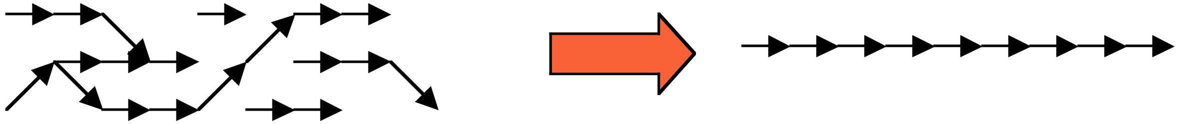
# Grouping

**To make sense of image elements they first have to be grouped into larger structures.**

**<u>Example</u>: Grouping noisy edge elements into a straight edge**

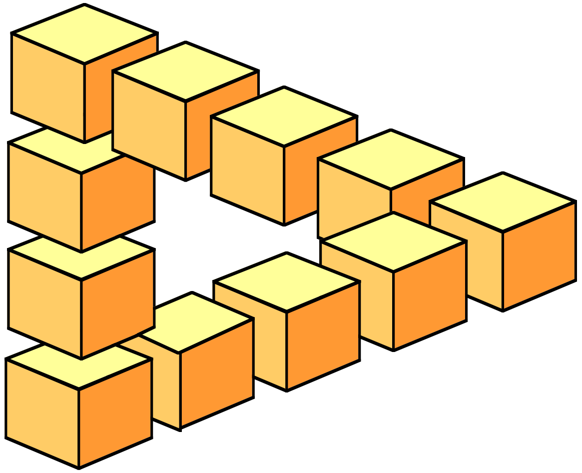


**<u>Essential problem</u>:**

**Obtaining globally valid results by local decisions**

**<u>Important methods</u>:**

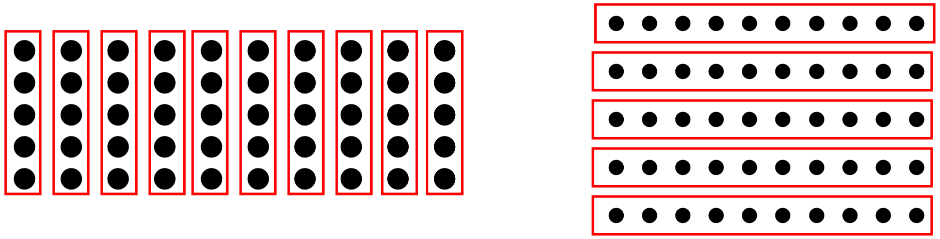
- **Fitting**
- **Clustering**
- **Hough Transform**
- **Relaxation**



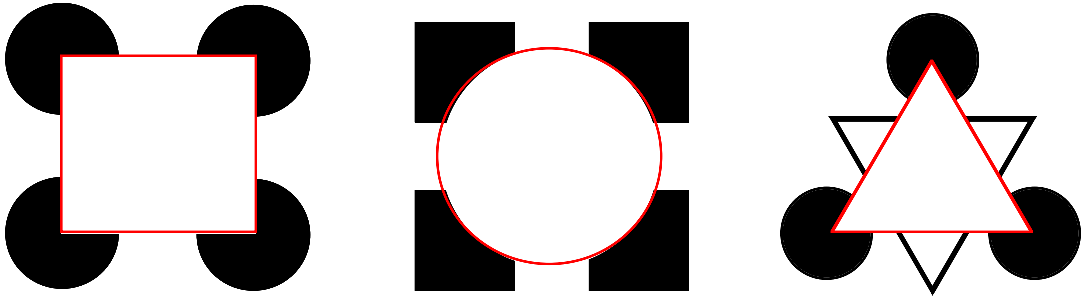
- **locally compatible**
- **globally incompatible**

# Cognitive Grouping

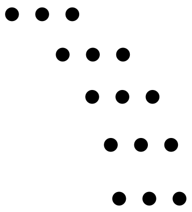
The human cognitive system shows remarkable grouping capabilities



grouping into rows or columns according to a distance criterion



grouping into virtual edges

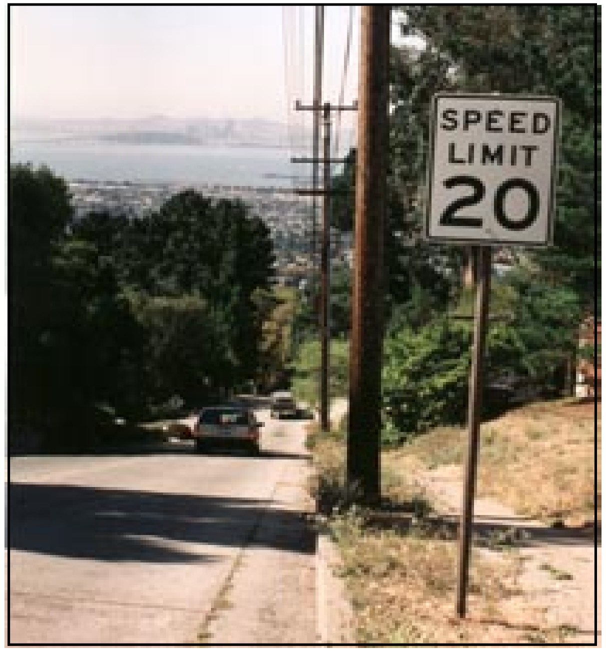


grouping into virtual motion

It is worthwhile wondering which cognitive grouping rules should also be followed by machine vision

# Fitting Straight Lines

**Why do we want to discover straight edges or lines in images?**



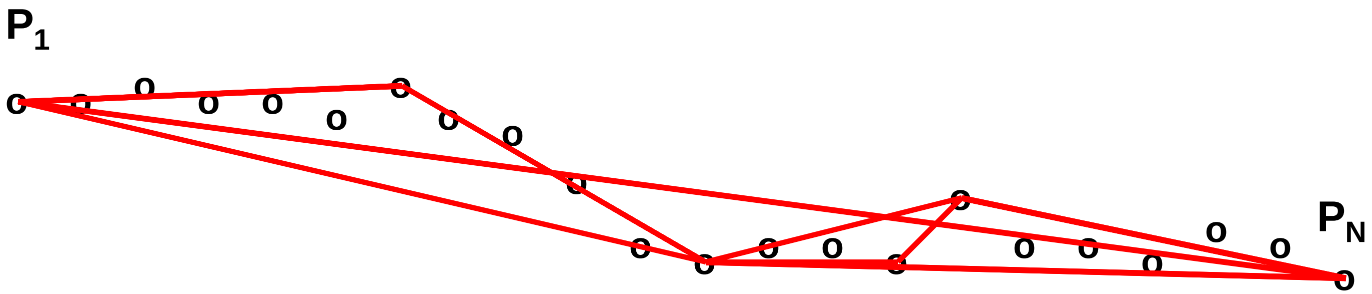
- **Straight edges occur abundantly in the civilized world.**
- **Approximately straight edges are also important to model many natural phenomena, e.g. stems of plants, horizon at a distance.**
- **Straightness in scenes gives rise to straighness in images.**
- **Straightness discovery is an example of constancy detection which is at the heart of grouping (and maybe even interpretation).**

**We will treat several methods for fitting straight lines:**

- **Iterative refinement**
- **Mean-square minimization**
- **Eigenvector analysis**
- **Hough transform**

# Straight Line Fitting by Iterative Refinement

**Example**: Fitting straight segments to a given object motion trajectory



**Algorithm**:

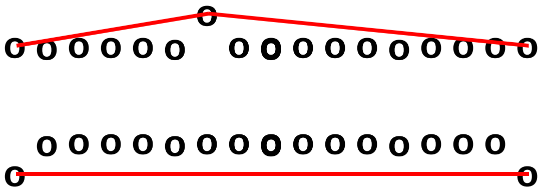
A: First straight line is $P_1P_N$

B: Is there a straight line segment $P_iP_k$ with an intermediate point $P_j$ $(i < j < k)$ whose distance from $P_iP_k$ is more than d? If no, then terminate.

C: Segment $P_iP_k$ into $P_iP_j$ and $P_jP_k$ and go to B.

**Advantage:** simple and fast

**Disadvantages:**
- strong effect of outliers
- not always optimal

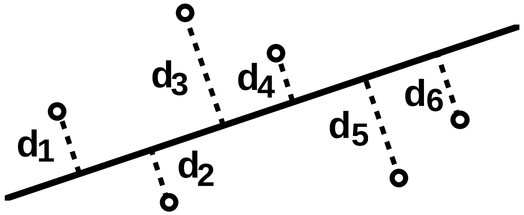


# Straight Line Fitting by Eigenvector Analysis (1)

**Given:** $(x_i\ y_i)\quad i = 1 \ldots N$

**Wanted:** Coefficients $c_0$, $c_1$ for straight line

$y = c_0 + c_1 x$ which minimizes $\Sigma\, d_i^2$



**Observation:**

The optimal straight line passes through the mean of the given points. Why?

Let (x´y´) be a coordinate system with the x´ axis parallel to the optimal straight line.

optimal straight line $x' = x_0'$

error $\Sigma\, d_i^2 = \Sigma\, (x_i' - x_0')^2$

condition for optimum $\delta/\delta x_0\, \{\Sigma\, (x_i' - x_0')^2\} = -2 \bullet \Sigma\, (x_i' - x_0') = 0$

$x_0' = 1/N \bullet \Sigma\, x_i'$

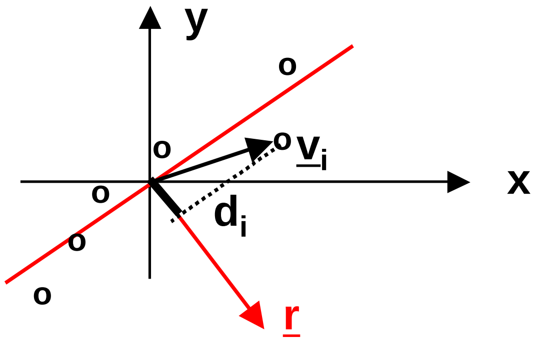
A new coordinate system may be chosen with the origin at the mean of the given points:

$$x_j' = x_j - \frac{\sum x_i}{N} \qquad y_j' = y_j - \frac{\sum y_i}{N}$$

Optimal straight line passes through origin, only direction is unknown.

# Straight Line Fitting by Eigenvector Analysis (2)

**After coordinate transformation the new problem is:**



**Given:** points $\underline{v}_i^T = [x_i\ y_i]$ with $\Sigma\, \underline{v}_i = \underline{0}$   i = 1 ... N

**Wanted:** direction vector $\underline{r}$ which minimizes $\Sigma\, d_i^2$

**Minimize**

$$d^2 = \sum_{i=1}^{N} d_i^2 = \sum_{i=1}^{N} (\underline{r}^T \underline{v}_i)^2 = \sum_{i=1}^{N} (\underline{r}^T \underline{v}_i)(\underline{v}_i^T \underline{r}) = \underline{r}^T S \underline{r}$$

**scatter matrix**

**Minimization with Lagrange multiplier λ:**

$$\underline{r}^T S \underline{r} + \lambda \underline{r}^T \underline{r} \Rightarrow \text{minimum} \qquad \text{subject to } \underline{r}^T \underline{r} = 1$$

**Minimizing $\underline{r}$ is <u>eigenvector</u> of S, minimum is <u>eigenvalue</u> of S.**

**For a 2D scatter matrix there exist 2 orthogonal eigenvectors:**

$\underline{r}_{min}$ orthogonal to optimal straight line

$\underline{r}_{max}$ parallel to optimal straight line

# Straight Line Fitting by Eigenvector Analysis (3)

Computational procedure:

- Determine mean $\underline{m}$ of given points with $m_x = 1/N\ \Sigma\ x_i$, $m_y = 1/N\ \Sigma\ y_i$, $i = 1 \dots N$
- Determine scatter matrix $S = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} = \begin{bmatrix} \Sigma\ (x_i-m_x)^2 & \Sigma\ (x_i-m_x)(y_i-m_y) \\ \Sigma\ (x_i-m_x)(y_i-m_y) & \Sigma\ (y_i-m_y)^2 \end{bmatrix}$
- Determine maximal eigenvalue

$$\lambda_{1,2} = \frac{S_{11}+S_{22}}{2} \pm \sqrt{\left(\frac{S_{11}+S_{22}}{2}\right)^2 - |S|} \qquad \lambda_{max} = \max \{\lambda_1, \lambda_2\}$$

- Determine direction of eigenvector corresponding to $\lambda_{max}$

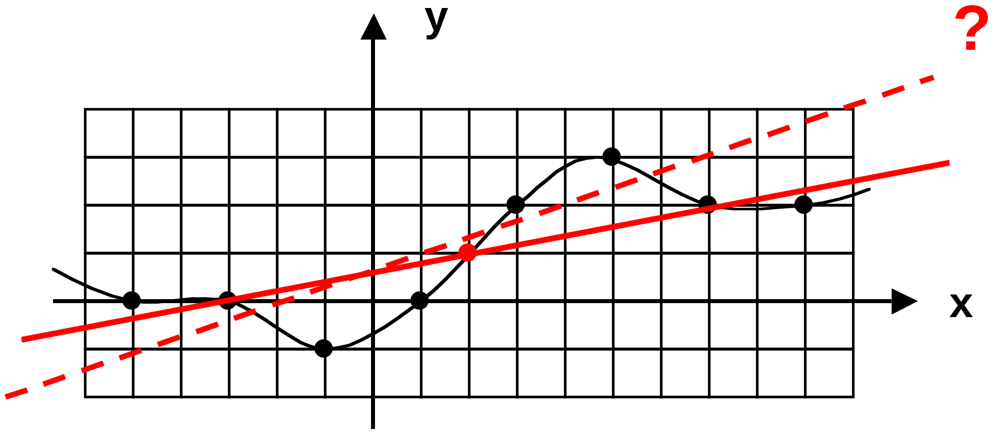
$S_{11}\ r_x + S_{12}\ r_y = \lambda_{max}\ r_x$ by definition of eigenvector => $r_y/r_x$

- Determine optimal straight line

$(y- m_y) = (x-m_x)\ (r_y/r_x) = (x-m_x)\ (\lambda_{max} - S_{11})/S_{12}$

# Example for Straight Line Fitting by Eigenvector Analysis

What is the best straight-line approximation of the contour?



Given points: { (-5 0) (-3 0) (-1 -1) (1 0) (3 2) (5 3) (7 2) (9 2) }

Center of gravity: $m_x = 2$ $m_y = 1$

Scatter matrix: $S_{11} = 168$ $S_{12} = S_{21} = 38$ $S_{22} = 14$

Eigenvalues: $\lambda_1 = 176{,}87$ $\lambda_2 = 5{,}13$

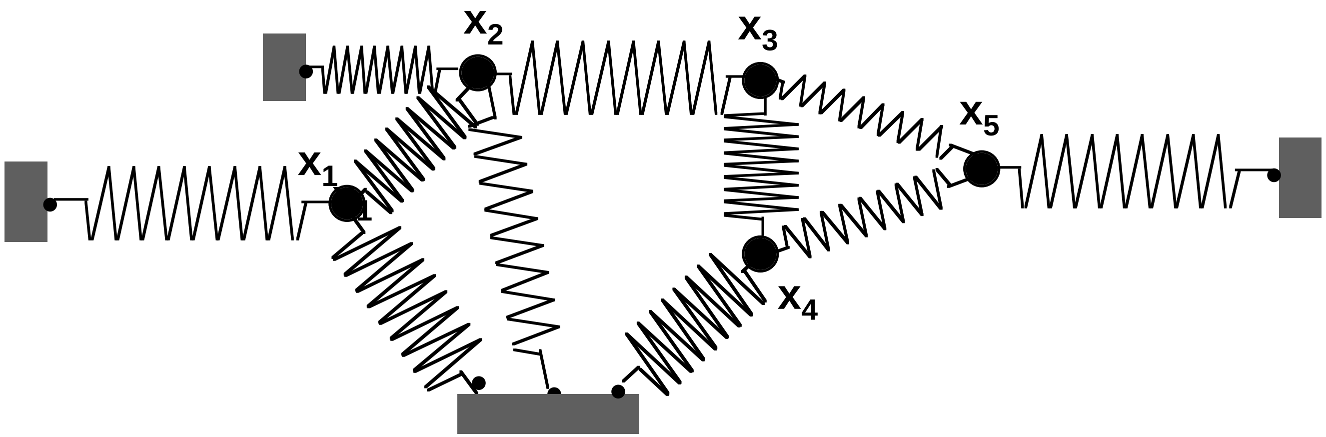
Direction of straight line: $r_y/r_x = 0{,}23$

Straight line equation: $y = 0{,}23\, x + 0{,}54$

# Grouping by Relaxation

Relaxation methods seek a solution by stepwise minimization ("relaxation") of constraints.

**Analogy with spring system:**



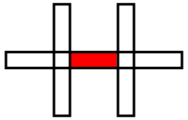
**Variables $x_i$ take on values (= positions) where springs are maximally relaxed corresponding to a state of global minimal energy. Hence relaxation is often realized by "energy minimization".**

# Contexts for Edge Relaxation

Iterative modification of edge strengths using context-dependent compatibility rules.

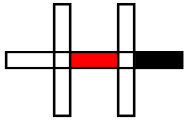
Context types:

-1



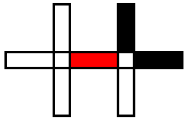
isolated edge

+1



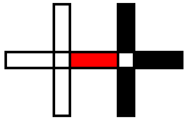
uncertain continuation

-1



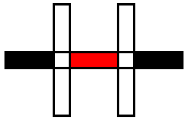
spurious continuation

-1



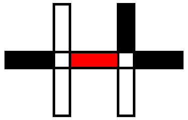
spurious continuation

+3



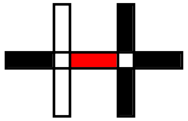
connecting edge

+2



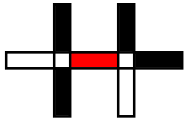
connecting edge

+2



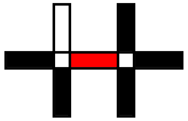
isolated edge

0



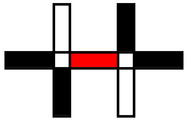
uncertain connection

0



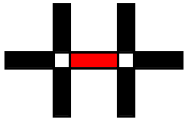
uncertain connection

0



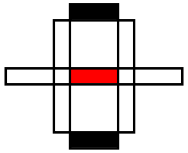
uncertain connection

0



uncertain connection

-1



competing edge

Each context contributes with weight $w_j = w_0 \cdot \{-1 \dots +2\}$ to an interative modification of the edge strength of the central element.

# Modification Rule for Edge Relaxation

| | |
|---|---|
| $P_i^k$ | edge strength in position i after iteration k |
| $Q_{ij}^k$ | strength of context j for position i after iteration k |
| $w_j$ | weight factor of context j |

$$Q_{ij}^k = \prod P_m^k \bullet \prod (1-P_n^k)$$

edge context strength

m, n ranging over all supporting and not supporting edge positions of context j, respectively.

$$P_i^{k+1} = P_i^k \frac{1+\Delta P_i^k}{1+P_i^k \Delta P_i^k}$$

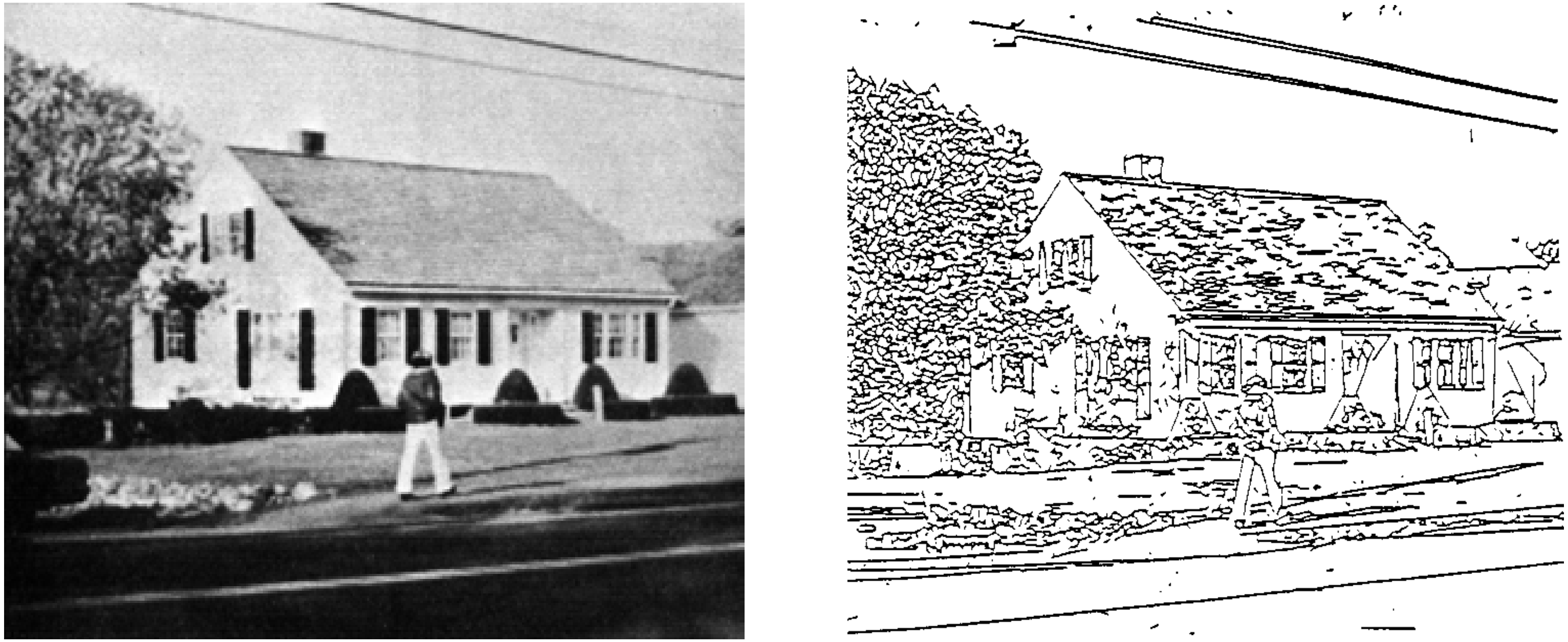
edge strength modification rule

$$\Delta P_i^k = \sum_{j=1}^{N} w_j Q_{ij}^k$$

edge strength increment

There is empirical evidence (but no proof) that for most edge images this relaxation procedure converges within 10 ... 20 iterations.

# Example of Edge-finding by Relaxation



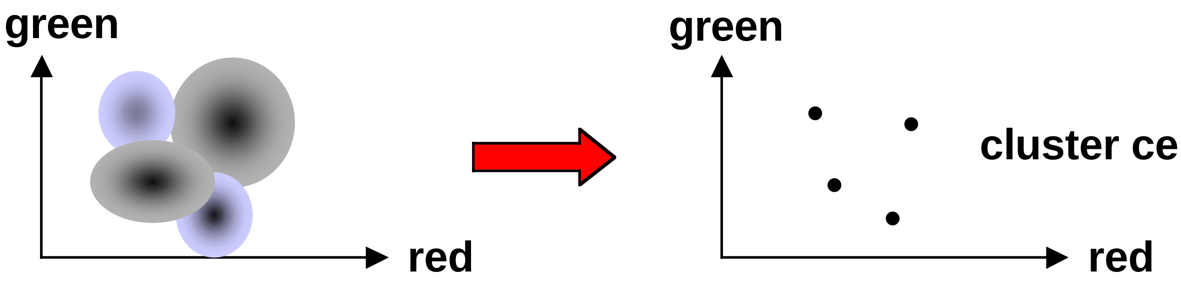
Landhouse scene from VISIONS project, 1982

# Histogram-based Segmentation with Relaxation (1)

**<u>Basic idea</u>:**

**Use relaxation to introduce a local similarity constraint into histogram-based region segmentation.**

**A Determine cluster centers by multi-dimensional histogram analysis**



**cluster centers $\underline{c}_i$, i = 1 ... N**

**B Label each pixel by cluster-membership probabilities $p_i$ , 1 = 1 ... N**

$$p_i = \frac{1/\,d_i}{\sum_{k=1}^{N} 1/\,d_k}$$

**$d_i$ is Euclidean distance between the feature vector of the pixel and cluster center $\underline{c}_i$**

# Histogram-based Labelling with Relaxation (2)

C Iterative relaxation of the $p_i(j)$ of all pixels j:

- equal labels of neighbouring pixels support each other
- unequal labels of neighbouring pixels inhibit each other

$$q_i(j) = \sum_{k \in D(j)} [w^+ p_i(k) - w^- (1 - p_i(k))]$$

D(j) is neighbourhood of pixel j

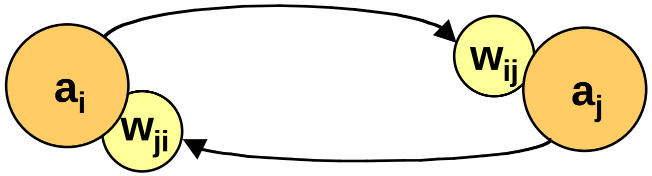
$$p_i'(j) = \frac{p_i(j) + q_i(j)}{\sum_n (p_n(j) + q_n(j))}$$

new probability $p_i'(j)$ of pixel j to belong to cluster i

D Region assignment of each pixel according to its maximal membership probability max $p_i$

E Recursive application of the procedure to individual regions

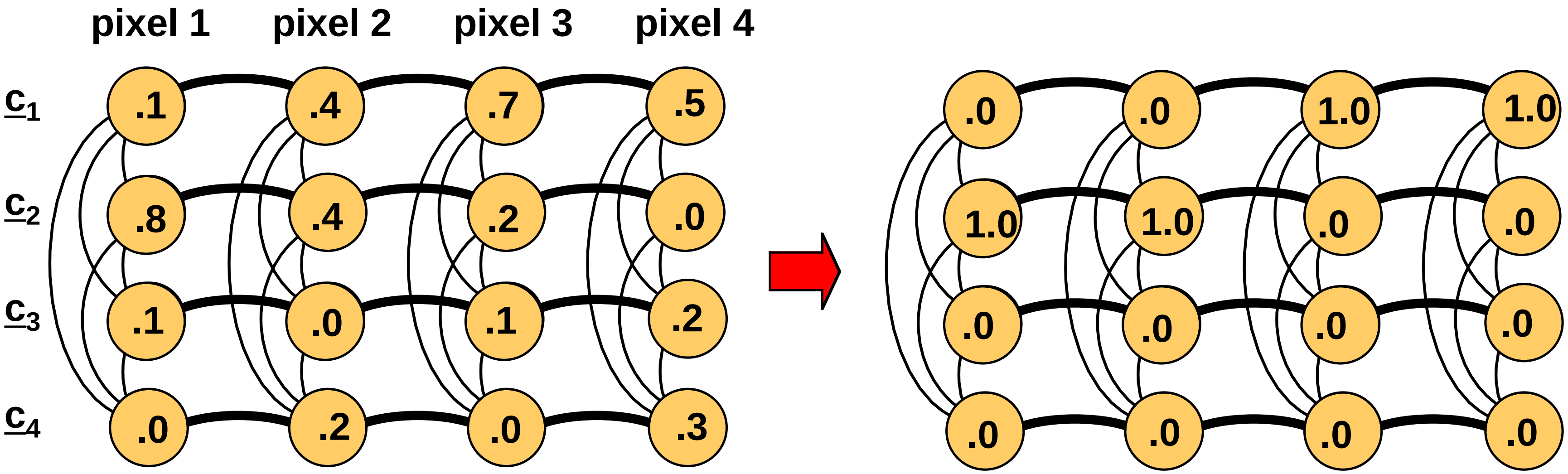
# Relaxation with a Neural Network



**Principle:**

cells influence each other´s activation via exciting or inhibiting weights

**Relaxation labelling of 4 pixels:**



| | pixel 1 | pixel 2 | pixel 3 | pixel 4 |
|---|---|---|---|---|
| $c_1$ | .1 | .4 | .7 | .5 |
| $c_2$ | .8 | .4 | .2 | .0 |
| $c_3$ | .1 | .0 | .1 | .2 |
| $c_4$ | .0 | .2 | .0 | .3 |

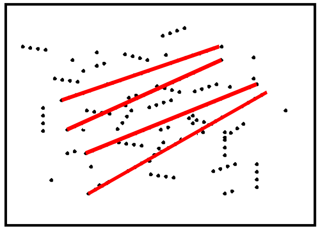
| | | | |
|---|---|---|---|
| .0 | .0 | 1.0 | 1.0 |
| 1.0 | 1.0 | .0 | .0 |
| .0 | .0 | .0 | .0 |
| .0 | .0 | .0 | .0 |

— bidirectional inhibiting connection

━ bidirectional exciting connection

# Hough Transform (1)

Robust method for fitting straight lines, circles or other geometric figures which can be described analytically.

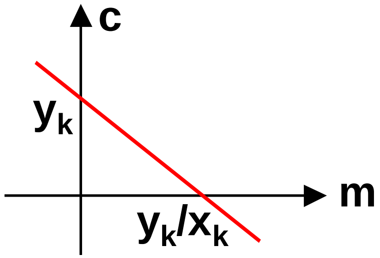


**Given:** Edge points in an image

**Wanted:** Straight lines supported by the edge points

An edge point $(x_k, y_k)$ supports all straight lines $y = mx + c$ with parameters m and c such that $y_k = mx_k + c$.

The locus of the parameter combinations for straight lines through $(x_k, y_k)$ is a straight line in parameter space.



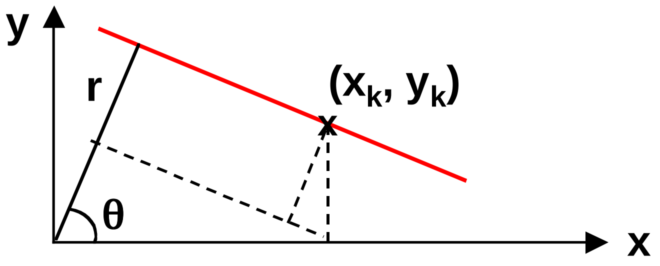
Principle of Hough transform for straight line fitting:

- Provide accumulator array for quantized straight line parameter combinations
- For each edge point, increase accumulator cells for all parameter combinations supported by the edge point
- Maxima in accumulator array correspond to straight lines in the image

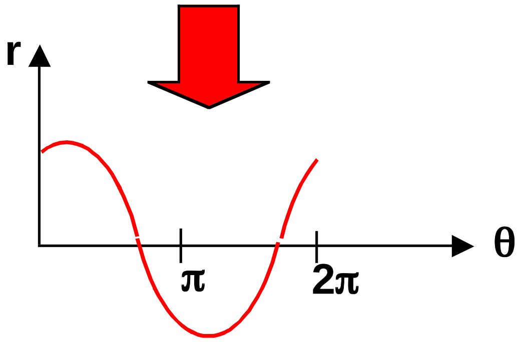
# Hough Transform (2)

For straight line finding, the parameter pair (r, θ) is commonly used because it avoids infinite parameter values:

$$x_k \cos\theta + y_k \sin\theta = r$$



Each edge point $(x_k, y_k)$ corresponds to a sinusoidal in parameter space:



Important improvement by exploiting direction information at edge points:

$(x_k, y_k, \varphi)$ ⇒ $x_k \cos\theta + y_k \sin\theta = r$ restricted to $\varphi - \delta \leq \theta \leq \varphi + \delta$

gradient direction

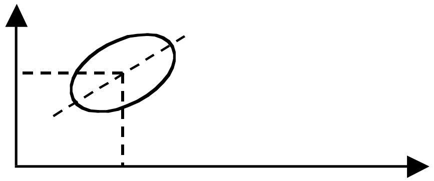
direction tolerance

# Hough Transform (3)

**Same method may be applied to other parameterizable shapes, e.g.**

- **circles** $(x_k-x_0)^2 + (y_k-y_0)^2 = r^2$ **3 parameters $x_0$, $y_0$, $r$**

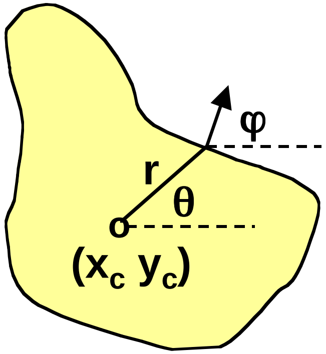
- **ellipses** $$\left(\frac{(x_k-x_0)\cos\gamma+(y_k-y_0)\sin\gamma}{a}\right)^2+\left(\frac{(y_k-y_0)\cos\gamma-(x_k-x_0)\sin\gamma}{b}\right)^2=1$$ **5 parameters $x_0$, $y_0$, $a$, $b$, $\gamma$**



**Accumulator arrays grow exponentially with number of parameters**
**=> quantization must be chosen with care**

# Generalized Hough Transform

- shapes are described by edge elements $(r\ \theta\ \varphi)$ relative to an arbitrary reference point $(x_c\ y_c)$
- $\varphi$ is used as index into $(\rho\ \theta)$ pairs of a shape description
- edge point coordinates $(x_k\ y_k)$ and gradient direction $\varphi_k$ determine possible reference point locations
- likely reference point locations are determined via maxima in accumulator array



$\varphi_1$: $\{(r_{11}\ \theta_{11})\ (r_{12}\ \theta_{12})\ \ldots\ \}$

$\varphi_2$: $\{(r_{21}\ \theta_{11})\ (r_{22}\ \theta_{12})\ \ldots\ \}$

$\vdots$

$\varphi_N$: $\{(r_{N1}\ \theta_{11})\ (r_{N2}\ \theta_{12})\ \ldots\ \}$

$$(x_k\ x_k\ \varphi_k) \Rightarrow \{(x_c\ y_c)\} = \{\ (x_k + r_i(\varphi)\cos\theta_i(\varphi),\ \ (x_k + r_i(\varphi)\sin\theta_i(\varphi))\ \}$$

counter cell in accumulator array