# Image Processing

Image Features

# Preliminaries

What are Image Features? **Anything**.

What they are used for?

- Some statements about image fragments (**patches**) – recognition
- Search for similar patches – matching

→ Both mean the ability to **compare** patches: either a patch to a template (recognition) or patches to each other (matching)

→ Image (patch) feature is a (real-valued) function $f(\text{"patch"})$

Problem: very high dimension, huge (potentially infinite) number of possible features → **Feature Selection**

Techniques: Principal Component Analysis, Clustering, …

# Simple Features

1. The feature of a patch is the patch itself (seen as vector) – “a full” description (raw data).

2. Features are values that were used for interest points detection:

   a) For instance the eigenvalues of the auto-correlation function from the Harris detector (or eigenvectors as well)

   b) For MSER-s – much more, since regions are detected → both coloring properties (mean color, standard deviation, stability …) and geometric ones (size, perimeter, curvature, convexity …) can be used.

# SIFT

First of all:

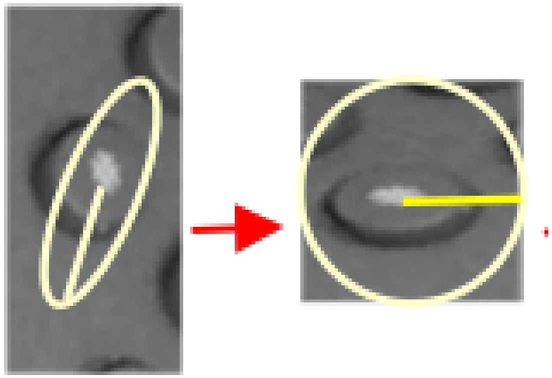
1. Interest points are detected

2. The patches around are **normalized** (often by an affine transformation, sometimes by a projective one)

A usual normalization:

- scale to a predefined size
- rotate to a predefined direction (e.g. “east”)

(auto-correlation can be used)



# SIFT

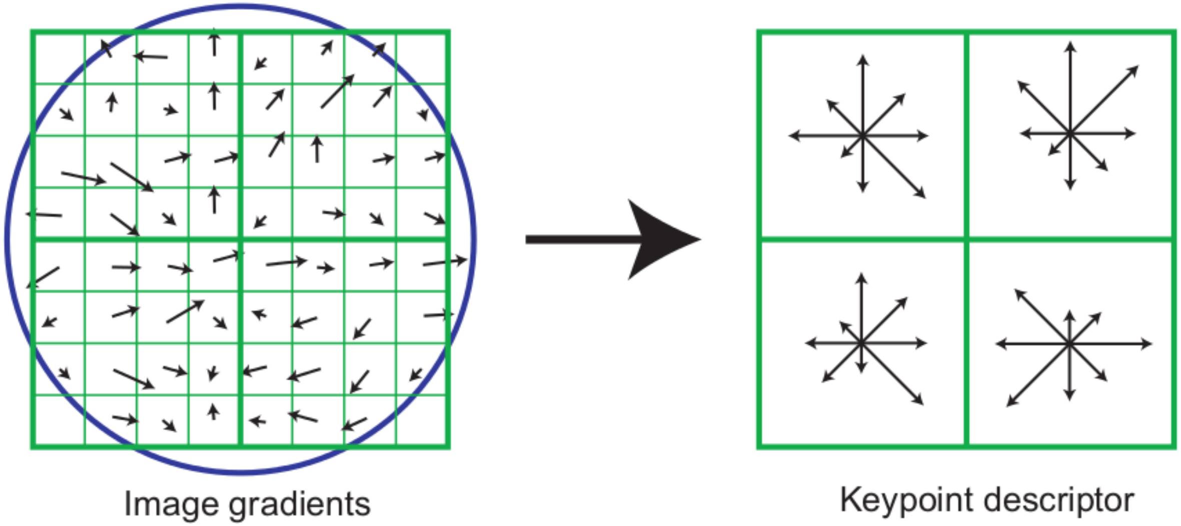


Figure 7: A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2x2 descriptor array computed from an 8x8 set of samples, whereas the experiments in this paper use 4x4 descriptors computed from a 16x16 sample array.

# SIFT

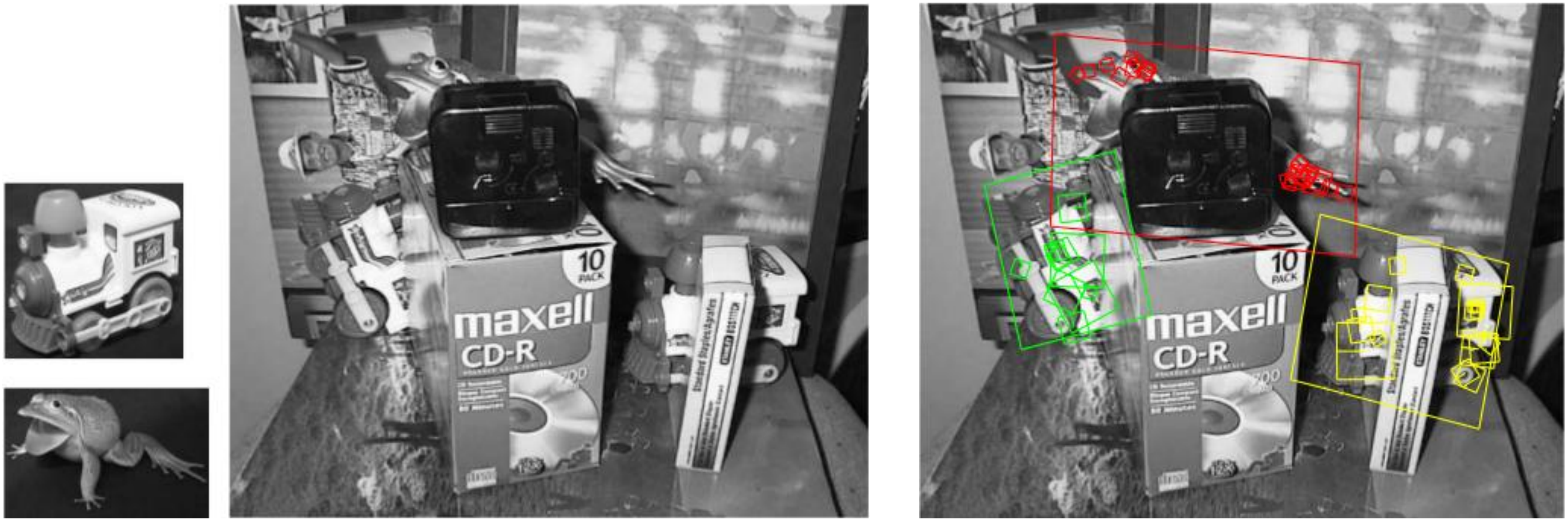


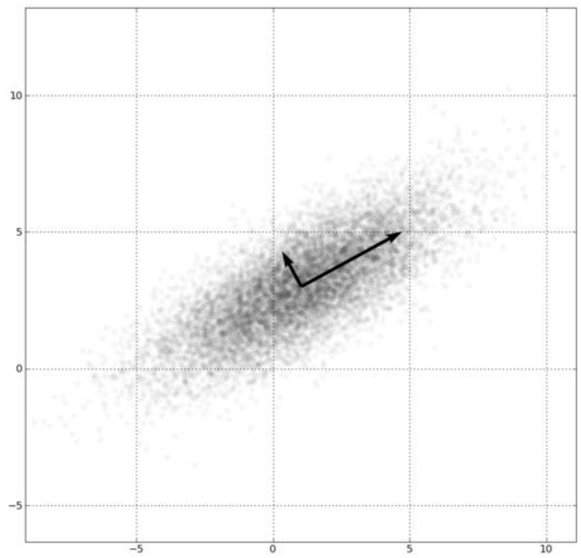
Figure 12: The training images for two objects are shown on the left. These can be recognized in a cluttered image with extensive occlusion, shown in the middle. The results of recognition are shown on the right. A parallelogram is drawn around each recognized object showing the boundaries of the original training image under the affine transformation solved for during recognition. Smaller squares indicate the keypoints that were used for recognition.

# Principal Component Analysis

**Problem** – high dimension:

- Feature is the (5x5) patch → feature vector is in $\mathbb{R}^{25}$
- SIFT is composed of 16 histograms of 8 directions → vector in $\mathbb{R}^{128}$

**Idea** – the feature space is represented in another **basis**.



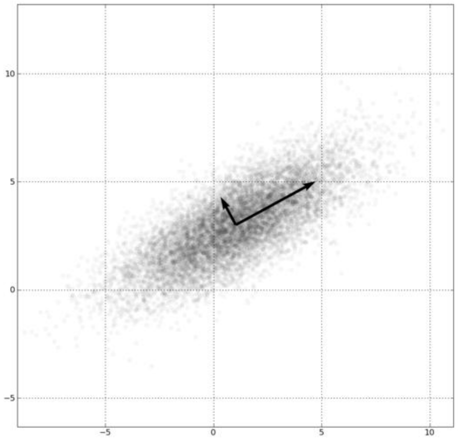
**Assumption**: the directions of small variances correspond to noise and can be neglected

**Approach**: project the feature space onto a linear subspace so that the variances in the projected space are maximal

# Principal Component Analysis

A simplified example: data are centered, the subspace is one-dimensional, i.e. it is represented by a vector $\|e\|^2=1$. Projection of an $x$ on $e$ is $\langle x,e\rangle$. Hence, the task is

$$\sum_l \langle x^l,e\rangle^2 \to \max_e \qquad \text{s.t. } \|e\|^2=1$$



Lagrangian:

$$\sum_l \langle x^l,e\rangle^2 + \lambda\big(\|e\|^2-1\big) \to \min_\lambda \max_e$$

Gradient with respect to $e$:

$$\sum_l 2\langle x^l,e\rangle\cdot x^l + 2\lambda e = 2e\sum_l x^l\otimes x^l + 2\lambda e = 0$$

$$e\cdot cov = \lambda e$$

# Principal Component Analysis

$$e \cdot cov = \lambda e$$

→ $e$ is an eigenvector and $\lambda$ is the corresponding eigenvalue of the **covariance matrix**. Which one?

For a pair $e$ und $\lambda$ the variance is

$$\sum_l \langle x^l, e \rangle^2 = e \cdot \sum_l x^l \otimes x^l \cdot e = e \cdot cov \cdot e = ||e||^2 \cdot \lambda = \lambda$$

→ chose the eigenvector corresponding to the **maximal** eigenvalue.

Similar approach: project the feature space into a subspace so that the summed squared distance between the points and their projections is minimal → the result is the same.

# Principal Component Analysis

Summary (for higher-dimensional subspaces):

1. Compute the covariance matrix $cov = \sum_l x^l \otimes x^l$
2. Find all eigenvalues and eigenvectors
3. Sort the eigenvalues in decreasing order
4. Choose $m$ eigenvectors for the $m$ first eigenvalues (in the order)
5. The $n \times m$ projection matrix consists of $m$ columns, each one is the corresponding eigenvector.

Are projections onto a **linear** subspace good? Alternatives?

# Function Spaces (recall)

Images are functions (continuous domain) $I : D \subset \mathbb{R}^2 \to C$

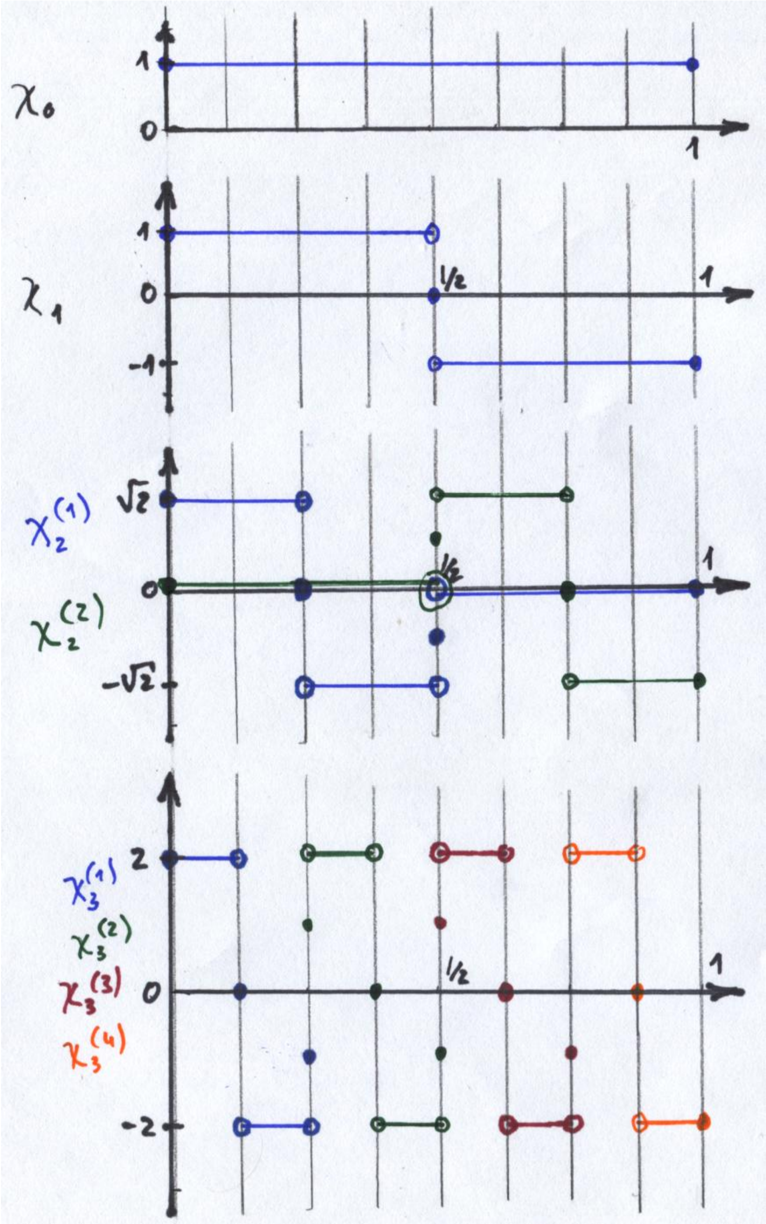
| | Vector<br>$v = (v_1, v_2 \dots v_n)$ | Function<br>$f(x),\ x \in \mathbb{R}$ |
|---|---|---|
| Domain | $\{1, 2 \dots n\}$ | $\mathbb{R}$ |
| Mapping | $\{1, 2 \dots n\} \to \mathbb{R}$ | $\mathbb{R} \to \mathbb{R}$ |
| Space | $\mathbb{R}^n$ | $\mathbb{R}^\infty$ |
| Scalar product | $\langle u, v \rangle = \sum_i u_i v_i$ | $\int f(x) g(x) dx$ |
| Length | $\sum_i v_i^2 = \langle v, v \rangle$ | $\int f(x)^2 dx$ |

The task is to decompose a patch $f(x)$ in **base-functions**, i.e.

$$f(x) = \int_y v(x, y) \lambda(y) dy$$

Example: Fourier transform ($\cos(yx)$ and $\sin(yx)$). Other useful bases?

# Haar base functions (1D)



$$\chi_0(s) = 1, 0 \leq s \leq 1$$

$$\chi_1(s) = \begin{cases} 1 & 0 \leq s < 1/2 \\ -1 & 1/2 < s \leq 1 \end{cases}$$

$$\chi_2^{(1)}(s) = \begin{cases} \sqrt{2} & 0 \leq s < 1/4 \\ -\sqrt{2} & 1/4 < s < 1/2 \\ 0 & \text{sonst} \end{cases}$$

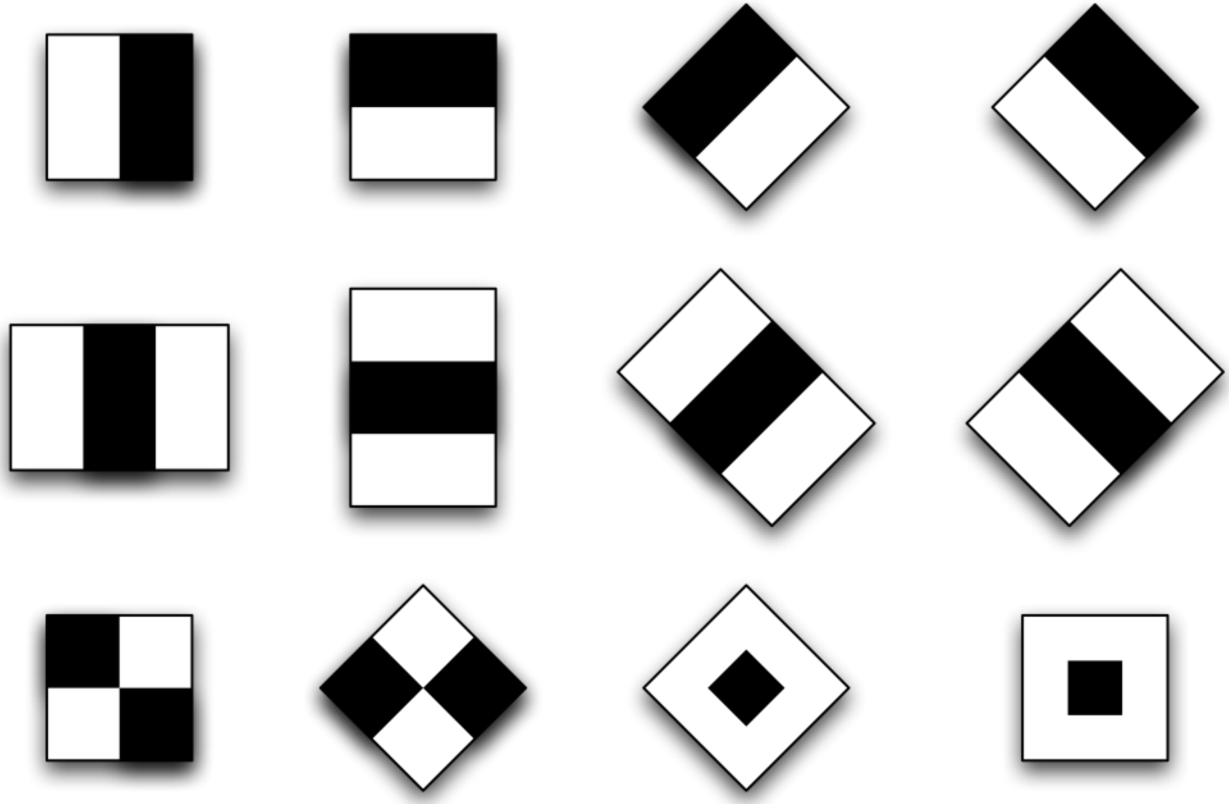
$$\chi_2^{(2)}(s) = \begin{cases} \sqrt{2} & 1/2 < s < 3/4 \\ -\sqrt{2} & 3/4 < s \leq 1 \\ 0 & \text{sonst} \end{cases}$$

$$\chi_3^{(1)}, \chi_3^{(2)}, \chi_3^{(3)}, \chi_3^{(4)}$$

etc.

# Haar base functions (2D)

Convolution kernels (black/white, ±1), responses are **Haar features**:



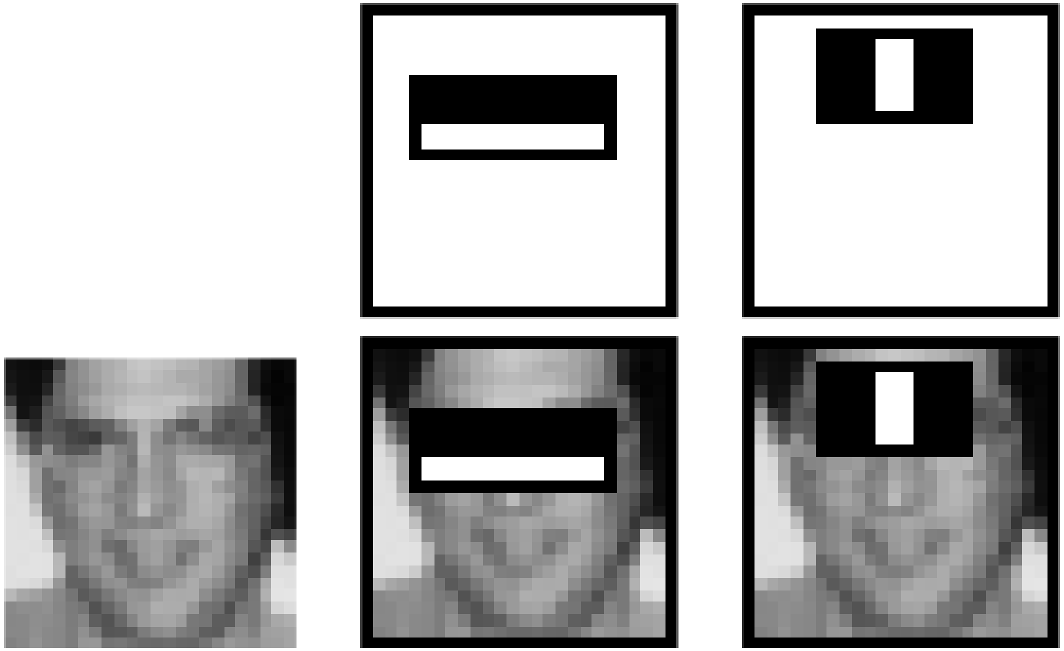
Can be computed very efficiently using the “Integral Image” approach (see the “Filtering” lecture).

# Viola & Jones (CVPR 2001)

Haar features – efficient computation

24x24 window, an arbitrary feature →180.000 features per position,

AdaBoost for learning (later)

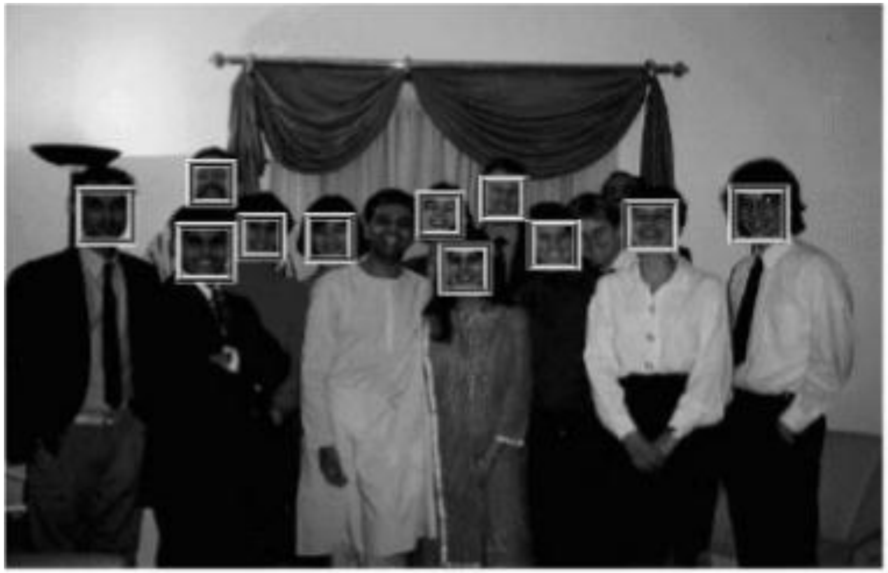


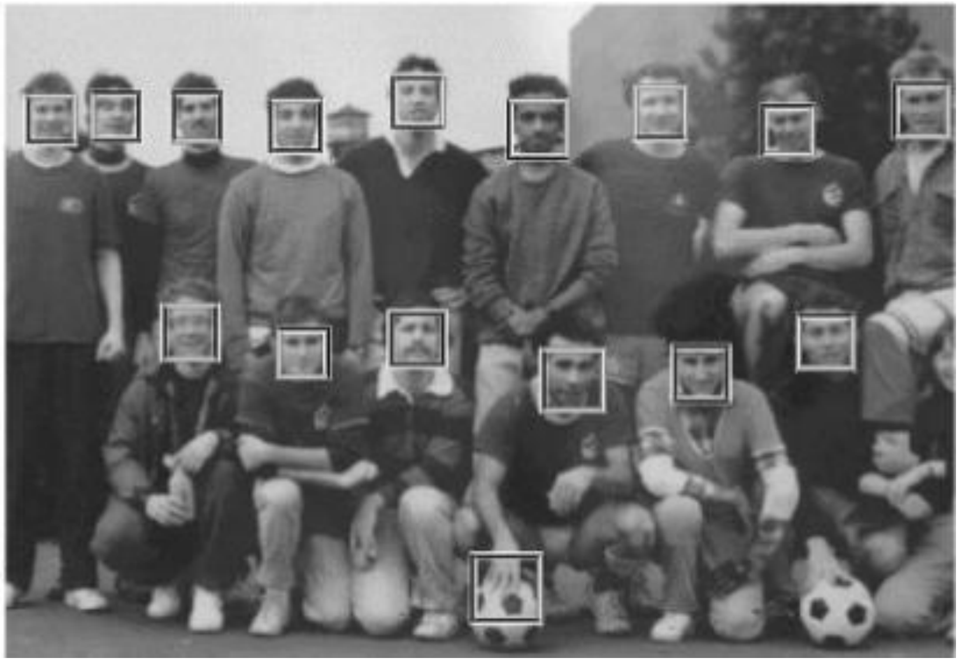
The first two best features

There are of course some more things in the paper.

# Viola & Jones (CVPR 2001)

Database: 130 Images, 507 faces





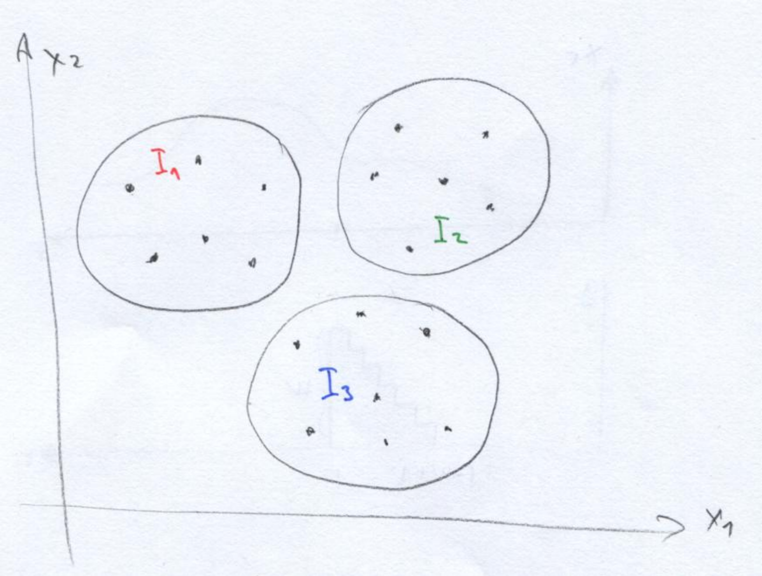
Recognition rate: about 7%

# Clustering

The task: partition a set of objects into "meaningful" subsets (clusters). The objects in a subset should be "similar".

Notations:

| | |
|---|---|
| Set of Clusters | $K$ |
| Set of indices | $I = \{1, 2, \dots, \|I\|\}$ |
| Feature vectors | $x^i,\ i \in I$ |



Partitioning

$$C = (I_1, I_2, \dots, I_{|K|}),\ I_k \cap I_{k'} = \emptyset \text{ for } k \neq k',\ \bigcup_k I_k = I$$

# Clustering

Let $x^i \in \mathbb{R}^n$ and each cluster has a “representative” $y^k \in \mathbb{R}^n$

The task reads:

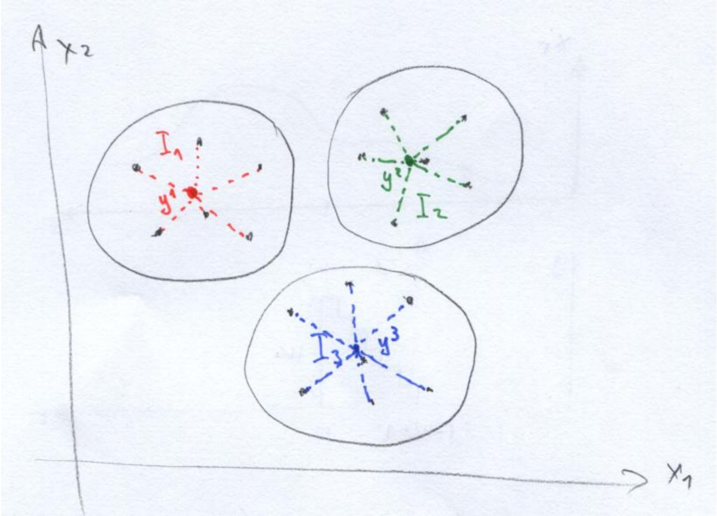
$$\sum_k \sum_{i \in I_k} \|x^i - y^k\|^2 \to \min_{C,y}$$

Alternative variant is to consider the clustering $C$ as a mapping $C : I \to K$ that assigns a cluster number to each $i \in I$

$$\sum_i \|x^i - y^{C(i)}\|^2 \to \min_{y,C}$$

$$\sum_i \min_k \|x^i - y^k\|^2 \to \min_y$$

# K-Means Algorithm



Initialize centers randomly,

Repeat **until convergence**:

1. Classify:

$$C(i) = \arg\min_{k'} \|x^i - y^{k'}\|^2 \quad \Rightarrow \quad i \in I_k$$

2. Update centers:

$$y^k = \arg\min_{y} \sum_{i \in I_k} \|x^i - y\|^2 = \frac{1}{|I_k|} \sum_{i \in I_k} x^i$$

- The task is NP
- converges to a local optimum (depends on the initialization)

# Some variants

Other distances, e.g. $||x^i - y^k||$ instead of $||x^i - y^k||^2$

In the K-Means algorithm the classification step remains the same, the update step – the geometric median of $x^i,\ i \in I_k$

$$y_k = \arg\min_y \sum_{i \in I_k} ||x^i - y||$$

(a bit complicated as the average ☹).

Another problem: features may be not additive ( $y^k$ does not exist)

Solution: K-Medioid Algorithm ( $y^k$ is a feature vector from the training set)

# A generalization

Observe (for the Euclidean distance):

$$\sum_i \|x^i - \bar{x}\|^2 \sim \sum_{ij} \|x^i - x^j\|^2$$

In what follows:

$$\sum_k \sum_{ij \in I_k} \|x^i - x^j\|^2 = \sum_k \sum_{ij \in I_k} d(i,j) \to \min_C$$

with a Distance Matrix $d$ that can be defined in very different ways.

Example: Objects are nodes of a weighted graph, $d(i,j)$ is the length of the shortest path from $i$ to $j$.

Distances for “other” objects (non-vectors):

- Edit (Levenshtein) distance between two symbolic sequences
- For graphs – distances based on graph isomorphism etc.

# Back to image features

Assumption: the set of all patches can be partitioned into subsets.

Each subset has a representative – an “ideal” patch.

All patches are noisy variants of the corresponding representatives.

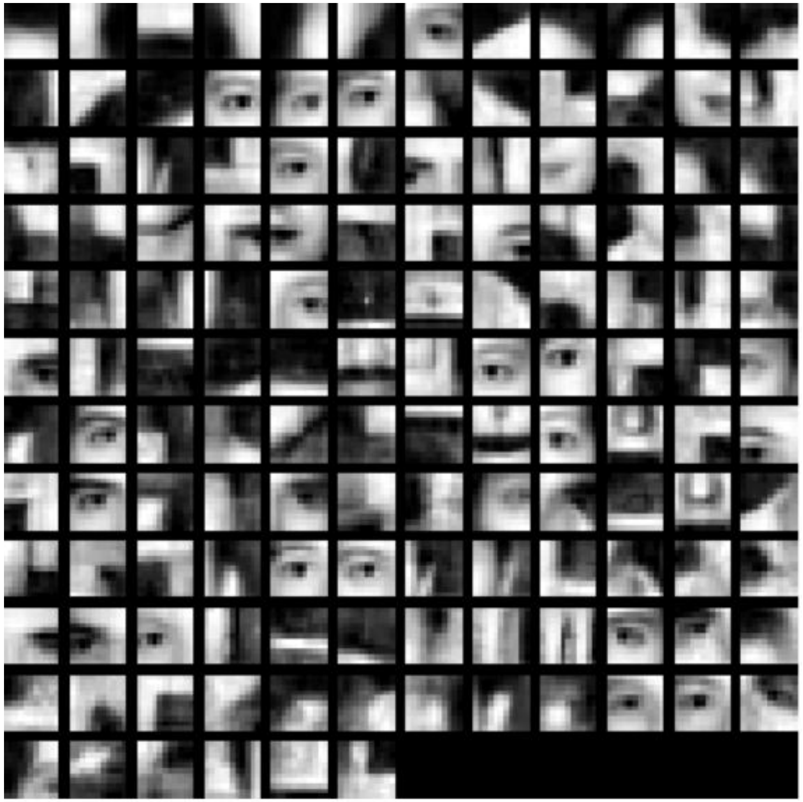
The feature of a patch is its representative.

The task is to find the partitioning and the representatives using a dataset of patches.

→ Clustering (patches are considered as vectors, e.g. squared Euclidean norm for distances).

# Visual words

Consider patches around interest points in a dataset and cluster them.



Dataset → Cluster centers

The feature is the number of the corresponding representative.

# Conclusion

- Simple features, SIFT, Haar-features
- Feature selection/reduction – PCA, Clustering

Literature:

- Alfred Haar in Göttingen: Zur Theorie der orthogonalen Funktionensysteme (Erste Mitteilung); On the Theory of Orthogonal Function Systems (First communication)
- David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints
- Viola & Jones: Rapid Object Detection using a Boosted Cascade of Simple Features (CVPR 2001)