Geometry for computer Vision

Lecture 3: Estimation theory

Lecture 3: Estimation theory

- DLT homography estimation
- Algebraic and geometric errors
- Maximum likelihood estimation
- RANSAC
- Voting techniques
- Mean-shift clustering
- Papers for next week

DLT

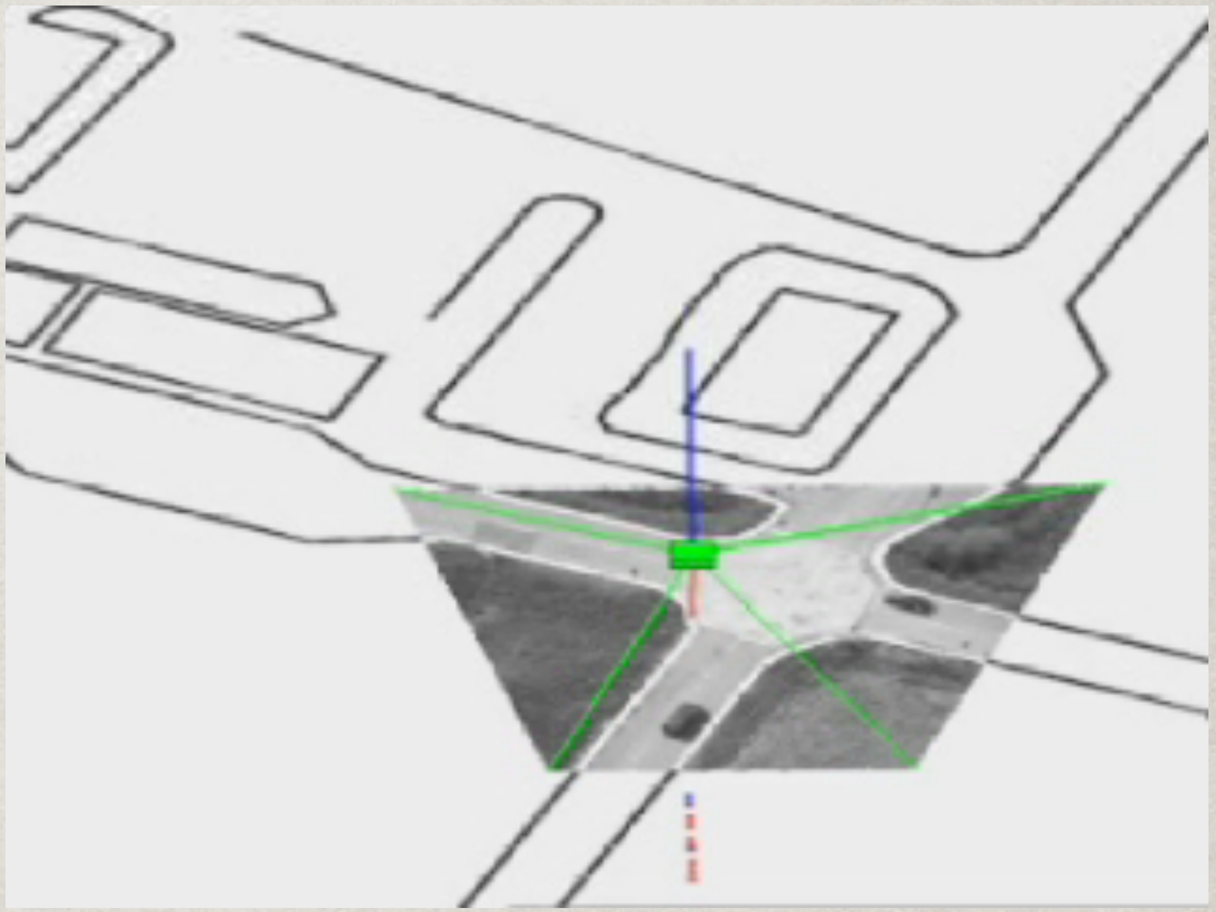
- Remember the homography from lecture 1?

$$\begin{pmatrix} y_1 \\ y_2 \\ 1 \end{pmatrix} \sim \mathbf{H} \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix}$$

- A simple way to estimate H from sets of correspondences $(x_1, x_2) \leftrightarrow (y_1, y_2)$ is to use the *Direct Linear Transformation* (DLT)

DLT example

Homograpy registration to map using tracked points



Extraction of rotation and translation from homography

Forssén, WITAS project 2000

DLT derivation

- Use the cross product with **y** to obtain

$$\mathbf{y} \sim \mathbf{H}\mathbf{x} \quad \Rightarrow \quad \mathbf{0} \sim \mathbf{y} \times \mathbf{H}\mathbf{x}$$

DLT derivation

- Use the cross product with $\mathbf{y}$ to obtain

$$\mathbf{y} \sim \mathbf{H}\mathbf{x} \quad \Rightarrow \quad \mathbf{0} \sim \mathbf{y} \times \mathbf{H}\mathbf{x}$$

- Decompose $\mathbf{H}$ in three row vectors

$$\mathbf{0} = \mathbf{y} \times \begin{pmatrix} — & \mathbf{h}^{1T} & — \\ — & \mathbf{h}^{2T} & — \\ — & \mathbf{h}^{3T} & — \end{pmatrix} \mathbf{x} = \mathbf{y} \times \begin{pmatrix} \mathbf{h}^{1T}\mathbf{x} \\ \mathbf{h}^{2T}\mathbf{x} \\ \mathbf{h}^{3T}\mathbf{x} \end{pmatrix}$$

DLT derivation

- Rewrite cross product as matrix product

$$\mathbf{0} = \mathbf{y} \times \begin{pmatrix} \mathbf{h}^{1T}\mathbf{x} \\ \mathbf{h}^{2T}\mathbf{x} \\ \mathbf{h}^{3T}\mathbf{x} \end{pmatrix} = \begin{pmatrix} 0 & -1 & y_2 \\ 1 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{h}^{1T}\mathbf{x} \\ \mathbf{h}^{2T}\mathbf{x} \\ \mathbf{h}^{3T}\mathbf{x} \end{pmatrix}$$

- Swap terms and factor out **h**-terms

$$\mathbf{0} = \begin{pmatrix} 0 & -1 & y_2 \\ 1 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x}^T\mathbf{h}^1 \\ \mathbf{x}^T\mathbf{h}^2 \\ \mathbf{x}^T\mathbf{h}^3 \end{pmatrix} = \begin{pmatrix} 0 & -\mathbf{x}^T & y_2\mathbf{x}^T \\ \mathbf{x}^T & 0 & -y_1\mathbf{x}^T \\ -y_2\mathbf{x}^T & y_1\mathbf{x}^T & 0 \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix}$$

DLT derivation

- Each point correspondence gives us two equations:

$$\mathbf{0} = \begin{pmatrix} 0 & 0 & 0 & -x_1 & -x_2 & -1 & y_2 x_1 & y_2 x_2 & y_2 \\ x_1 & x_2 & 1 & 0 & 0 & 0 & -y_1 x_1 & -y_1 x_2 & y_1 \end{pmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix}$$

$$\text{or} \quad \mathbf{Mh} = \mathbf{0}$$

- If we have 4 points we get 8 equations, and can solve for $\mathbf{H}$ up to scale.

- For more points we can use least squares.

SVD solution

$$\mathbf{Mh} = \mathbf{0}$$

- Using the *Singular Value Decomposition* (SVD) we can decompose $\mathbf{M}$ into

$$\mathbf{U}\begin{pmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_N \end{pmatrix}\mathbf{V}^T\mathbf{h} = \mathbf{0}$$

- By choosing $\mathbf{V}^T\mathbf{h} = (0 \ \ldots \ 1)^T$ we find the smallest residual

SVD solution

- By choosing $\mathbf{V}^T\mathbf{h} = (0\ \ldots\ 1)^T$ we find the smallest residual.

- Thus **h** should be proportional to the last row of **V**.

- **SVD** solves the problem

$$\mathbf{h}^* = \arg\min_{\mathbf{h}} ||\mathbf{M}\mathbf{h}|| \quad \text{s.t.} \quad ||\mathbf{h}|| = 1$$

Algebraic Error

$$\mathbf{Mh} = \mathbf{0}$$

- SVD minimises the sum of squared residuals

$$\epsilon^2 = \sum_k r_k^2 \,, \quad \text{where} \quad r_k = \mathbf{m}_k \mathbf{h}$$

- The error that we happen to minimise when we solve an over-determined system is called the *algebraic error*.

- Usually contrasted with the *geometric error,* i.e. what we really want to minimise.

Algebraic Error

- Assume i.i.d. noise on the measured points

$$x_1 = \hat{x}_1 + \epsilon_1 \qquad \epsilon_k \in \mathcal{N}(0, \sigma)$$

$$x_2 = \hat{x}_2 + \epsilon_2$$

- Recall the first residual row

$$r_k = \mathbf{m}_k \mathbf{h} = \begin{pmatrix} 0 & 0 & 0 & -x_1 & -x_2 & -1 & y_2 x_1 & y_2 x_2 & y_2 \end{pmatrix} \mathbf{h}$$

- In the noise free case this should be zero

$$r_k = \mathbf{m}_k \mathbf{h} = \begin{pmatrix} 0 & 0 & 0 & -\hat{x}_1 & -\hat{x}_2 & -1 & \hat{y}_2 \hat{x}_1 & \hat{y}_2 \hat{x}_2 & \hat{y}_2 \end{pmatrix} \mathbf{h}$$

- This leaves us with

$$r_k = \begin{pmatrix} 0 & 0 & 0 & -\epsilon_1 & -\epsilon_2 & -1 & \epsilon_3 \hat{x}_1 + \hat{y}_2 \epsilon_1 + \epsilon_1 \epsilon_3 & \epsilon_4 \hat{x}_2 + \hat{y}_2 \epsilon_2 + \epsilon_2 \epsilon_4 & \epsilon_4 \end{pmatrix} \mathbf{h}$$

Algebraic Error

$$\mathbf{r} = \begin{pmatrix} 0 & 0 & 0 & -\epsilon_1 & -\epsilon_2 & -1 & \epsilon_3\hat{x}_1 + \hat{y}_2\epsilon_1 + \epsilon_1\epsilon_3 & \epsilon_4\hat{x}_2 + \hat{y}_2\epsilon_2 + \epsilon_2\epsilon_4 & \epsilon_4 \\ \epsilon_1 & \epsilon_2 & 1 & 0 & 0 & 0 & -\epsilon_3\hat{x}_1 - \hat{y}_1\epsilon_3 - \epsilon_1\epsilon_3 & -\epsilon_3\hat{x}_2 - \hat{y}_1\epsilon_2 + \epsilon_2\epsilon_3 & \epsilon_3 \end{pmatrix} \mathbf{h}$$

- Noise on columns 7 and 8 is counted more!
- Columns 3 and 6 are noise free!

Hartley normalisation

- Hartley normalisation gives a more even weight on all columns

$$\hat{\mathbf{x}} \sim \begin{pmatrix} \sqrt{2}/s & 0 & -\sqrt{2}\mu_1/s \\ 0 & \sqrt{2}/s & -\sqrt{2}\mu_2/s \\ 0 & 0 & 1 \end{pmatrix} \mathbf{x}$$

- s - average distance to origin
- μ_1, μ_2 - mean in first and second coordinate

Hartley normalisation

- If we have found a homography that maps normalised points

$$\hat{\mathbf{y}} \sim \tilde{\mathbf{H}}\hat{\mathbf{x}} \quad \text{where} \quad \hat{\mathbf{y}} = \mathbf{N}_y\mathbf{y} \quad \text{and} \quad \hat{\mathbf{x}} = \mathbf{N}_x\mathbf{x}$$

- We can find the mapping for the original points as

$$\mathbf{H} = \mathbf{N}_y^{-1}\tilde{\mathbf{H}}\mathbf{N}_x$$

Why?

- Further improvements by row&col weighting

$$\mathbf{M}\mathbf{h} = 0 \quad \Rightarrow \quad \mathbf{W}_1\mathbf{M}\mathbf{W}_2\mathbf{h} = 0$$

Maximum Likelihood

- Instead of the algebraic error, it would be better to maximise

$$p\left(\mathbf{h}|\left\{\mathbf{x}_k,\mathbf{y}_k\right\}\right)$$

Maximum Likelihood

- Instead of the algebraic error, it would be better to maximise

$$p\left(\mathbf{h}|\left\{\mathbf{x}_k,\mathbf{y}_k\right\}\right)$$

- Mathematically it is however easier to look for a **h** that maximises

$$p\left(\left\{\mathbf{x}_k,\mathbf{y}_k\right\}|\,\mathbf{h}\right)$$

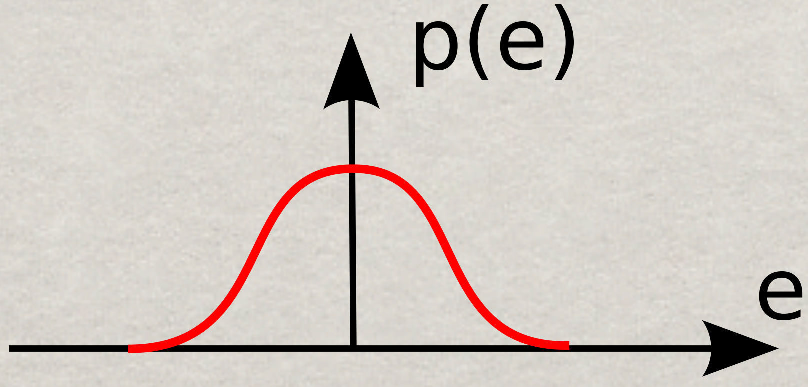
- This is called *Maximum Likelihood* (ML)

Maximum Likelihood

- The error in direct measurements is often easy to model.
- E.g. empirically from measurements with ground truth.

Maximum Likelihood

- The error in direct measurements is often easy to model.
- E.g. empirically from measurements with ground truth.
- It is e.g. reasonable to model errors in pixel locations as localised and unbiased.



Maximum Likelihood

- Assume no errors in $\mathbf{y}$, but errors in $\mathbf{x}$ that are Gaussian and independent:

$$p\left(\{\mathbf{x}_k\}\left|\mathbf{H},\{\mathbf{y}_k\}\right.\right)=\prod_k \frac{1}{2\pi\sigma^2}\exp\left(-d^2\left(\mathbf{x}_k,\mathbf{H}\mathbf{y}_k\right)/2\sigma^2\right)$$

- $d(\mathbf{x}_k,\mathbf{H}\mathbf{y}_k)$ is the Euclidean distance in image 1.

Maximum Likelihood

- Assume no errors in $\mathbf{y}$, but errors in $\mathbf{x}$ that are Gaussian and independent:

$$p\left(\{\mathbf{x}_k\}\,|\mathbf{H},\{\mathbf{y}_k\}\right) = \prod_k \frac{1}{2\pi\sigma^2}\exp\left(-d^2\left(\mathbf{x}_k,\mathbf{H}\mathbf{y}_k\right)/2\sigma^2\right)$$

$$p\left(\{\mathbf{x}_k\}\,|\mathbf{H},\{\mathbf{y}_k\}\right) = \frac{1}{2\pi\sigma^2}\exp\left(-\sum_k d^2\left(\mathbf{x}_k,\mathbf{H}\mathbf{y}_k\right)/2\sigma^2\right)$$

- We could instead find the $\mathbf{H}$ that minimises:

$$-\log p\left(\{\mathbf{x}_k\}\,|\mathbf{H},\{\mathbf{y}_k\}\right) \propto \sum_k d^2\left(\mathbf{x}_k,\mathbf{H}\mathbf{y}_k\right)$$

Maximum Likelihood

- The cost function $J(\mathbf{H}) = \sum_k d^2(\mathbf{x}_k, \mathbf{H}\mathbf{y}_k)$
- is a non-linear least-squares problem.
- Can be solved by gradient descent, starting in an initial guess $\mathbf{H}_0$ close to the correct solution.
- $\mathbf{H}_0$ is typically found using normalised DLT.

Maximum Likelihood

- Maximum Likelihood = Least Squares IF:
  - Gaussian noise
  - i.i.d
  - in one image (the other is error free)
- For errors in both images we need to optimise over both $\mathbf{H}$ and the undistorted points $\{\hat{\mathbf{x}}_k, \hat{\mathbf{y}}_k\}$

Maximum Likelihood

- Reprojection error

$$\sum_{k=1}^{K} d(\mathbf{x}_k, \hat{\mathbf{x}}_k)^2 + d(\mathbf{y}_k, \mathbf{H}^{-1}\hat{\mathbf{x}}_k)^2$$

- 2K+9 parameters. Solved with e.g. Levenberg-Marquardt. Expensive if many points.

- A simple approximation is the 9 parameter *symmetric transfer error*:

$$\sum_{k=1}^{K} d(\mathbf{x}_k, \mathbf{H}\mathbf{y}_k)^2 + d(\mathbf{y}_k, \mathbf{H}^{-1}\mathbf{x}_k)^2$$

Maximum Likelihood

- ML solutions can be derived for other parameter estimation problems as well.
- All have in common that a *reprojection error,* i.e. an error in the measurements, needs to be derived.
- ML solutions are called the *gold standard* in the Hartley&Zisserman book.

Problems with linear methods

- Example: LS line estimation from points:

$$\begin{pmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & 1 \\ x_K & y_K & 1 \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = 0$$

Problems with linear methods

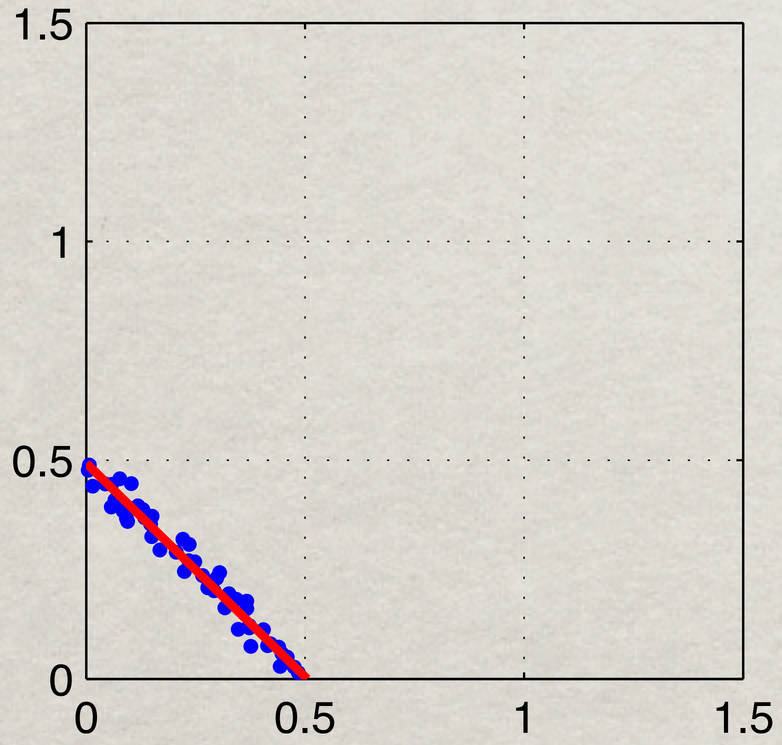
- Example: LS line estimation from points:

$$\begin{pmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & 1 \\ x_K & y_K & 1 \end{pmatrix} \begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix} = 0$$

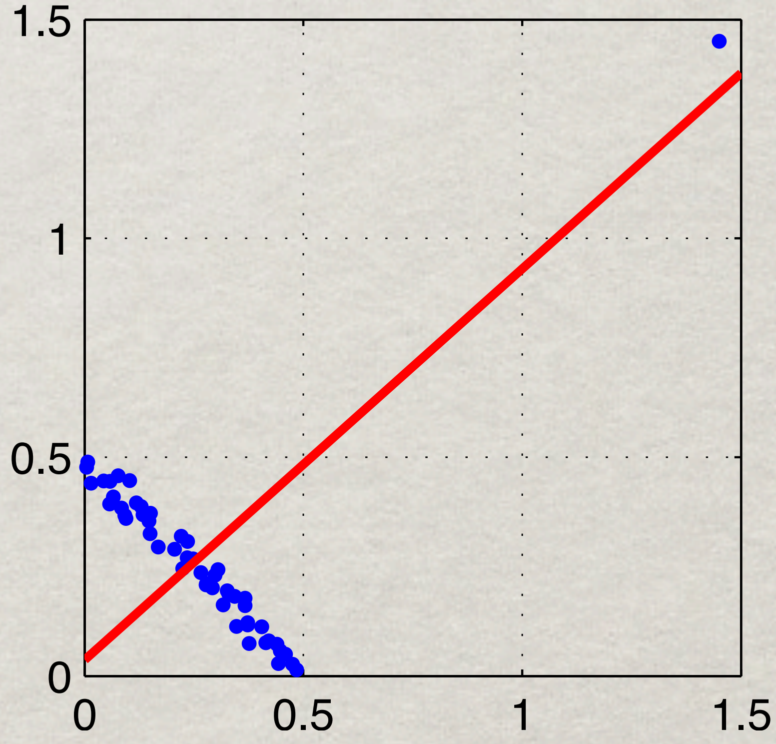
- Remember errror analysis from before: Column weighting with $1/\sigma$ also helps here.
- But, there is a bigger problem...

The problem with LS

- What if some measurements are very wrong, i.e. they measure something else?



LS for additive uniform noise



LS after adding one outlier

A solution

- Random Sample Consensus (RANSAC) Fischler and Bolles 1981.
- **Hypothesize**
- **Verify**
- **Loop**

Ransac

- Random Sample Consensus (RANSAC) Fischler and Bolles 1981.
- **Hypothesize** pick a few samples and estimate solution
- **Verify** test the solution, by evaluating the likelihood
- **Loop** keep doing this and store the best solution

Ransac for a homography (from H&Z)

1. Detect interest points
2. Select a set of putative correspondences
3. Randomly select 4 correspondences and compute $\mathbf{H}$ using DLT
4. Score $\mathbf{H}$ by counting number of *inliers*

$$d_{\mathrm{sym}}(\mathbf{x}_k, \mathbf{y}_k | \mathbf{H}) < t$$

5. Repeat 3 and 4.
6. Choose $\mathbf{H}$ with highest score.
7. Run ML on inlier set.

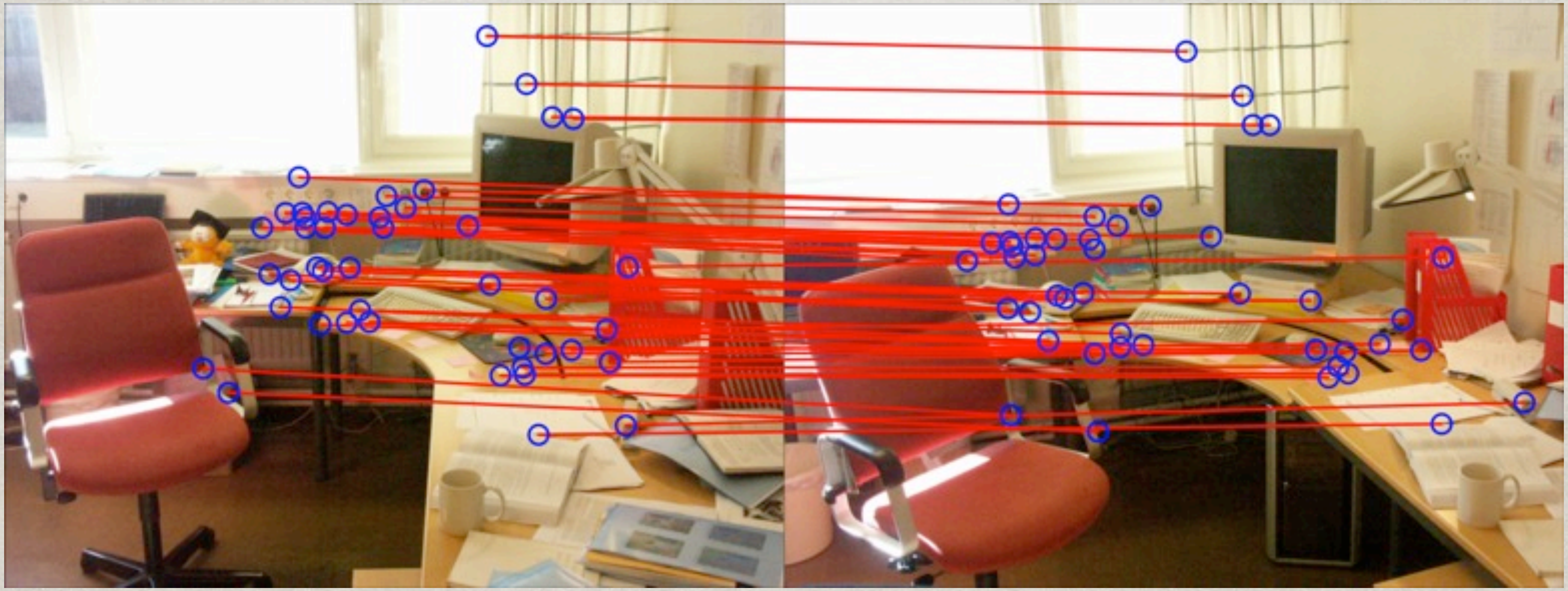
Ransac

- Same thing can be done for the fundamental matrix **F**

Putative correspondences

Ransac

- Same thing can be done for the fundamental matrix **F**



Inliers after RANSAC

Ransac for a homography (from H&Z)

- The algorithm in the book is outdated (but its a good introduction).
- Lecture 6 will cover more up-to date techniques.
- Two issues:

    1. How many RANSAC iterations?
    2. Threshold value?

Number of samples

- w - fraction of inliers
- s - number of points in minimal sample
- p - probability of finding an uncontaminated sample (we can never be sure!)
- N - number of samples used

$$(1 - w^s)^N = 1 - p$$

- Solving for N gives us

$$N = \log(1 - p) / \log(1 - w^s)$$

Number of samples

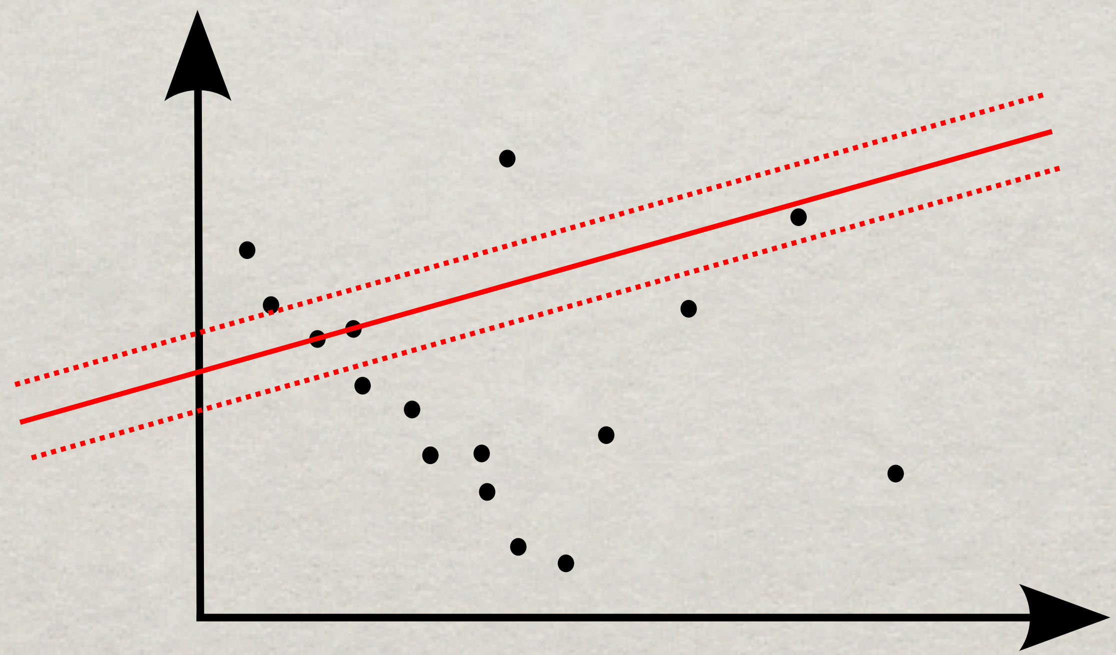
$$N = \log(1 - p) / \log(1 - w^s)$$

s	w=0.95	w=0.90	w=0.80	w=0.75	w=0.70	w=0.60	w=0.50
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

Number of samples

$$N = \log(1-p)/\log(1-w^s)$$

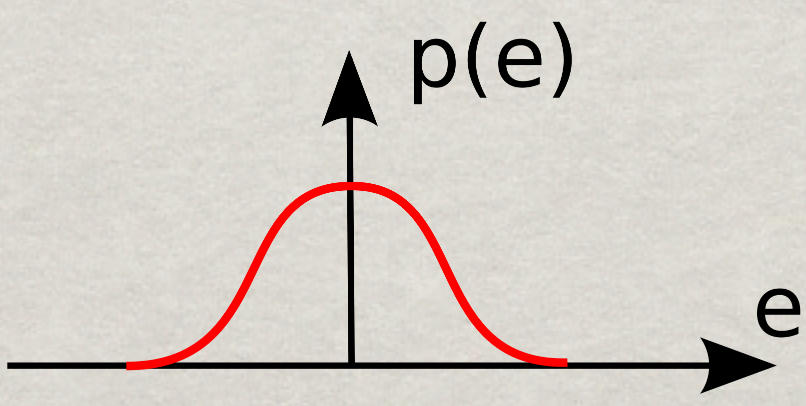
- In practise, we have inlier noise, and then this heuristic is wildly optimistic



Threshold value

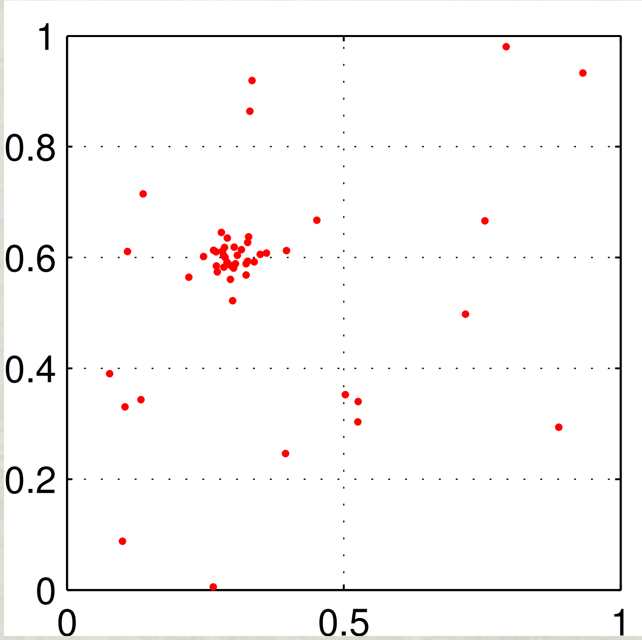
- Preferrably, we should not score the hypotheses based on number of inliers, but on the likelihood of the model.

- From this follows that we should sum the likelihoods of the errors...



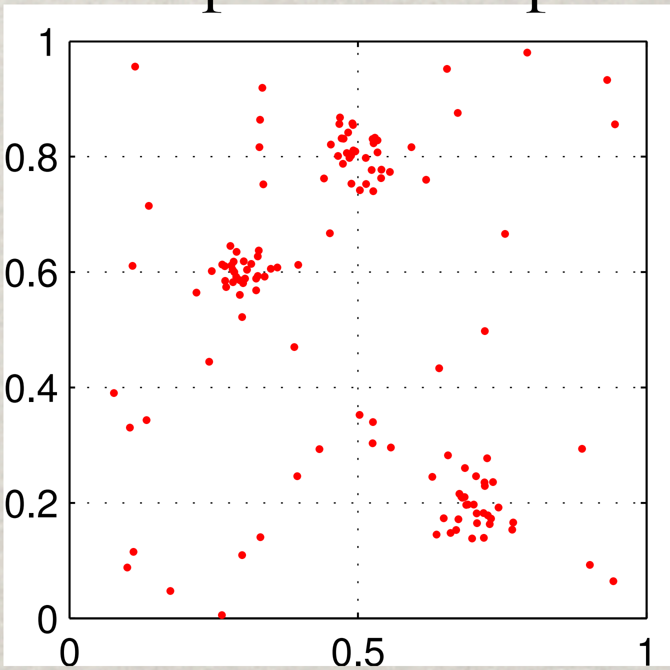
Strong and weak robustness

- **Weak robustness**
  one cluster and <50% outliers
  * RANSAC
  * L_1 optimisation
    median, LP, LmedS,...



model parameter space

- **Strong robustness**
  several clusters, and outliers
  * voting (histograms/GHT)
  * mean-shift,...

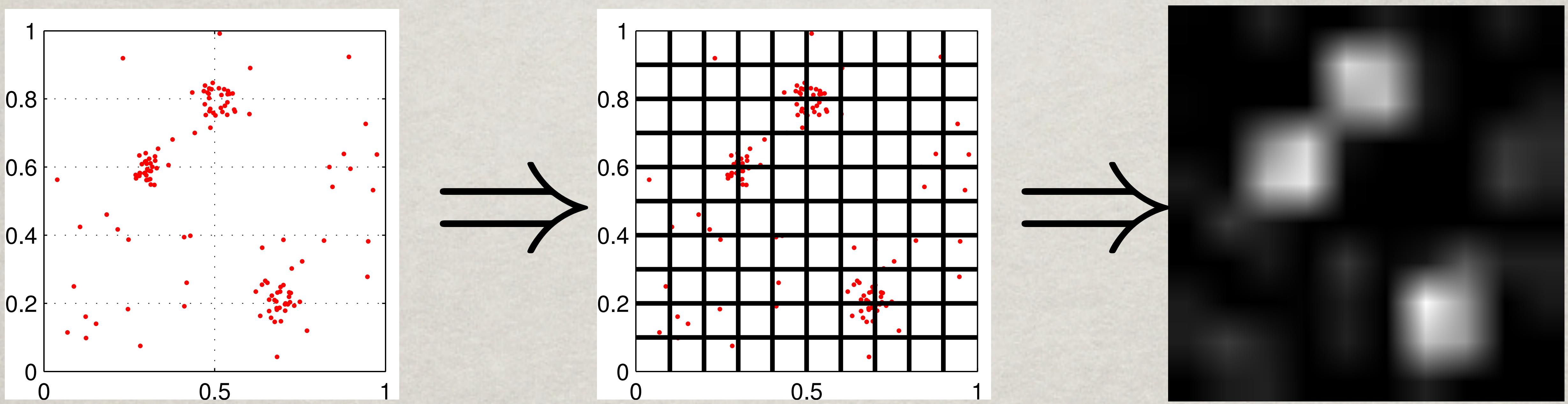


Voting techniques

- For some problems, we can define a grid over possible parameter values, and evaluate the likelihood at each grid location.
- Channel Clustering (Forssén, 2004)
- Approximations:
    1. Histograms
    2. Hough Transform
    3. Generalised Hough Transform (GHT)

Voting techniques

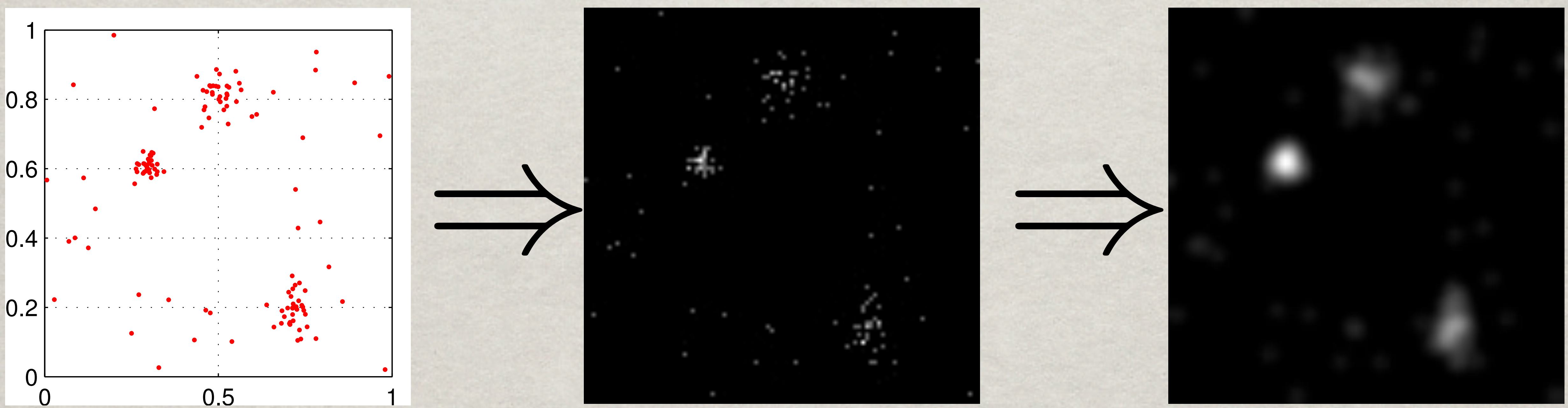
- Histogramming and GHT simplifies this to just letting each sample cast a vote in a cell.



- Similarly, the Hough transform paints a line in the grid cell space...

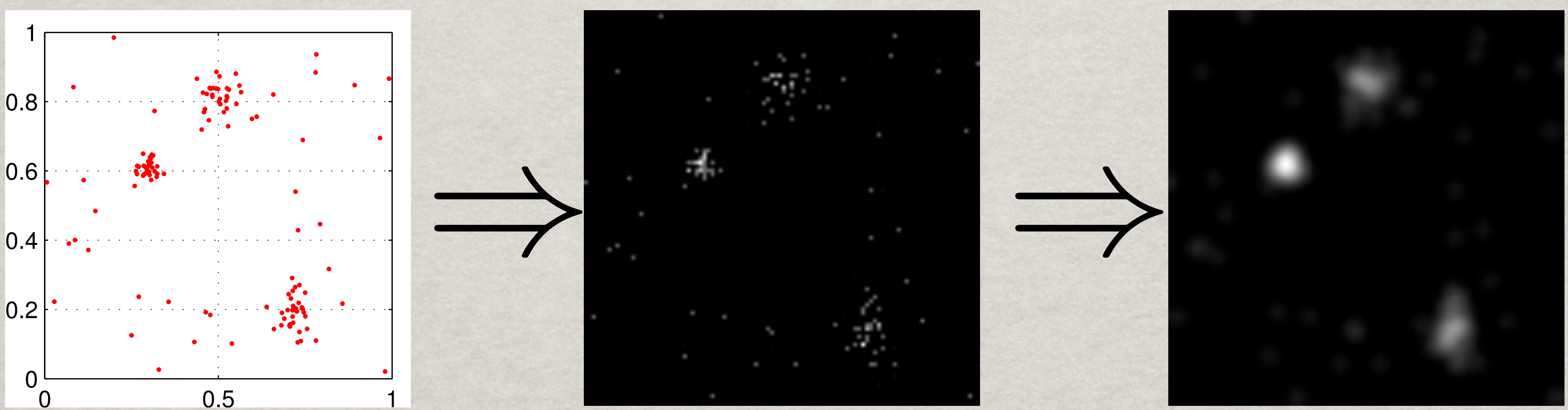
Voting techniques

- Increased number of cells, followed by low-pass filtering gives us better accuracy, and reduces the risk of missing a peak.



Channel Clustering

- Since the blurring reduces the bandwidth we can sample more sparsely, and even afford to properly evaluate the likelihood.
- Accurate peaks from a decoding scheme (Forssén, 2004)



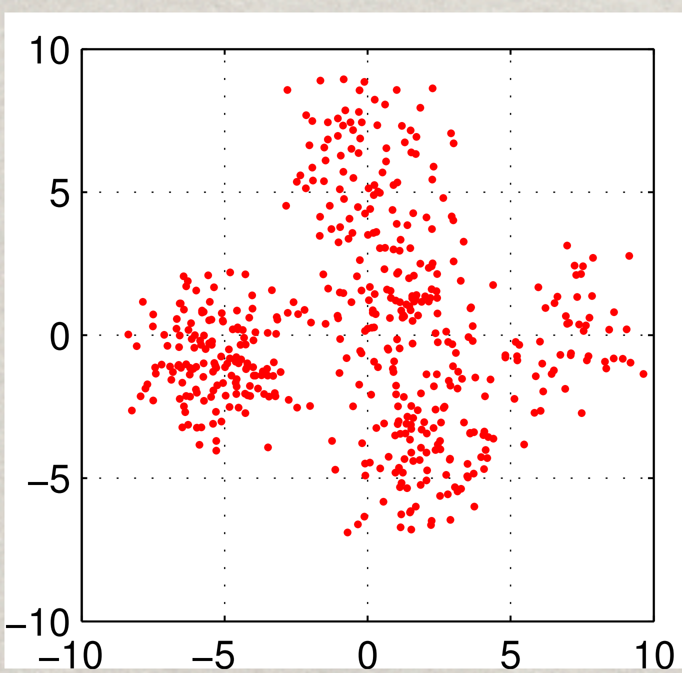
Mean-shift Clustering

- Algorithm illustration (Cheng, 1995)

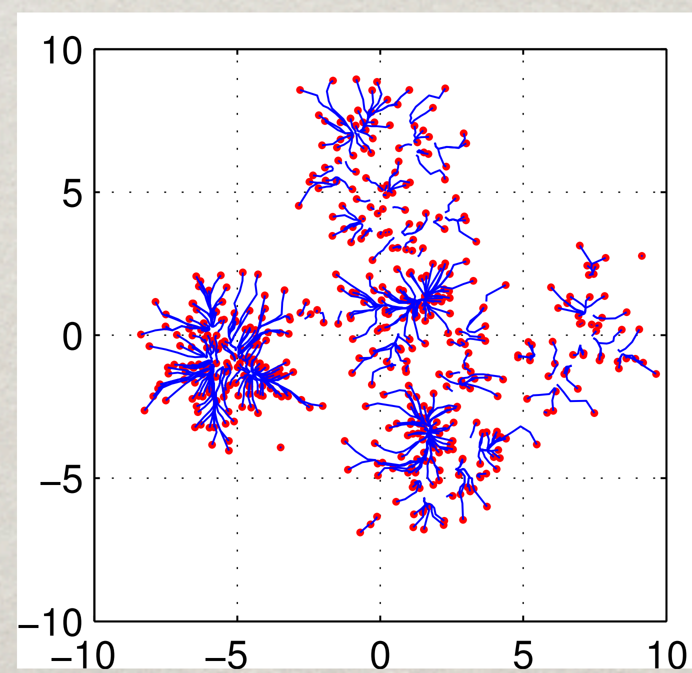
1. Start in each data point $\mathbf{m}_n = \mathbf{x}_n$
2. Move to poisition of local average

$$\mathbf{m}_n \leftarrow \text{mean}_{\mathbf{x}_n \in S(\mathbf{m}_n)}(\mathbf{x}_n)$$

3. Repeat 2 until convergence



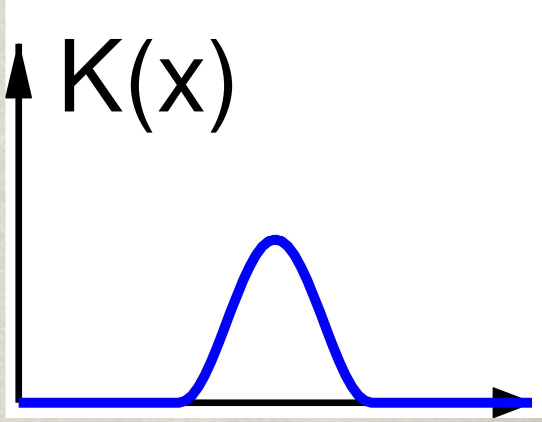
$\Rightarrow$

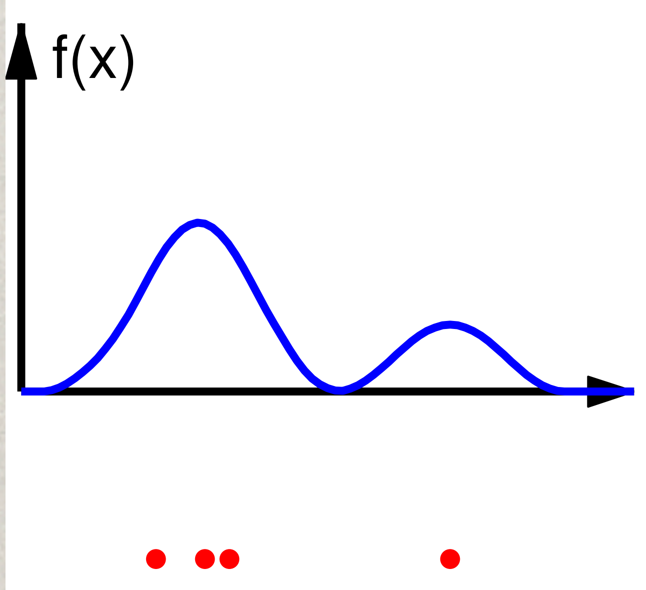


Mean-shift Clustering

- Mean-shift is gradient ascent (with a particular step length) on the cost function

$$f(\mathbf{m}) = \frac{1}{N}\sum_{n=1}^{N} K(||\mathbf{x}_n - \mathbf{m}||)$$

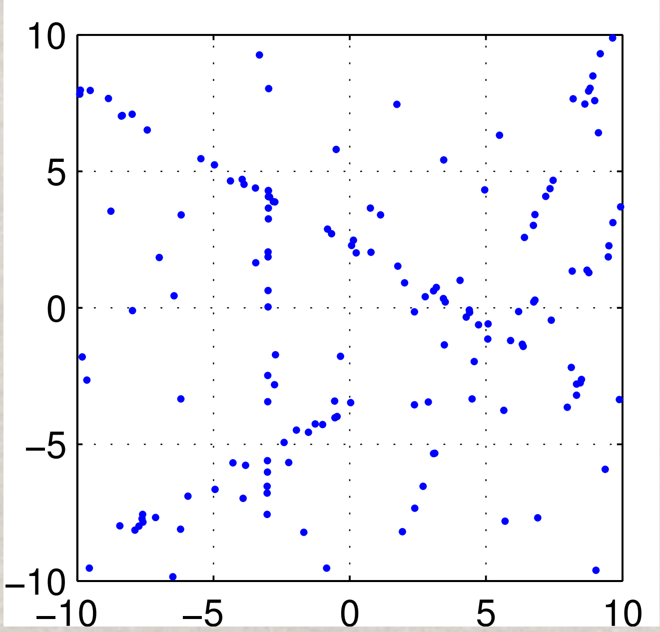




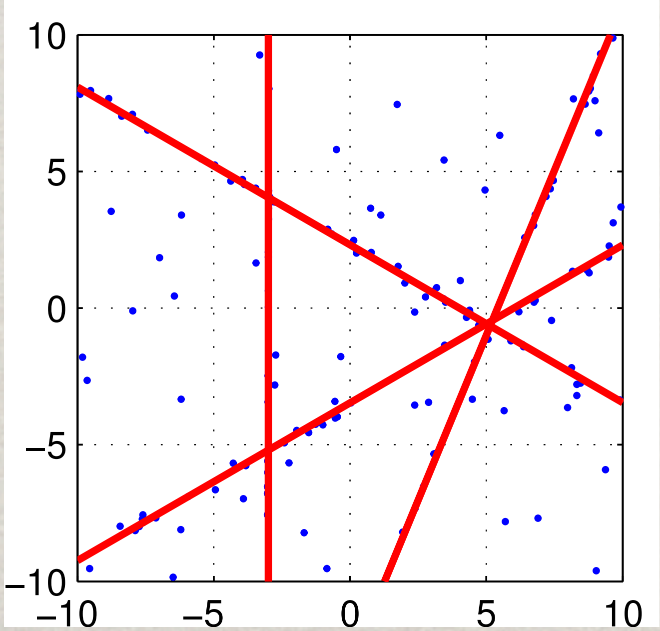
- If we set K to the error likelihood, mean-shift is ML

Mean-shift Clustering

- Example 1: (Cheng 95)
  1. Pick 300 2D points in an edge image
  2. Generate all (44 850) pairs of points
  3. Each pair gives a sample (ρ_k, φ_k)
  4. Cluster in (ρ, φ) space

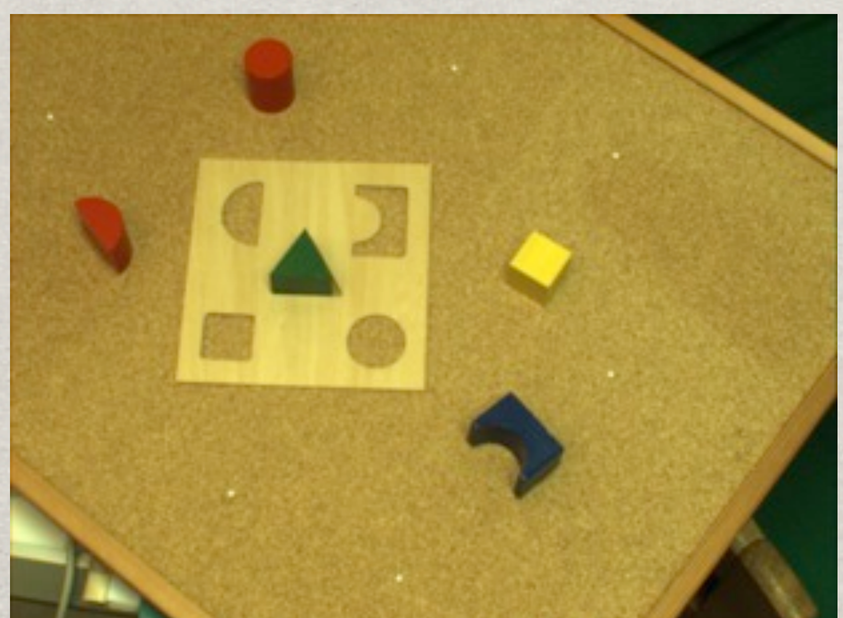


$\Rightarrow$

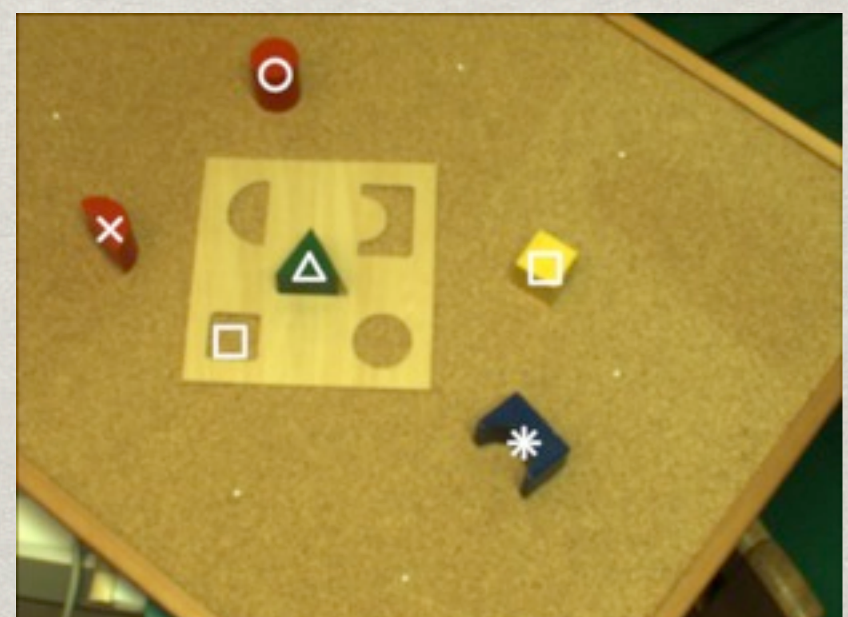


Mean-shift Clustering

- Example 2: Pose Estimation (Viksten, ICRA2009)
  - Extract local invariant features (e.g. SIFT or MSER)
  - Let each pair of features cast a vote on the pose of an object $\mathbf{x}_k = (x_0, y_0, \alpha, s, \varphi, \theta, \texttt{type})$
  - Cluster the votes using mean-shift



$\Rightarrow$



For Next week...

- Papers to read:

    1. Mendoca and Cippolla, A *Simple Technique for Self-Calibration,* CVPR99

    2. Costeira and Kanade, *A Multibody Factorization Method for Independently Moving Objects*, sections 1-3

For Next week...

- For those taking the course for credits:
  - Prepare two topics for discussion on the paper. E.g. something you disagree with, or do not understand. Remember to explain *how* and *why*!
- We will leave room in the second half of the lecture for the discussion.