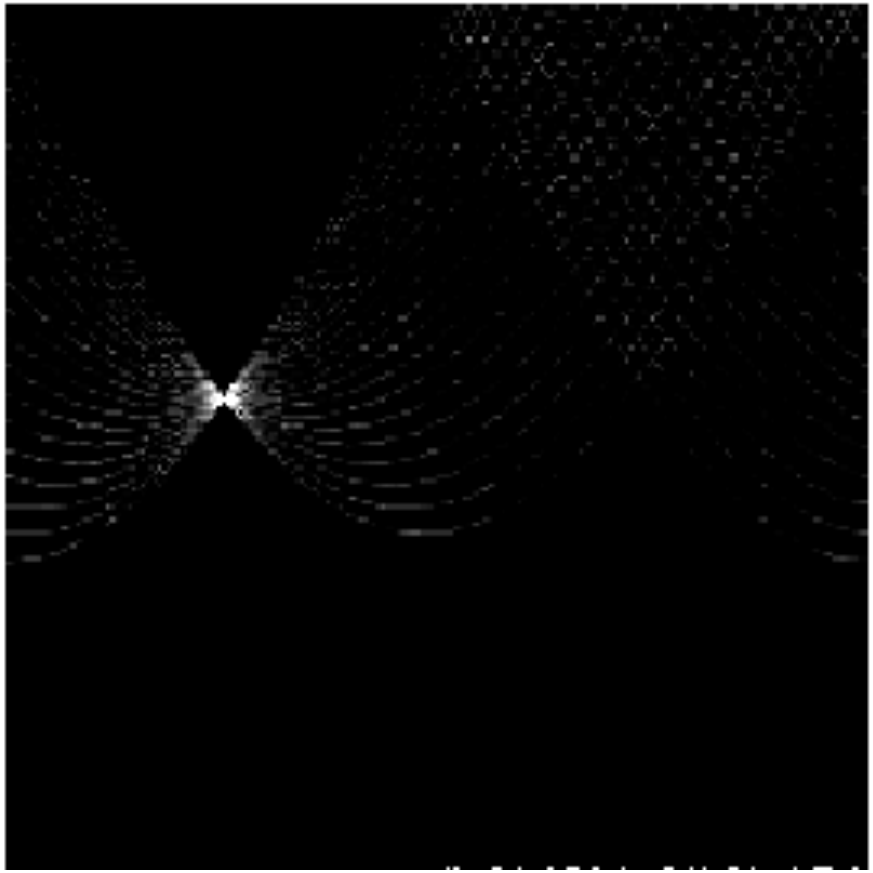


Adapted from slides by S. Seitz

Basic illustration



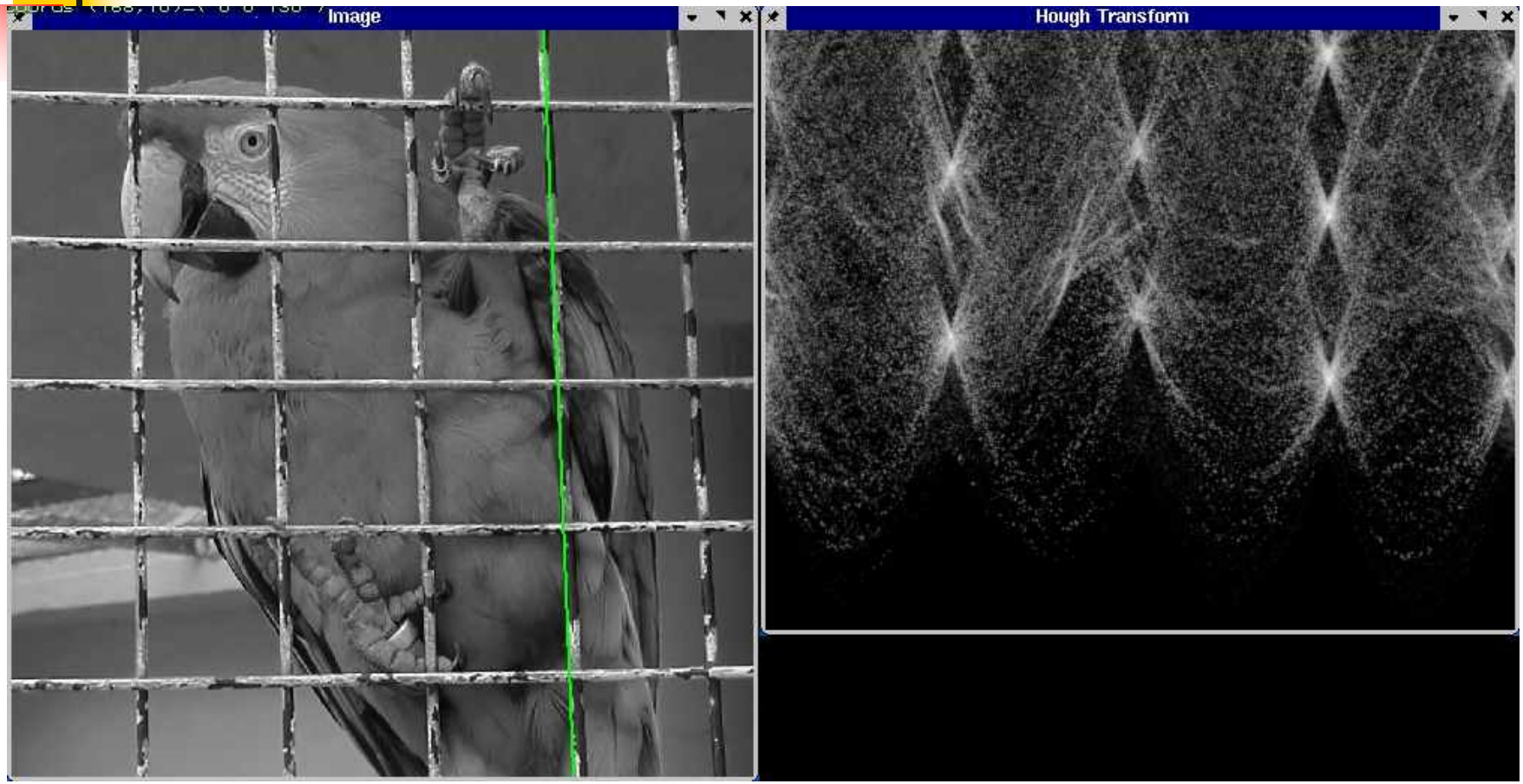
features



votes

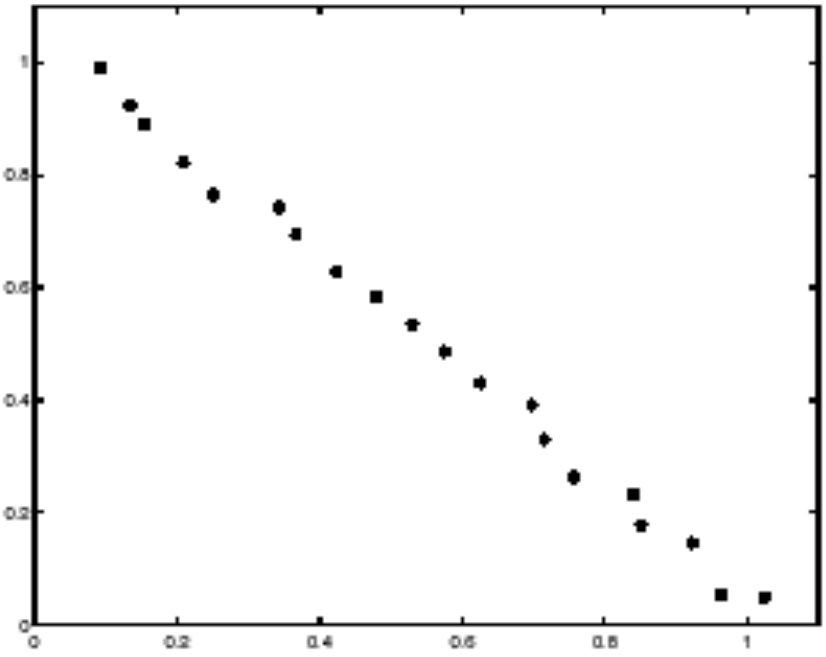
http://ostatic.com/files/images/ss_hough.jpg

A more complicated image

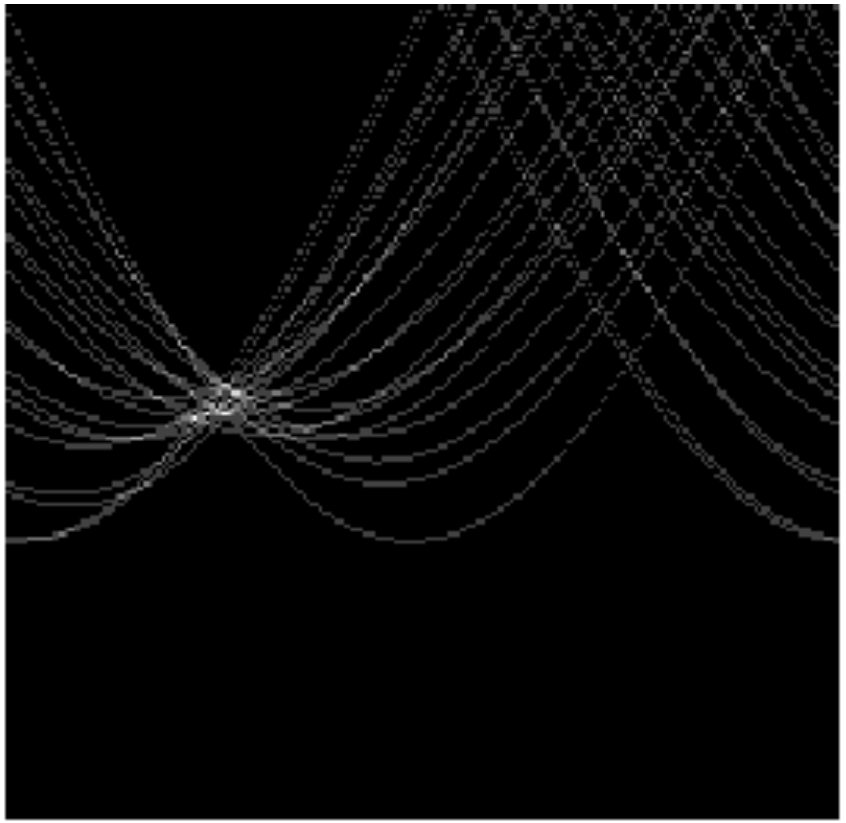


http://ostatic.com/files/images/ss_hough.jpg

Effect of noise

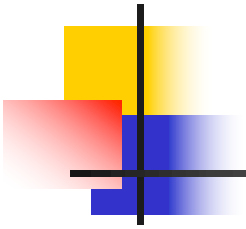


features



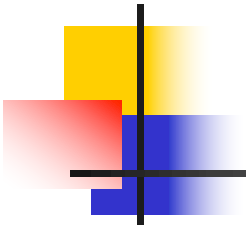
votes

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



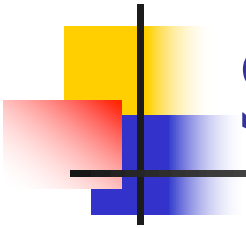
Dealing with noise

- Requires appropriate resolution of discretization of the grid in the parameter space
  - Too coarse:
    - large votes in a cell for accumulating too many different lines correspond to a single bucket
  - Too fine:
    - may miss lines as some points may not be exactly collinear voting different buckets.



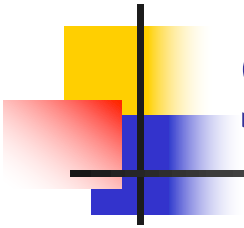
Dealing with noise

- Smoothing accumulator array.
- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude



Summary

- Feature matching.
  - Distance function / Similarity measure.
  - Different policies
    - Use of a threshold value.
    - Nearest Neighbor.
    - Nearest neighbor with distance ratio (NNDR).
  - Use of indexing / hashing for efficient computation.
  - Special distance functions for comparing histograms.
    - Kullback-Leiber Divergence (KLD)
    - Earth Mover's Distance (EMD)



Summary

- Model fitting
  - Prior knowledge of the model useful.
  - Need to consider goodness of fit.
  - Different techniques for line fitting
    - Least squares
    - Total least squares.
    - RANSAC
    - Hough Transform.