Image Analysis - Lecture 3

Texture and Segmentation

Kalle Åström

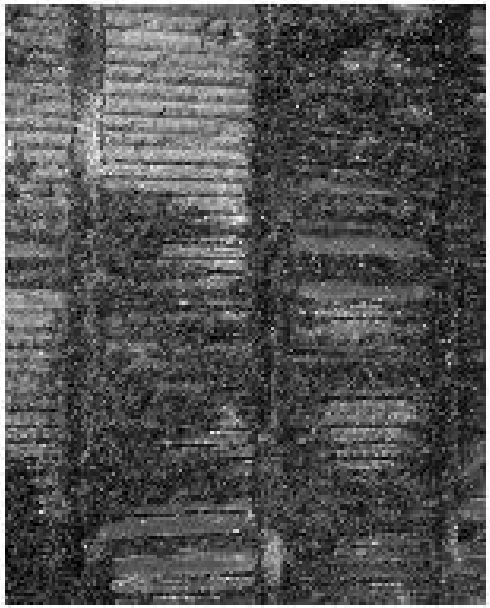
November 6, 2013

Lecture 3

Contents

- Texture
  - Textons
  - Filter Banks
  - Gabor filters
- Segmentation
  - What is segmentation?
  - Clustering

Texture



Texture (ctd.)

Texture is easy to recognize, but difficult to explain.
A leaf is an object, but foliage is a texture

- Texture recognition
- Texture synthesis
- Shape from texture

Repetition: Features

Examples of features in images are

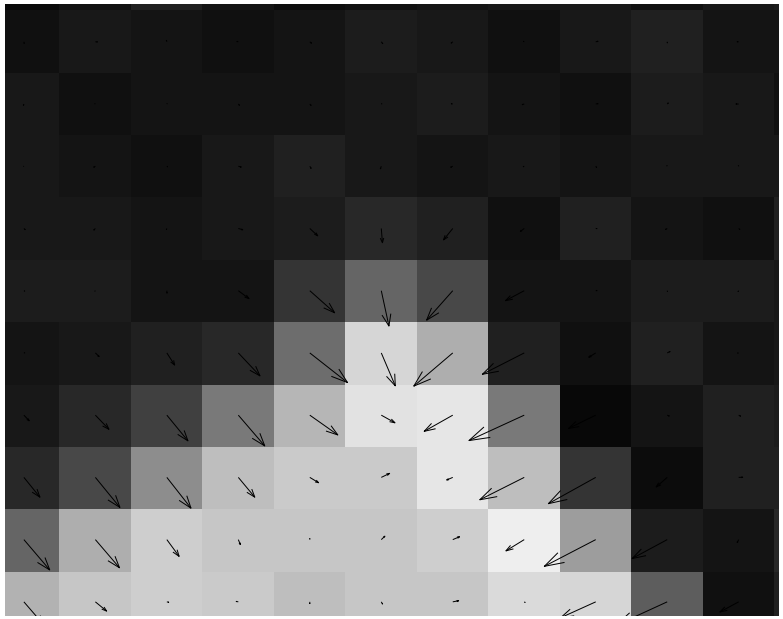
- Edges, where the intensity gradients are large.
- Ridges, the centre of dark or light stripes.
- Corners or other interesting points that can be tracked reliably.

Repetition: Feature detection

Use convolution of appropriate filter (elongated) gaussian, first derivatives of (elongated) gaussians), higher order derivatives of (elongated) gaussians, specific patterns, ...

- Local maxima
- (Later use also interpolation to achieve sub-pixel accuracy, week 3 in the course)
- Thresholding
- (Later use also machine learning/classification, week 5 in the course)

Illustration of orientations



Illustrations of the partial derivatives $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$

The Orientation Tensor

Construct the matrix

$$M = \begin{bmatrix} W_{xx} & W_{xy} \\ W_{xy} & W_{yy} \end{bmatrix} = \begin{bmatrix} (\frac{\partial f}{\partial x})^2 * G_b & (\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}) * G_b \\ (\frac{\partial f}{\partial x}\frac{\partial f}{\partial y}) * G_b & (\frac{\partial f}{\partial y})^2 * G_b \end{bmatrix},$$

where G_b denotes the Gaussian function with parameter b.
M - orientation tensor.
Note: We construct a matrix for every pixel.

Properties of the orientation tensor

The matrix M has the following properties:

- (Flat) Two small eigenvalues in a region - flat intensity.
- (Flow) One large and one small eigenvalue - edges and flow regions.
- (Texture) Two large eigenvalues - corners, interest points, texture regions.

This can be used in algorithms for segmenting the image into (flat, flow, texture).

Textons

Texture often contains regular patterns of parts, often called **textons**.

Idea: If you can detect the textons and measure where they are and how they are distributed?

Problem: There is no useful definition of textons that can be used for detection.

By convolution with small patterns, a strong response signals existence of that small pattern. This could be thought of as texton-detection.

What patterns should we use?

Filter banks: Edge detection

For **edge detection** we used two filters $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$.
After non-linear transformation (squaring) and summation we obtained

$$\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 = |\nabla f|^2 .$$

which is used to detect edges (classify pixels as belonging to the texture 'edge').

Filter banks: Corner detection

For **corner/interest point detection** we used two filters $a_1 = \frac{\partial f}{\partial x}$ and $a_2 = \frac{\partial f}{\partial y}$.
After non-linear transformation (a_1^2, $a_1 a_2$ and a_2^2), further smoothing and another non-linear transformation we obtained

$$f_{cr} = (k + \frac{1}{k})|\det(M)| - |\mathrm{trace}(M)^2 - 2\det(M)|$$

that was used to detect interest points (classify pixels as belonging to the texture 'corner').

Filter banks: Ridge detection

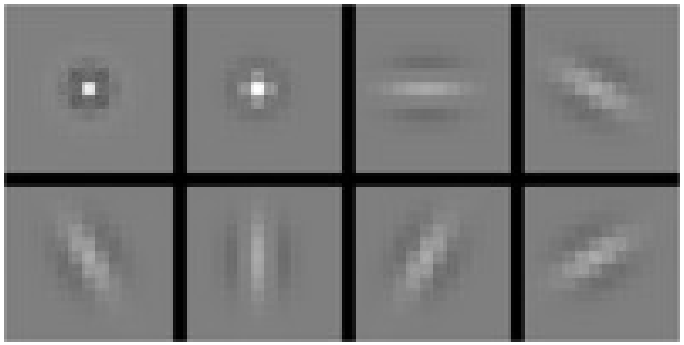
For **ridge detection** we used three filters
$\frac{\partial^2 f}{\partial x^2}$, $\frac{\partial^2 f}{\partial x \partial y}$ and $\frac{\partial^2 f}{\partial y^2}$.
After mean value filtering and a non-linear transformation we obtain a measure that can be used to detect ridges (classify pixels as belonging to the texture 'ridge').

Filter banks: Texture classification

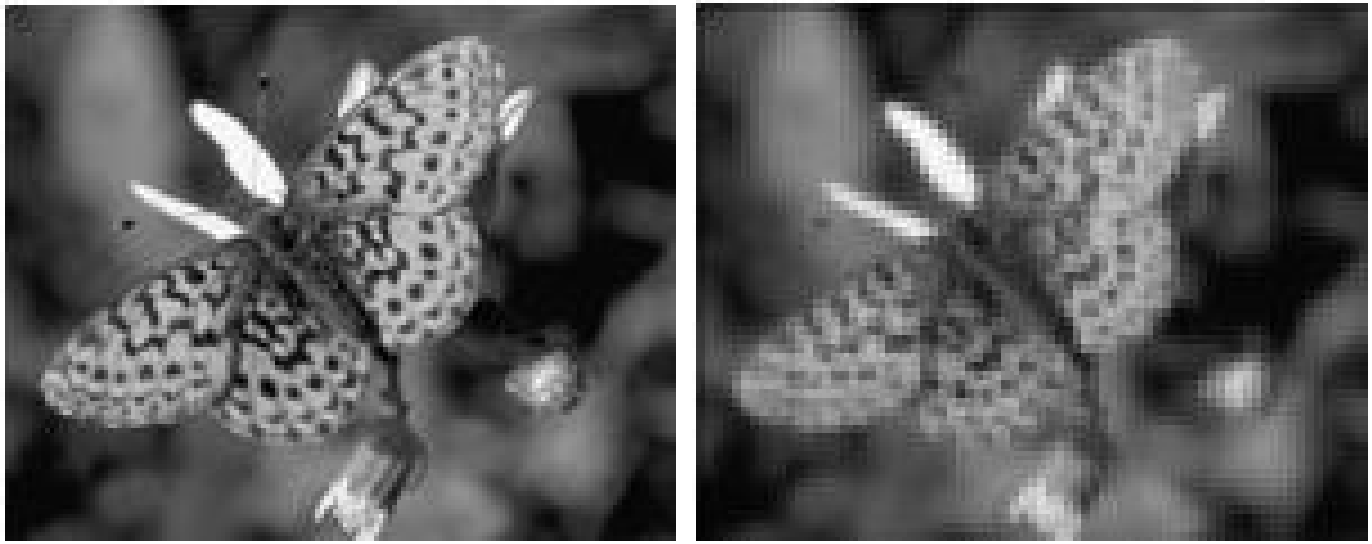
- Several filters $(f * h_1, \ldots, f * h_n)$.
- Non-linear transformation, e.g. squares, absolute values, taking the positive or negative part of a signal, thresholding.
- Classification (more theory on classification in later lectures).

Examples of filters and filter banks

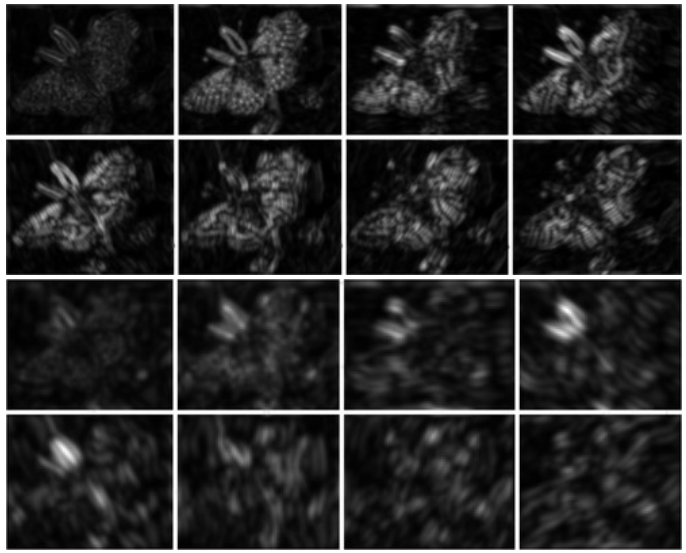
- **Spots:**
- **Bars:**
- **Gabor filters:**



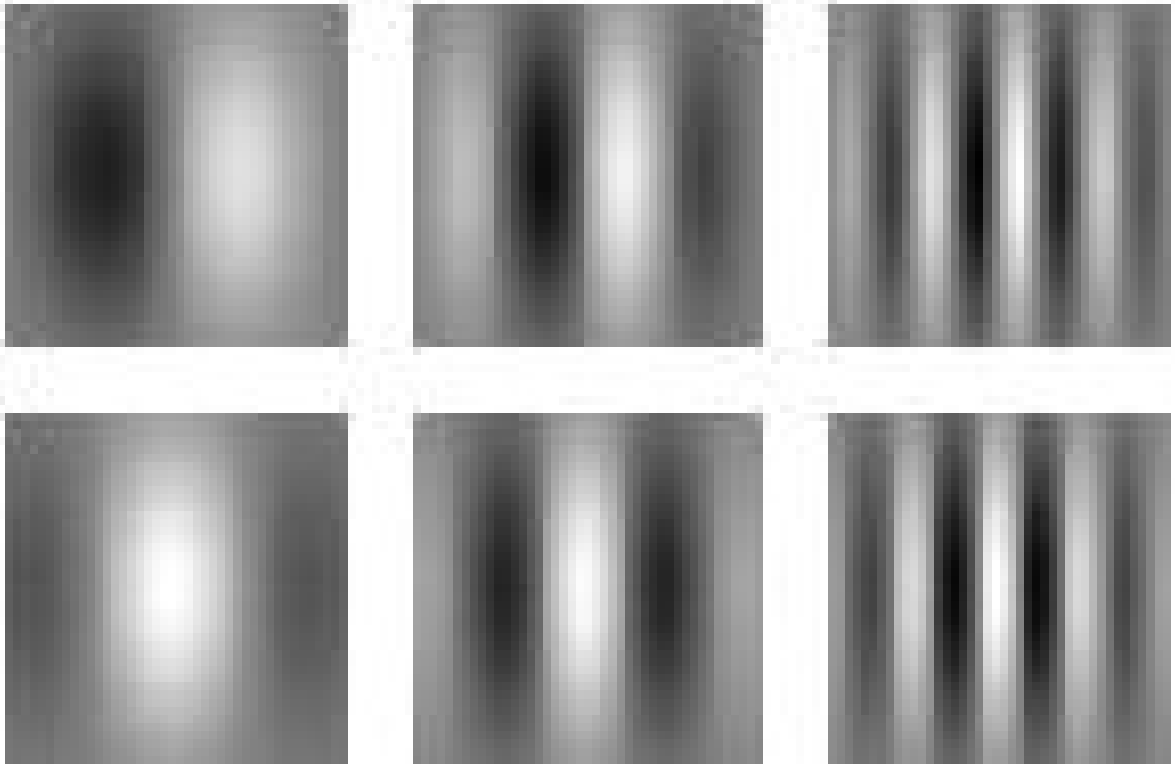
Search (for texture) on different scales



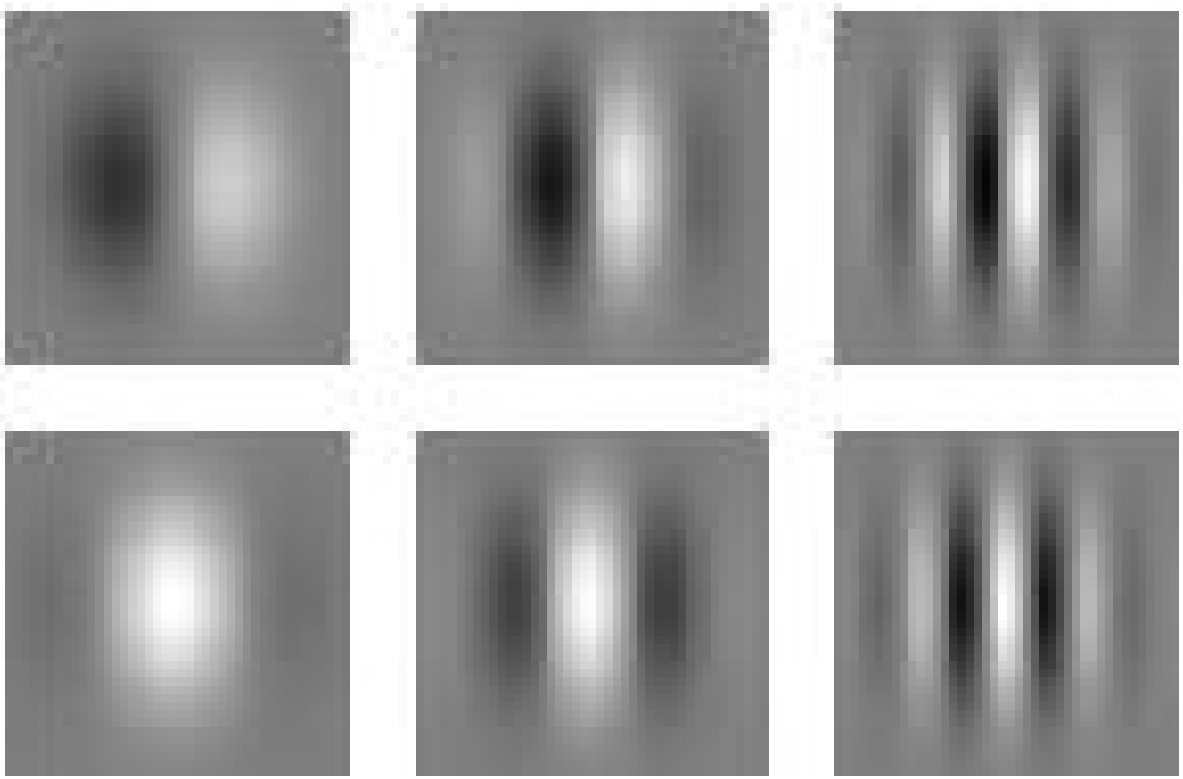
Response for different filters



Gabor filter



Gabor filters (ctd.)



Non-linear transformations

You can try several non-linear transformations, e.g.

- Squaring, polynomials
- Absolute value
- Thresholding (in particular taking the positive and negative parts of a signal).
- (Later: Use machine learning, week 5 in the course)

Mean value filtering

After non-linear transformation, often it is a good idea to form the mean over a region, e.g. using mean value filtering. Similar to what we did with the orientation tensor.

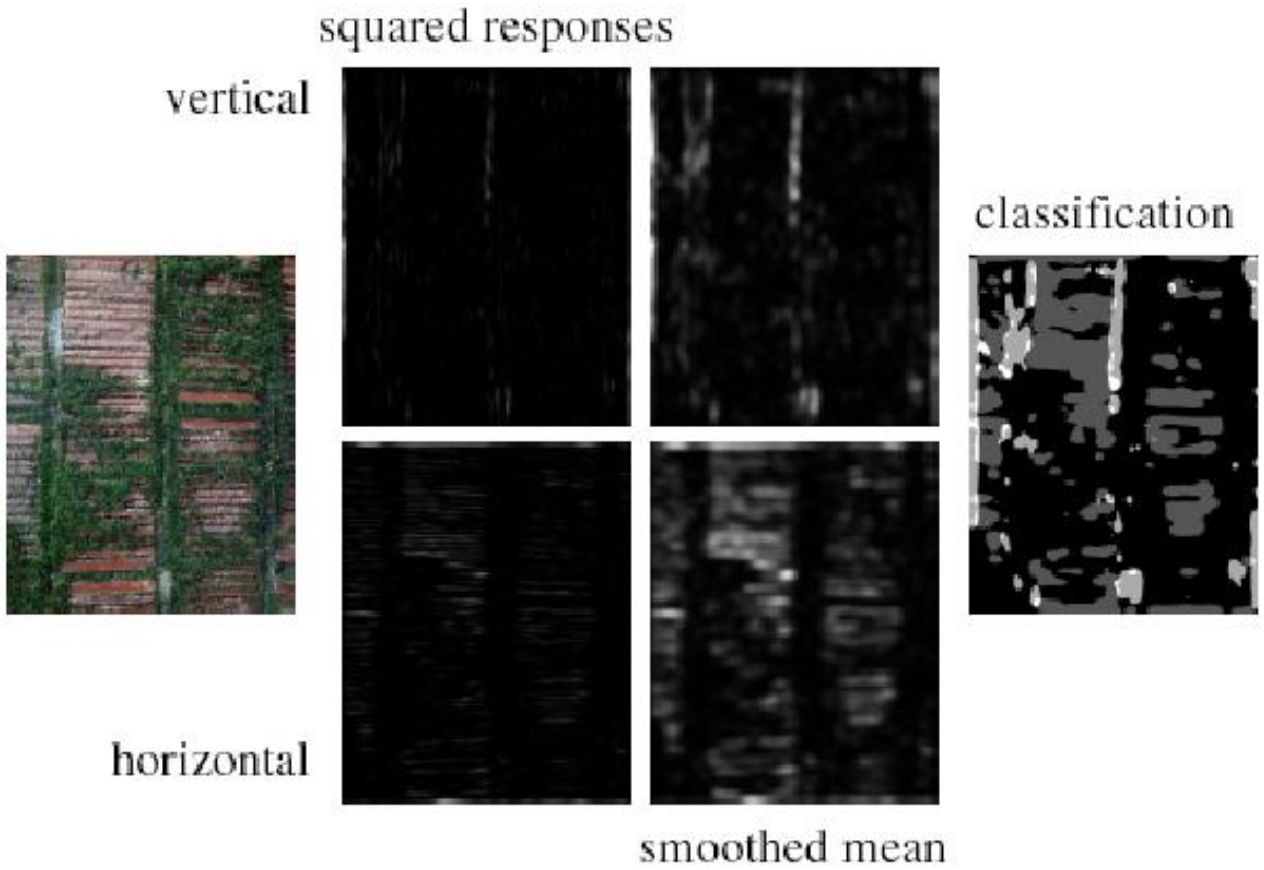
General idea

The result is a number of values (channels, features) for every pixel.
The goal is now to classify pixels in different classes (textures).
Classification will be further studied in a later lecture.
One could possibly use information from neighboring pixels as well.

Choice of features

One possibility is to take the mean and variance of the channel within a region.
It might be desirable that the response in the channels were uncorrelated.

Example of texture classification



What is segmentation?

Goal: Segment the image into pieces/segments, i.e. regions that belong to the same object or that has the same properties. Can also be seen as a problem of 'grouping' of pieces (pixels, regions) together.

Examples of segmentation

Some typical segmentation problems are:

- Cut an image sequence into shots
- Find manufactured parts in an industrial environment
- Find humans in images and video
- Find houses in satellite images
- Find faces in images

Example: OCR.
Example: Image interpretation
Example: Road user analysis
Example: Medical Image Analysis, e.g. Cell analysis

Segmentation using clustering

Using clustering:

- Segment images into pieces
- Fit lines to a set of points
- Fit a fundamental matrix to image pairs

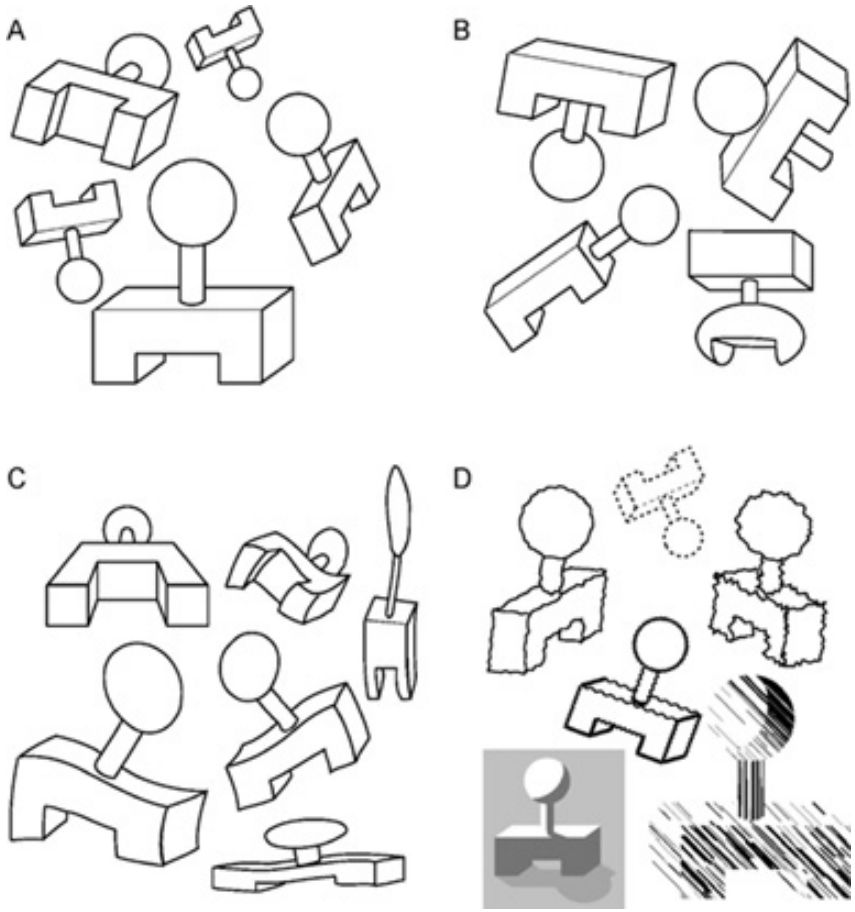
In some cases it is easier to view segmentation as the problem of putting pieces together. This is usually called **grouping** (less precise) or **clustering** (which has a precise meaning in the field of pattern recognition).

Gestalt laws

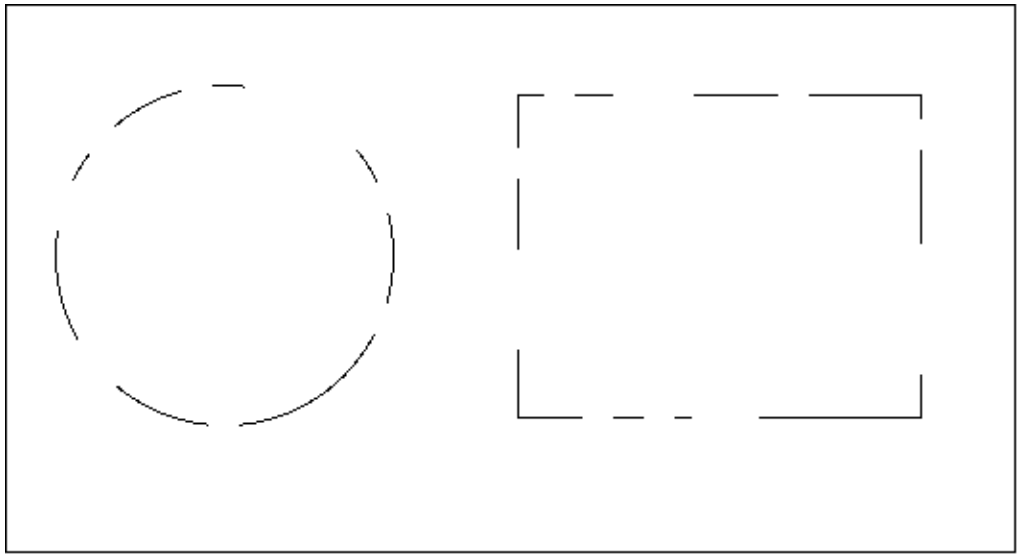
Around 1900 the 'Gestalt' theory was developed by psychologists in Germany, the Berlin school. They developed a descriptive theory of mind and brain. Some principles that they discovered for human grouping of features are:

- Proximity
- Similarity
- Same fate
- Same region
- Closedness
- Symmetry
- Parallelism

Invariance



Closure



Split and Merge

Goal: To segment the image R into n regions $R_1, R_2, \ldots, R_n$.
Use a criteria for what a region is: P

- $\cup R_i = R$
- R_i connected
- $R_i \cap R_j = \emptyset, \quad i \neq j \quad (R_i$ disjoint)
- $P(R_i)$=TRUE, $i = 1, \ldots, n$
- $P(R_i \cup R_j)$=FALSE, $i \neq j$.

Pixel-aggregation

Example

$P(R_i)$=TRUE if the difference in intensities is ≤ 3 gray-levels. Initialize with some points and add further points in each iteration.

■

Split and Merge (ctd.)

Idea: Merge areas that are "similar" and split areas that are "different".
The criteria P determines what is similar and what is different.

Algorithm

(Split and Merge) Given an image R and a property P.

1. *Start with the division $R = \{R_1\}$*
2. *If $P(R_i)$ =FALSE: Split R_i into four smaller regions (SPLIT)*
3. *If $P(R_i \cup R_j)$=TRUE for two adjacent regions R_i and R_j: Merge R_i and R_j to one region (MERGE)*
4. *Continue with step 2 and 3 until convergence*

Foreground Background estimation

- Study an image sequence
- Introduce a model for background and estimate its parameters
- Estimate which pixels deviate from the background model

Example: Road user analysis.

Video shots

- Study an image sequence
- For every pair of images in a sequence
  - Calculate the distance between the images
  - Cut the sequence into shots where the distance is too large

Image metrics

Examples of metrics are:

- Form the difference between the images and calculate some norm, e.g. the 2-norm (the square root of the sum of the squares)
- Histogram comparison. Calculate the histogram of the two images. Calculate a distance based on the histograms.
- Block comparison. Divide the image into a number of blocks (e.g. 8×8 pixels) and compare these.
- Edge comparison. Find edges in both images and see if they are roughly in the same position (context) in both images.

Clustering

Goal: Partition a set on n feature vectors with d components

$$x_1, \ldots x_n$$

i.e. $x_i \in \mathbb{R}^d$, in groups (clusters) such that all examples in the same group are similar.

Classification

Goal: Given a number of examples, that already are partitioned into groups

$$(x_1, f_1), \ldots (x_n, f_n)$$

construct a function

$$R : x \longrightarrow f$$

such that $R(x_i) = f_i$ as good as possible
Here $x_i \in \mathbb{R}^n$ are examples and $f_i \in \mathbb{Z}$ is a number representing, which class/group it belongs to.

The Clustering Problem

Input: n examples $x_1, \ldots x_n$.
Output: A mapping

$$c : \{1, \ldots, n\} \mapsto \{1, \ldots k\}.$$

Example: **K-means algorithm**
Choose k cluster centres $m_1, \ldots, m_k$ and a clustering function c that minimises

$$f(c, m) = \sum_{i=1}^{n} |x_i - m_{c(i)}|^2.$$

K-means

The problem

$$\min_{c,m} f(c,m) = \min_{c,m} \sum_{i=1}^{n} |x_i - m_{c(i)}|^2$$

is a non-linear optimization problem that can be solved with many methods. However, most only give a local optima.

K-means implementation

One popular implementation is the following

1. Randomly choose k cluster centra (e.g. k examples)
2. **Optimize** c**:** For every example x_i assign $c(i)$ such that $|x_i - m_{c(i)}|$ is minimized, i.e.

$$c = \text{argmin}_c f(c, m).$$

3. **Optimize** m**:** For every group j change m_j as the centre of mass of corresponding examples, i.e.

$$m = \text{argmin}_m f(c, m).$$

Hierarchical methods

A popular clustering method is hierarchical clustering.
Start with one cluster or regard each point as a separate cluster.
Reduce one after one or merge one after one.
Stop when you are finished.

1. Start with n clusters
2. Choose error criteria, e.g. $f(c, m)$ as above
3. Merge the two clusters that minimizes the error criteria (or split the cluster with the biggest error criteria)
4. If the number of clusters is more than k go to 3.

The result is a clustering for every number of clusters, k, between 1 and n. This can often be represented in a so called dendrogram, where you map error criteria and clustering.

There are **many** other methods

- EM-method (Similar to k-means, but better (and slower))
- k-mediods
- adaptive resonance theory
- fuzzy clustering
- auto-class
- mode-separator
- Self Organizing Maps
- Agglomerative hierarchical clustering
- Divisive hierarchical clustering
- Iso-map

Graph Theory

A graph $G = (V, E)$ consists of vertices(nodes) V and edges E.
Every edge connects two vertices.
In a directed graph, every edge has an orientation.
In a weighted graph, every edge has a weight (a number).
A graph is connected if one can 'walk' between all pairs of vertices through one or several edges.
Every graph can be split into a disjoint set of connected components.

Matrix representation

Weighted graphs can be represented as a matrix. A weighted edge between vertex i and vertex j with v is represented by matrix element (i, j).
For un-directed graphs, half the weight is put at position (i, j) and half in (j, i).
Connected components - blocks in block diagonal matrices.

Affinity measures

When solving clustering problems with graph theoretical methods one need a closeness measure $v_{i,j}$, for every pair of nodes (i, j). A large number means that they are close. A small number means that they are different.
The affinity measure depends on which problem one has. Usual ingredients are

- Distance - e.g. $aff(x, y) = e^{-(x-y)^T(x-y)/(2\sigma_d^2)}$
- Intensity - e.g. $aff(x, y) = e^{-(I(x)-I(y))^T(I(x)-I(y))/(2\sigma_I^2)}$
- Color - e.g. $aff(x, y) = e^{-dist(c(x),c(y))^2/(2\sigma_c^2)}$
- Texture - e.g. $aff(x, y) = e^{-(f(x)-f(y))^T(f(x)-f(y))/(2\sigma_f^2)}$

Eigenvectors and segmentation

Assume that w_n is a vector of ones for those elements that belong to a particular cluster and zeros otherwise. Then the sum of all weights for edges within a cluster is

$$w_n^T A w_n.$$

By maximizing $w_n^T A w_n$ with the constraint $w_n^T w_n = 1$ one might argue that we maximize clustering.
Maxima with this problem corresponds to stationary points of the Lagrange function.

Eigenvectors and segmentation

Maximize $w_n^T A w_n$ with constraint $w_n^T w_n = 1$.
Study the Lagrange function

$$L(w_n, \lambda) = w_n^T A w_n + \lambda(w_n^T w_n - 1).$$

Differentiate and divide with two gives

$$A w_n = -\lambda w_n.$$

This is an eigenvalue problem.

Cut-methods for segmentation

- Popular method: Normalized cuts. Similar to previos graph method, but objective is modified to avoid small clusters. See Section 14.5.5. Works poorly sometimes.
- Even more popular method: Graph cuts. Works very well. See "Graph cuts homepage" on the internet for recent applications and tutorials. Next time.
- Very efficient methods exist to find global minima for graph cuts

Masters thesis suggestion of the day: Road user

There are several interesting and useful applications of algorithms for automatic detection and tracking of road users (cars, bicycles, pedestrians). Two such uses are (i) to increase safety in traffic and (ii) decrease the environmental load of traffic.

Recommended reading

- Forsyth & Ponce: **9. Texture**, **14. Segmentation**.
- Szeliski: **5. Segmentation**.

Review - Lecture 3

- Texture recognition
  - Filter bank
  - Nonlinear transformations
  - Mean value filtering
  - Classification
- Segmentation
  - Gestalt laws
  - Clustering
  - Graph theoretical clustering