Image Analysis

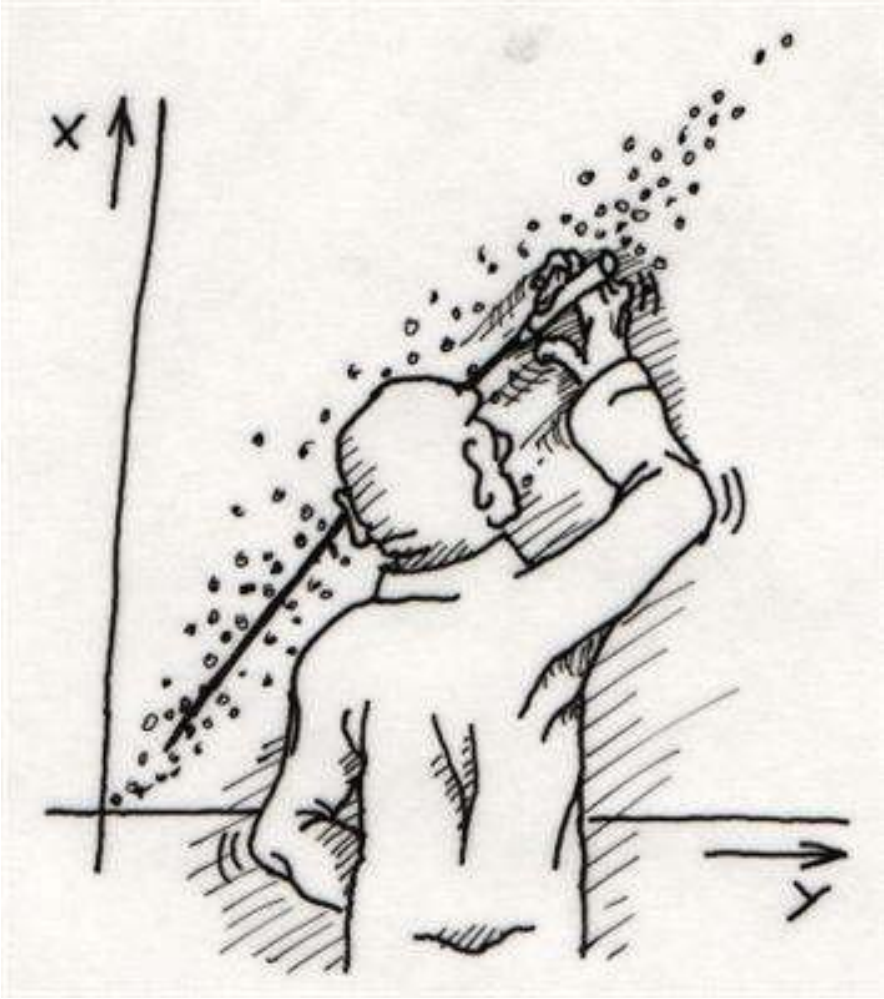
Segmentation by Fitting a Model

Christophoros Nikou

cnikou@cs.uoi.gr

Images taken from:
D. Forsyth and J. Ponce. Computer Vision: A Modern Approach, Prentice Hall, 2003.
Computer Vision course by Svetlana Lazebnik, University of North Carolina at Chapel Hill (http://www.cs.unc.edu/~lazebnik/spring10/).

Fitting

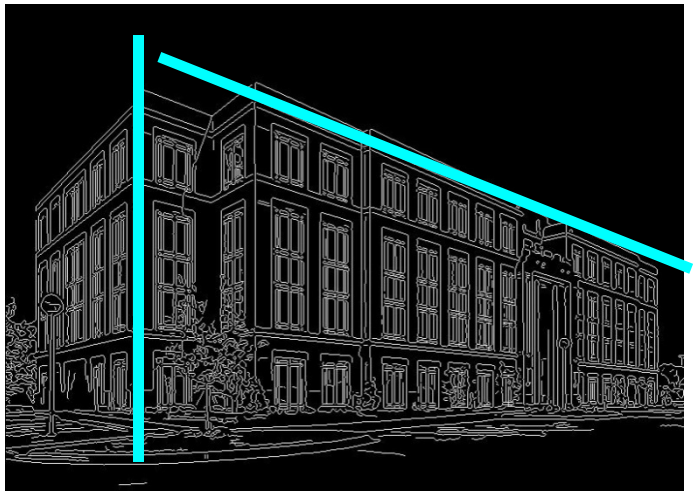


Fitting: motivation

- We've learned how to detect edges, corners, blobs. Now what?

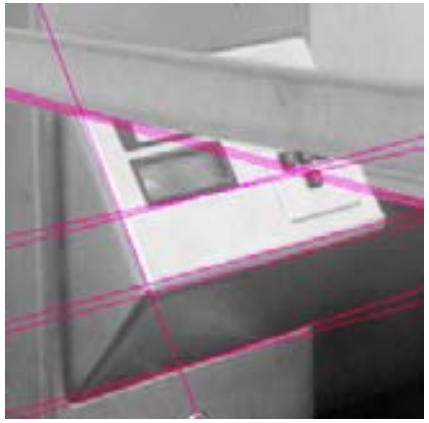
- We would like to form a higher-level, more compact representation of the features in the image by grouping multiple features according to a simple model.

9300 Harris Corners Pkwy, Charlotte, NC



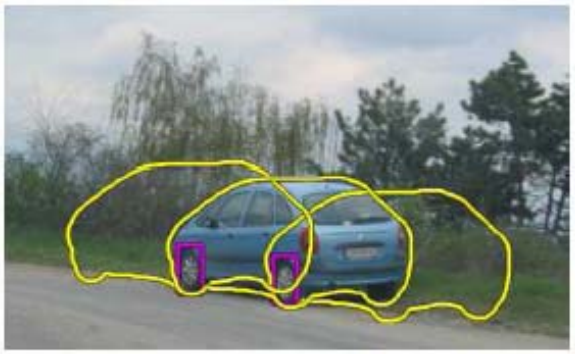
Fitting: motivation (cont.)

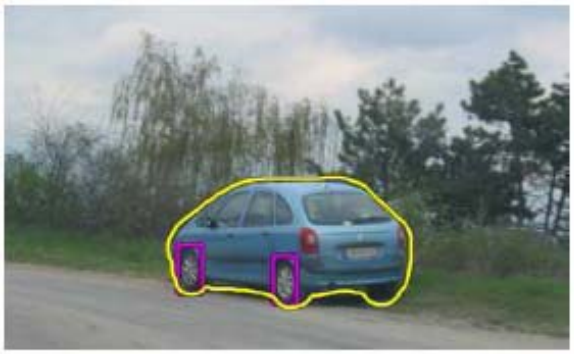
- Choose a parametric model to represent a set of features.



simple model: lines

simple model: circles





complicated model: car

Fitting: issues

- Choose a parametric model to represent a set of features.
- Membership criterion is not local
  - Can't tell whether a point belongs to a given model just by looking at that point.
- Three main questions:
  - What model represents this set of features best?
  - Which feature to which of several models?
  - How many models are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

Fitting: issues (cont.)

Case study: Line detection

- **Noise** in the measured feature locations.
- **Extraneous data:** clutter (outliers), multiple lines.
- **Missing data:** occlusions.

Fitting: issues (cont.)

- If we know which points belong to the line, how do we find the “optimal” line parameters?
  - Least squares.
- What if there are outliers?
  - Robust fitting, RANSAC.
- What if there are many lines?
  - Voting methods: RANSAC, Hough transform.
- What if we’re not even sure it’s a line?
  - Model selection.

Least squares line fitting

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

$$E = \sum_{i=1}^{n} (y_i - m x_i - b)^2$$

- Is there any problem with this formulation?
- We will find the solution first and we will discuss it afterwards.

Least squares line fitting (cont.)

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

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

$$= (Y - XB)^T (Y - XB) = Y^T Y - 2(XB)^T Y + (XB)^T (XB)$$

$$\frac{dE}{dB} = 2X^T XB - 2X^T Y = 0 \Leftrightarrow X^T XB = X^T Y$$

Least squares solution to $XB=Y$: $B = (X^T X)^{-1} X^T Y$

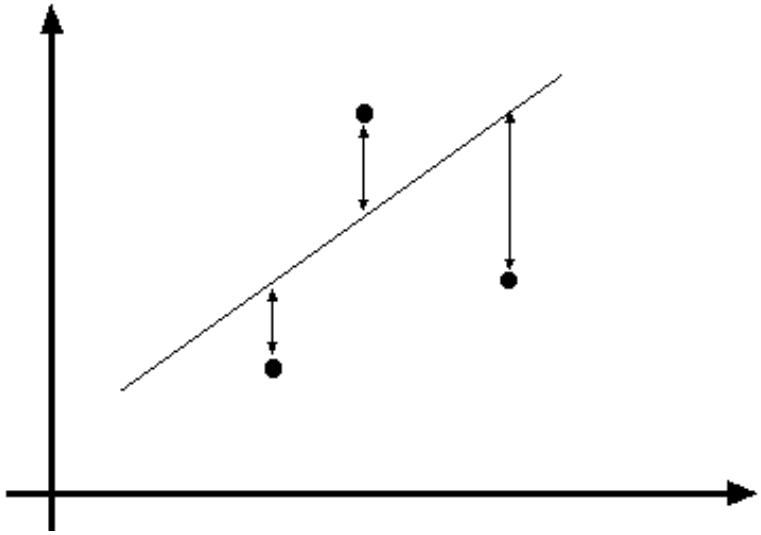
Least squares line fitting (cont.)

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

$$\Leftrightarrow B = \begin{bmatrix} m \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \frac{1}{n} \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{bmatrix}$$

Least squares line fitting (cont.)

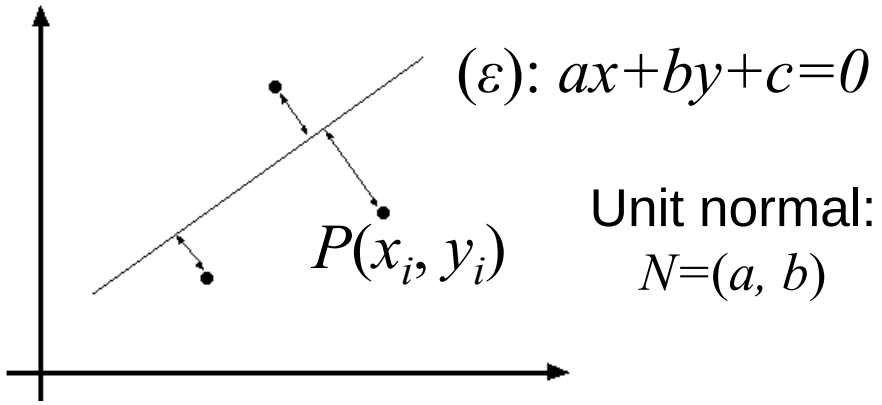
- Problems
  - Not rotation-invariant.
  - It measures the error of the vertical coordinate only.
  - Fails completely for vertical lines.



Total least squares

- Distance between point $P(x_n, y_n)$ and line (ε): $ax+by+c=0$:

$$d(P,\varepsilon)=\frac{|ax_n+by_n+c|}{\sqrt{a^2+b^2}}$$

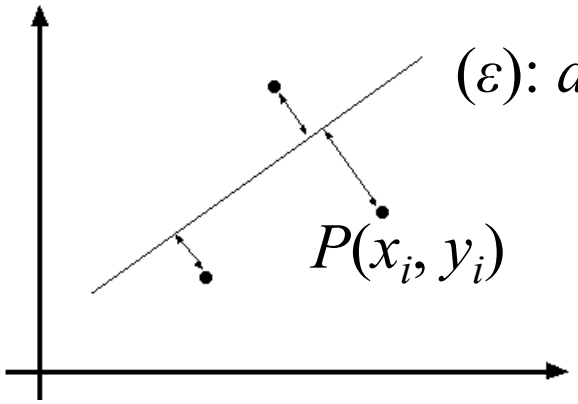


- We impose the constraint $a^2+b^2=1$ because $\lambda ax+\lambda by+\lambda c=0$ is the same line.

Total least squares (cont.)

- Find the parameters of the line (a, b, c) that minimize the sum of squared perpendicular distances of each point from the line:

$$E(a,b,c) = \sum_{i=1}^{n} (ax_i + by_i + c)^2 \quad \text{subject to } a^2+b^2=1.$$



Total least squares (cont.)

$$E(a,b,c)=\sum_{i=1}^{n}(ax_i+by_i+c)^2$$

$$\frac{\partial E}{\partial c}=\sum_{i=1}^{n}2(ax_i+by_i+c)=0 \qquad \Leftrightarrow c=-\frac{a}{n}\sum_{i=1}^{n}x_i-\frac{b}{n}\sum_{i=1}^{n}y_i=-a\bar{x}-b\bar{y}$$

Substituting c into $E(a,b,c)$ and introducing the constriant:

$$E(a,b,\lambda)=\sum_{i=1}^{n}\left(a(x_i-\bar{x})+b(y_i-\bar{y})\right)^2+\lambda\left(1-a^2+b^2\right)$$

Total least squares (cont.)

$$\frac{\partial E}{\partial a} = 0 \Leftrightarrow a\left(\frac{1}{n}\sum_{i=1}^{n} x_i^2 - (\bar{x})^2\right) + b\left(\frac{1}{n}\sum_{i=1}^{n} x_i y_i - \bar{x}\,\bar{y}\right) = \lambda a$$

$$\frac{\partial E}{\partial b} = 0 \Leftrightarrow a\left(\frac{1}{n}\sum_{i=1}^{n} x_i y_i - \bar{x}\,\bar{y}\right) + b\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2 - (\bar{y})^2\right) = \lambda b$$

which may be written in matrix-vector form:

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

Total least squares (cont.)

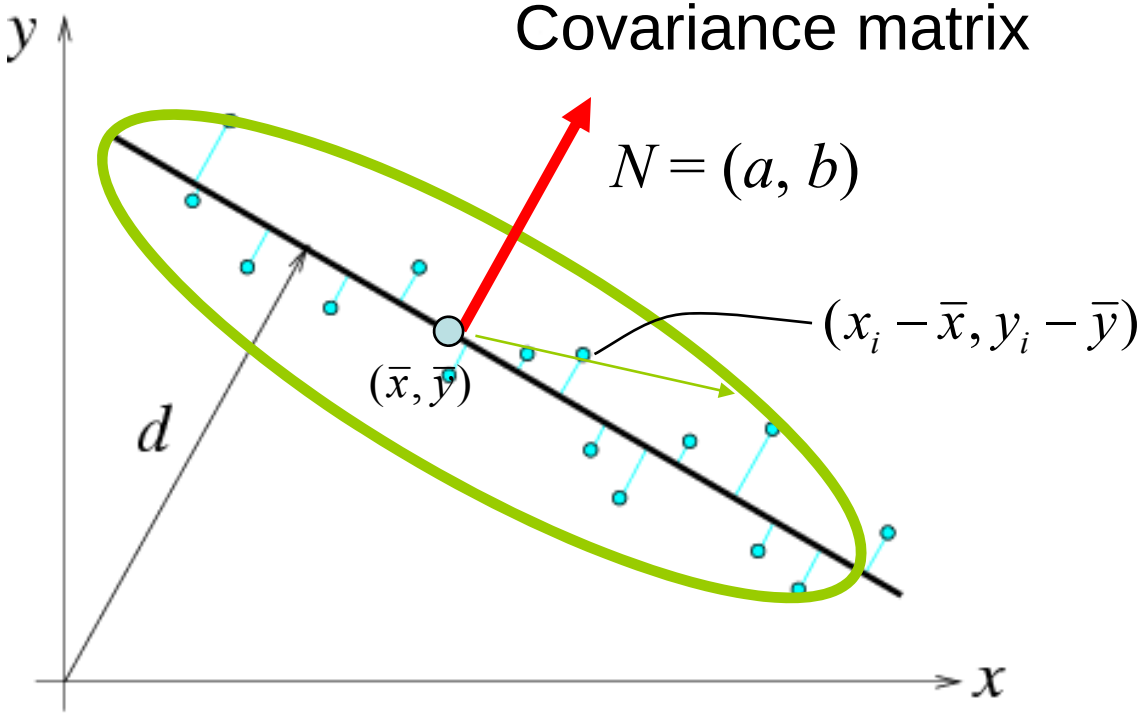
$$\begin{bmatrix} \sum_{i=1}^{n}(x_i-\bar{x})^2 & \sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y}) \\ \sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y}) & \sum_{i=1}^{n}(y_i-\bar{y})^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \lambda \begin{bmatrix} a \\ b \end{bmatrix} \Leftrightarrow Aw = \lambda w$$

- The solution is the eigenvector of A corresponding to the smallest eigenvalue (minimization of E) of the covariance matrix.
- The second eigenvalue (the largest) corresponds to a perpendicular direction.

Total least squares (cont.)

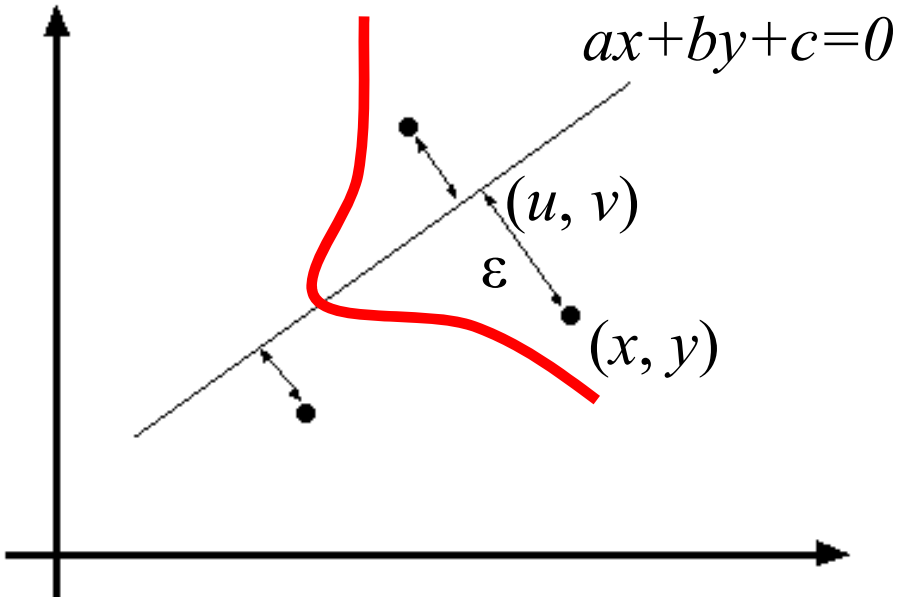
$$\begin{bmatrix} \sum_{i=1}^{n}(x_i-\bar{x})^2 & \sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y}) \\ \sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y}) & \sum_{i=1}^{n}(y_i-\bar{y})^2 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \lambda \begin{bmatrix} a \\ b \end{bmatrix}$$

Covariance matrix



Least squares as a maximum likelihood problem

- **Generative model**: line points are corrupted by Gaussian noise in the direction perpendicular to the line



$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} + \varepsilon \begin{pmatrix} a \\ b \end{pmatrix}$$

point on the line

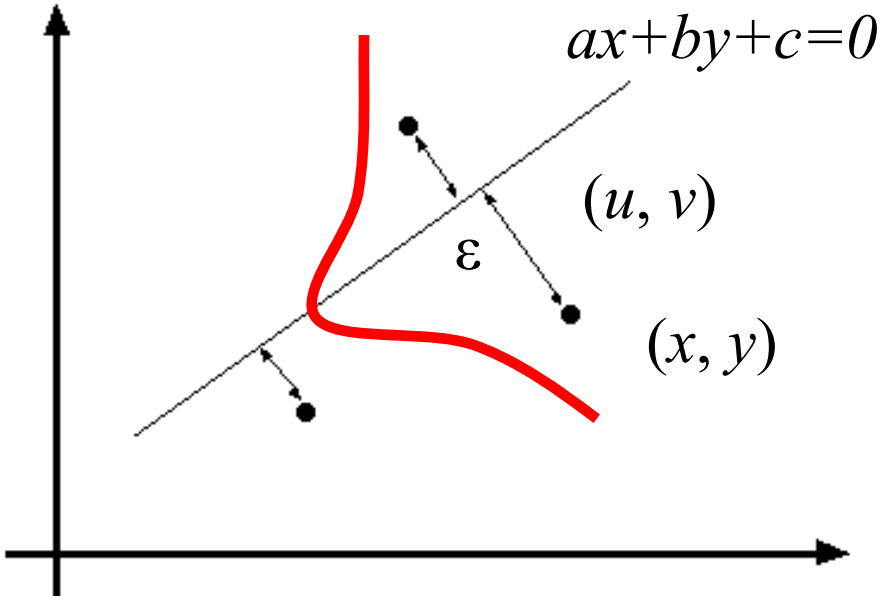
noise: zero-mean Gaussian with std. dev. σ

normal direction

Least squares as a maximum likelihood problem (cont.)

- **Generative model**:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} u \\ v \end{pmatrix} + \varepsilon \begin{pmatrix} a \\ b \end{pmatrix}$$



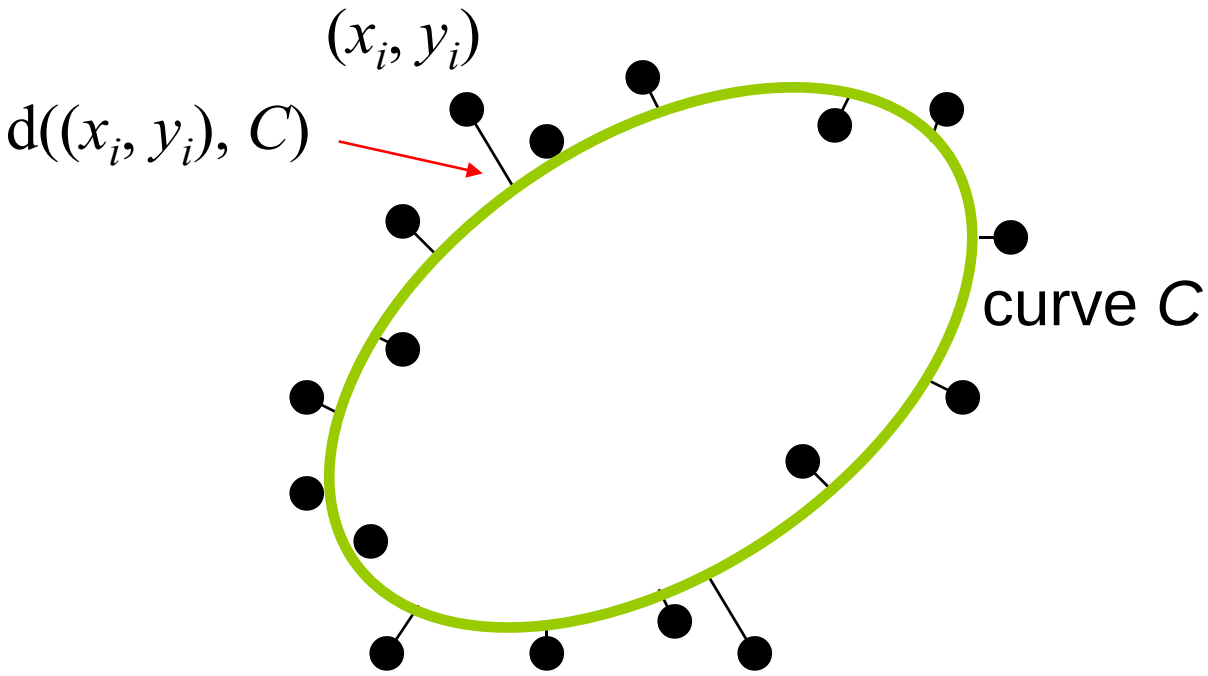
Likelihood of points given line parameters (a, b, c):

$$P(x_1,\ldots,x_n \mid a,b,d) = \prod_{i=1}^{n} P(x_i \mid a,b,c) \propto \prod_{i=1}^{n} \exp\left(-\frac{(ax_i+by_i+c)^2}{2\sigma^2}\right)$$

Log-likelihood: $L(x_1,\ldots,x_n \mid a,b,c) = -\frac{1}{2\sigma^2}\sum_{i=1}^{n}(ax_i+by_i+c)^2$

Least squares for general curves

- We would like to minimize the sum of squared *geometric distances* between the data points and the curve.



Least squares for conics

- Equation of a 2D general conic:

  $C(\mathbf{a}, \mathbf{x}) = \mathbf{a} \cdot \mathbf{x} = ax^2 + bxy + cy^2 + dx + ey + f = 0,$

  $\mathbf{a} = [a, b, c, d, e, f], \quad \mathbf{x} = [x^2, xy, y^2, x, y, 1]$

- *Algebraic distance:* $C(\mathbf{a}, \mathbf{x})$.
- Algebraic distance minimization by linear least squares:

$$\begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2^2 & x_2y_2 & y_2^2 & x_2 & y_2 & 1 \\ \vdots & & & & \ddots & \vdots \\ x_n^2 & x_ny_n & y_n^2 & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = 0$$

Least squares for conics

- Least squares system: D**a** = 0
- Need constraint on **a** to prevent trivial solution
- Discriminant: $b^2 - 4ac$
  - Negative: ellipse
  - Zero: parabola
  - Positive: hyperbola
- Minimizing squared algebraic distance subject to constraints leads also to a generalized eigenvalue problem.

A. Fitzgibbon, M. Pilu, and R. Fisher, *Direct least-squares fitting of ellipses*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 21(5), 476--480, May 1999 .

Who came from which line?

- Assume we know how many lines there are - but which lines are they?
  - easy, if we know who came from which line
- Three strategies
  - Incremental line fitting
  - K-means
  - Probabilistic (later!)

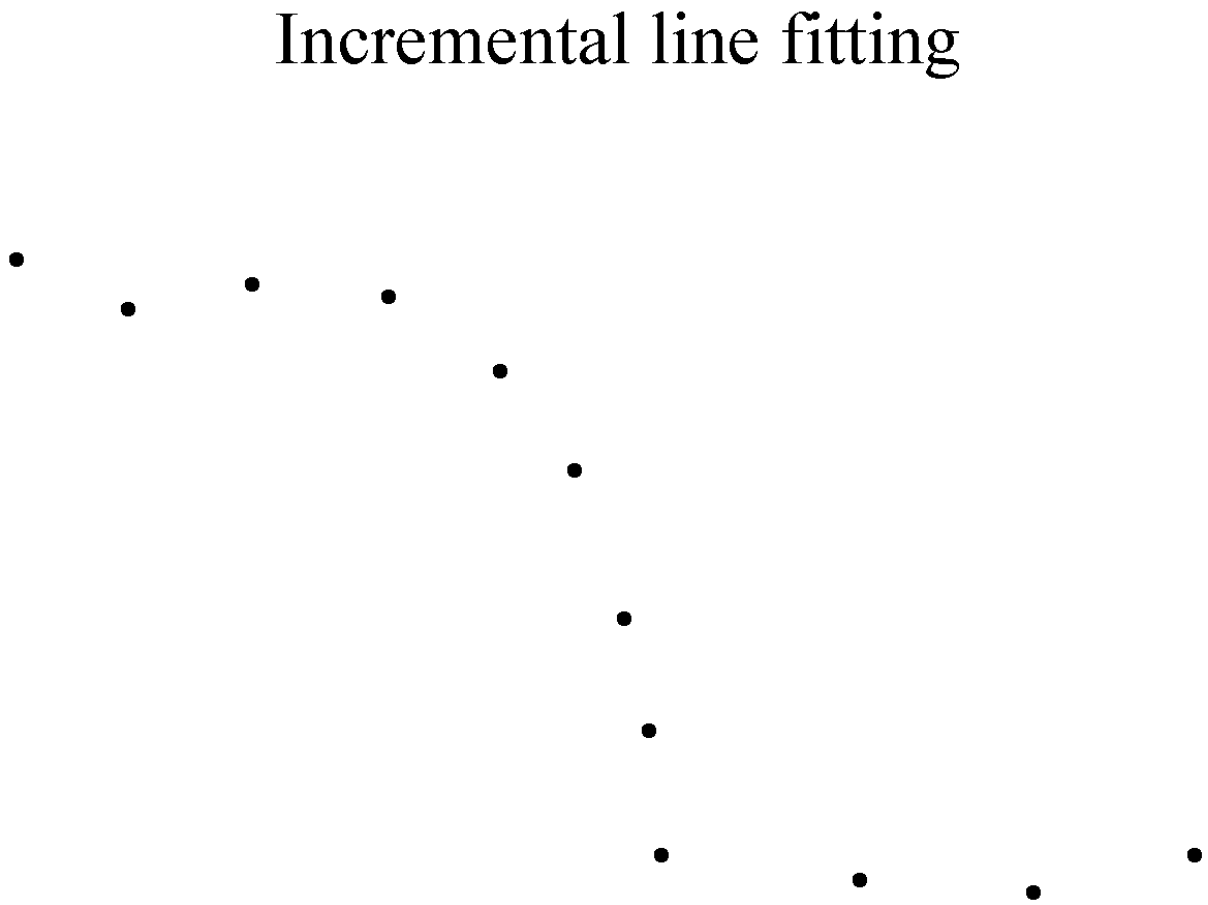
Incremental line fitting

A really useful idea; it's the easiest way to fit a set of lines to a curve.

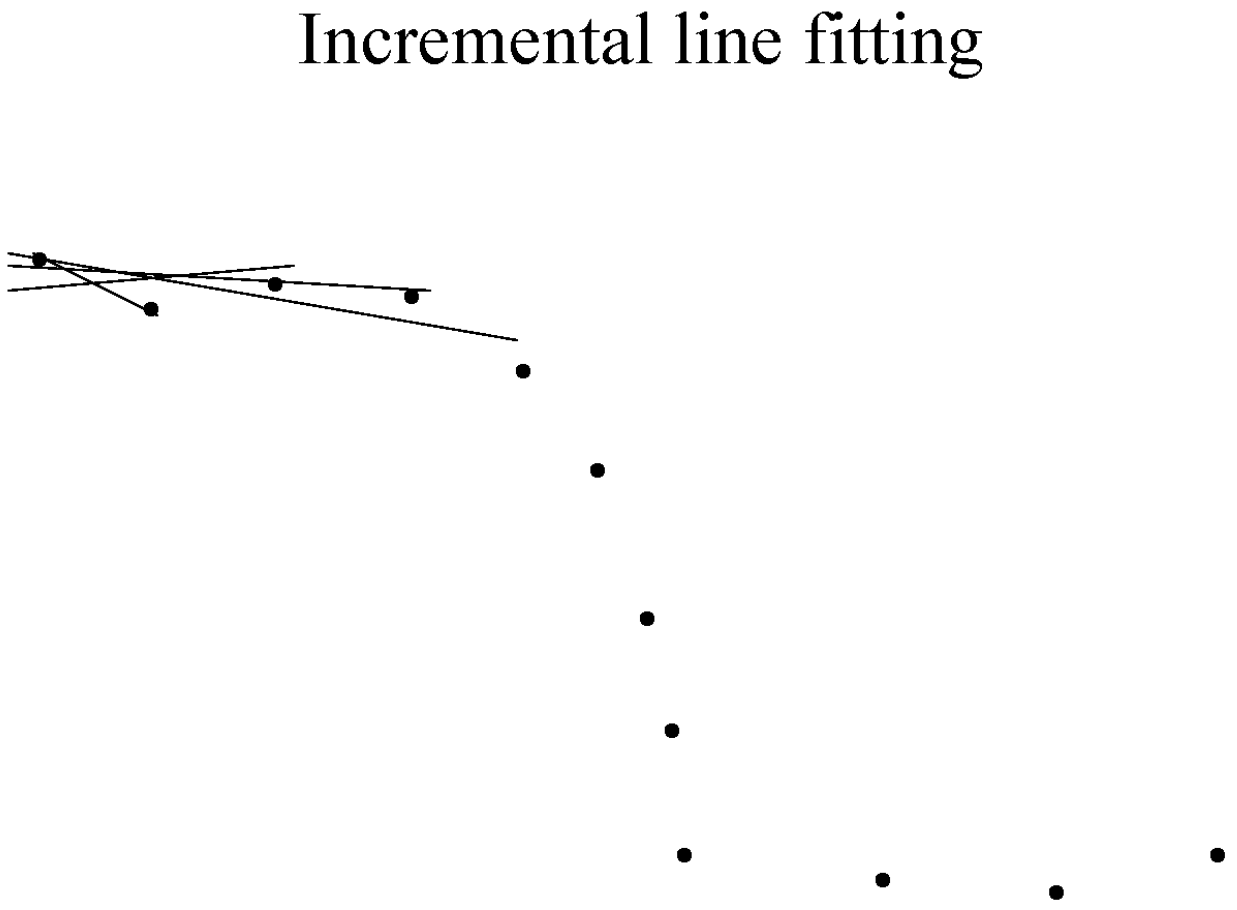
Algorithm 15.1: Incremental line fitting by walking along a curve, fitting a line to runs of pixels along the curve, and breaking the curve when the residual is too large

```
Put all points on curve list, in order along the curve
Empty the line point list
Empty the line list
Until there are too few points on the curve
   Transfer first few points on the curve to the line point list
   Fit line to line point list
   While fitted line is good enough
      Transfer the next point on the curve
         to the line point list and refit the line
   end
   Transfer last point(s) back to curve
   Refit line
   Attach line to line list
end
```

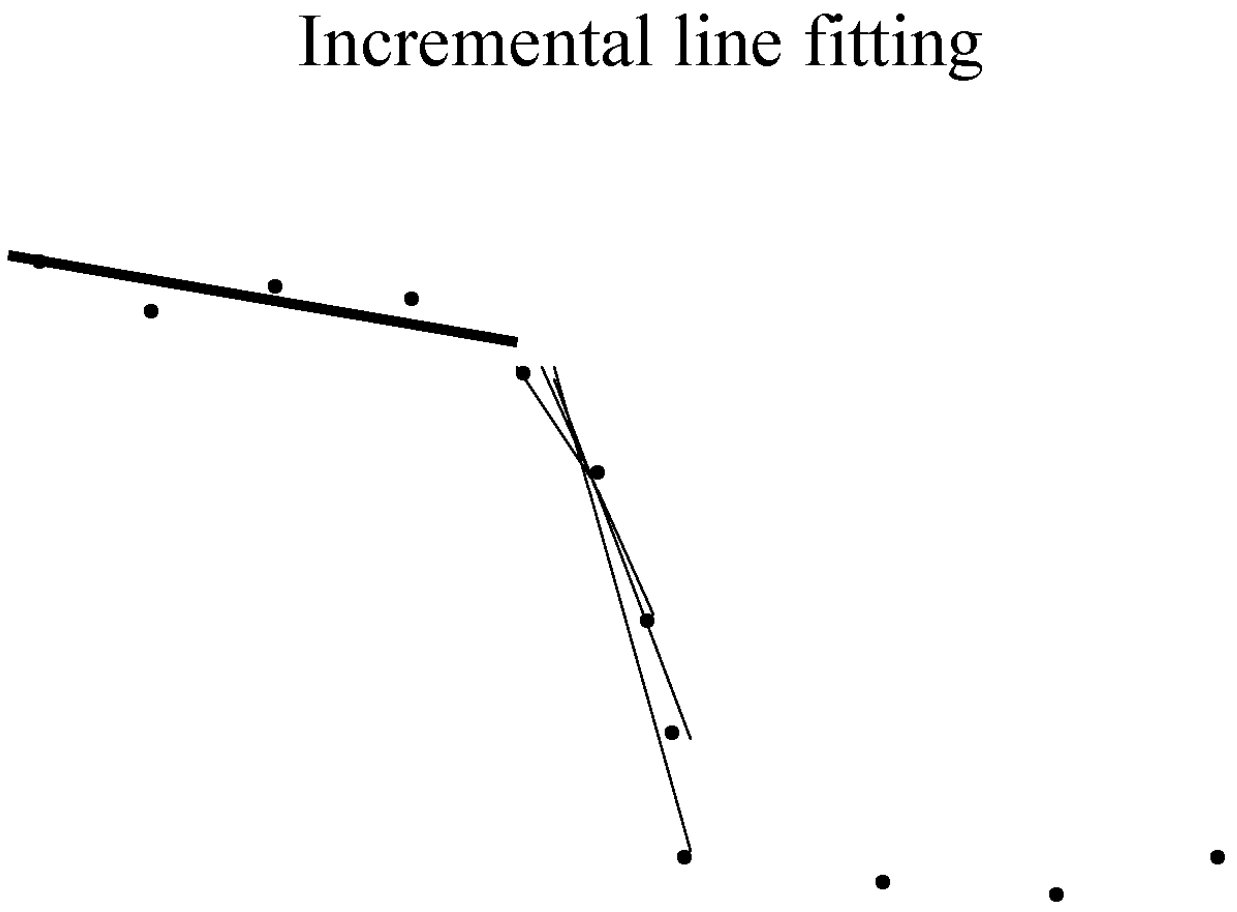
Incremental line fitting (cont.)



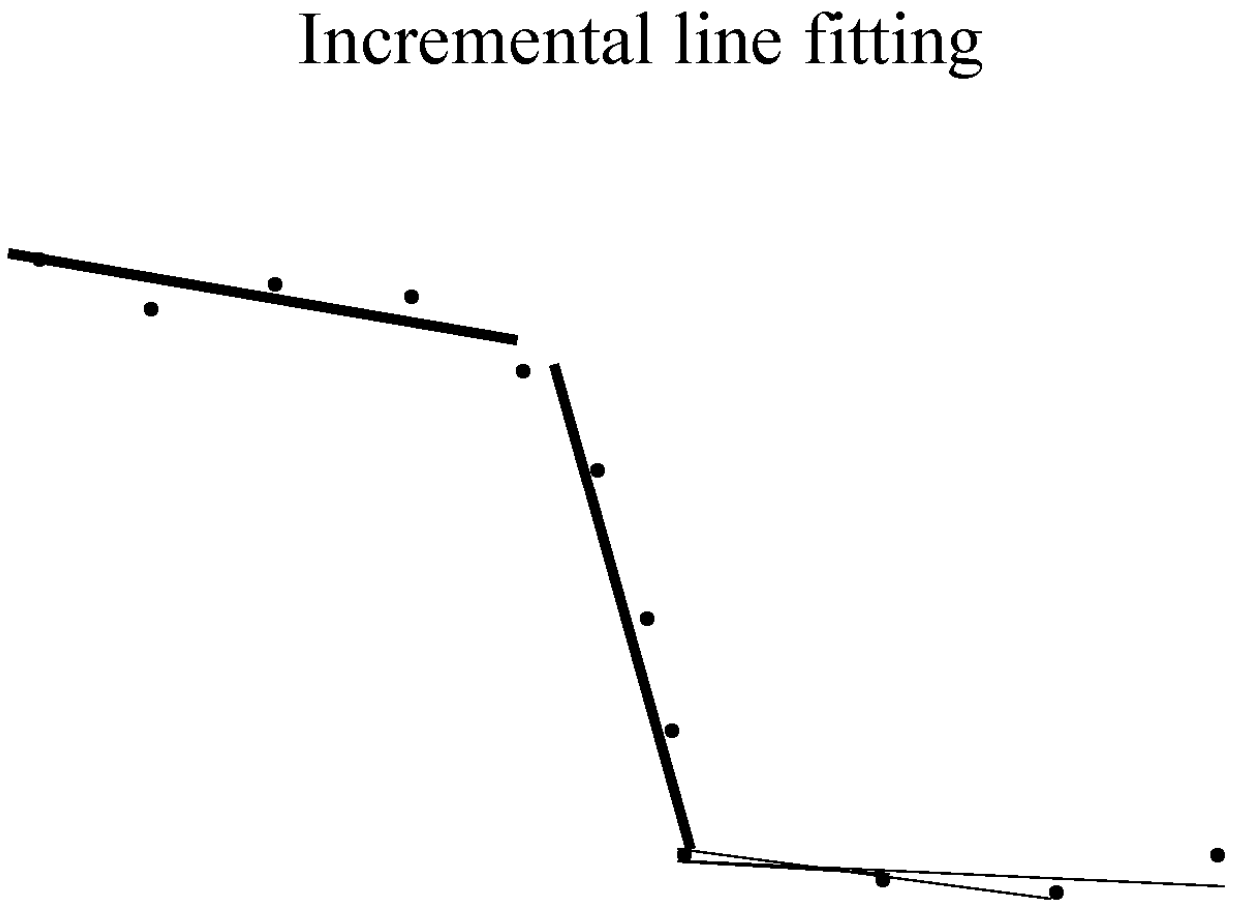
Incremental line fitting (cont.)



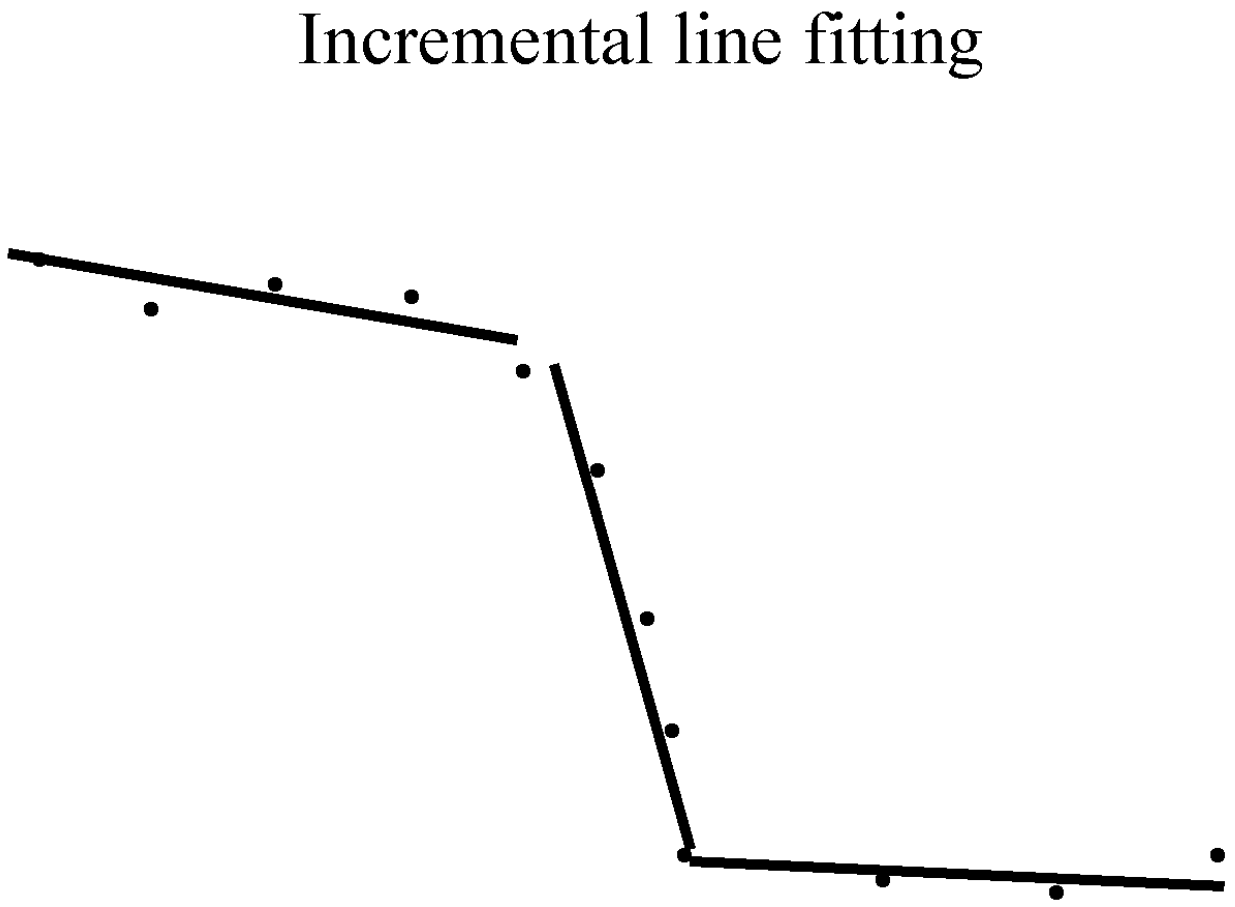
Incremental line fitting (cont.)



Incremental line fitting (cont.)



Incremental line fitting (cont.)



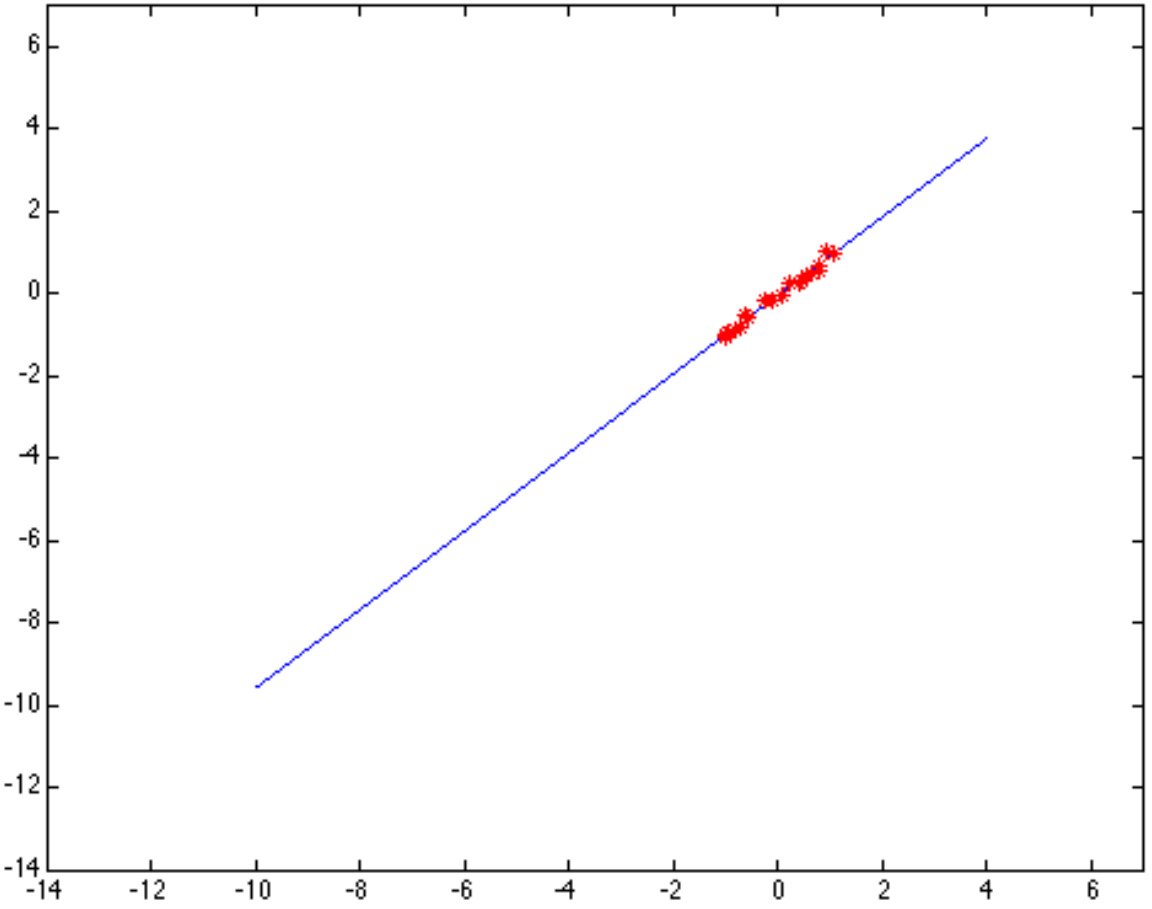
Allocating points to lines by K-means

Algorithm 15.2: K-means line fitting by allocating points to the closest line and then refitting.

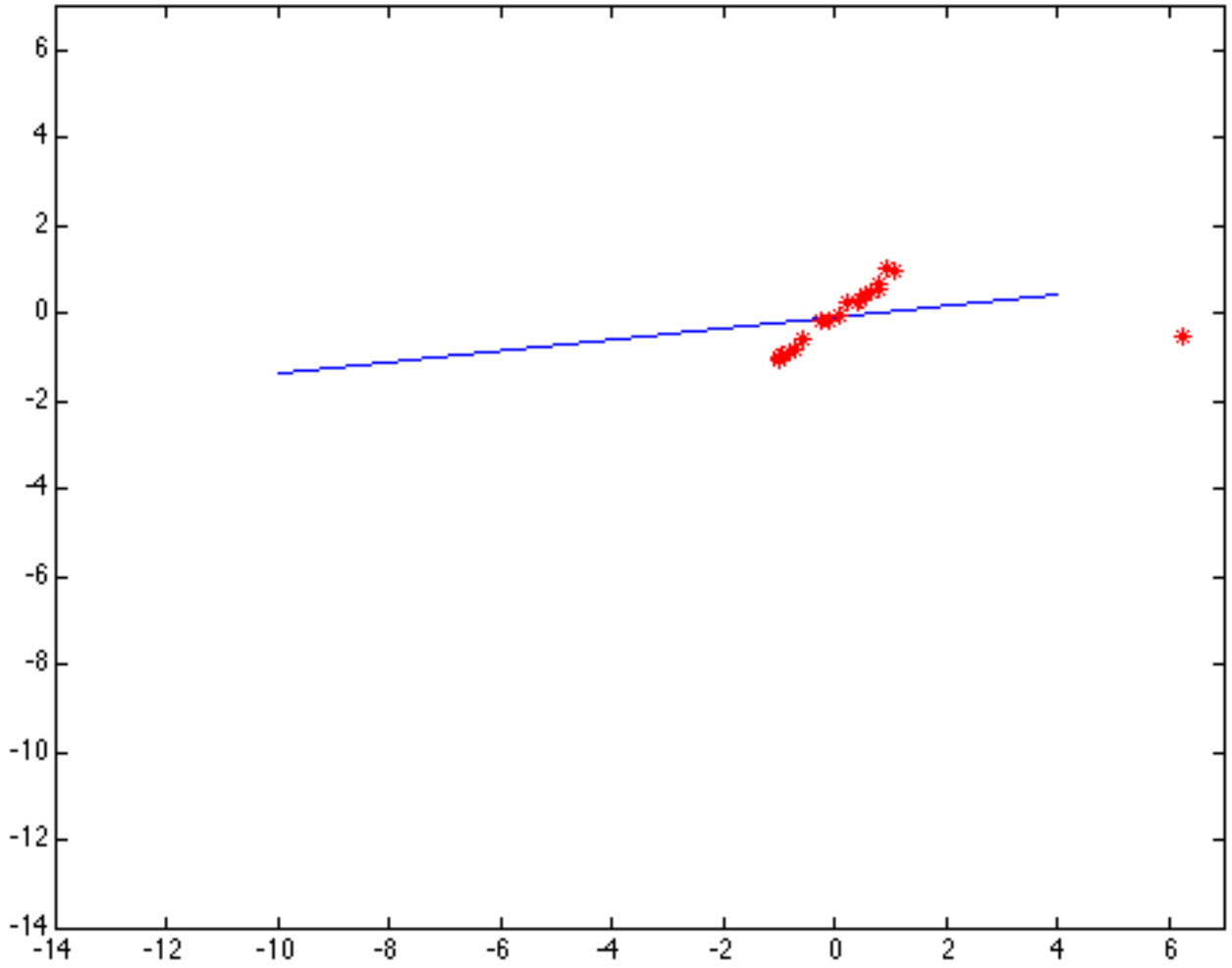
Hypothesize k lines (perhaps uniformly at random)
or
Hypothesize an assignment of lines to points
  and then fit lines using this assignment

Until convergence
  Allocate each point to the closest line
  Refit lines
end

Least squares and outliers

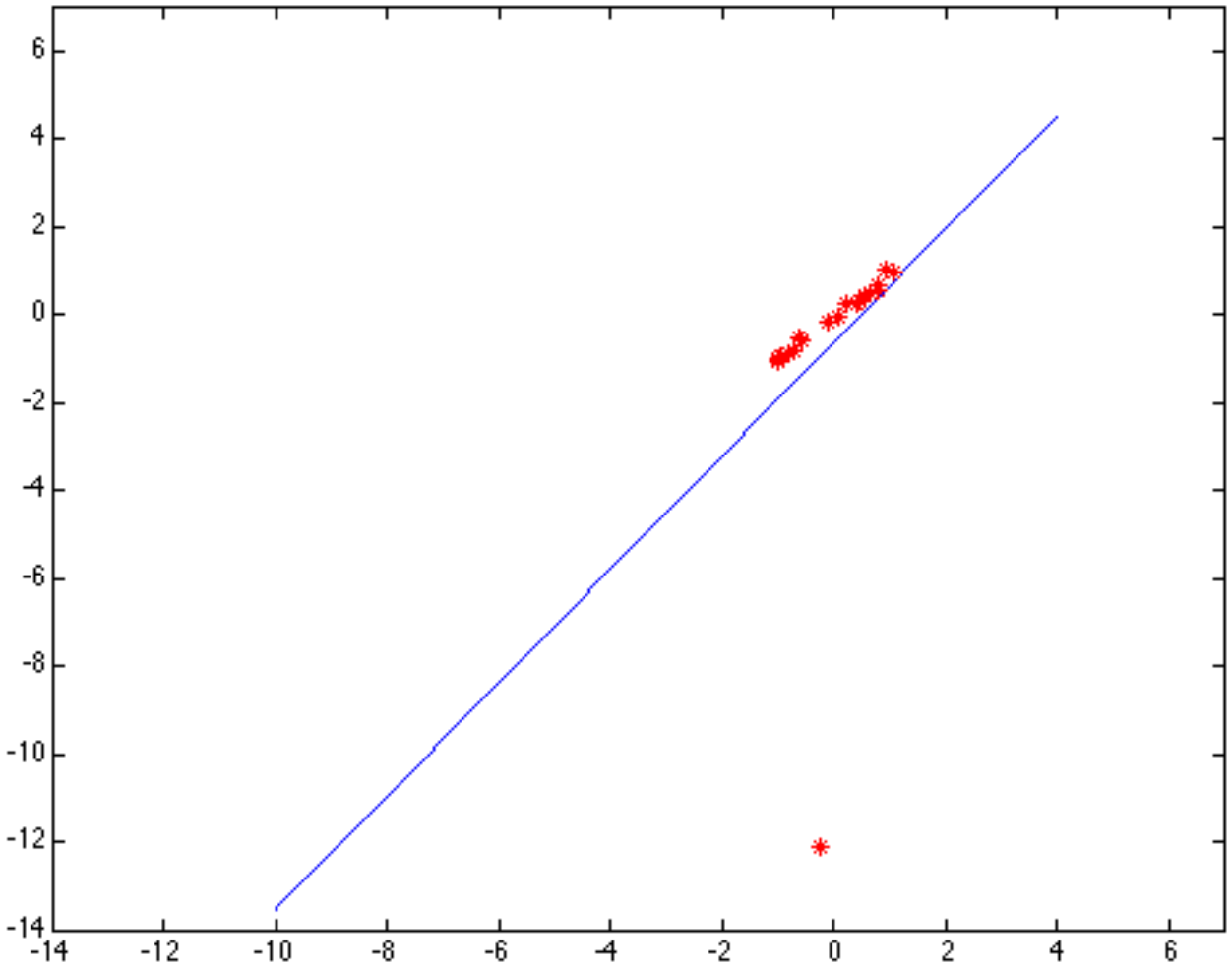


Least squares and outliers (cont.)



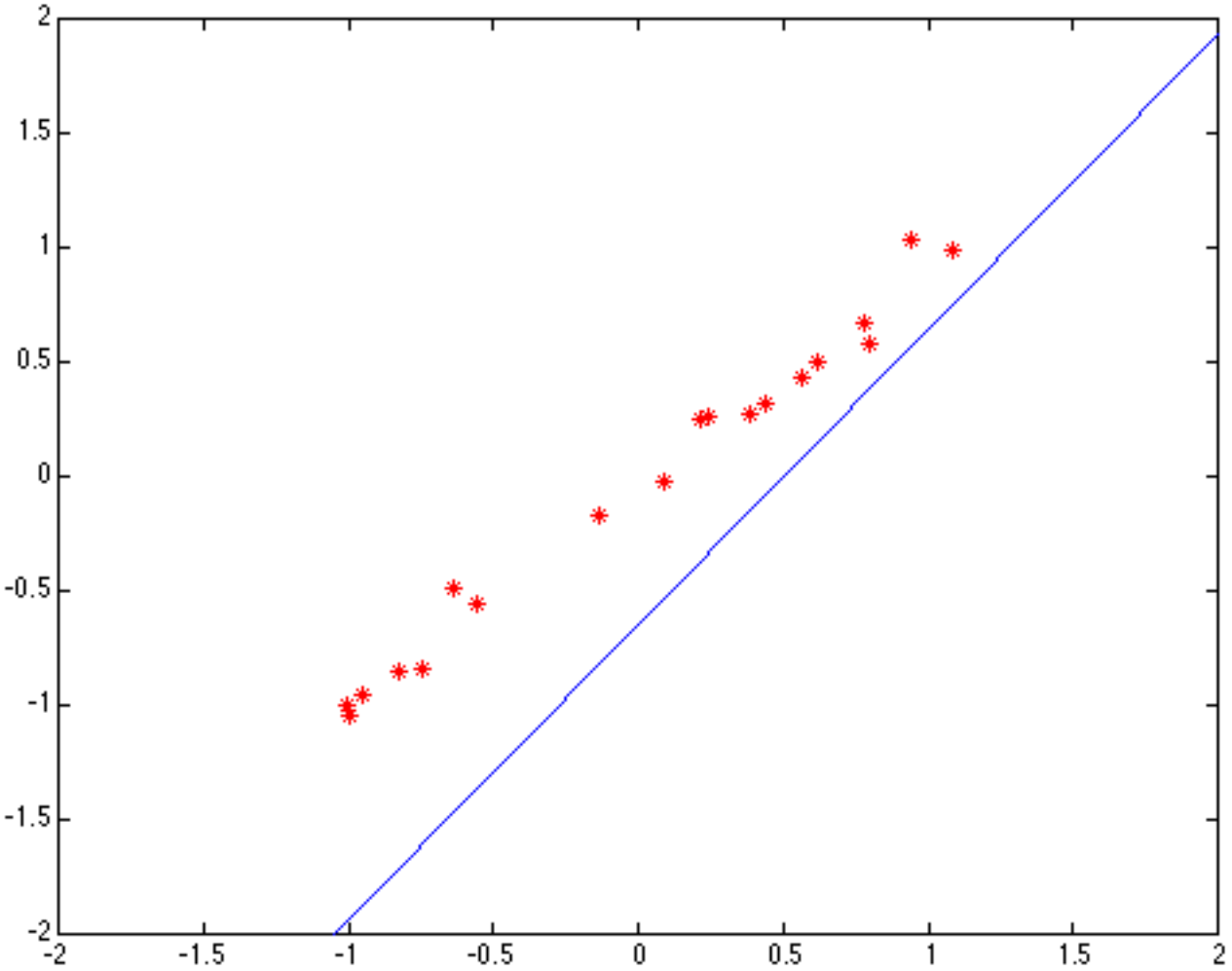
Problem: squared error heavily penalizes outliers.

Least squares and outliers (cont.)



The y-coordinate of a point corrupted. The error seems small…

Least squares and outliers (cont.)



Zoom into the previous example to highlight the error.

Robust estimators

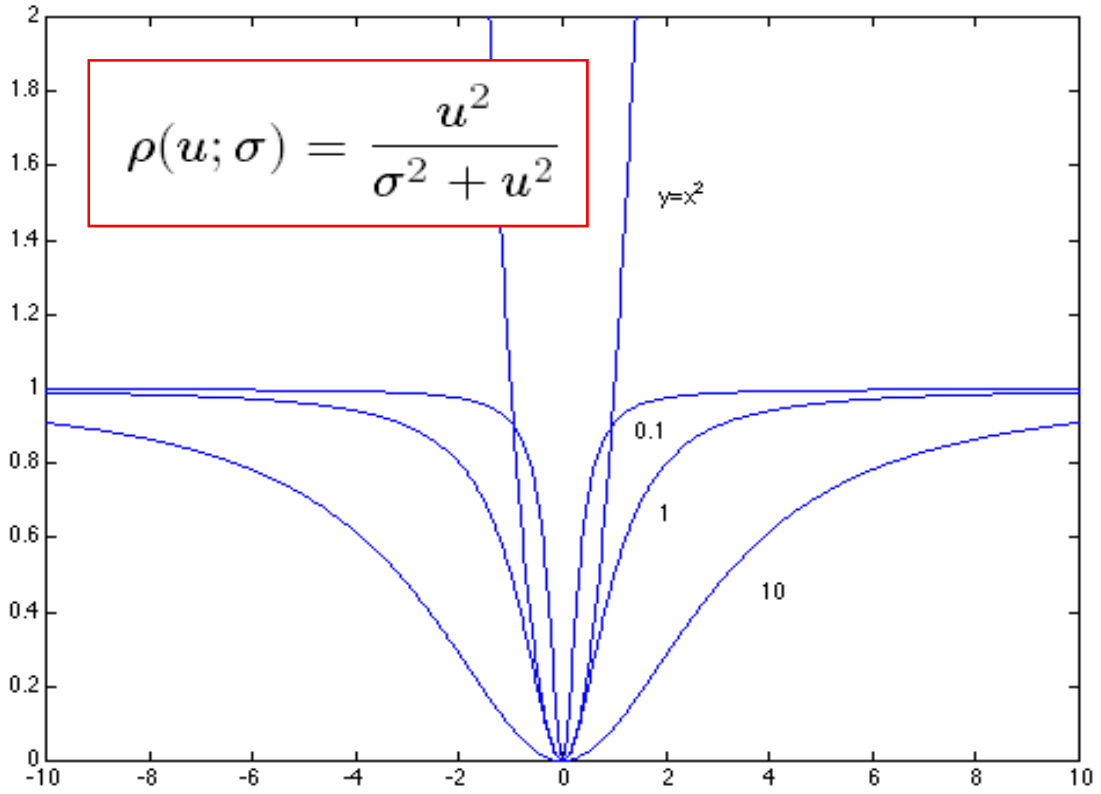
- In the least squares case, we minimize the residual $r(x_i, \theta)$ with respect to the model parameters θ:

$$\min_{\theta} \sum_{i} r^2\left(x_i; \theta\right)$$

- Outliers contribute too much to the estimation of the parameters θ.
- A solution is to replace the square with another function less influenced by outliers.
- This function should behave like the square for "good" points.

Robust estimators (cont.)

$$\min_{\theta} \sum_{i} \rho\left(r\left(x_i, \theta\right); \sigma\right)$$



The robust function ρ behaves like a squared distance for small values of its argument but saturates for larger values of it.

Notice the role of parameter σ. We have to select it carefully.

Robust estimators (cont.)

- Estimation of the model parameters is now obtained by:

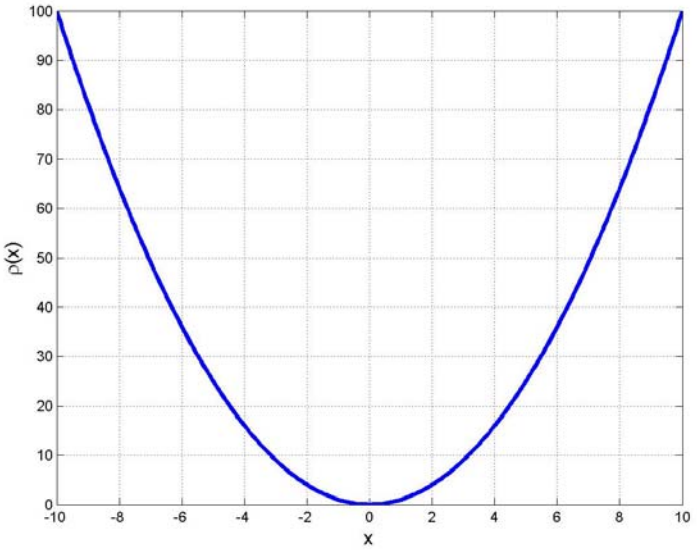
$$\sum_i \frac{\partial \rho\left(r\left(x_i,\theta\right);\sigma\right)}{\partial \theta} = 0 \Leftrightarrow \sum_i \frac{\partial \rho\left(r\left(x_i,\theta\right);\sigma\right)}{\partial r}\frac{\partial r\left(x_i,\theta\right)}{\partial \theta} = 0$$

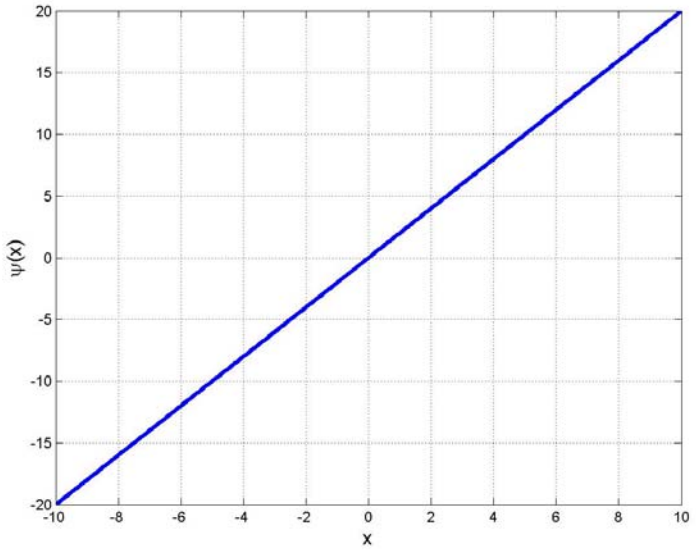
$$\Leftrightarrow \sum_i \psi\left(r\left(x_i,\theta\right);\sigma\right)\frac{\partial r\left(x_i,\theta\right)}{\partial \theta} = 0, \quad \text{with} \quad \psi(x;\sigma) = \frac{d\,\rho(x;\sigma)}{dx}$$

where $\psi(x;\sigma)$ is the influence function of the robust estimator characterizing the influence of the residuals to the solution.

Robust estimators (cont.)

Standard least squares estimator and its influence function



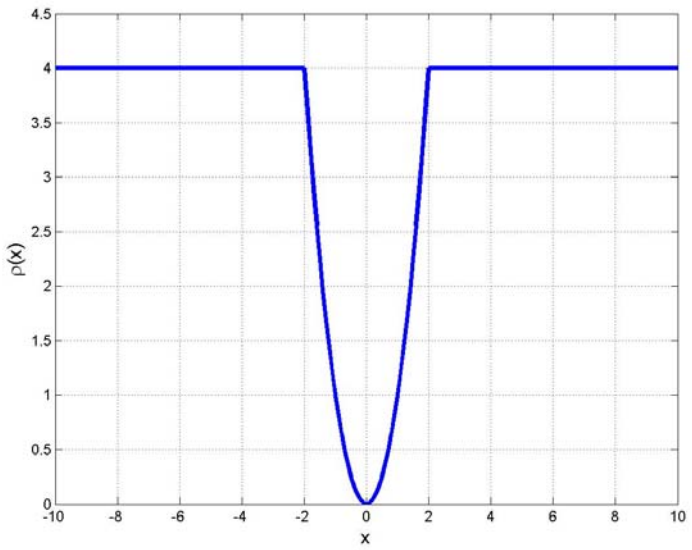


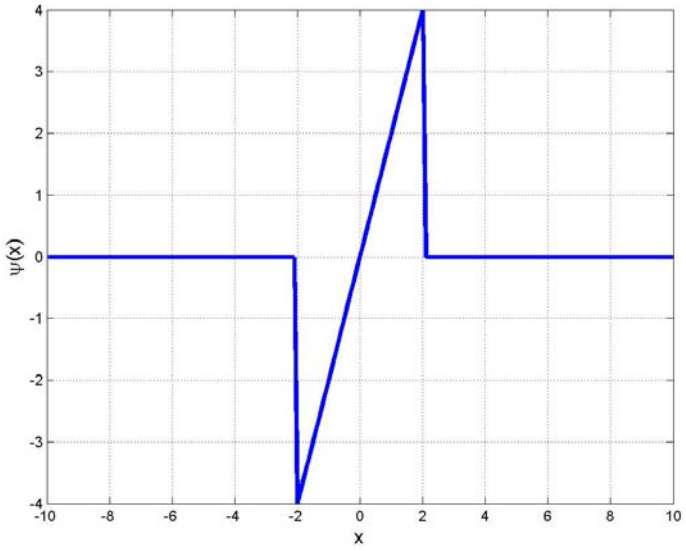
$$\rho(x) = x^2$$

$$\psi(x) = 2x$$

Robust estimators (cont.)

Truncated least squares estimator and its influence function



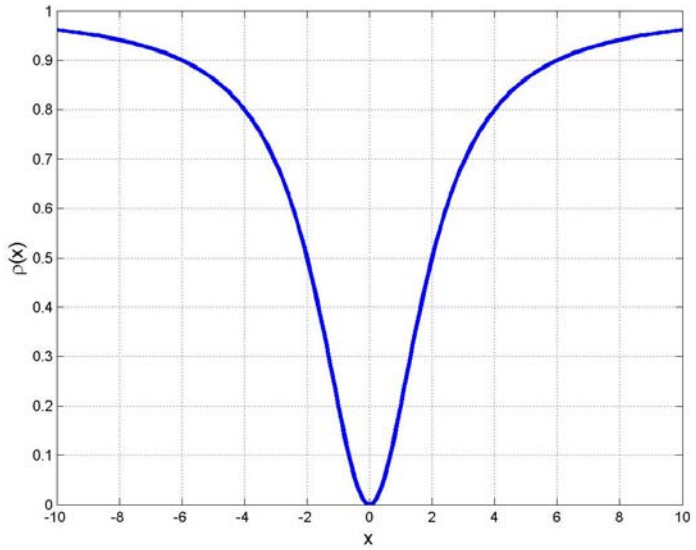


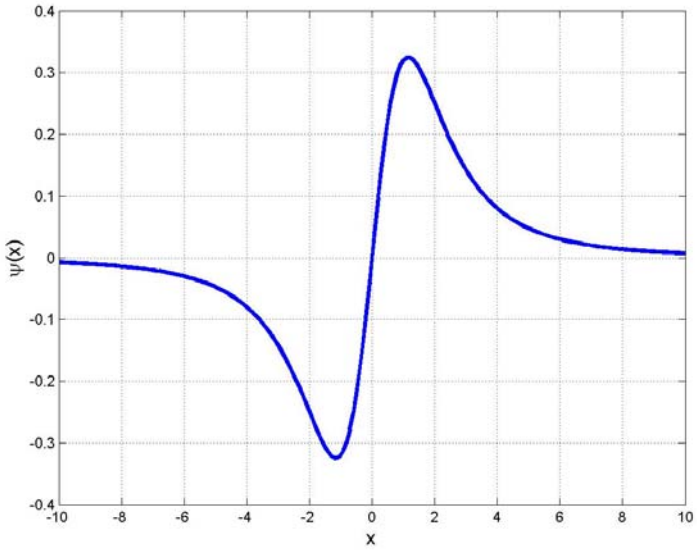
$$\rho(x) = \begin{cases} x^2, & \text{if } |x| \leq \sigma \\ \sigma^2, & \text{if } |x| > \sigma \end{cases}$$

$$\psi(x) = \begin{cases} 2x, & \text{if } |x| \leq \sigma \\ 0, & \text{if } |x| > \sigma \end{cases}$$

Robust estimators (cont.)

Geman-McClure estimator and its influence function

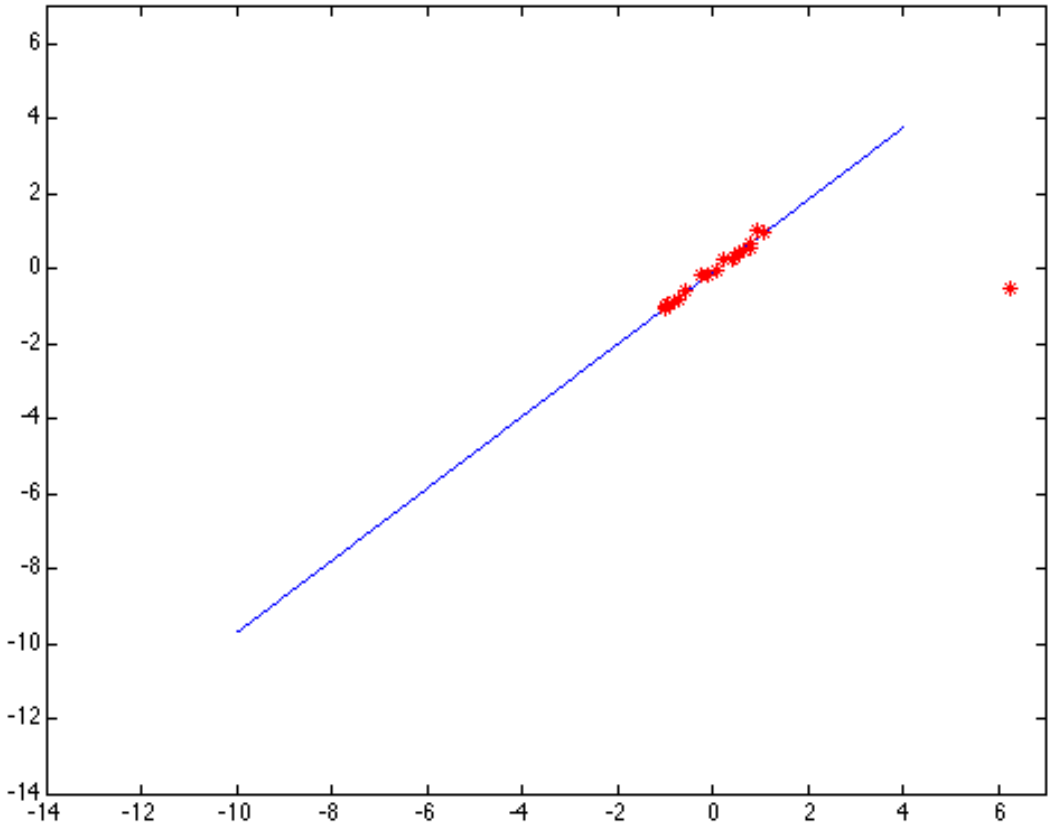




$$\rho(x) = \frac{x^2}{\sigma^2 + x^2}$$

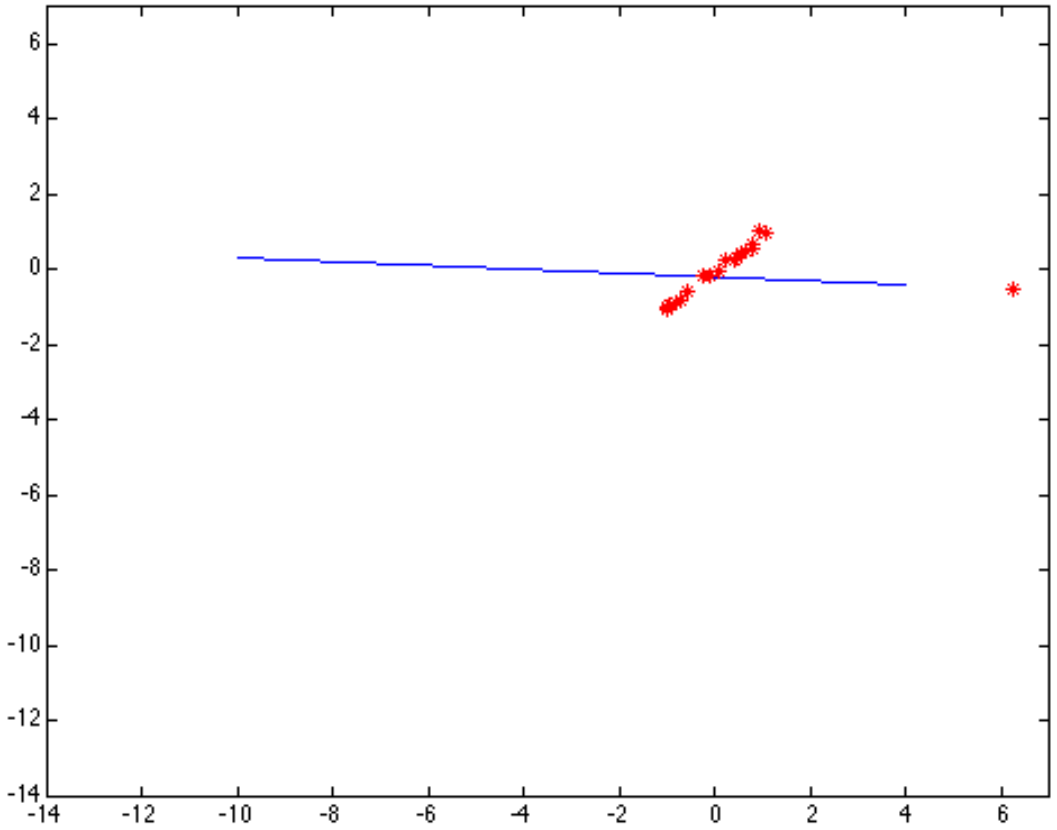
$$\psi(x) = \frac{2\sigma^2 x}{(\sigma^2 + x^2)^2}$$

Robust estimators (cont.)



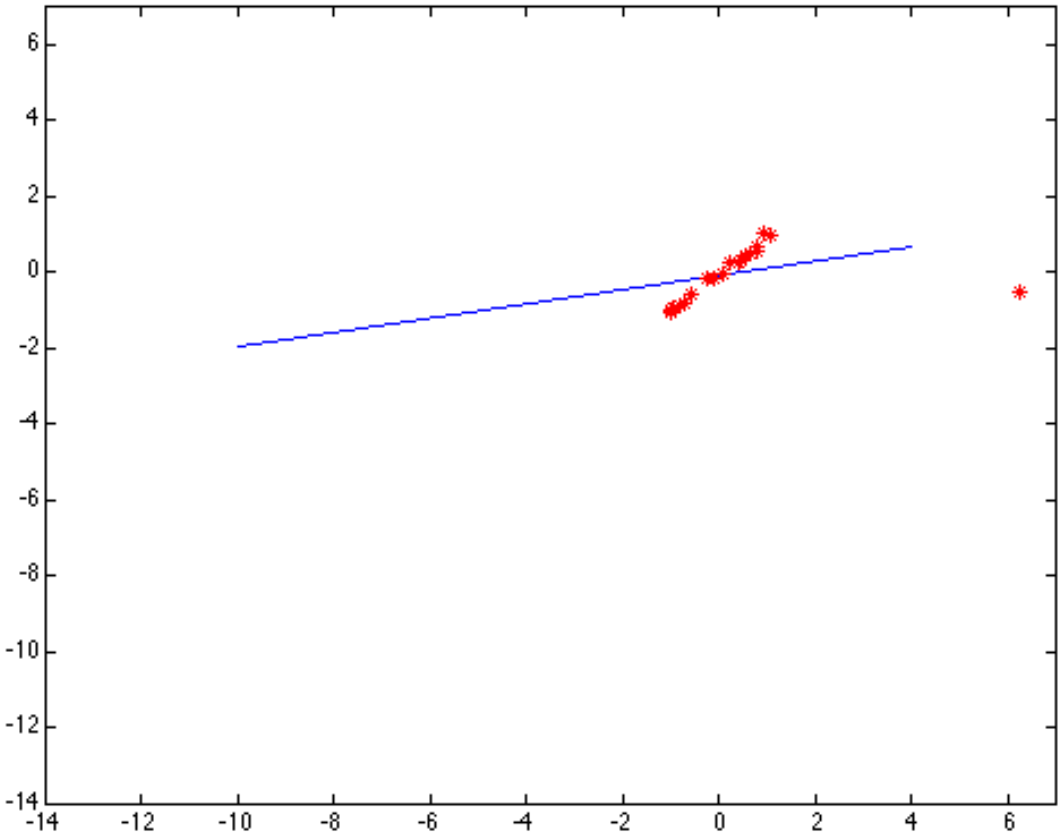
Correct selection of σ: the effect of the outlier is eliminated

Robust estimators (cont.)



Small σ: the error value is almost the same for every point and the fit is very poor. All the points are considered as outliers.

Robust estimators (cont.)



Large σ: the error value is like least squares and the fit is very poor. All the points are considered as “good”.

Robust estimation (cont.)

- Robust fitting is a nonlinear optimization problem that must be solved iteratively.
- Least squares solution can be used for initialization.
- Adaptive choice of scale base on the median absolute deviation (MAD) of the residuals at each iteration (t):

$$MAD^{(t)} = \underset{i}{\text{median}} \left\{ r\left(x_i; \theta^{(t)}\right) - \underset{k}{\text{median}} \left\{ r\left(x_k; \theta^{(t)}\right) \right\} \right\}$$

$$\sigma^{(t)} = 1.4826 \cdot MAD^{(t)}$$

- It is based on the MAD of normally distributed points with standard deviation of 1.

P. Rousseeuw and A. Leory. Robust regression and outlier detection. J. Wiley 1987.

RANSAC

- Robust fitting can deal with a few outliers – what if we have many of them?
- Random sample consensus (RANSAC)
  - 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 and R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM, Vol 24, pp 381-395, 1981.

RANSAC (cont.)

- Insight
  - Search for “good”: points in the data set.
  - Suppose we want to fit a line to a point set degraded by at 50% by outliers.
  - Choosing randomly 2 points has a chance of 0.25 to obtain “good” points.
  - We can identify them by noticing that a many other points will be close to the line fitted to the pair.
  - Then we fit a line to all these “good” points.

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

RANSAC (cont.)

- Repeat k times:
  - Draw n points uniformly at random.
  - Fit line to these n points.
  - Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than t).
  - If there are d or more inliers, accept the line (and the consensus set) and refit to all inliers.
- Select the largest consensus set.

Parameters k, n, t and d are crucial.

RANSAC (cont.)

- Number of samples n and number of iterations k.
  - The minimum number to fit the model (2 points for lines).
  - If w is the fraction of "good" points, the expected number of draws required to get a "good" sample is:

$$E[\boldsymbol{k}]=1 \text{ x p(one good sample in one draw)}$$
$$+ \quad 2 \text{ x p(one good sample in two draws)}$$
$$+ \ldots = w^n + 2(1-w^n)\, w^n + 3(1-w^n)^2\, w^n + \ldots = w^{-n}.$$

We add a few standard deviations to this number:

$$\sigma_k = \frac{\sqrt{1-w^n}}{w^n}$$

RANSAC (cont.)

- Number of iterations k (alternative approach).
  - We look for a number of samples that guarantees a that at least one of them is free from outliers with a probability of p.
  - Probability of seeing a bad sample of points in one draw:

$$(1 - w^n)$$

  - and the probability of seeing only bad samples in k draws is:

$$1 - p = (1 - w^n)^k$$

Solving for k:

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$

RANSAC (cont.)

- The number of iterations k required to ensure, with a probability p=0.99, that at least one sample has no outliers for a given size of sample, n, and proportion of outliers 1-w.

$$k = \frac{\log(1-p)}{\log(1-w^n)}$$

	proportion of outliers 1-w						
n	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

RANSAC (cont.)

- Threshold t to decide if a point is close to the line or not.
  - We generally specify it as a part of the modeling process.
  - The same as in the maximum likelihood formulation of line fitting (parameter σ).
  - Look similar data sets, fit some lines by eye and determine it empirically.

RANSAC (cont.)

- Number of points d that must be close to the line (the size of the consensus set).
  - A rule of thumb is to terminate if the size of the consensus set is similar to the number of inliers believed to be in the data set, given the assumed proportion of outliers.
  - For N data points and if the percentage of inliers is w:

$$d = wN$$

RANSAC (cont.)

- Pros
  - Simple and general.
  - Applicable to many different problems.
  - Often works well in practice.
- Cons
  - Lots of 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.
  - Can fail for extremely low inlier ratios.
  - We can often do better than brute-force sampling.

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

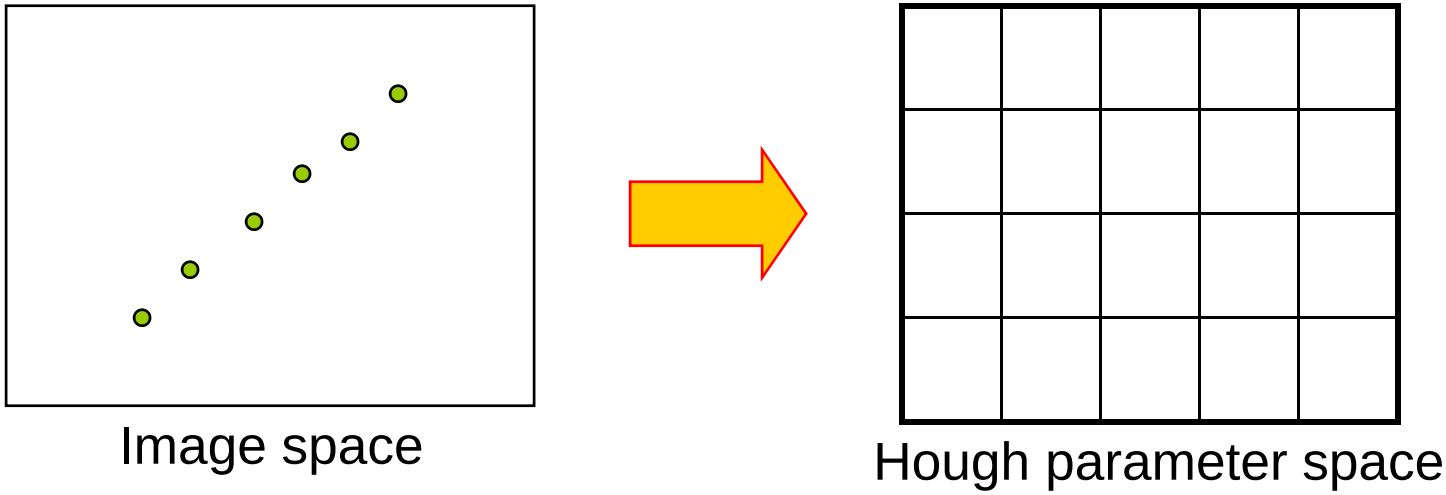


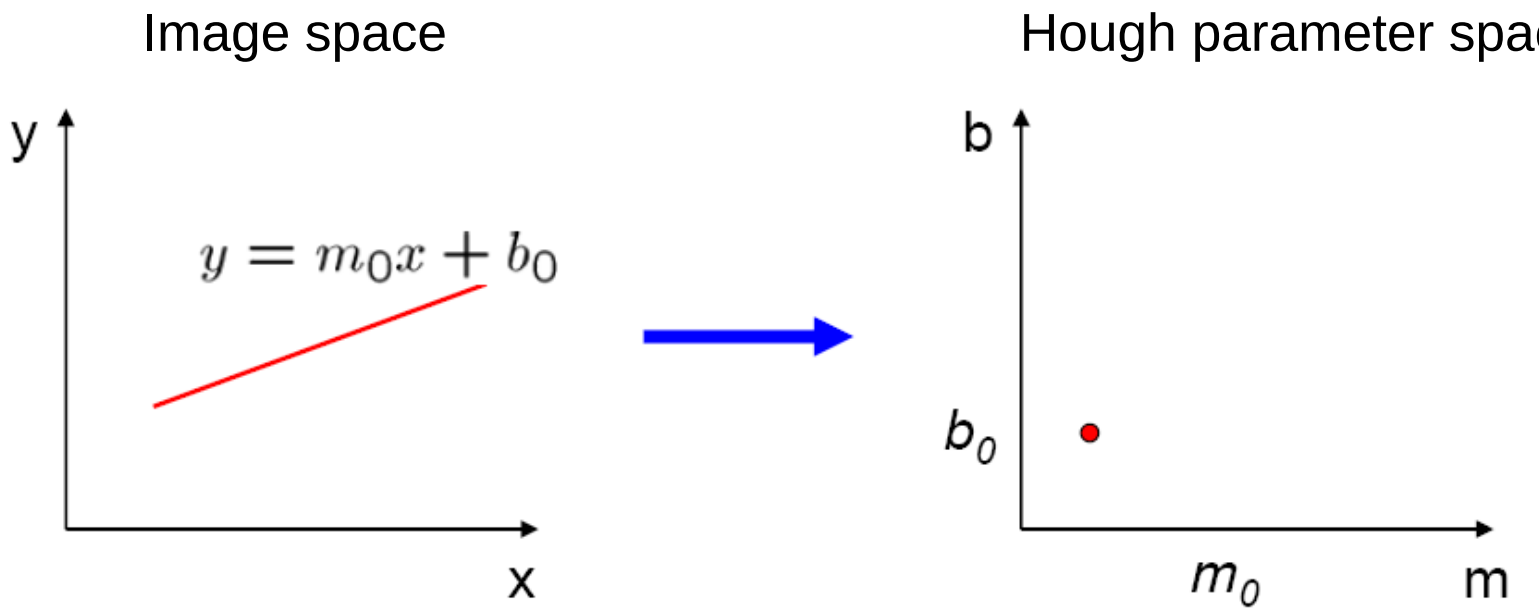
Image space

Hough parameter space

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

Hough transform (cont.)

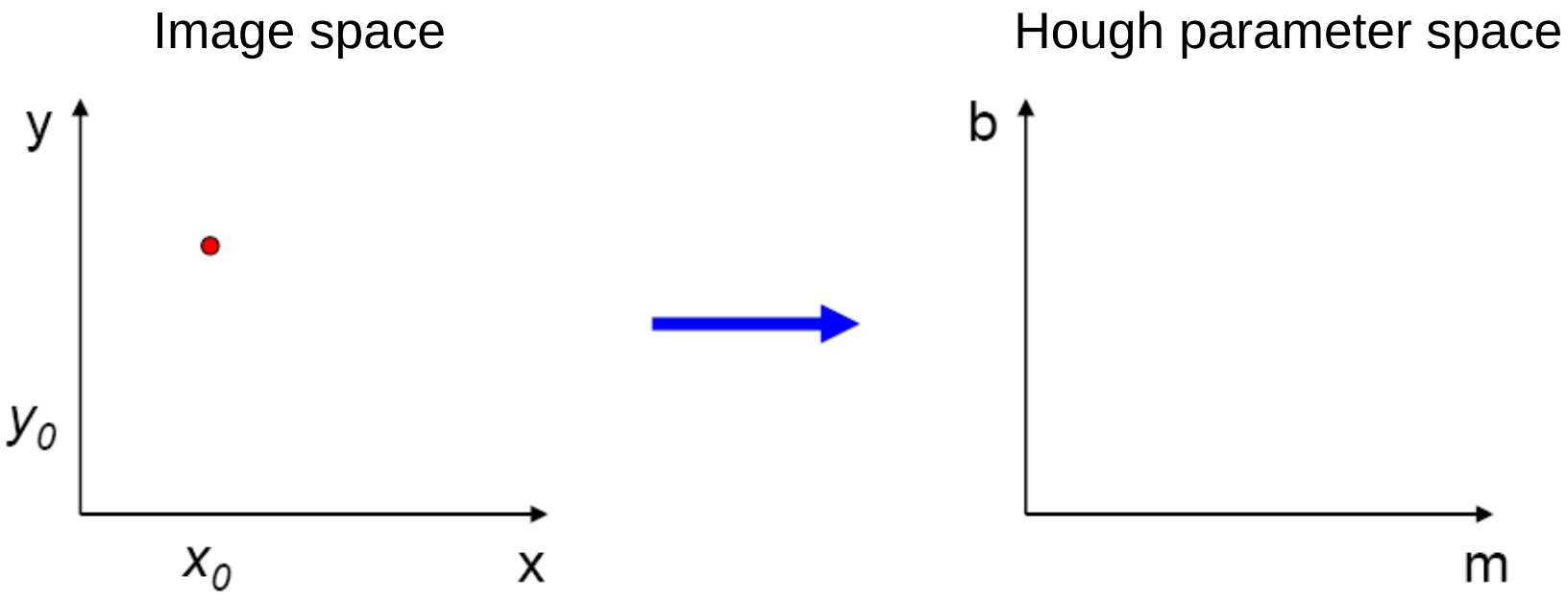
- A line in the image corresponds to a point in Hough space



Source: S. Seitz

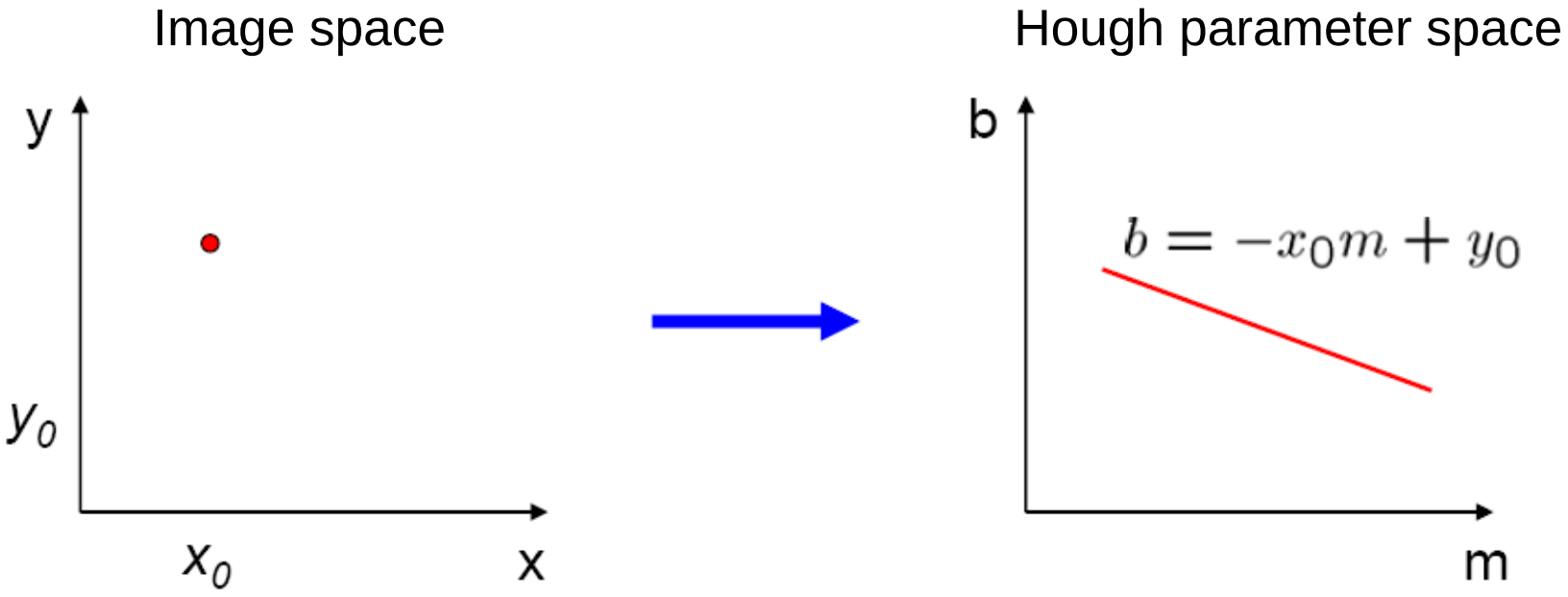
Hough transform (cont.)

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



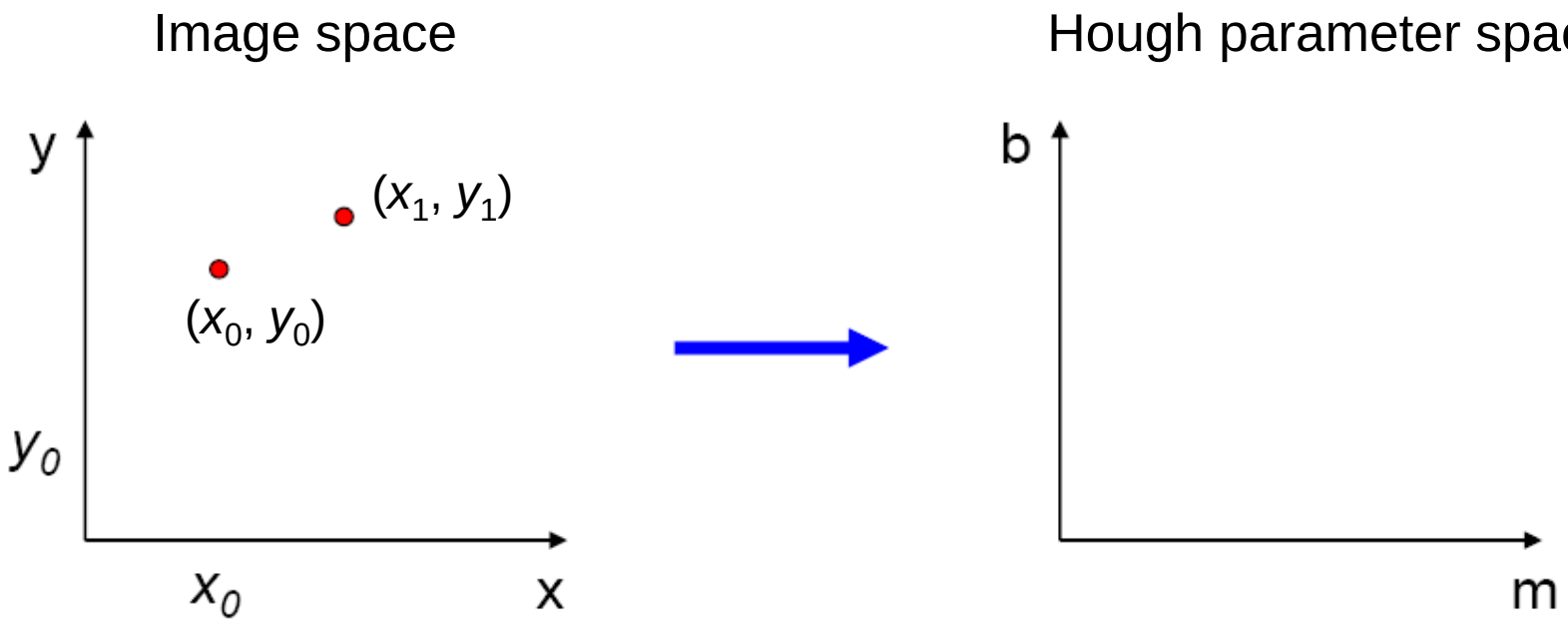
Hough transform (cont.)

- 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



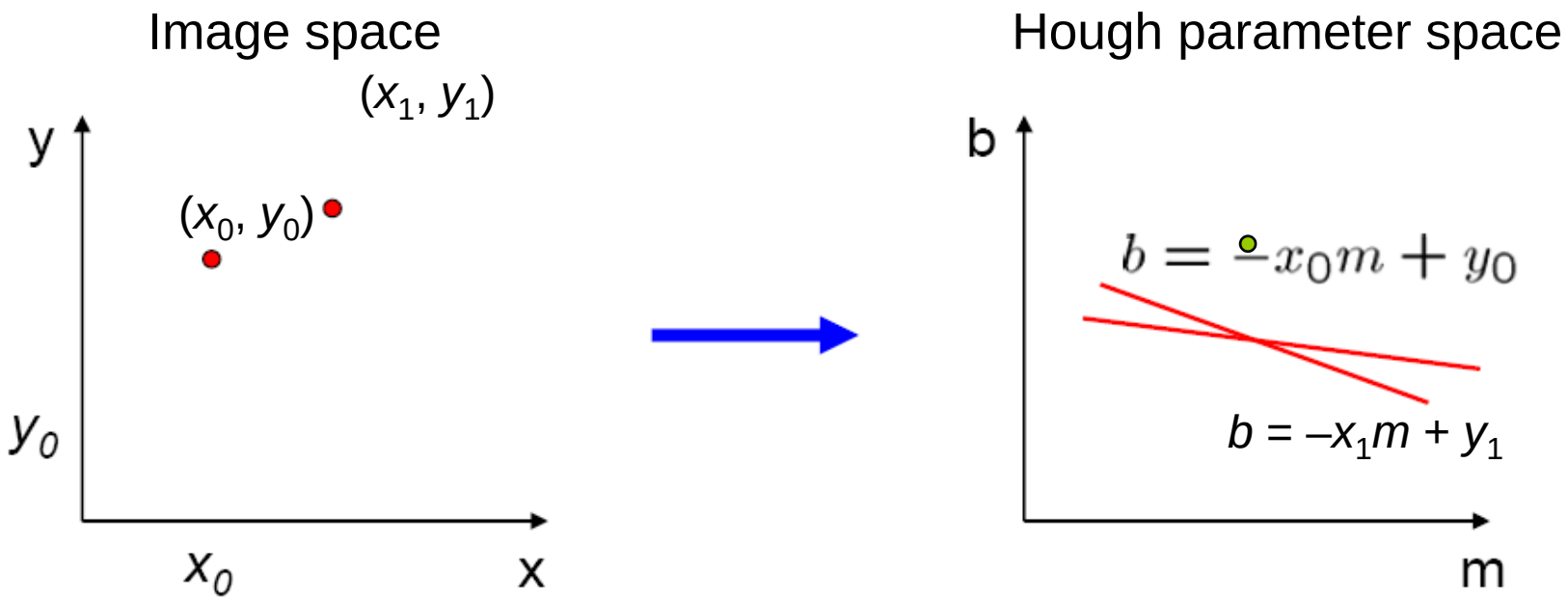
Hough transform (cont.)

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



Hough transform (cont.)

- 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 $b = -x_1m + y_1$

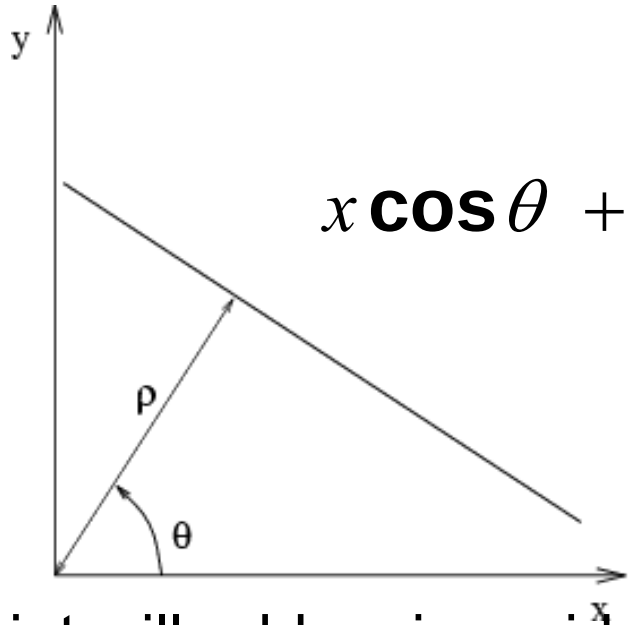


Hough transform (cont.)

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

Hough transform (cont.)

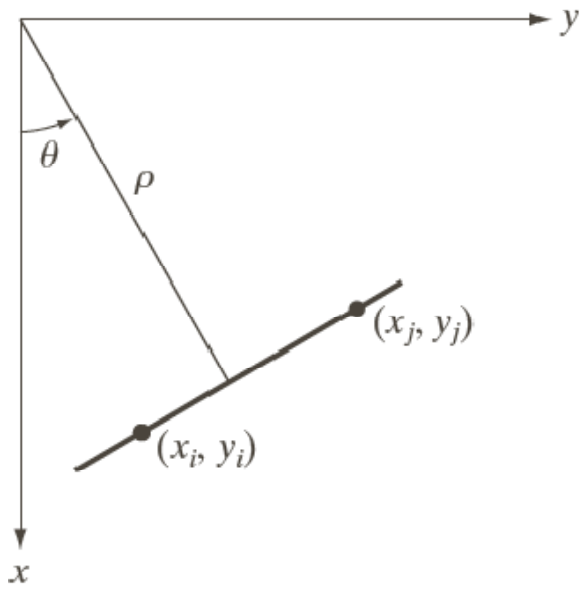
- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m
- Alternative: polar representation

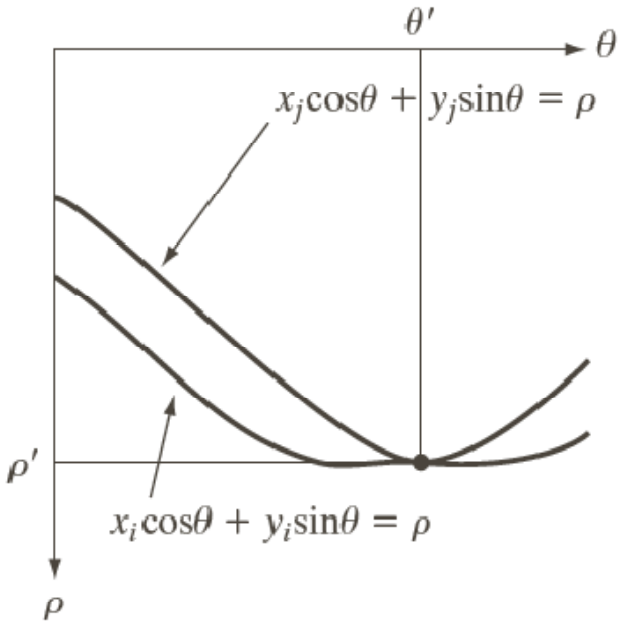


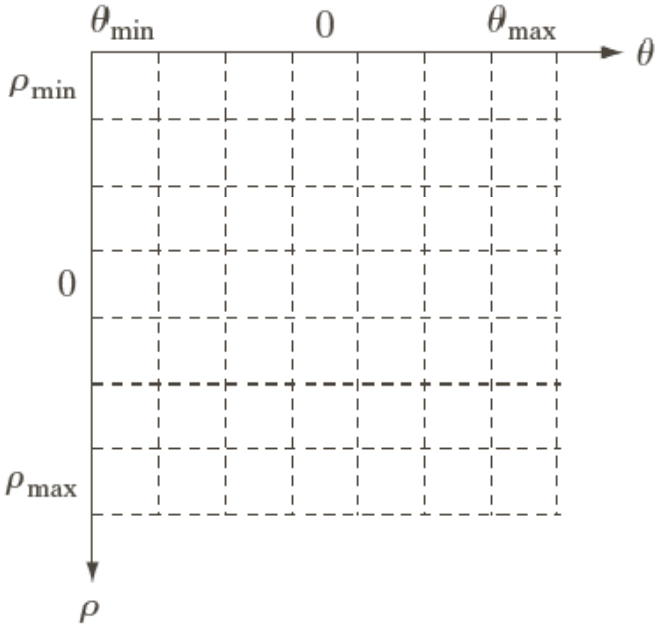
$$x\cos\theta + y\sin\theta = \rho$$

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

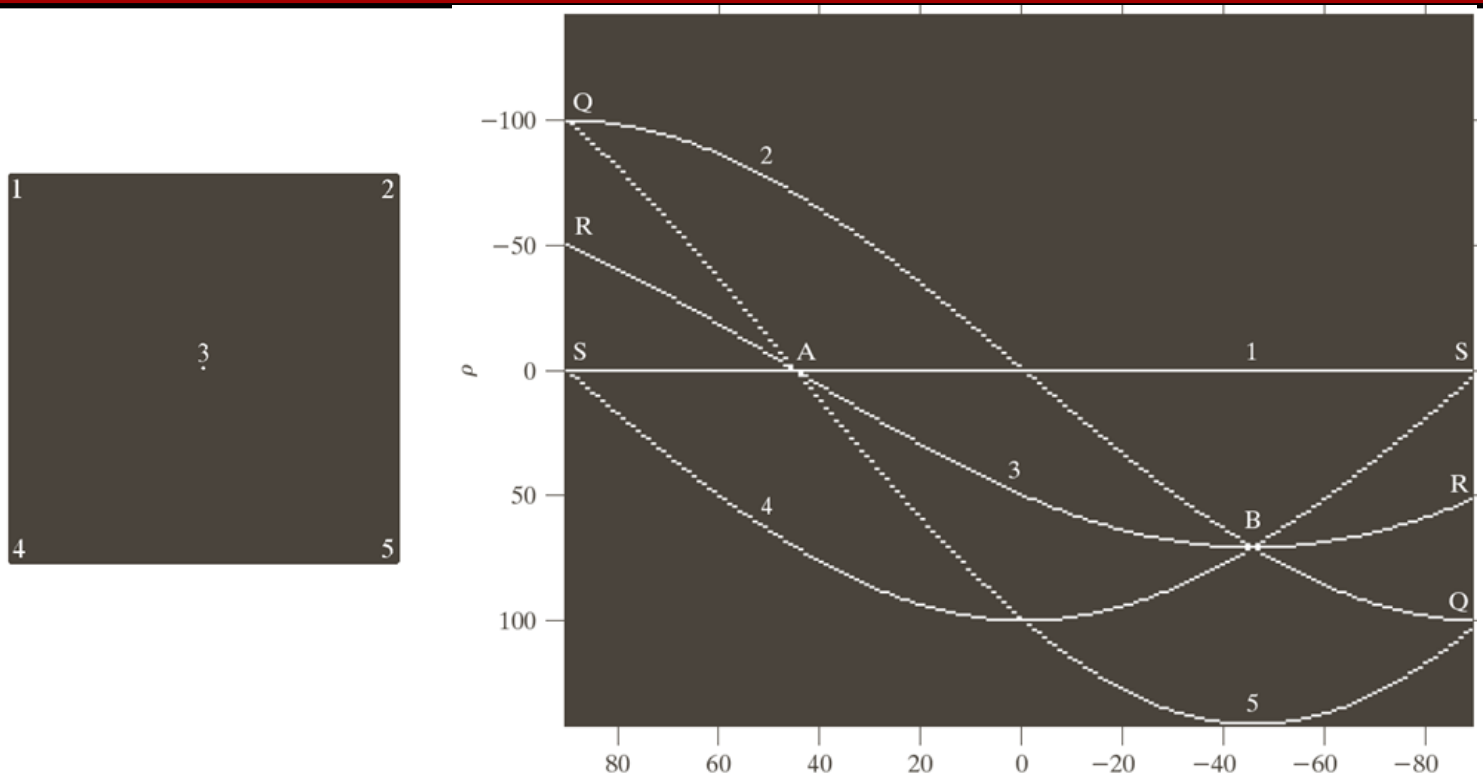
Hough transform (cont.)





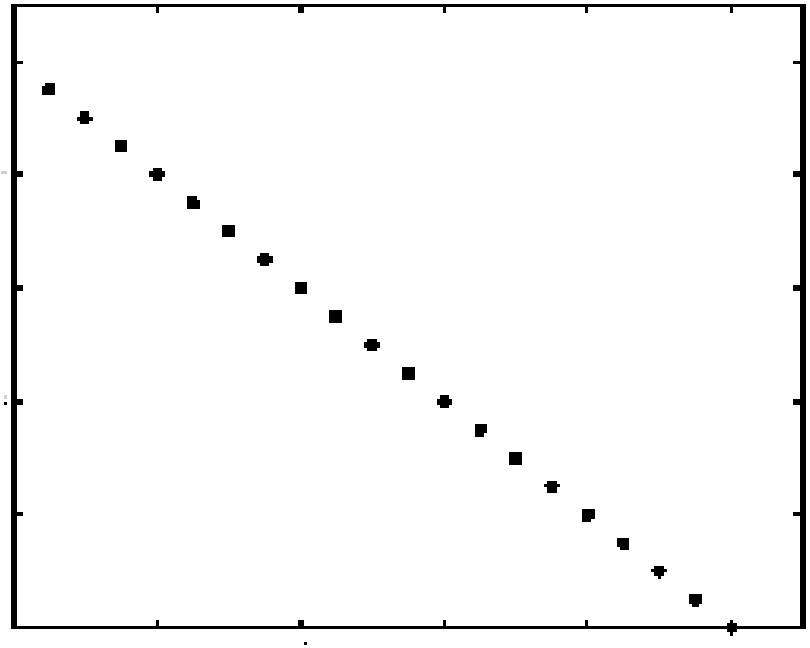


Hough transform (cont.)

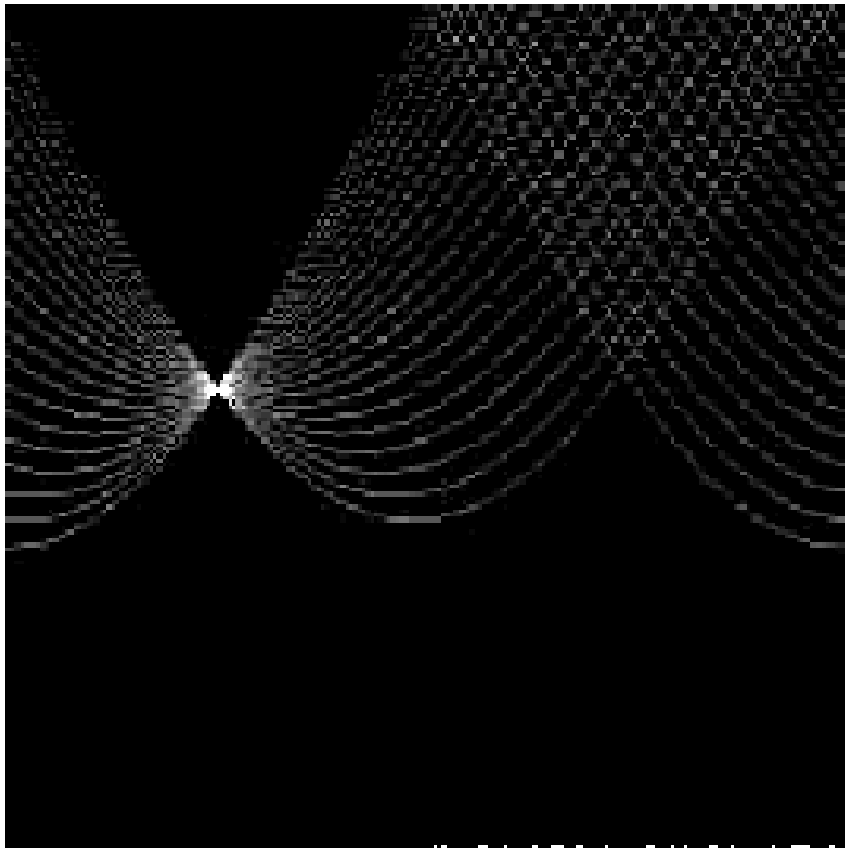


- A: intersection of curves corresponding to points 1, 3, 5.
- B: intersection of curves corresponding to points 2, 3, 4.
- Q, R and S show the reflective adjacency at the edges of the parameter space. They do not correspond to points.

Hough transform (cont.)



features



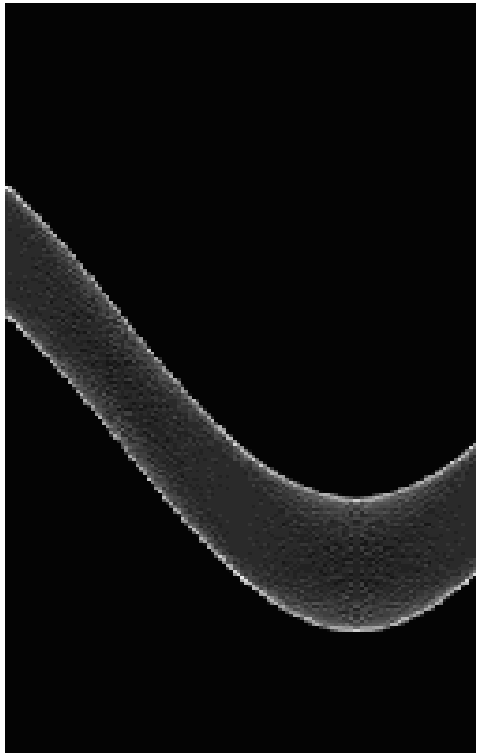
votes

- 20 points drawn from a line and the Hough transform.
- The largest vote is 20.

Hough transform (cont.)

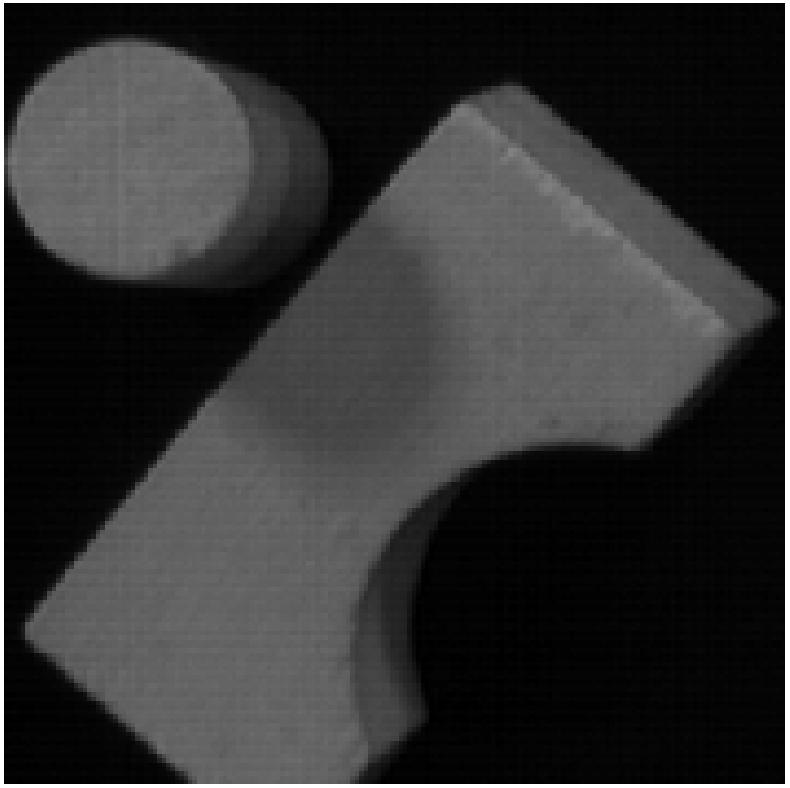
Square

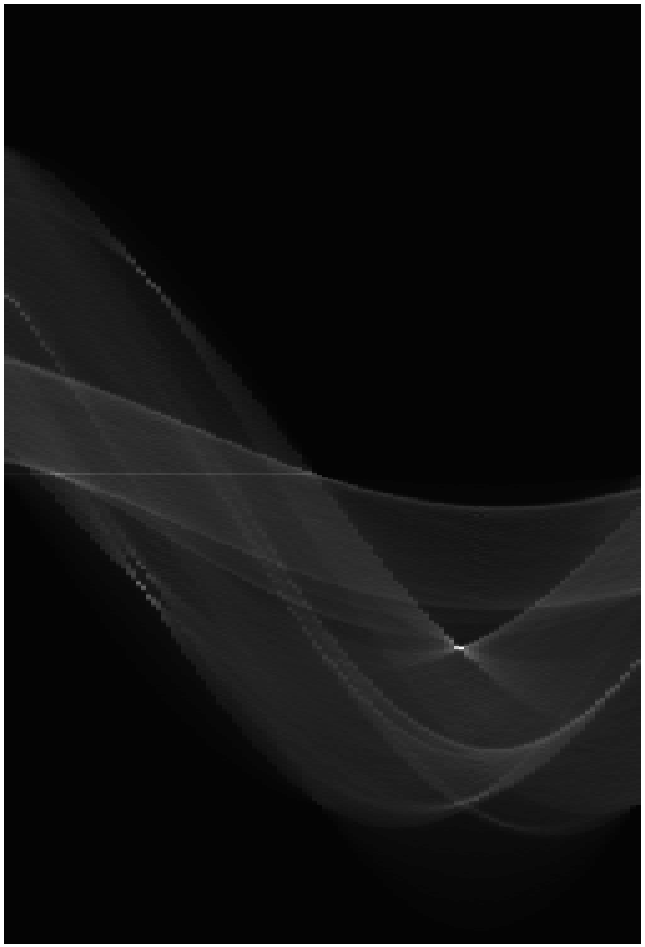
Circle



- The four sides of the square get the most votes.
- In the circle, all of the lines get two votes and the tangent lines get one vote.

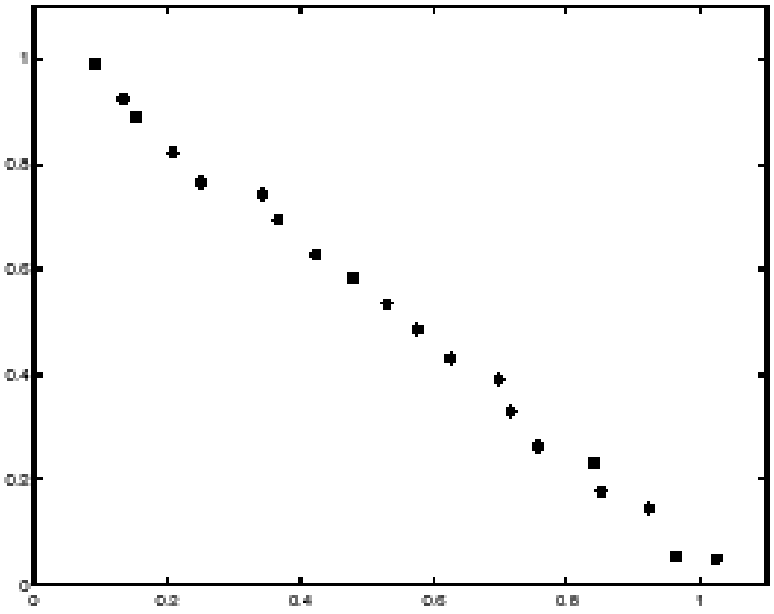
Hough transform (cont.)



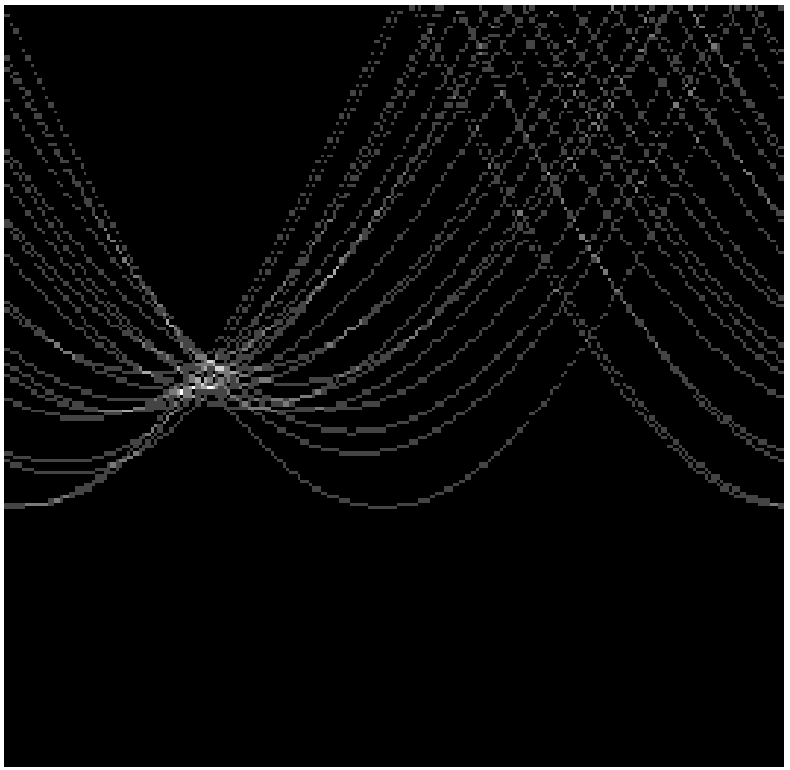


- Several lines

Hough transform (cont.)



features

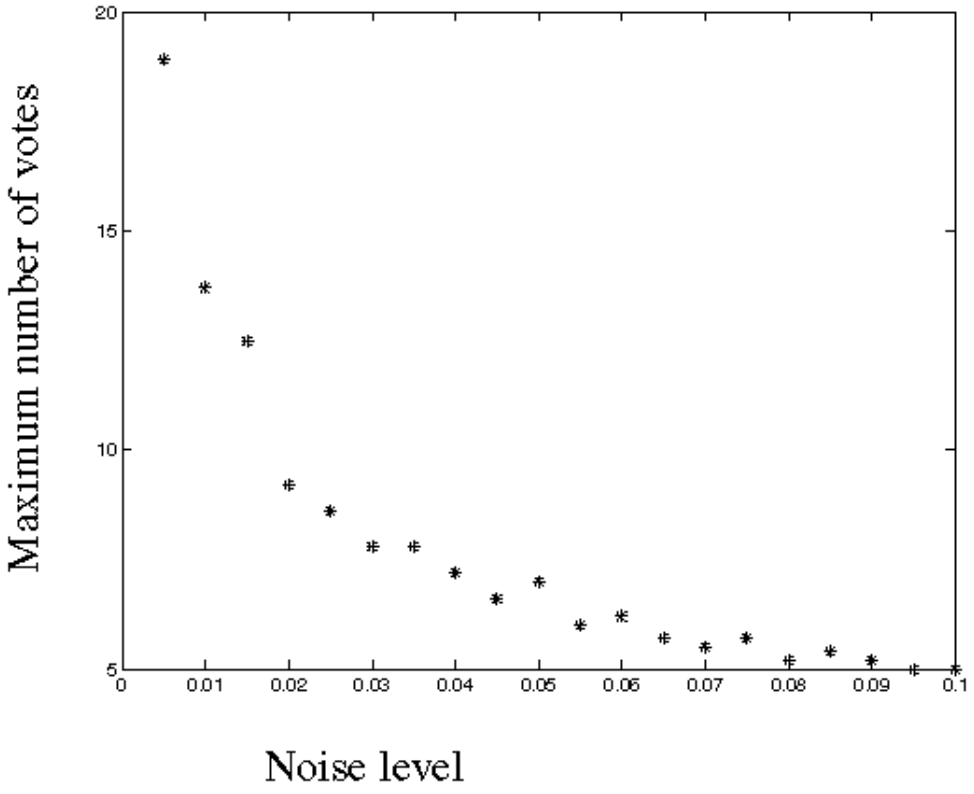


votes

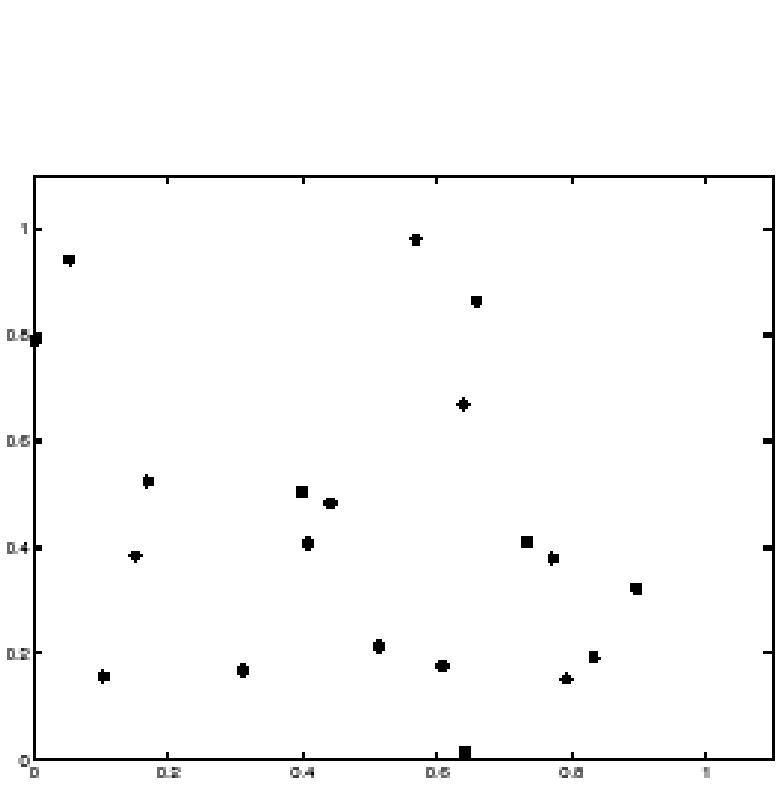
- Random perturbation in [0, 0.05] of each point.
- Peak gets fuzzy and hard to locate.
- Max number of votes is now 6.

Hough transform (cont.)

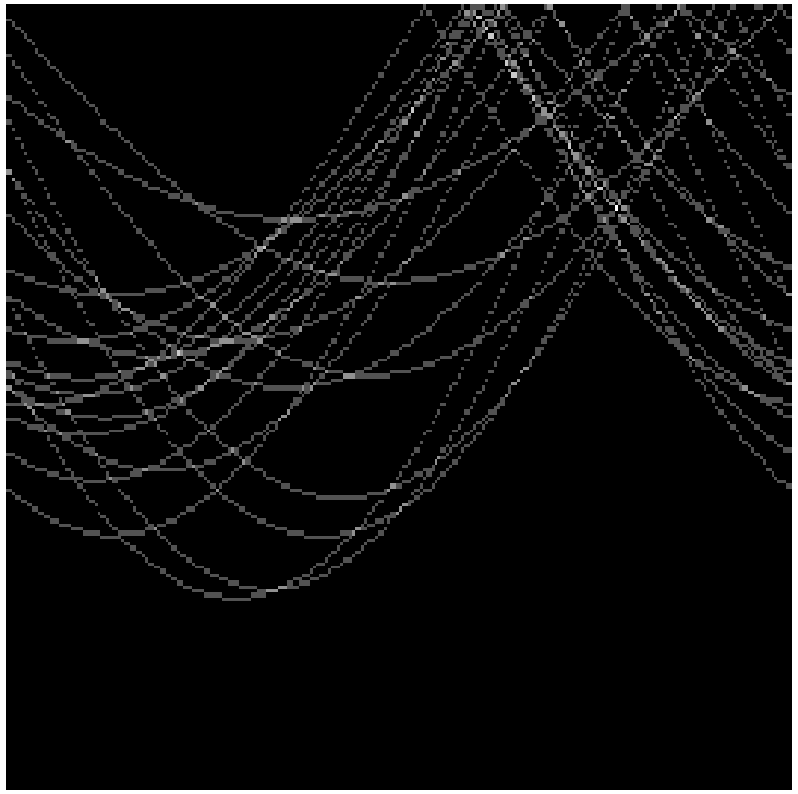
- Number of votes for a line of 20 points with increasing noise:



Hough transform (cont.)



features

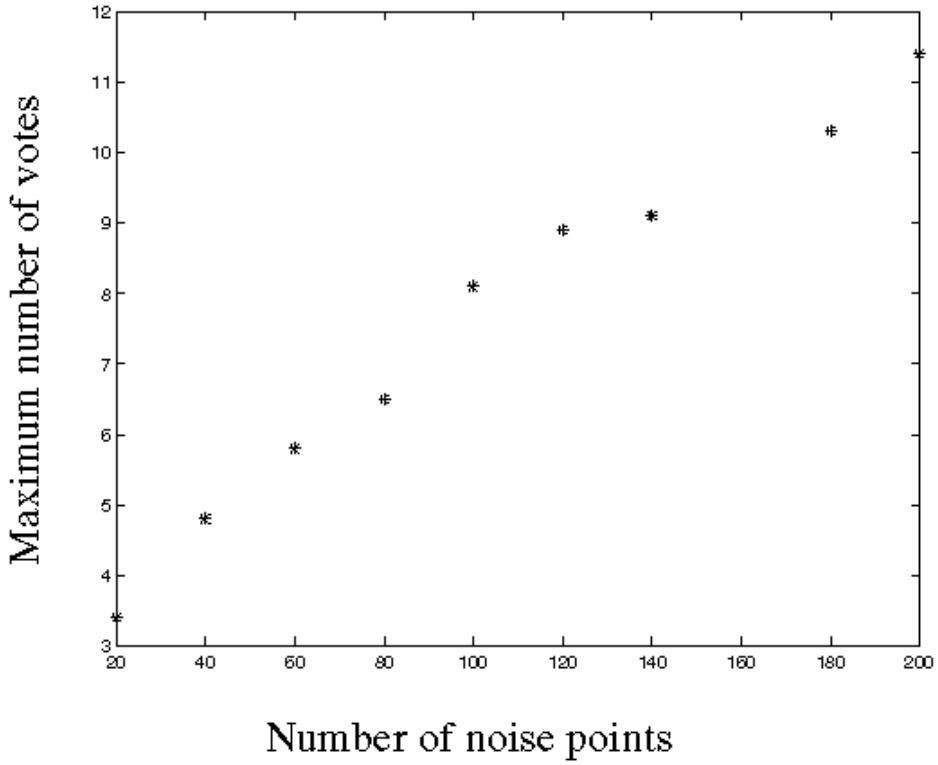


votes

- Random points: uniform noise can lead to spurious peaks in the accumulator array. Max votes = 4.

Hough transform (cont.)

- As the level of uniform noise increases, the maximum number of votes increases too and makes the task difficult.



Practical details

- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude.
- 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 also neighboring bins (smoothing in accumulator array).

Hough transform: Pros

- Can deal with non-locality and occlusion.
- Can detect multiple instances of a model in a single pass.
- Some robustness to noise: noise points unlikely to contribute consistently to any single bin.

Hough transform: 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.