Computer Vision I

View-Based Modeling - 15.05.2013

TECHNISCHE UNIVERSITÄT DARMSTADT

vi visual inference

Announcements

TECHNISCHE UNIVERSITÄT DARMSTADT

- **No class next week!**
  - Project review meeting

- Homework assignment 1:
  - Is out...
  - Signup for the “Testat” current read only -- will go online again soon

- **Exam**
  - 21. August 2013, 10:00 - 12:00

Wrap up - Basics of Digital Image Processing

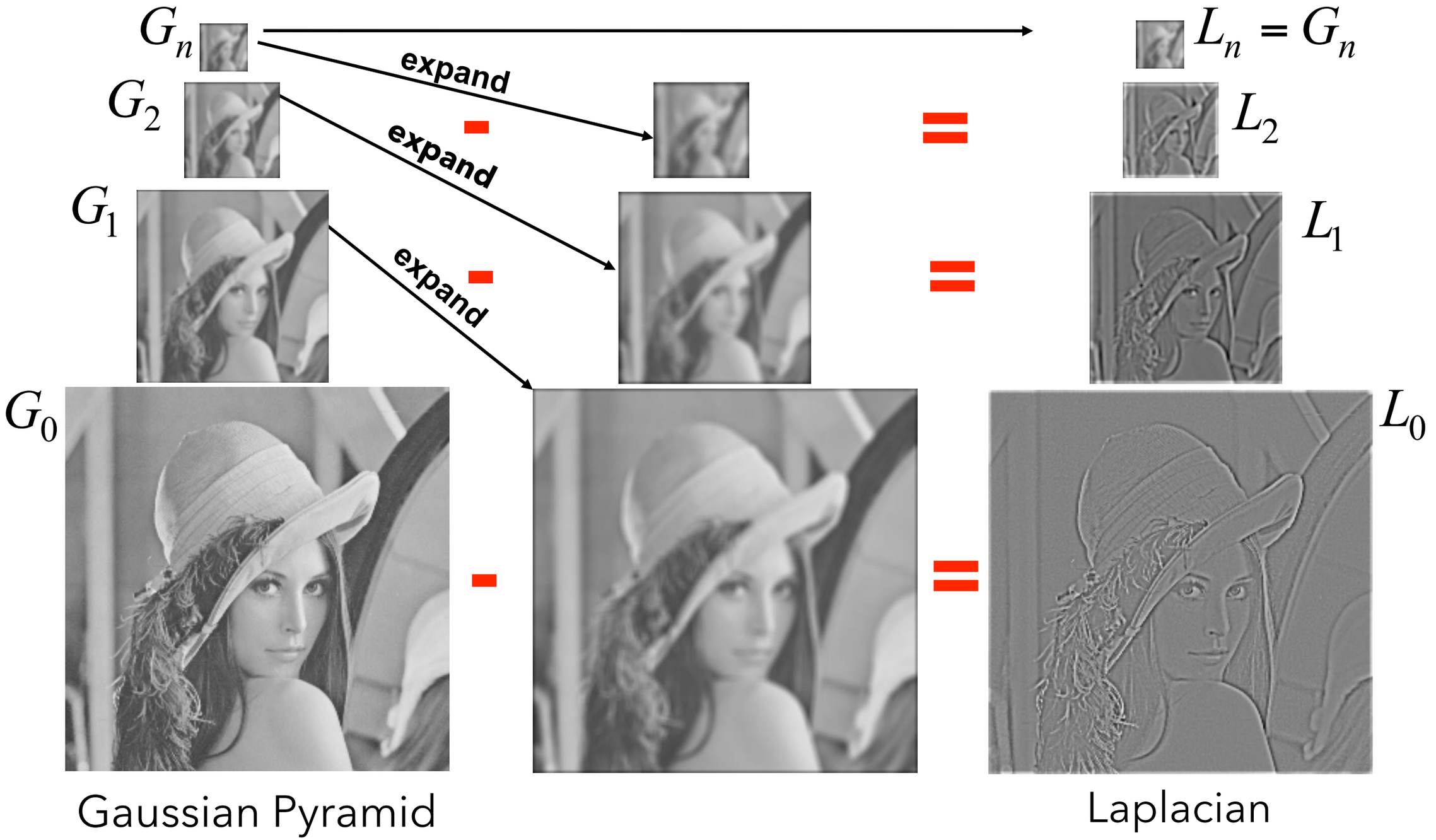
TECHNISCHE UNIVERSITÄT DARMSTADT

- Images
- Filtering
  - Linear filtering
  - Non-linear filtering & morphology
- Multi-scale image representation
  - Gaussian pyramid
  - Laplacian pyramid
- Edge detection
  - ‘Recognition using line drawings’
  - Image derivatives (1st and 2nd order)

Laplacian pyramid

TECHNISCHE UNIVERSITÄT DARMSTADT

$$L_i = G_i - \text{expand}(G_{i+1}) \qquad G_i = L_i + \text{expand}(G_{i+1})$$



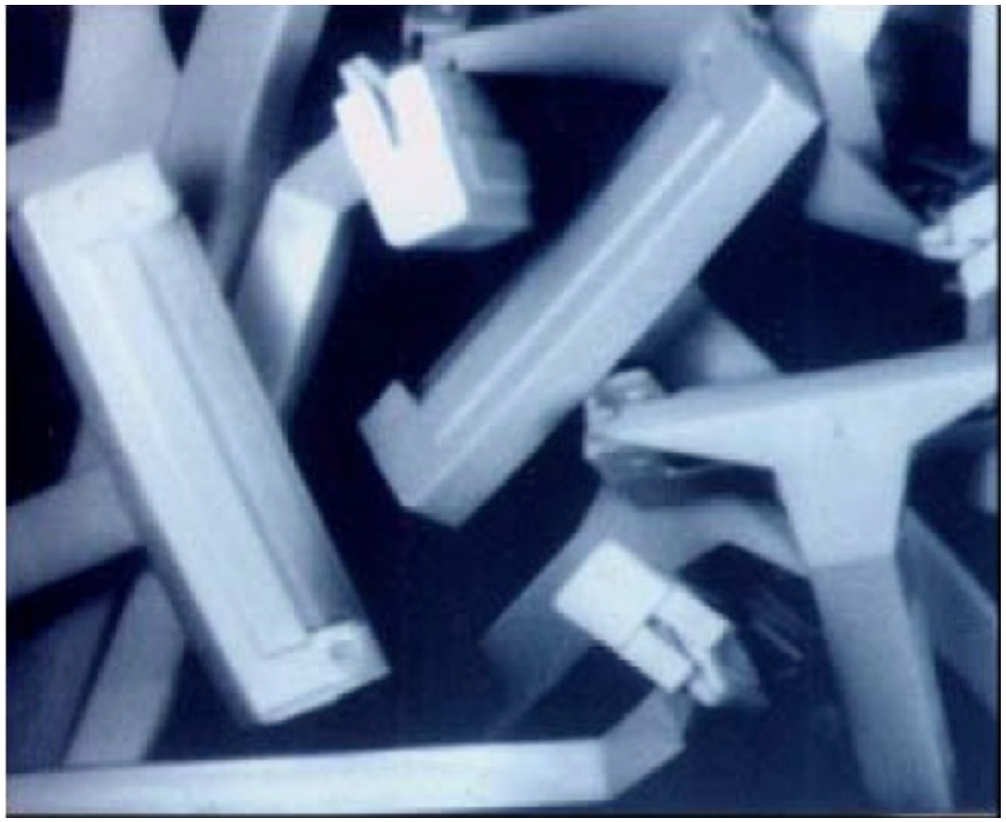
Laplacian pyramid

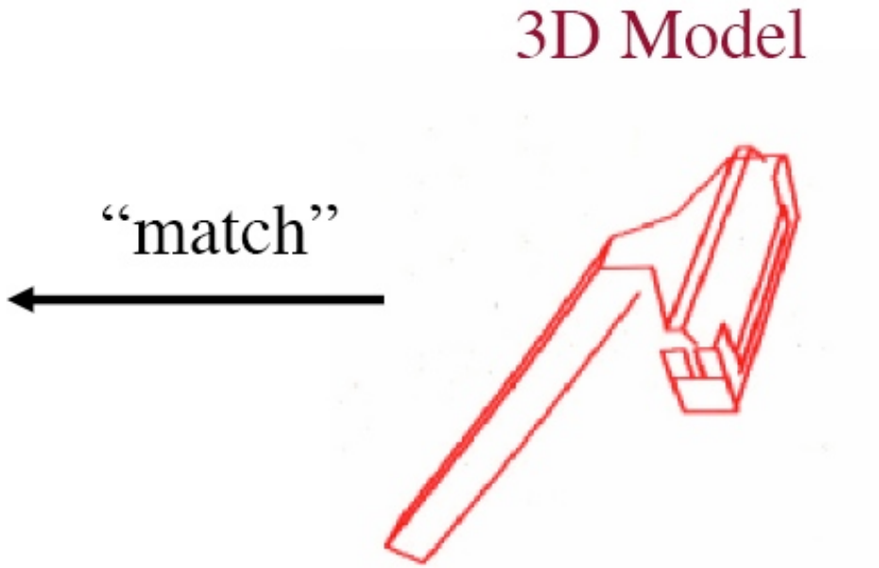
TECHNISCHE UNIVERSITÄT DARMSTADT

- The Laplacian pyramid provides a simple frequency decomposition into subbands.
- Each level / subband contains only image structure of a particular range of spatial frequencies:
  - E.g., the finest level contains all the high-frequency detail.
- Observation: We can get back the original image by reversing the decomposition.

Example of Recognition & Localization

- Simple approach [David Lowe]



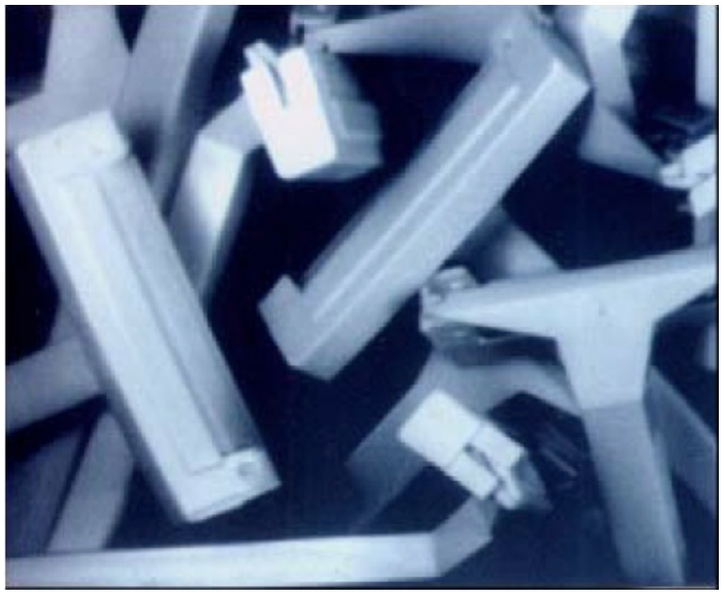


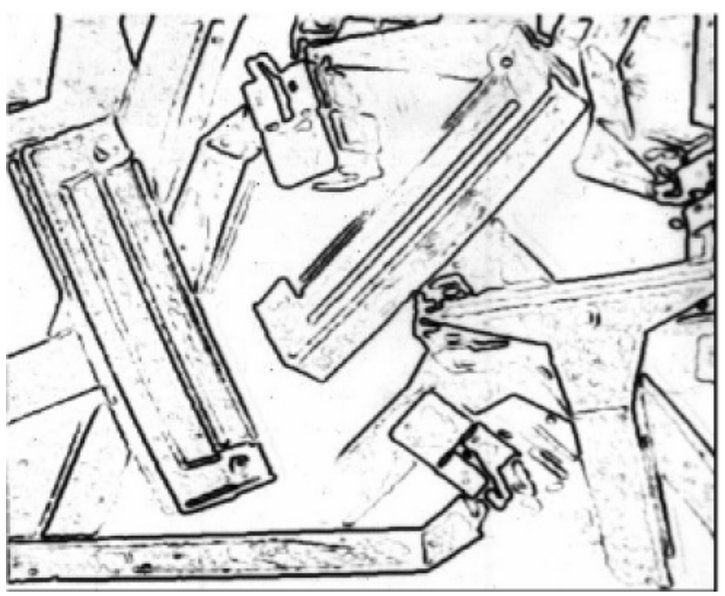
Parameters: 3D position and orientation

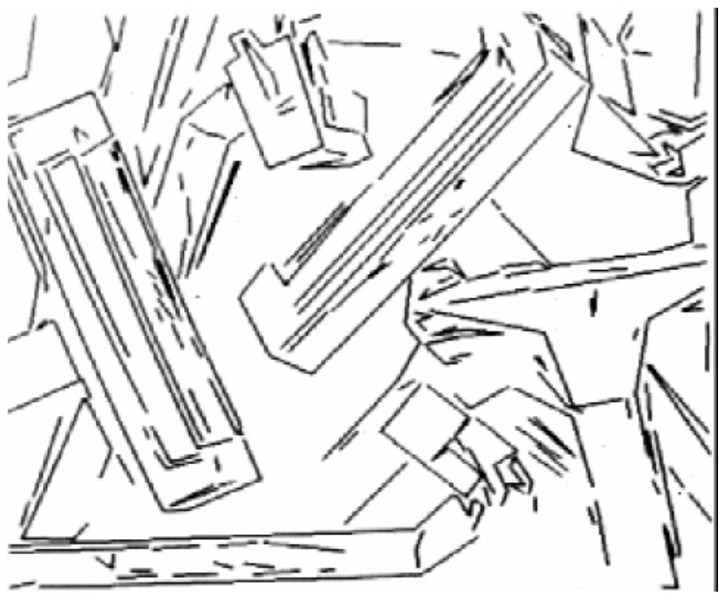
Example of Recognition & Localization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Simple approach [David Lowe]
  - 1. “Filter” image to **find brightness changes**
  - 2. **“Fit” lines** to the raw measurements



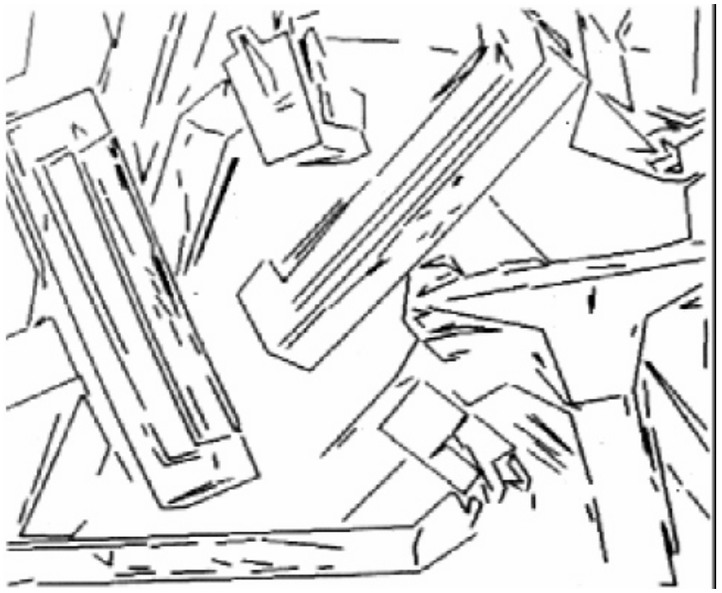


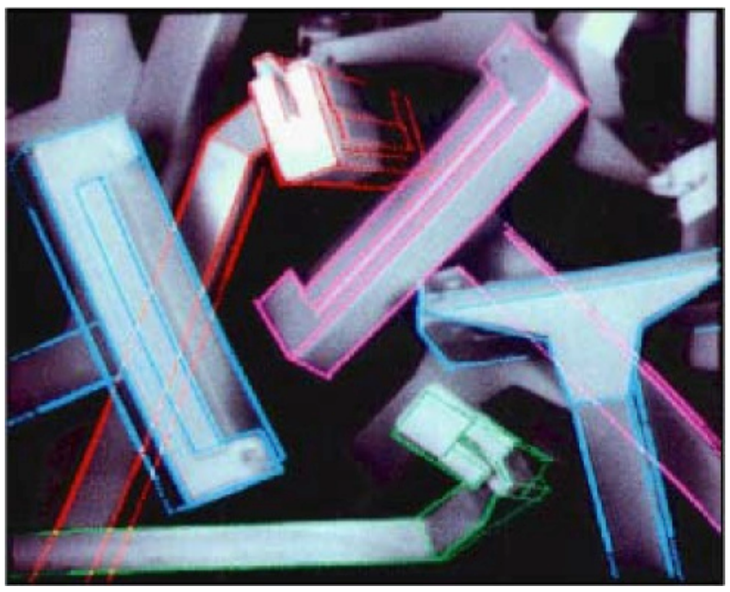


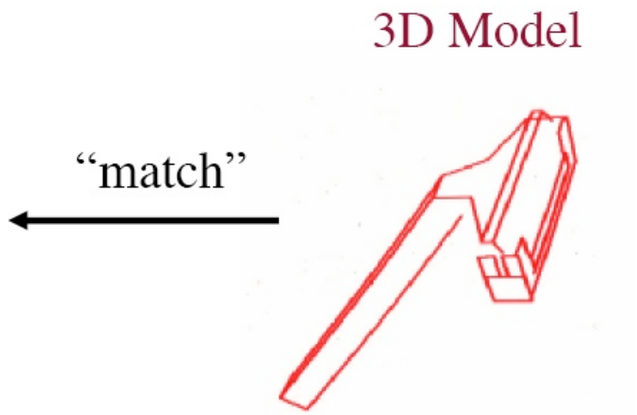
Example of Recognition & Localization

TECHNISCHE UNIVERSITÄT DARMSTADT

- Simple approach [David Lowe]
  - 3. "Project" model into the image and **"match" to lines** (solving for 3D pose)

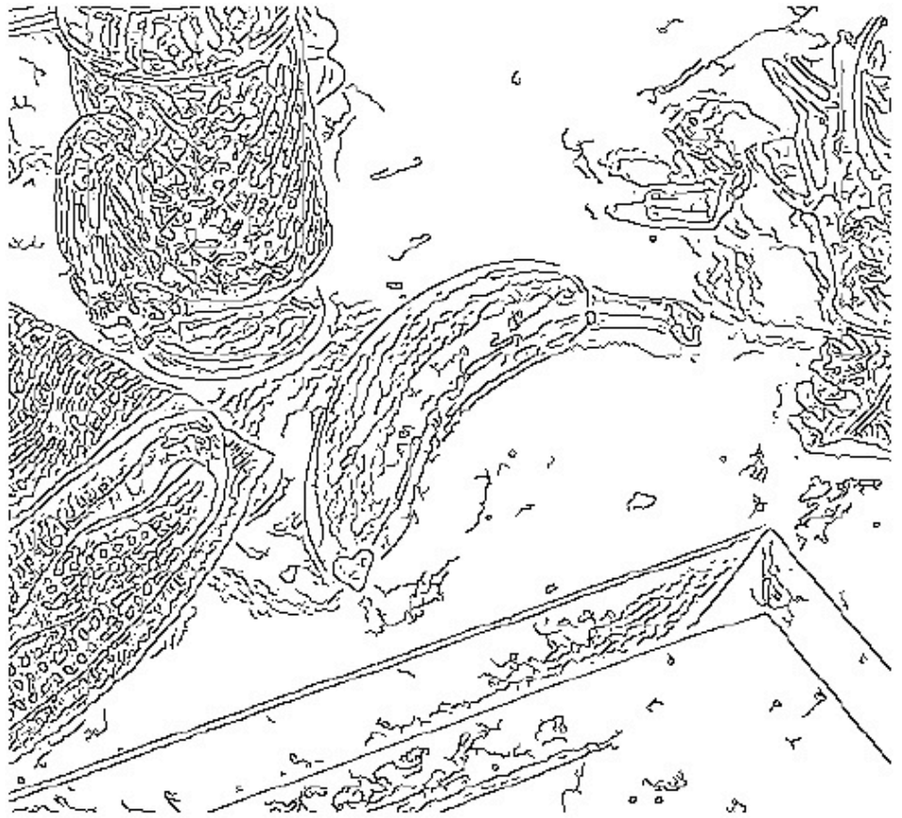






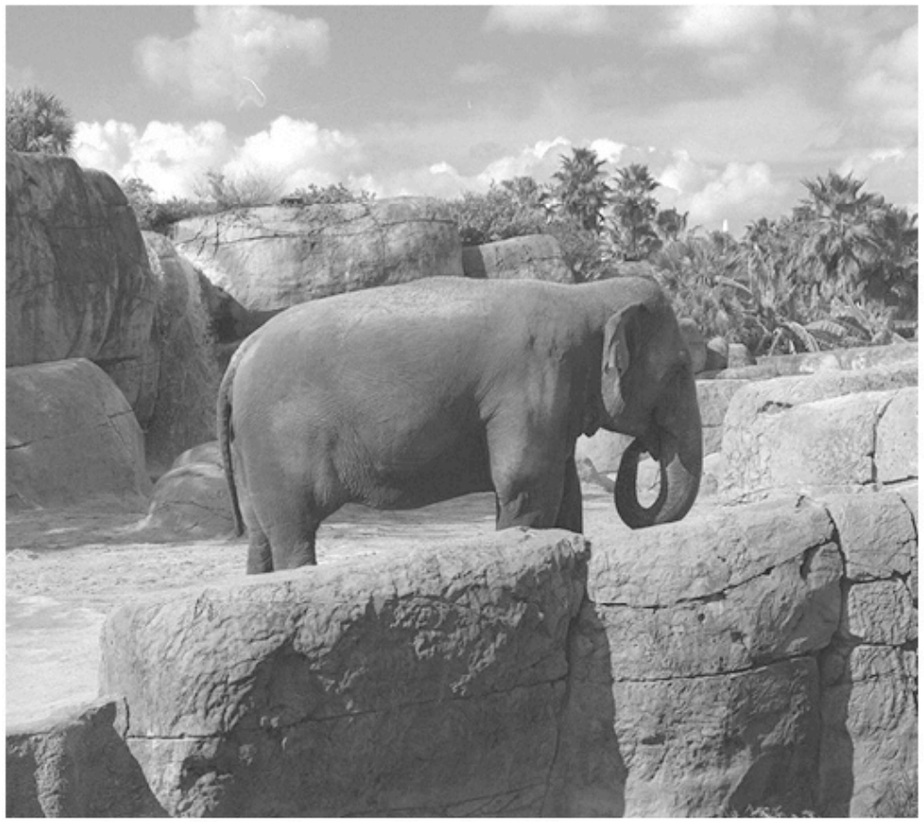
Line drawing vs. edge detection

TECHNISCHE UNIVERSITÄT DARMSTADT

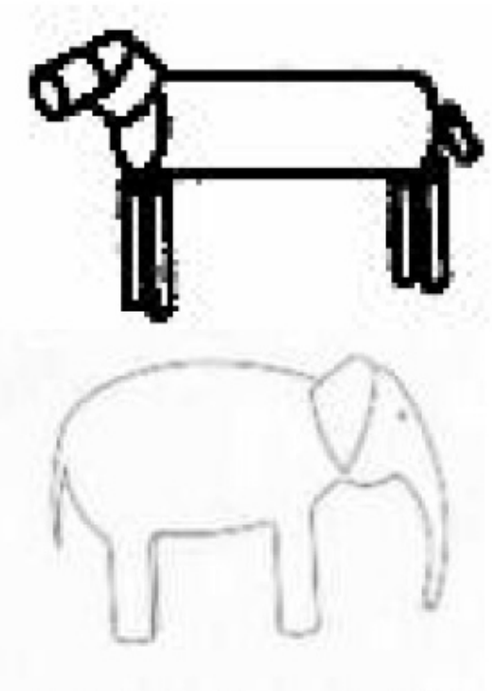


Models vs. edge detection

TECHNISCHE UNIVERSITÄT DARMSTADT



University of South Florida



Match “model” to measurements?

Search and Recognition

TECHNISCHE UNIVERSITÄT DARMSTADT

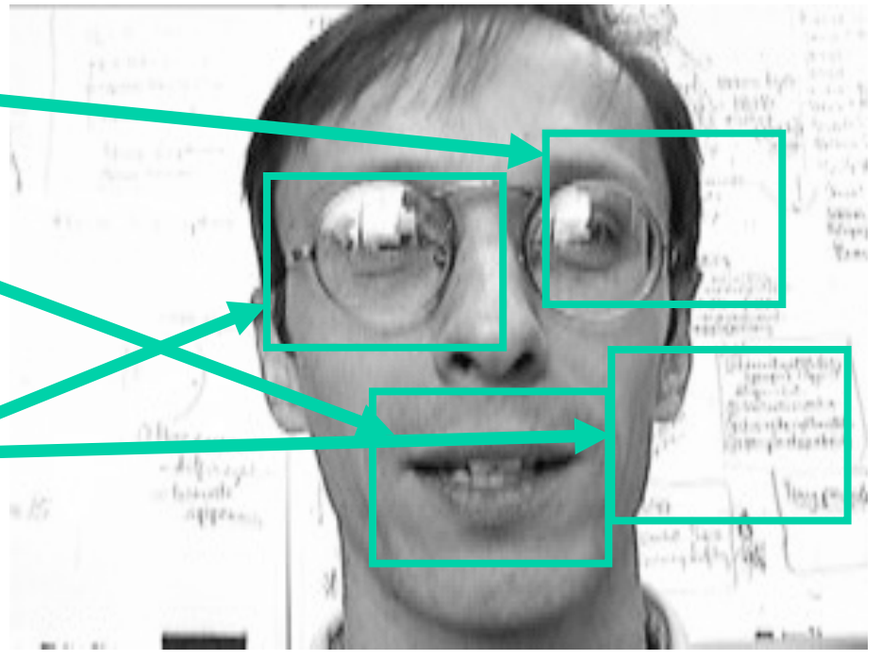
- What if we do not have a line-based object model?
  - Edges may not help us.

- How can we find the mouth?
- How can we recognize the “expression”?

Naïve View-Based Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

Database of mouth “templates”



- Search every image region (at every scale).
- Compare each template; chose the best match.

View-Based Methods

TECHNISCHE UNIVERSITÄT DARMSTADT

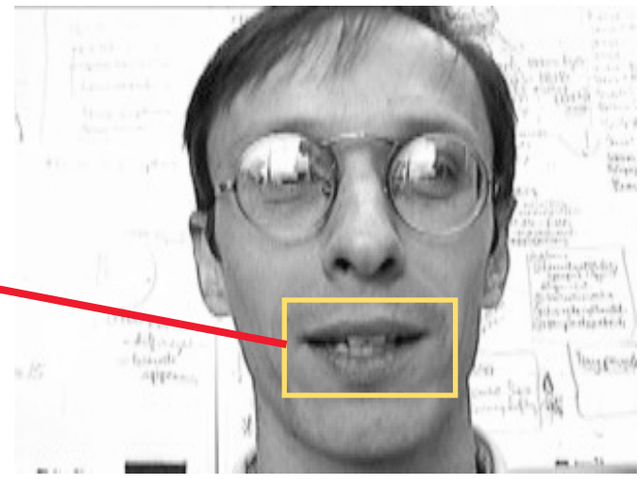
- Idea: Represent objects by their appearance in an ensemble of images, including different poses, illuminations, configurations of shape, …
- Approach covered here:
  - Subspace methods (also called “Eigen methods”)

Images as Vectors

TECHNISCHE UNIVERSITÄT DARMSTADT

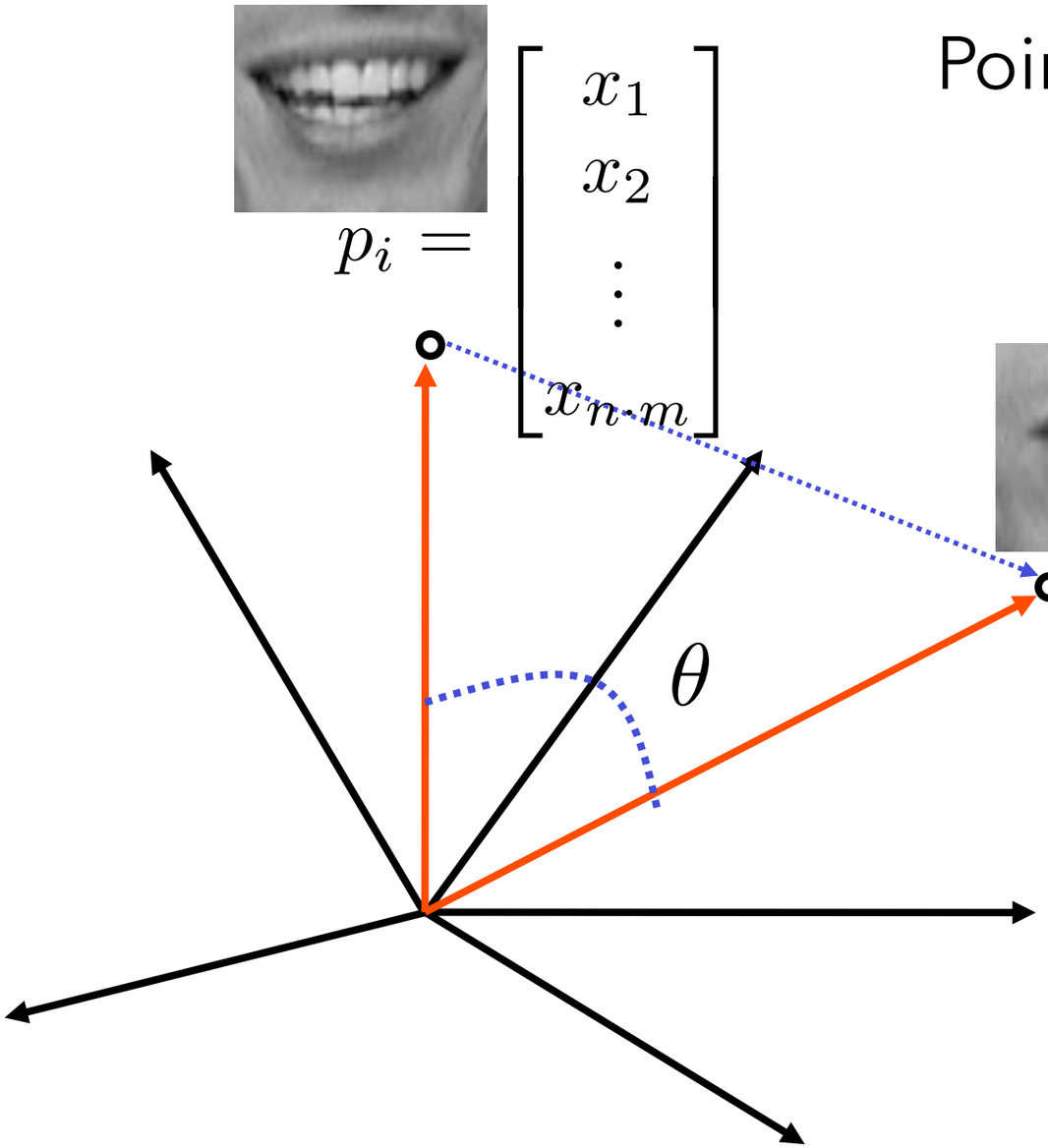
$$= \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n \cdot m} \end{bmatrix}$$

$$= \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n \cdot m} \end{bmatrix}$$



e.g. standard lexicographic ordering

Images as Points



Points in n x m dimensional space

Matching involves deciding how far apart they are in this space.

Filters are Templates

TECHNISCHE UNIVERSITÄT DARMSTADT

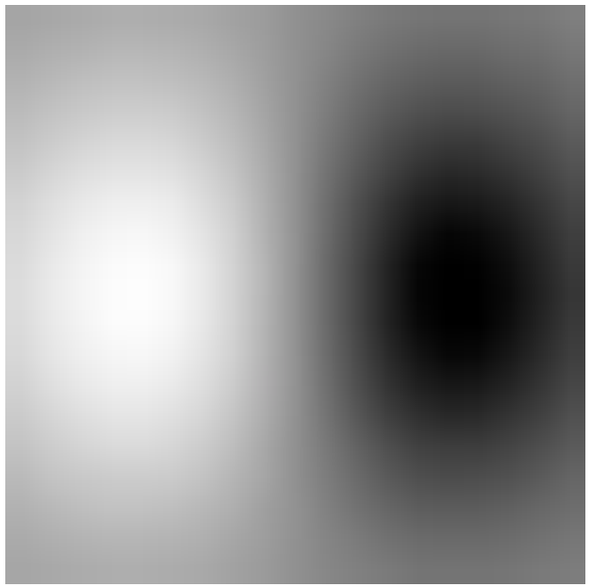
- Remember image filtering from last lecture.
- Applying a filter at some pixel can be seen as taking a dot-product between the image and some vector.
- Filtering the image is a set of dot products:

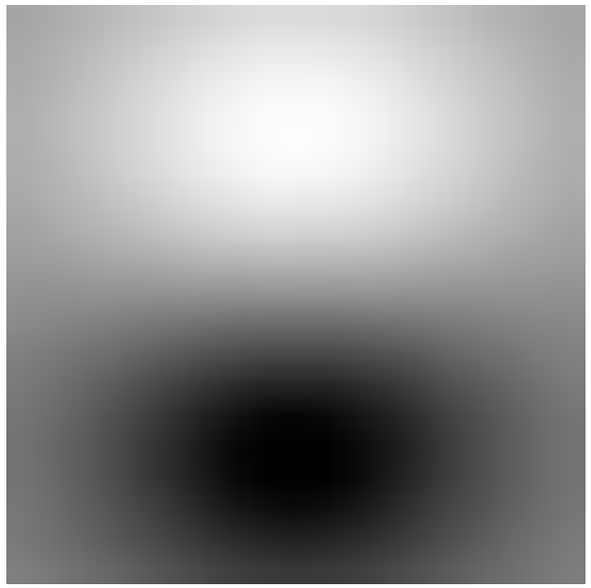
$$f[m,n] = (I * g)[m,n] = \sum_{k,l} I[m-k, n-l] g[k,l]$$

Filters as Templates

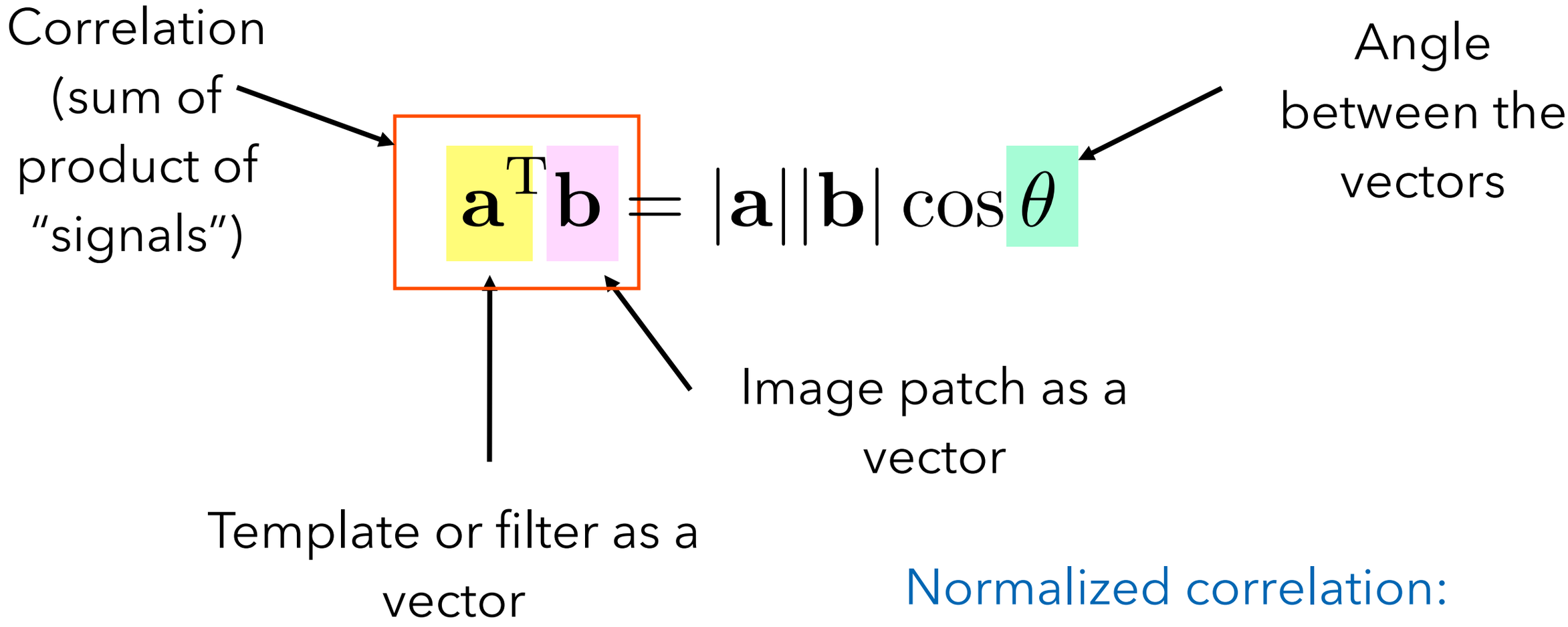
TECHNISCHE UNIVERSITÄT DARMSTADT

- Insight:
  - Filters look like the effects they are intended to find.
  - Filters find effects they look like.
  - We can use filters to match a template against an image.





Template Matching



Normalized correlation:

$$\frac{\mathbf{a}^T\mathbf{b}}{|\mathbf{a}||\mathbf{b}|} = \cos\theta$$

Dot Product

TECHNISCHE UNIVERSITÄT DARMSTADT

- When we filter, we measure the angle (cosine of it, really) between the filter template and the image patch.

$$p_i = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n \cdot m} \end{bmatrix}$$

$$p_j = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n \cdot m} \end{bmatrix}$$

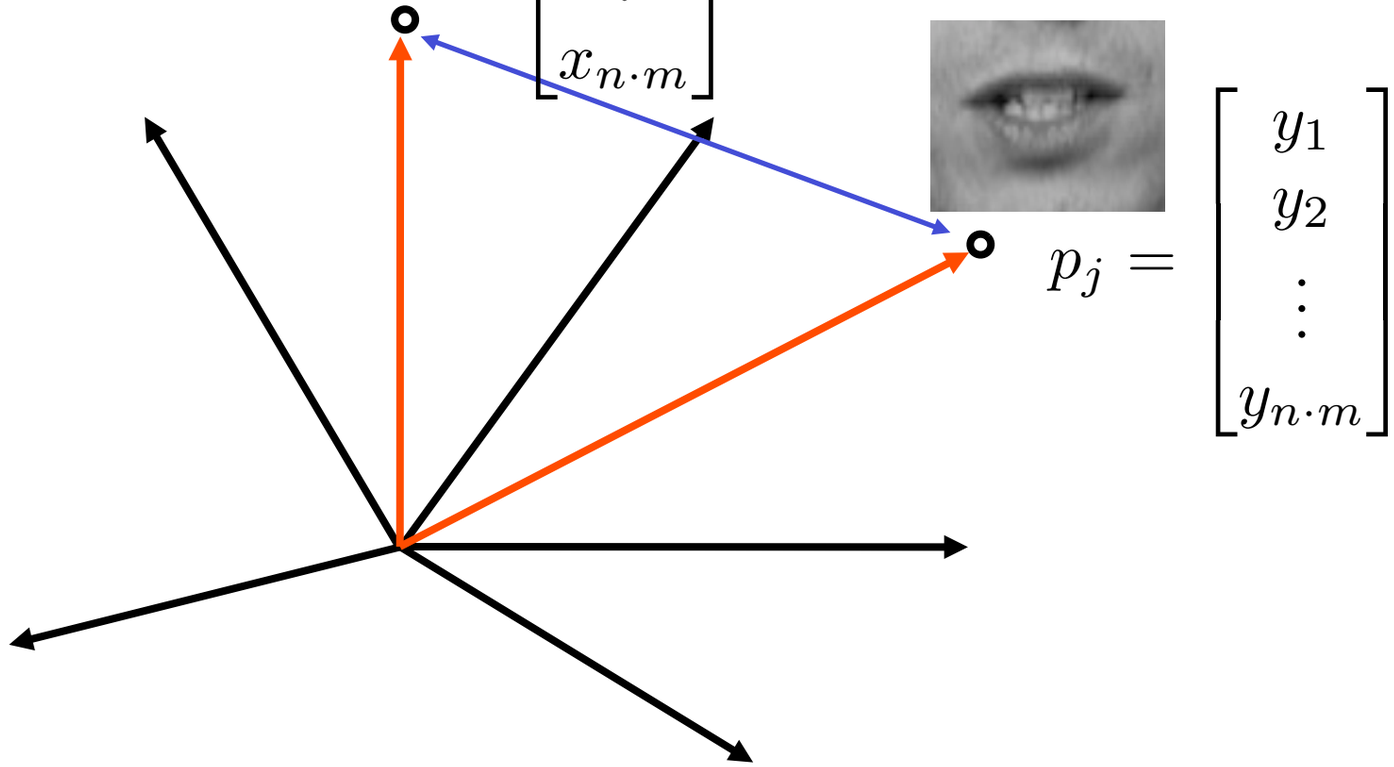
SSD Matching

- An alternative is to minimize the Sum of Squared Differences (SSD):

$$E(I,T) = \sum_{i,j} \left(I(i,j) - T(i,j)\right)^2$$

- Distance metric

$$p_i = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n \cdot m} \end{bmatrix}$$



$$p_j = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n \cdot m} \end{bmatrix}$$

Template Methods

TECHNISCHE UNIVERSITÄT DARMSTADT

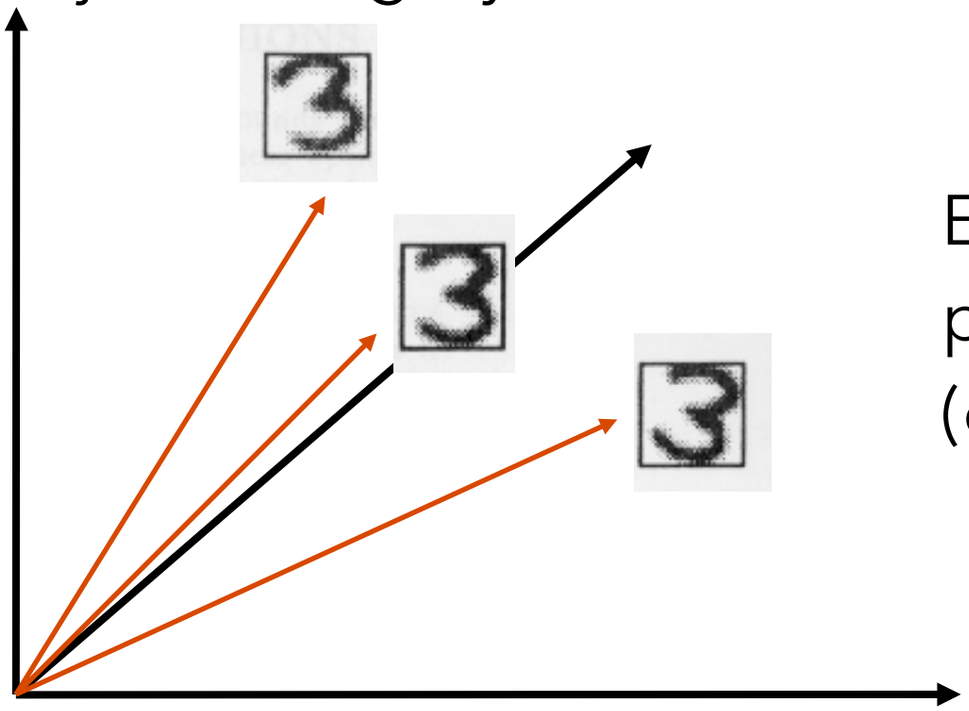
- Image templates (simplest view-based method – straw man):
  - Keep an image of every object from different viewing directions, lighting conditions, etc.
  - Nearest neighbor cross-correlation matching with images in model database (or robust matching for clutter & occlusion).
- Obvious problems:
  - Storage and computation costs become unreasonable as the number of objects increases.
  - May require very large ensemble of 'training' images.

Fleet & Szeliski

Subspace Methods

TECHNISCHE UNIVERSITÄT DARMSTADT

- How can we find more efficient representations for the ensemble of views, and more efficient methods for matching?
  - Idea: images are not random… especially images of the same object (category) have similar appearance



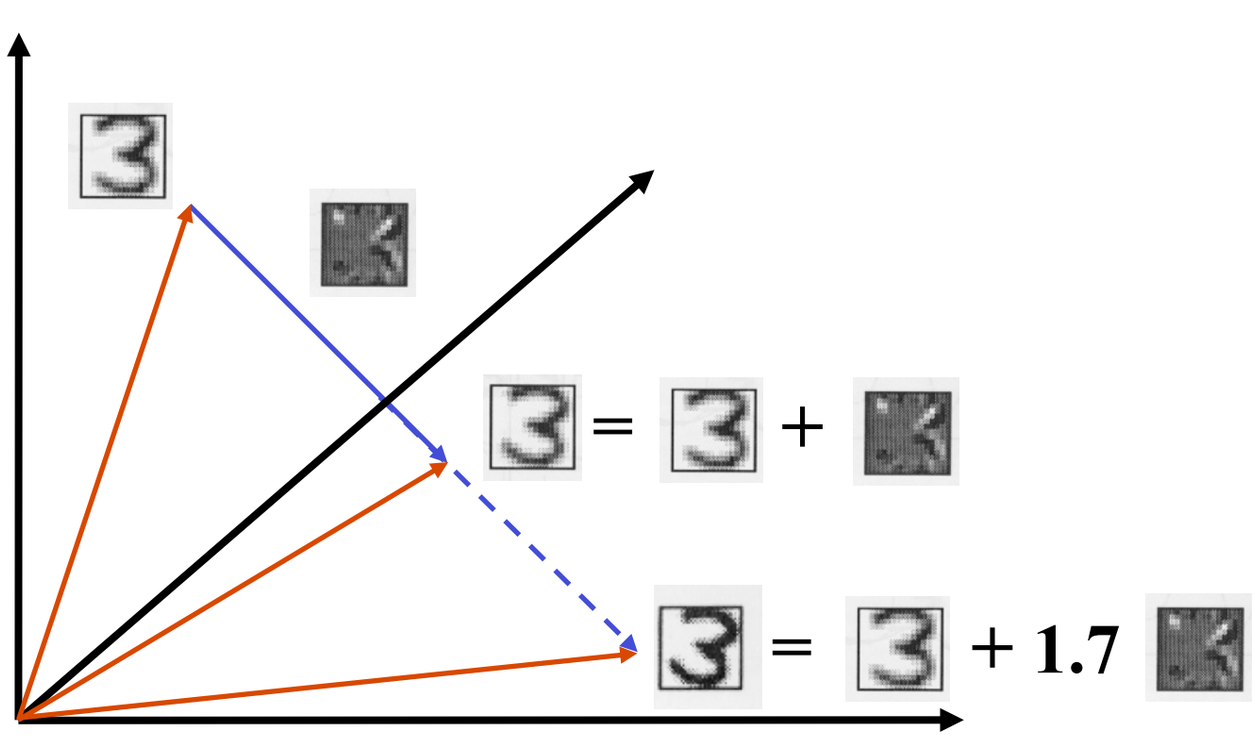
E.g., let images be represented as points in a high-dimensional space (e.g., one dimension per pixel)

Fleet & Szeliski

Linear Dimensionality Reduction

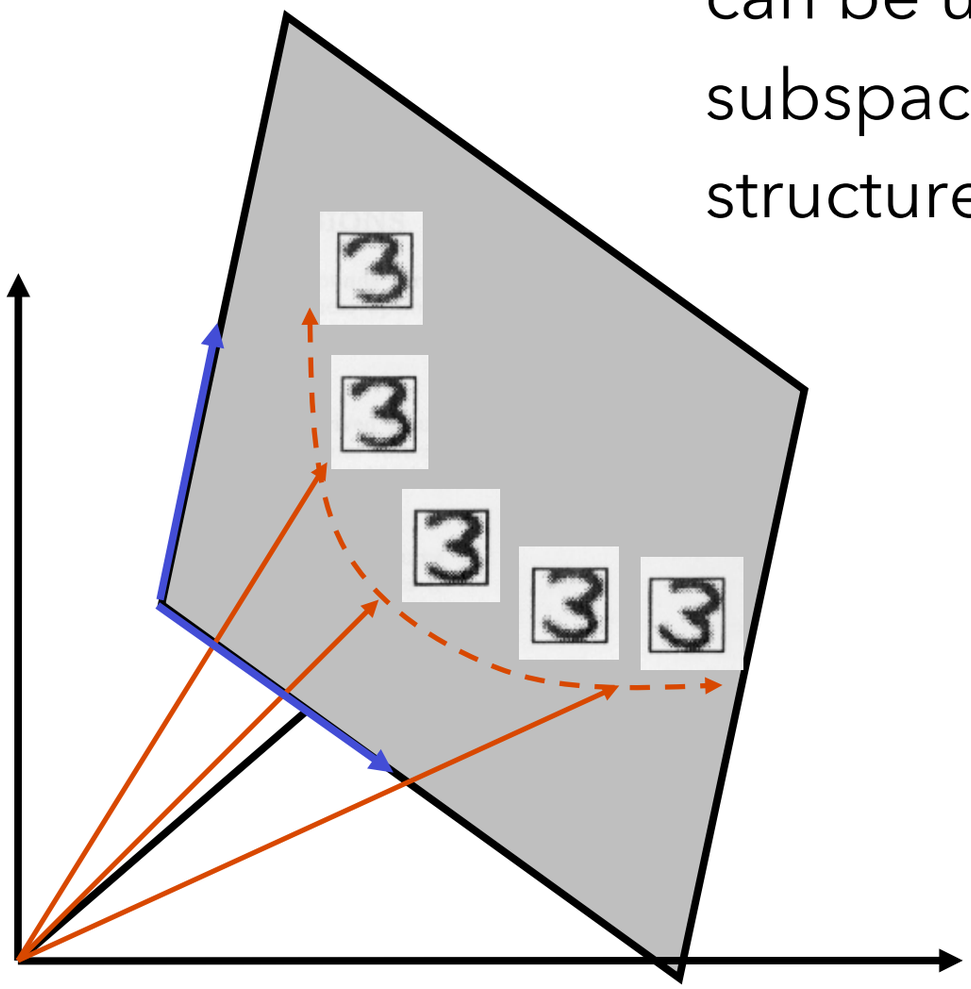
TECHNISCHE UNIVERSITÄT DARMSTADT

- Given that differences are structured, we can use 'basis images' to transform images into other images in the same space.



Fleet & Szeliski

Linear Dimensionality Reduction



What linear transformations of the images can be used to define a lower-dimensional subspace that captures most of the structure in the image ensemble?

Fleet & Szeliski

Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

- Find a lower dimensional representation that captures the variability in the data.
- Search using this low dimensional model.

Goal

TECHNISCHE UNIVERSITÄT DARMSTADT

Data point n:

$$\mathbf{x}^n \in \mathbb{R}^M$$

Low-dimensional representation:

$$\mathbf{a}^n \in \mathbb{R}^D \qquad D << M$$

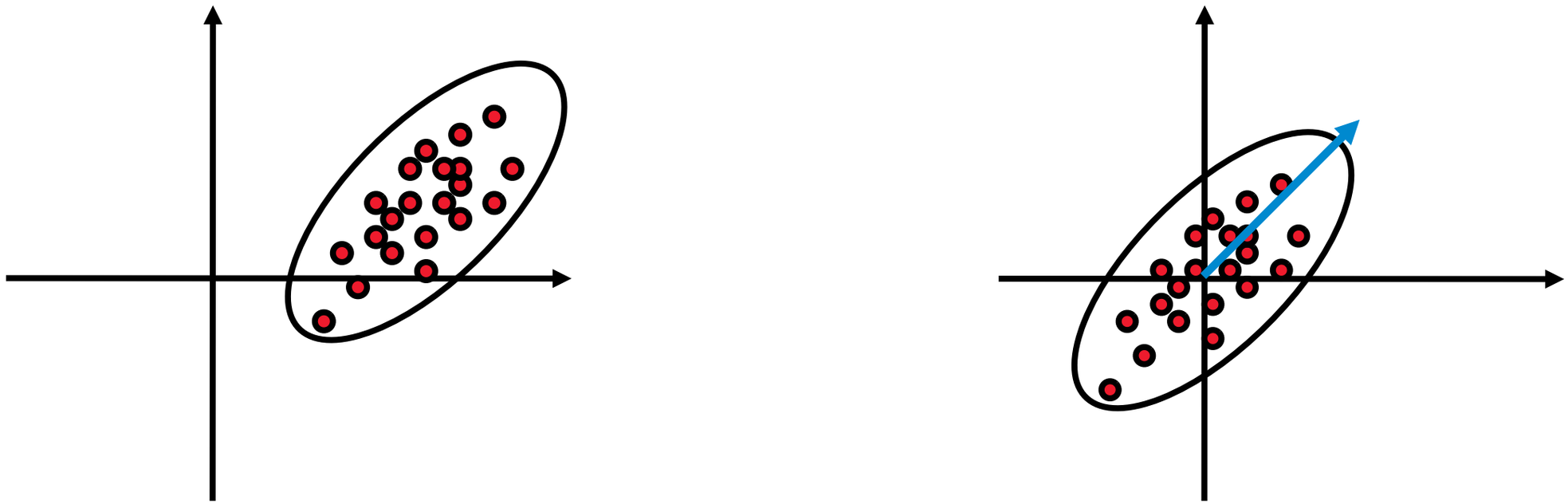
Mapping: $\mathbf{x}^n \rightarrow \mathbf{a}^n$

- Restrict this mapping to be a linear function:

$$\boldsymbol{a}^n = \mathbf{B}\mathbf{x}^n, \qquad \mathbf{B} \in \mathbb{R}^{D \times M}$$

- **Note:**
  - From now on, I will use boldface symbols to denote vectors, e.g.: $\mathbf{x}, \mathbf{y}, \ldots$

Intuition



If I give you the mean and one vector to represent the data, what vector would you choose?

Why?

Recall: Basis Representations

TECHNISCHE UNIVERSITÄT DARMSTADT

We can always write a vector as:

$$\mathbf{x} = \sum_{i=1}^{M} a_i \mathbf{u}_i \qquad \text{where} \qquad \mathbf{u}_i^{\mathrm{T}} \mathbf{u}_j = \delta_{ij}$$

Kronecker delta = 1 if $i = j$, 0 otherwise.

orthonormal

Example:

$$\begin{bmatrix} 3 \\ 7 \end{bmatrix} = 3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 7 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Recall: Basis Representations

TECHNISCHE UNIVERSITÄT DARMSTADT

$$
\begin{aligned}
\mathbf{x} &= a_1\mathbf{u_1} + a_2\mathbf{u_2} \\
a_1 &= \mathbf{u}_1^{\mathrm{T}}(\mathbf{x} - a_2\mathbf{u}_2) \\
&= \mathbf{u}_1^{\mathrm{T}}\mathbf{x} - a_2\mathbf{u}_1^{\mathrm{T}}\mathbf{u}_2 \\
&= \mathbf{u}_1^{\mathrm{T}}\mathbf{x}
\end{aligned}
$$

More generally:

$$a_i = \underbrace{\mathbf{u}_i^{\mathrm{T}}\mathbf{x}}$$

Scalar coefficient

projection

Decomposition

$$\mathbf{x}^n = \sum_{i=1}^{D} a_i \mathbf{u}_i + \sum_{j=D+1}^{M} b_j \mathbf{u}_j$$

Approximation $\tilde{\mathbf{x}}^n$

Error

Want the D bases that minimize the mean squared error over the training data

$$\arg\min E(\mathbf{u}_1, \ldots, \mathbf{u}_D) = \sum_{n=1}^{N} ||\boldsymbol{x}^n - \tilde{\boldsymbol{x}}^n||^2$$

Minimizing the Error

TECHNISCHE UNIVERSITÄT DARMSTADT

- Rewrite the error (assuming a single basis vector):

$$E(\mathbf{u}) = \sum_{n=1}^{N} ||\mathbf{x}^n - \tilde{\mathbf{x}}^n||^2$$

$$= \sum_{n=1}^{N} ||\mathbf{x}^n - (\mathbf{u}^{\mathrm{T}}\mathbf{x}^n)\mathbf{u}||^2$$

$$= \sum_{n=1}^{N} ||\mathbf{x}^n||^2 - 2(\mathbf{u}^{\mathrm{T}}\mathbf{x}^n)^2 + (\mathbf{u}^{\mathrm{T}}\mathbf{x}^n)^2 \cdot \mathbf{u}^{\mathrm{T}}\mathbf{u}$$

$$= \sum_{n=1}^{N} ||\mathbf{x}^n||^2 - (\mathbf{u}^{\mathrm{T}}\mathbf{x}^n)^2 = \sum_{n=1}^{N} ||\mathbf{x}^n||^2 - (a_n)^2$$

Minimizing the Error

- Rewrite the error (assuming a single basis vector):

$$E(\mathbf{u}) = \sum_{n=1}^{N} ||\mathbf{x}^n||^2 - (\mathbf{u}^{\mathrm{T}}\mathbf{x}^n)^2 = \sum_{n=1}^{N} ||\mathbf{x}^n||^2 - (a_n)^2$$

  - Minimizing the error is equivalent to maximizing the variance of the projection:

$$\frac{1}{N}\sum_{n=1}^{N} a_n^2 \qquad \text{(assuming mean 0)}$$

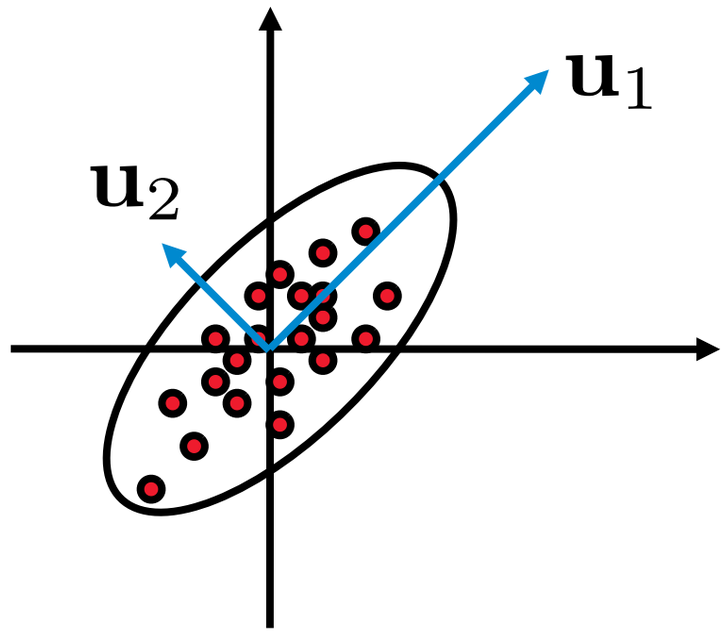
- We can ensure a zero mean projection by subtracting the mean from every data point:

$$\mathbf{x}^n - \bar{\mathbf{x}}$$

Intuition

$$\mathbf{x}^n - \bar{\mathbf{x}} = \sum_{i=1}^{D} a_i \mathbf{u}_i + \sum_{j=D+1}^{M} b_j \mathbf{u}_j$$

$$\tilde{\mathbf{x}}^n = \sum_{i=1}^{D} a_i \mathbf{u}_i + \bar{\mathbf{x}}$$



Projecting onto $\mathbf{u}_1$ captures the majority of the variance and hence projecting onto it minimizes the error

How do we find the axis of largest variance?

Statistics Review: Univariate

TECHNISCHE UNIVERSITÄT DARMSTADT

Sample mean:

(univariate case)

$$\bar{x} = \langle x \rangle = E[x] = \frac{1}{N} \sum_{n=1}^{N} x^n$$

Sample variance:

$$\begin{aligned} \hat{\sigma}^2 &= \left\langle (x - \bar{x})^2 \right\rangle = \mathrm{Var}[x] = \frac{1}{N} \sum_{n=1}^{N} (x^n - \bar{x})^2 \\ &= \left\langle x^2 \right\rangle - \langle 2\bar{x}x \rangle + \left\langle \bar{x}^2 \right\rangle = \left\langle x^2 \right\rangle - \bar{x}^2 \end{aligned}$$

Statistics Review: Multivariate

TECHNISCHE UNIVERSITÄT DARMSTADT

Sample mean:

(multivariate case)

$$\bar{\mathbf{x}} = \langle \mathbf{x} \rangle = E[\mathbf{x}] = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}^n$$

Sample covariance:

$$\begin{aligned} \mathrm{cov}(x_i, x_j) = \sigma_{ij} &= \langle (x_i - \bar{x}_i)(x_j - \bar{x}_j) \rangle \\ &= \langle x_i x_j \rangle - \langle x_i \bar{x}_j \rangle - \langle \bar{x}_i x_j \rangle + \langle \bar{x}_i \bar{x}_j \rangle \\ &= \langle x_i x_j \rangle - \bar{x}_i \bar{x}_j \\ \mathrm{cov}(x_i, x_i) = \sigma_i^2 &= \langle x_i^2 \rangle - \bar{x}_i^2 \end{aligned}$$

Statistical Correlation

TECHNISCHE UNIVERSITÄT DARMSTADT

- The covariance of two variates X_i and X_j of a random vector $\mathbf{X}$ provides a measure of how strongly correlated these variables are, and the derived quantity is:

$$\mathrm{cor}(x_i, x_j) = \frac{\mathrm{cov}(x_i, x_j)}{\sigma_i \sigma_j}$$

Covariance Matrix

TECHNISCHE UNIVERSITÄT DARMSTADT

- The covariance matrix describes the covariance between all components of a random vector, e.g.:

$$\mathbf{C} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

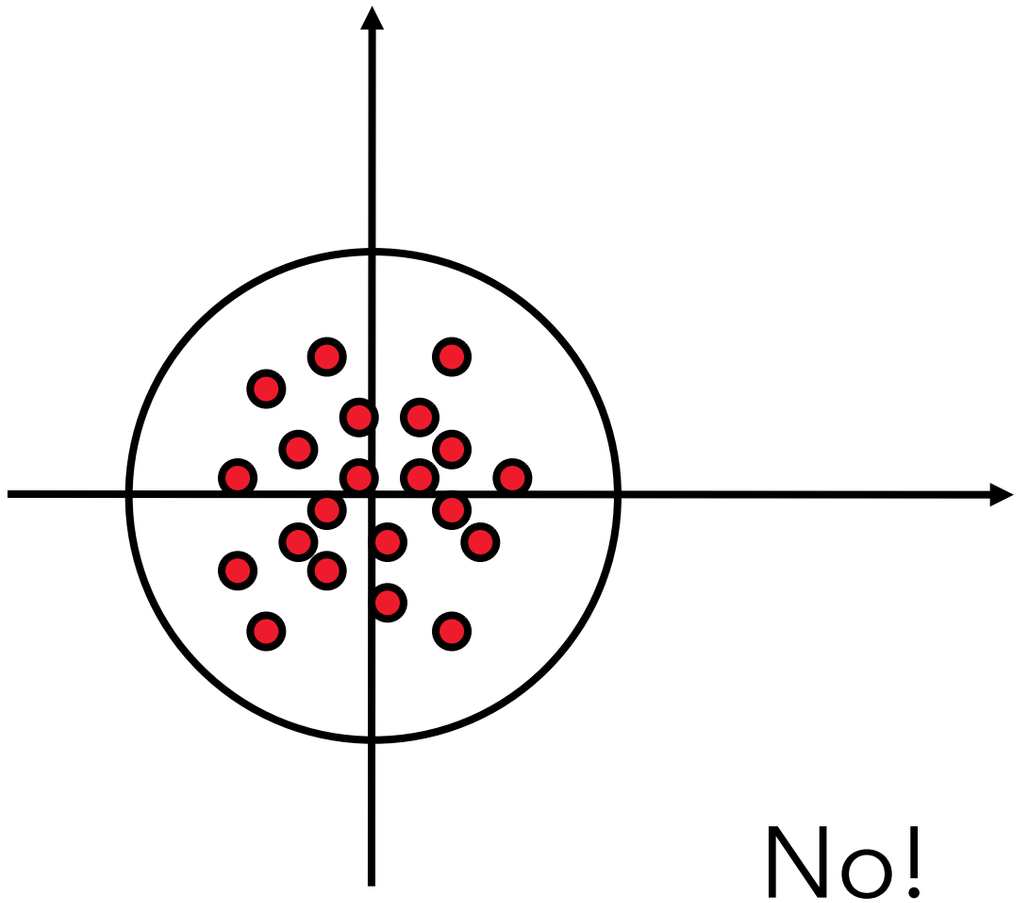
$$\hat{\mathbf{C}} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^{\mathrm{T}}$$

Recall: Outer Product

TECHNISCHE UNIVERSITÄT DARMSTADT

$$\mathbf{x}\mathbf{x}^{\mathrm{T}} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} x_1 x_1 & x_1 x_2 \\ x_2 x_1 & x_2 x_2 \end{bmatrix}$$

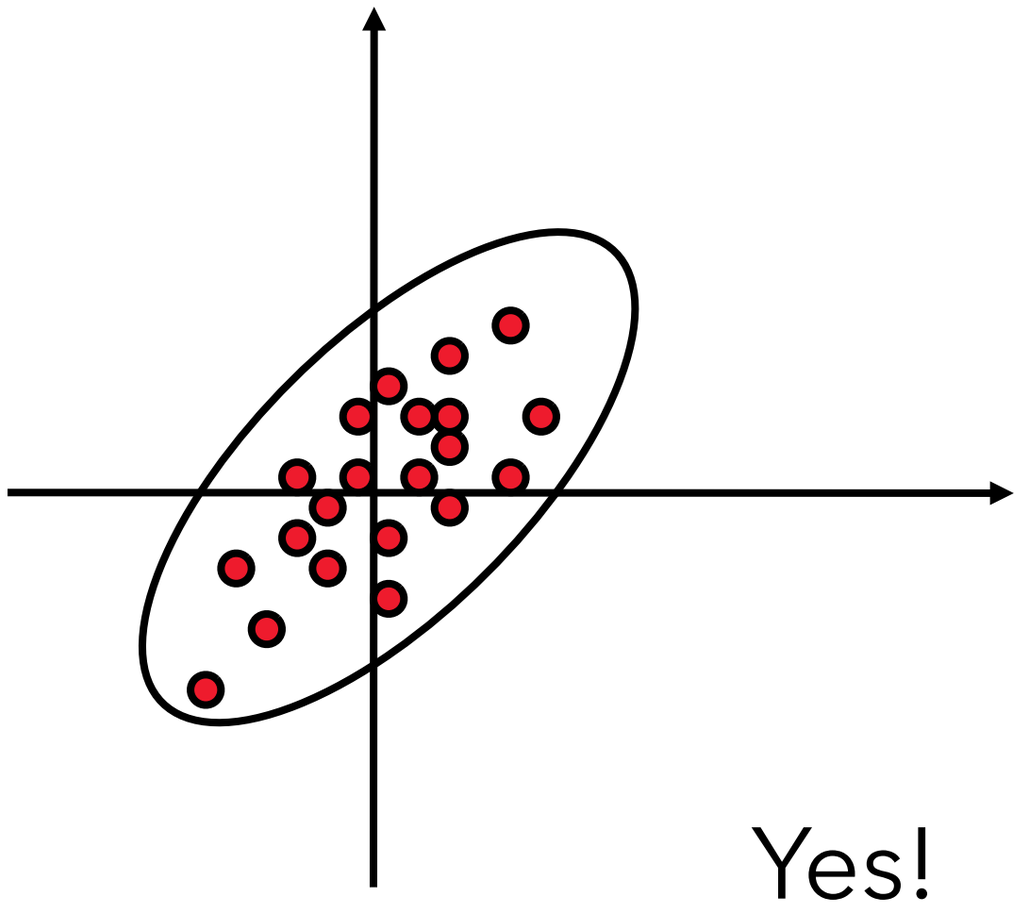
Correlated?



No! $\mathbf{C} = \begin{bmatrix} c & 0 \\ 0 & c \end{bmatrix}$

Correlated?

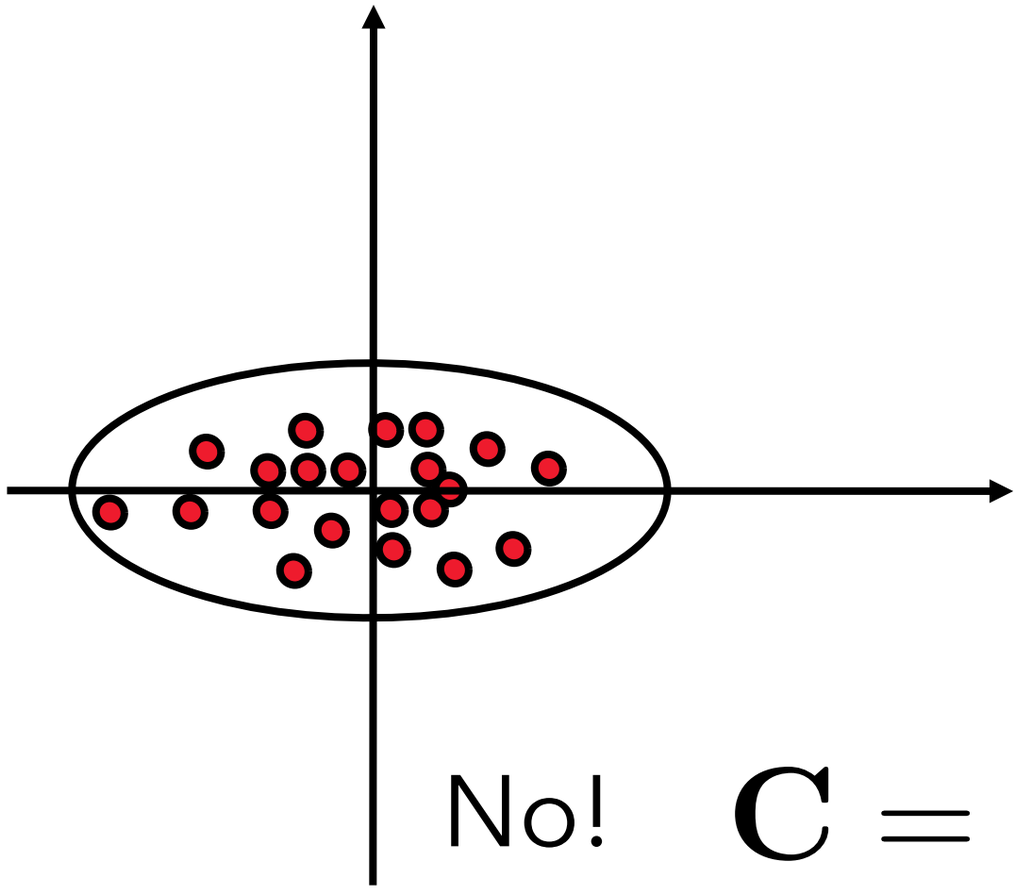
TECHNISCHE UNIVERSITÄT DARMSTADT



Yes! $\mathbf{C} = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$

Correlated?

TECHNISCHE UNIVERSITÄT DARMSTADT

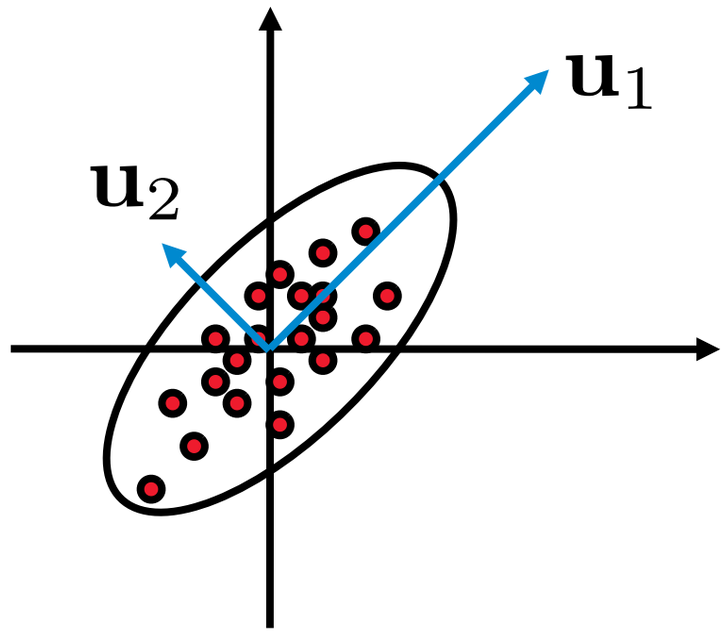


No! $\mathbf{C} = \begin{bmatrix} c & 0 \\ 0 & d \end{bmatrix}, \quad c > d$

Intuition

$$\mathbf{x}^n - \bar{\mathbf{x}} = \sum_{i=1}^{D} a_i \mathbf{u}_i + \sum_{j=D+1}^{M} b_j \mathbf{u}_j$$

$$\tilde{\mathbf{x}}^n = \sum_{i=1}^{D} a_i \mathbf{u}_i + \bar{\mathbf{x}}$$



Data is correlated in its original coordinate system!

Note that axes of the projection are orthogonal and **decorrelate** the data; i.e. in the coordinate frame of these axes, the data is uncorrelated.

(side note: this only works for Gaussians).

Formulation of problem

TECHNISCHE UNIVERSITÄT DARMSTADT

- Let $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N] \in \mathbb{R}^{M \times N}$ be a matrix of N vectors in M-dimensional input space.
- Let $\mathbf{u} \in \mathbb{R}^M$ be a direction (a vector of length 1) in the input space.
- The projection of the n-th vector $\mathbf{x}^n$ onto the vector $\mathbf{u}$ can be calculated as:

$$a_n = \mathbf{u}^{\mathrm{T}} \mathbf{x}^n = \sum_{n=1}^{M} x_i^n u_i$$

- The goal is to find a direction $\mathbf{u}$ that maximizes the variance of the projections of all input vectors $\mathbf{x}^n, \quad n = 1 \ldots N$

Mean & Variance

TECHNISCHE UNIVERSITÄT DARMSTADT

- The mean of the projection

$$\bar{a} = \frac{1}{N}\sum_{n=1}^{N} a_n = \frac{1}{N}\sum_{n=1}^{N} \mathbf{u}^{\mathrm{T}}\mathbf{x}^n = \frac{1}{N}\mathbf{u}^{\mathrm{T}}\left[\sum_{n=1}^{N} \mathbf{x}^n\right] = \mathbf{u}^{\mathrm{T}}\bar{\mathbf{x}}$$

- ... is the projection of the mean.

Mean & Variance

TECHNISCHE UNIVERSITÄT DARMSTADT

- Variance of the projection:

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(a_n - \bar{a})^2 = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{u}^{\mathrm{T}}\mathbf{x}^n - \mathbf{u}^{\mathrm{T}}\bar{\mathbf{x}})^2$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{u}^{\mathrm{T}}(\mathbf{x}^n - \bar{\mathbf{x}})\right)^2 = \frac{1}{N}\sum_{n=1}^{N}\mathbf{u}^{\mathrm{T}}(\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^{\mathrm{T}}\mathbf{u}$$

$$= \mathbf{u}^{\mathrm{T}}\left[\frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^{\mathrm{T}}\right]\mathbf{u} = \mathbf{u}^{\mathrm{T}}\mathbf{C}\mathbf{u}$$

covariance matrix

Maximizing variance

TECHNISCHE UNIVERSITÄT DARMSTADT

- Maximize σ^2 under the constraint that $||\mathbf{u}|| = 1$.
  - Maximize the following function (Langrangian):

$$F(u;\lambda) = \mathbf{u}^{\mathrm{T}}\mathbf{C}\mathbf{u} - \lambda(||\mathbf{u}||^2 - 1) = \mathbf{u}^{\mathrm{T}}\mathbf{C}\mathbf{u} - \lambda(\mathbf{u}^{\mathrm{T}}\mathbf{u} - 1)$$

$$\frac{\partial F(u;\lambda)}{\partial \mathbf{u}} = 2\mathbf{C}\mathbf{u} - 2\lambda\mathbf{u} = 0$$

$$\Leftrightarrow \quad \mathbf{C}\mathbf{u} = \lambda\mathbf{u}$$

with $||\mathbf{u}|| = 1$

- Compute eigenvectors and eigenvalues of $\mathbf{C}$:
- The largest eigenvalue = maximal variance.
- Corresponding eigenvector = direction with the max. variance

Eigendecomposition

TECHNISCHE UNIVERSITÄT DARMSTADT

- Let the eigenvectors and eigenvalues of $\mathbf{C}$ be $\mathbf{u}_k$ and λ_k for $k \leq M$, i.e., $\mathbf{C}\mathbf{u}_k = \lambda_k \mathbf{u}_k$ with $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_M$

- In matrix form: $\mathbf{C}\mathbf{U} = \mathbf{U}\Lambda$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_M)$ and $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_M]$.

- Because $\mathbf{U}$ is orthonormal (we usually assume that the eigenvectors have unit norm) we know that $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{I}$.

- This means that we can decompose $\mathbf{C}$ as $\boxed{\mathbf{C} = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}}$

Principal Component Analysis

TECHNISCHE UNIVERSITÄT DARMSTADT

- $\mathbf{C}$ is real, symmetric, and positive-definite. Thus we can decompose it as:

$$\mathbf{C} = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_M \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_M \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^{\mathrm{T}} \\ \mathbf{u}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{u}_M^{\mathrm{T}} \end{bmatrix}$$

Orthonormal columns

eigenvectors

eigenvalues

- This is the eigendecomposition of $\mathbf{C}$

Principal Component Analysis

TECHNISCHE UNIVERSITÄT DARMSTADT

- Observation: If $\lambda_k \approx 0$ for $k > D$ for some $D << M$ then we can use the subset of the first D eigenvectors to define a basis for approximating the data vectors.

$$\mathbf{x}^n - \bar{\mathbf{x}} = \sum_{i=1}^{D} a_i \mathbf{u}_i + \sum_{j=D+1}^{M} b_j \mathbf{u}_j$$

$$\mathbf{x}^n \approx \tilde{\mathbf{x}}^n = \bar{\mathbf{x}} + \sum_{i=1}^{D} a_i \mathbf{u}_i \quad \text{where} \quad a_i = \mathbf{u}_i^{\mathrm{T}}(\mathbf{x}^n - \bar{\mathbf{x}})$$

- This representation has the minimal mean squared error (MSE) of all linear representations of dimension D :

$$\min E(\mathbf{u}_1, \ldots, \mathbf{u}_D) = \sum_{n=1}^{N} ||\mathbf{x}^n - \tilde{\mathbf{x}}^n||^2$$

Principal Component Analysis

TECHNISCHE UNIVERSITÄT DARMSTADT

- Now we know how we can represent our data in a lower dimensional space in a principled fashion.
  - We compute the mean of the data, and subtract it.
  - We compute the covariance matrix, decompose it, and choose a the first D eigenvalues.
  - This gives us an (eigen)basis for representing the data:

$$\mathbf{a}^n = \mathbf{B}^{\mathrm{T}}(\mathbf{x}^n - \bar{\mathbf{x}}) \qquad \text{where} \qquad \mathbf{B} = [\mathbf{u}_1, \ldots, \mathbf{u}_D]$$
$$\tilde{\mathbf{x}}^n = \bar{\mathbf{x}} + \mathbf{B}\mathbf{a}^n$$

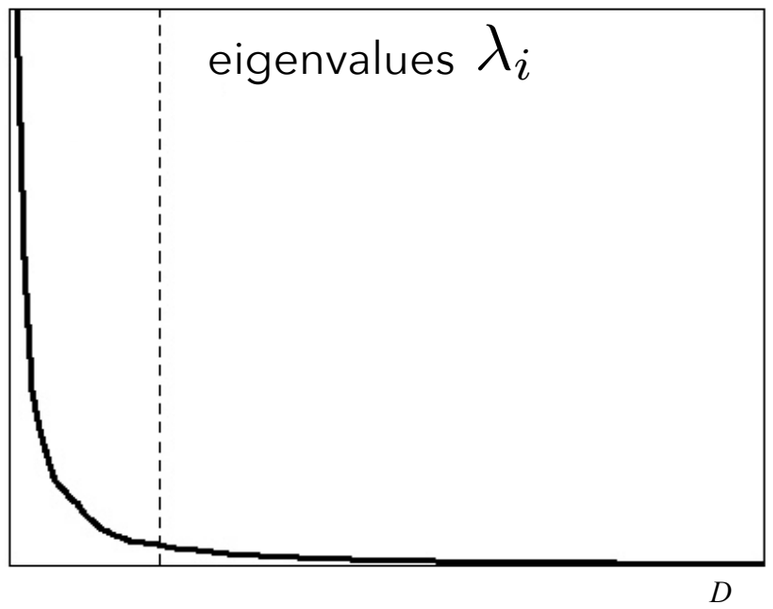
  - It is common to also normalize the variance of each dimension.
- Question:
  - How do we choose D?

Choosing D

TECHNISCHE UNIVERSITÄT DARMSTADT

- Larger D leads to better approximation.
- There are at least 2 good possibilities for choosing D:
  - First, choose D based on application performance, i.e. choose the smallest D that makes the application work well enough.
  - Second, choose D so that the eigenbasis captures some fraction of the variance, say $\eta = 0.9$
    - Remember: Eigenvalue λ_i describes the marginal variance captured by $\mathbf{u}_i$
    - Choose D using

$$\sum_{i=1}^{D} \lambda_i \geq \eta \sum_{i=1}^{M} \lambda_i$$



PCA in Practice

$$p_i = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n \times m} \end{bmatrix}$$

- Suppose we want to find a low-dimensional representation for mouth images.
- Let's assume the images are of moderate size: 128×128
- How "big" is the covariance matrix?
  - The data vector has $2^7 \cdot 2^7 = 2^{14}$ dimensions.
  - So the covariance matrix has 2^{28} entries.
  - Takes up 2^{31} Bytes, i.e. 2GB.
  - Too large to handle efficiently.
- What can we do to reduce this?

More Efficiency

TECHNISCHE UNIVERSITÄT DARMSTADT

- Observation 1:
  - While we may work in spaces that have 100000s of dimensions, we often only have 100s or 1000s of examples.
- Observation 2:
  - We do not need to compute all eigenvectors and eigenvalues. Computing a few dozen of the largest ones usually suffices.
- But how do we exploit this?

Rewrite PCA

TECHNISCHE UNIVERSITÄT DARMSTADT

- Data vectors in matrix form: $\mathbf{X} = [\mathbf{x}^1, \ldots, \mathbf{x}^N]$

- First subtract the mean: $\bar{\mathbf{x}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}^n$

$$\hat{\mathbf{X}} = \mathbf{X} - [\bar{\mathbf{x}}, \ldots, \bar{\mathbf{x}}]$$

- Express the covariance matrix differently:

$$\hat{\mathbf{C}} = \frac{1}{N} \sum_{n=1}^{N} (\mathbf{x}^n - \bar{\mathbf{x}})(\mathbf{x}^n - \bar{\mathbf{x}})^{\mathrm{T}} = \frac{1}{N} \hat{\mathbf{X}} \hat{\mathbf{X}}^{\mathrm{T}}$$

Singular Value Decomposition

TECHNISCHE UNIVERSITÄT DARMSTADT

- SVD is a generalization of the eigendecomposition to rectangular matrices $\hat{\mathbf{X}} \in \mathbb{R}^{M \times N}$

$$\hat{\mathbf{X}} = \mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}} = \underbrace{\begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \dots & \mathbf{u}_M \end{bmatrix}}_{M \times M} \underbrace{\begin{bmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & \end{bmatrix}}_{M \times N} \underbrace{\begin{bmatrix} \mathbf{v}_1^{\mathrm{T}} \\ \mathbf{v}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{v}_N^{\mathrm{T}} \end{bmatrix}}_{N \times N}$$

Orthonormal columns
left-singular vectors

singular values
(non-negative)

Orthonormal rows
right-singular vectors

How to use SVD for PCA

TECHNISCHE UNIVERSITÄT DARMSTADT

- Decompose the data matrix $\hat{\mathbf{X}}$ with SVD:

$$
\begin{aligned}
\frac{1}{N}\hat{\mathbf{X}}\hat{\mathbf{X}}^{\mathrm{T}} &= \frac{1}{N}\mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}\left(\mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}\right)^{\mathrm{T}} \\
&= \frac{1}{N}\mathbf{U}\mathbf{S}\mathbf{V}^{\mathrm{T}}\mathbf{V}\mathbf{S}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} \\
&= \frac{1}{N}\mathbf{U}\mathbf{S}\mathbf{S}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} \\
&= \mathbf{U}(\frac{1}{N}\mathbf{S}^2)\mathbf{U}^{\mathrm{T}} \\
&= \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}
\end{aligned}
$$

How to use SVD for PCA

TECHNISCHE UNIVERSITÄT DARMSTADT

- If we perform SVD on the data matrix (after subtracting the mean) then:
  - The left-singular vectors give us the eigenvectors of the covariance matrix.
  - The singular values let us easily compute the eigenvalues of the covariance matrix:

$$\lambda_i = \frac{1}{N} s_i^2$$

- **Advantage:** We never have to explicitly build and store the covariance matrix!

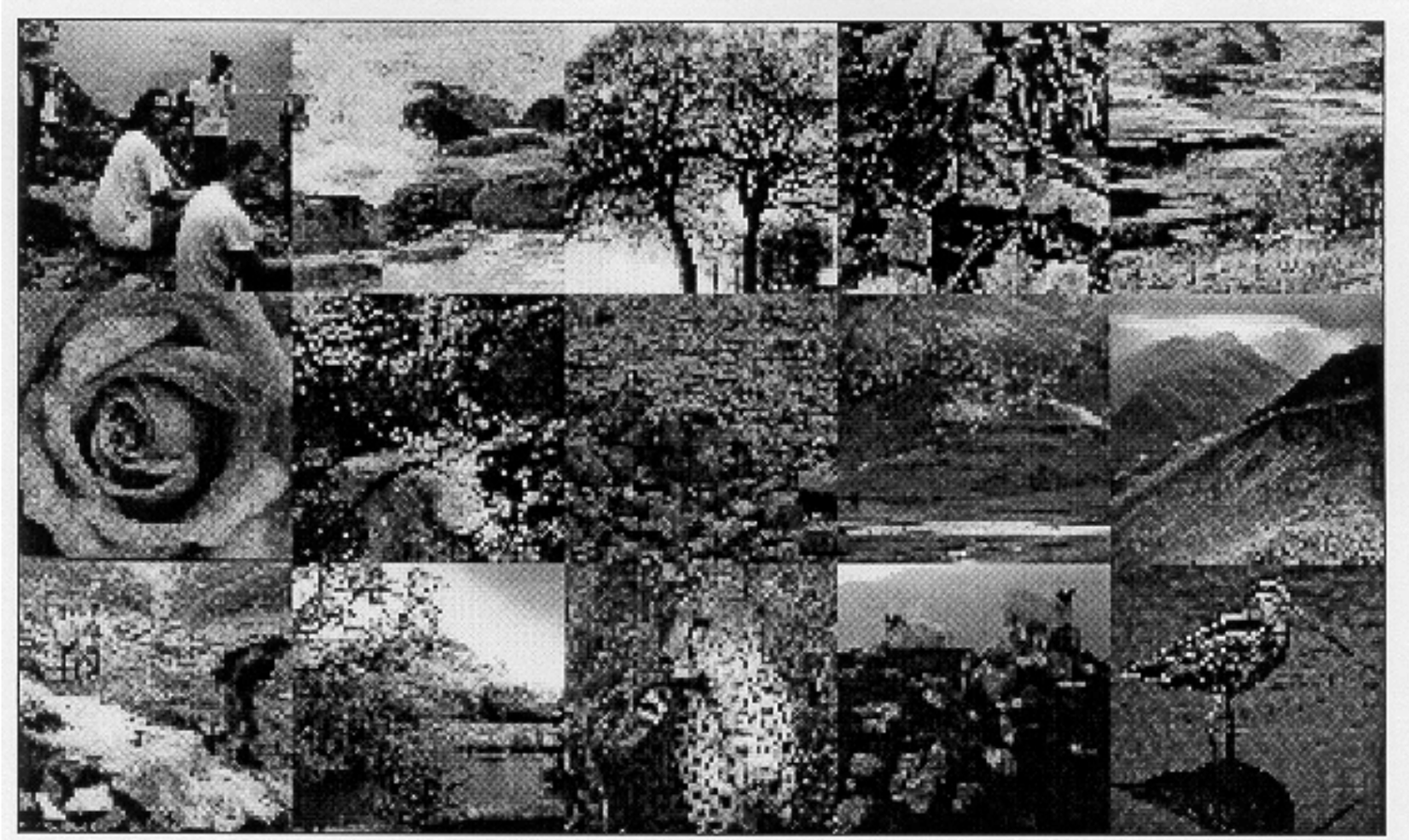
How does this help us?

TECHNISCHE UNIVERSITÄT DARMSTADT

- $\mathbf{S} \in \mathbb{R}^{M \times N}$ is of manageable size now, and so is $\mathbf{V} \in \mathbb{R}^{N \times N}$
- But $\mathbf{U} \in \mathbb{R}^{M \times M}$ is still much too large.
- **Observation 1:** We only need the first N left-singular vectors, because only N singular values can be non-zero.
  - "Economy" decomposition, makes it all manageable memory-wise and computationally.
- Observation 2: We will typically need even fewer than N singular values and left-singular vectors.
  - Special variants that only compute a partial decomposition.

Generic Image Ensembles

TECHNISCHE UNIVERSITÄT DARMSTADT



Fleet & Szeliski

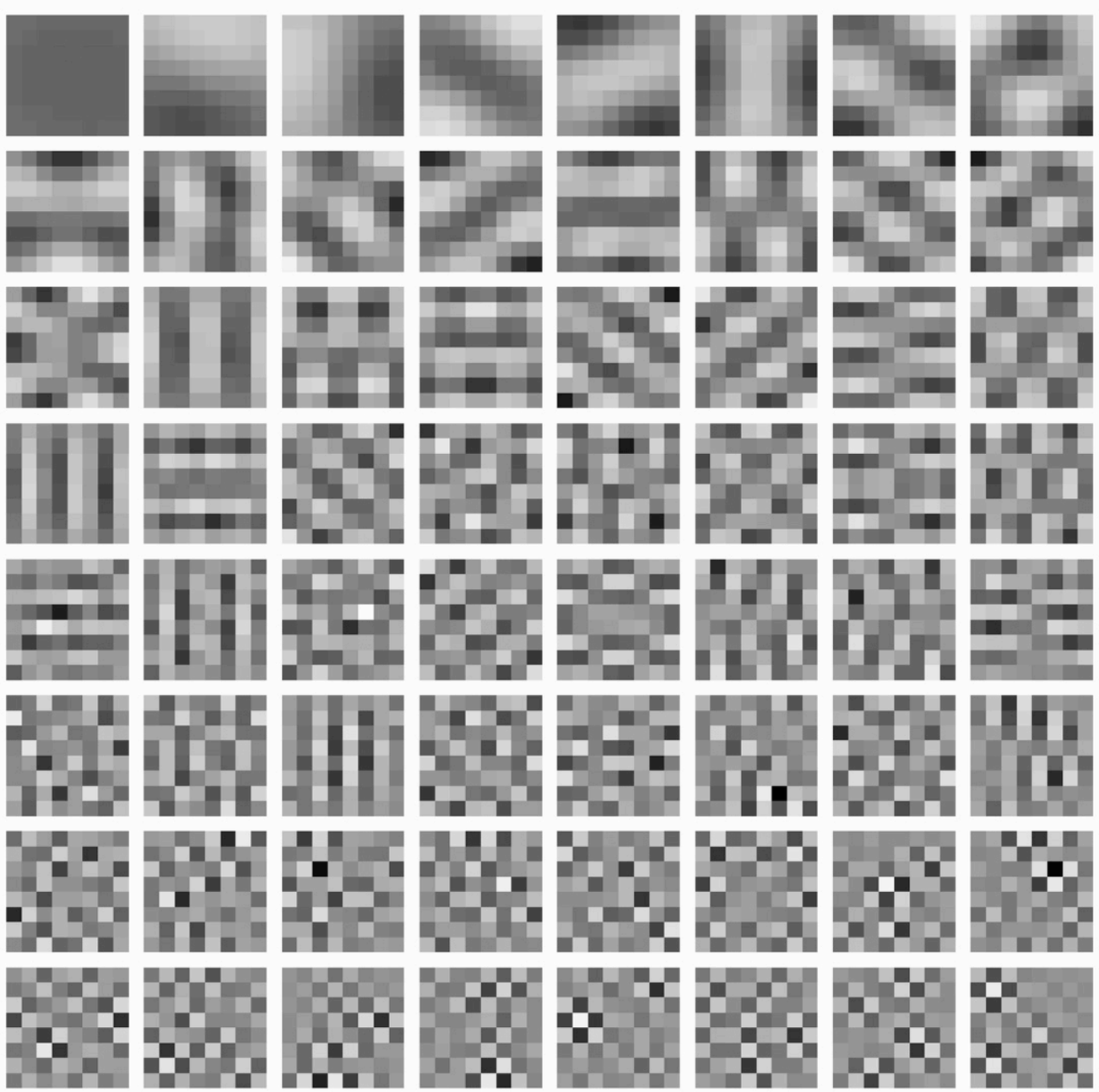
Generic Image Ensembles

TECHNISCHE UNIVERSITÄT DARMSTADT

Is there a low dimensional model describing natural images?

Fleet & Szeliski

PCA of Natural Image Patches



What do these bases look like?

8X8 image patches

Fleet & Szeliski

Search and Recognition

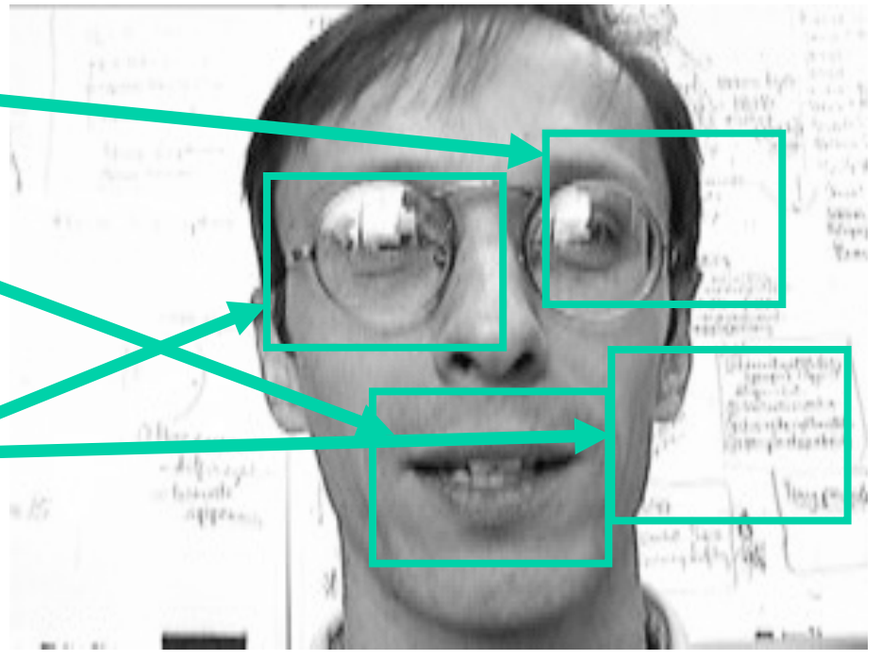
TECHNISCHE UNIVERSITÄT DARMSTADT

- How can we find the mouth?
- How can we recognize the “expression”?

Naïve View-Based Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

Database of mouth “templates”

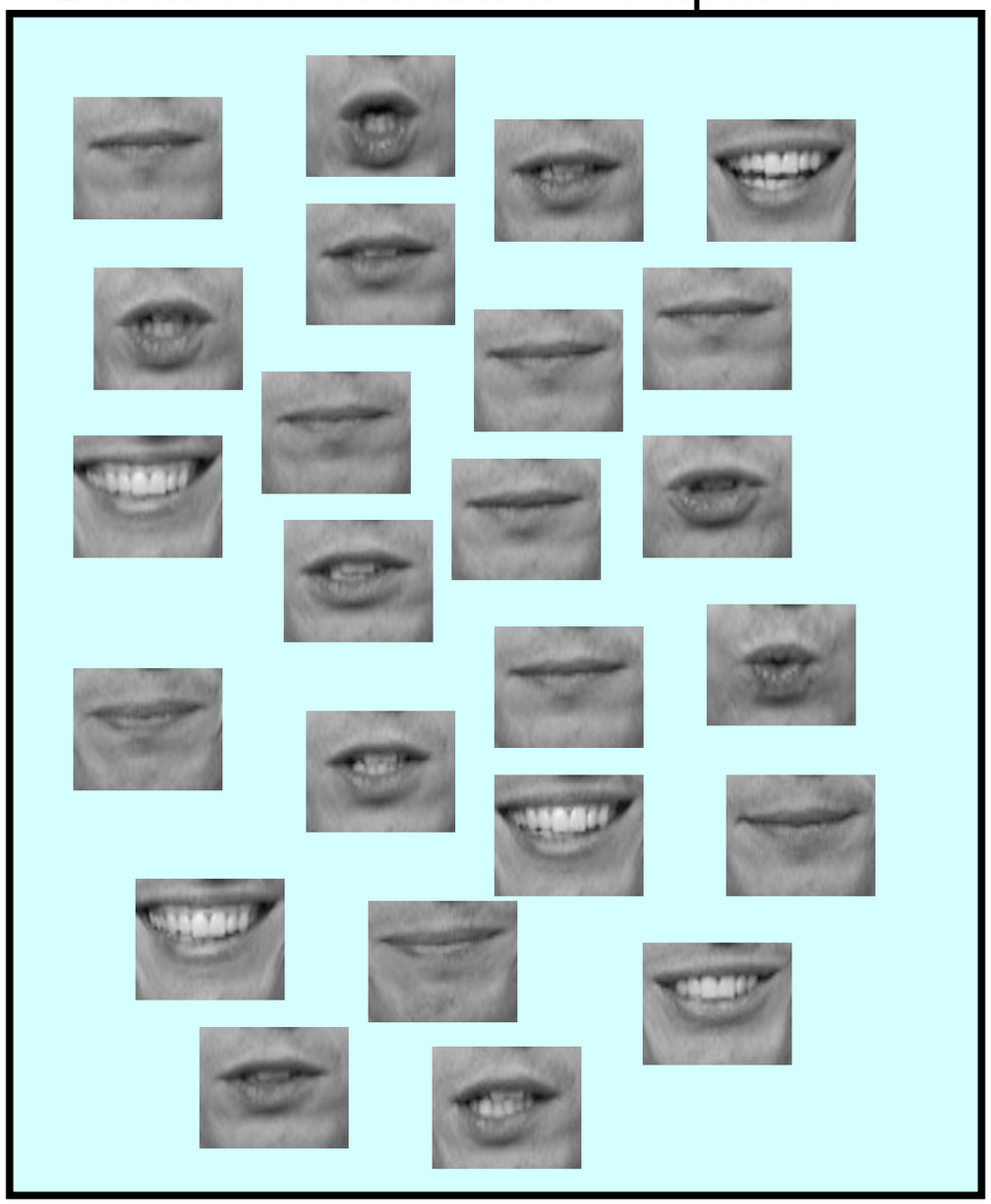


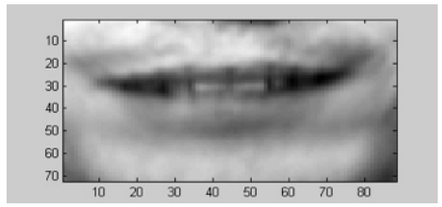
- Search every image region (at every scale).
- Compare each template; chose the best match.

View-Based Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

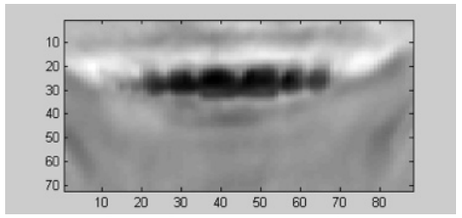
Database of mouth “templates”

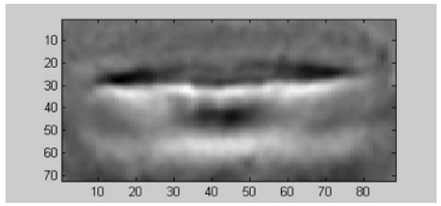


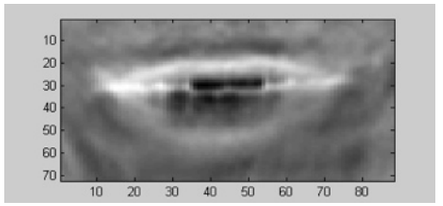


Mean

First three eigenvectors:



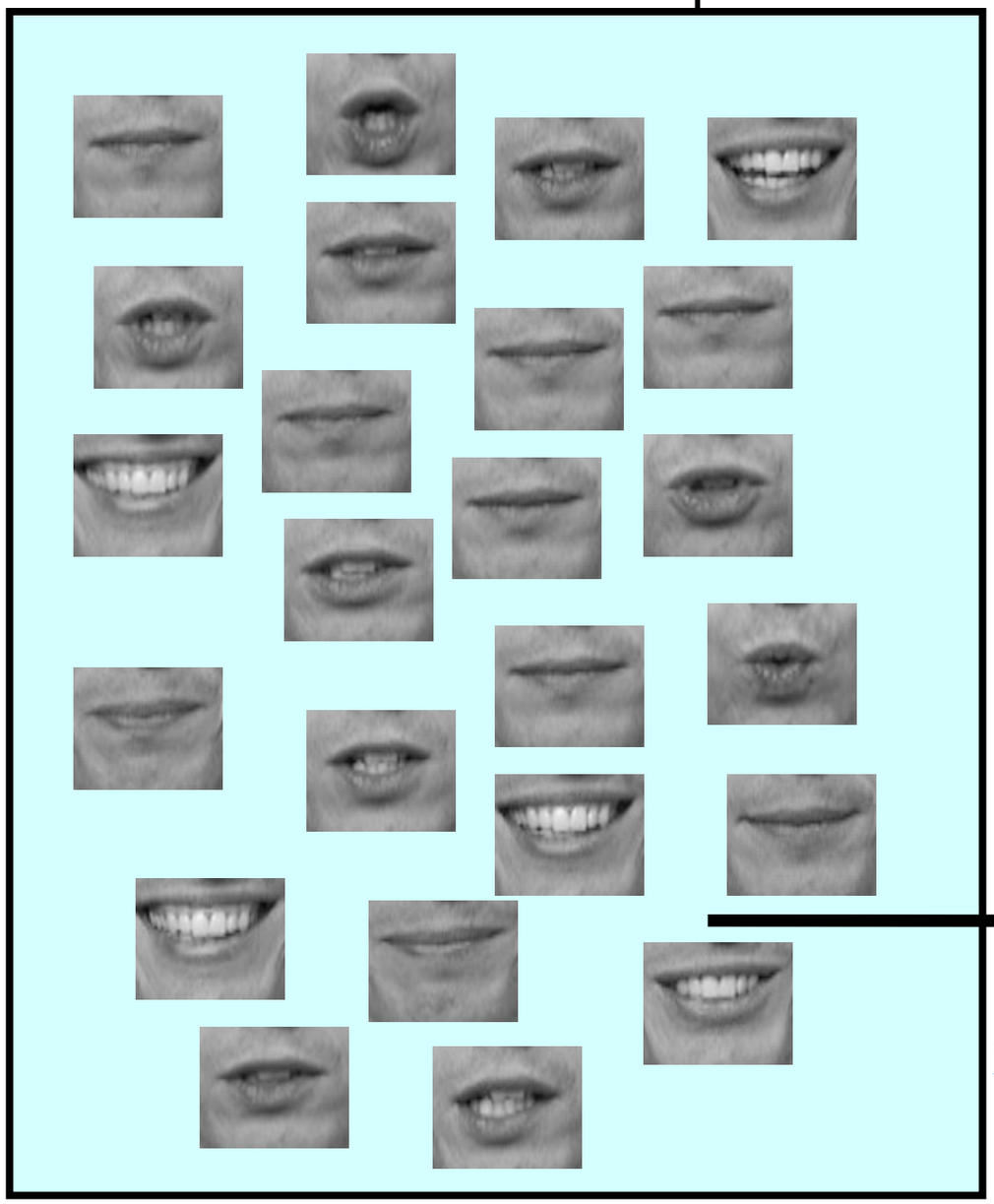


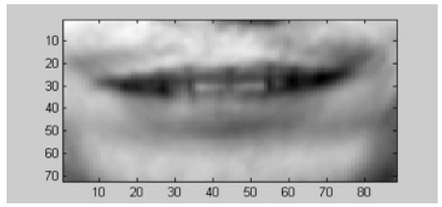


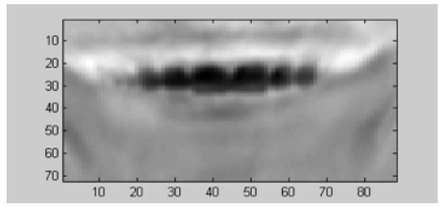
View-Based Approach

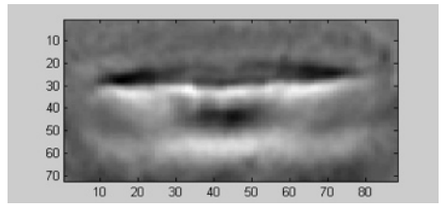
TECHNISCHE UNIVERSITÄT DARMSTADT

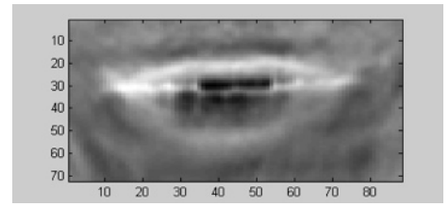
Database of mouth “templates”



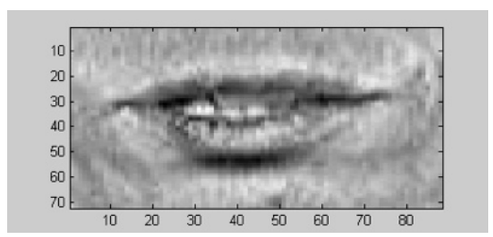








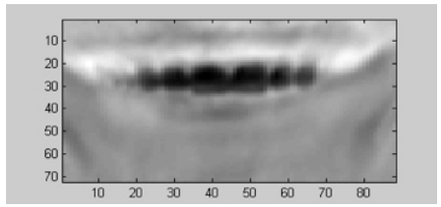
project

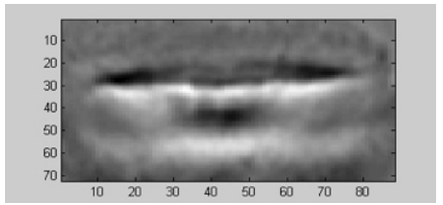


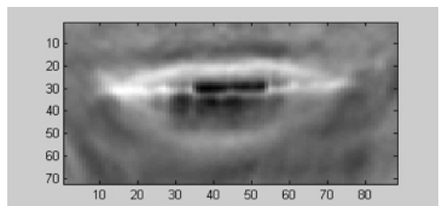
Subtract mean

$$\begin{bmatrix} 715.2646 \\ -376.2406 \\ -331.9533 \end{bmatrix}$$

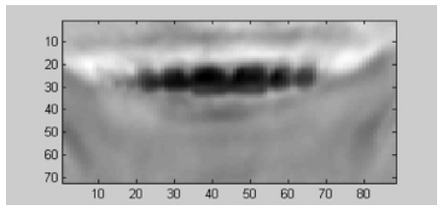
Mouth Space

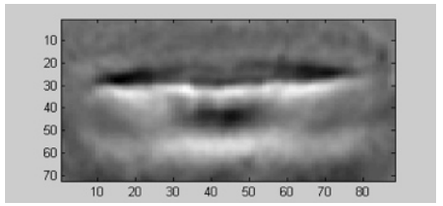

10
20
30
40
50
60
70
10 20 30 40 50 60 70 80

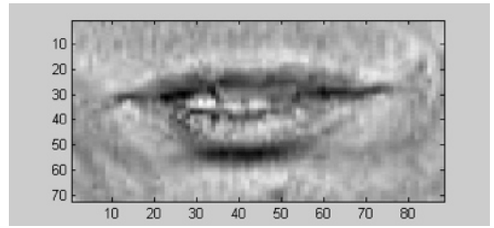

10
20
30
40
50
60
70
10 20 30 40 50 60 70 80

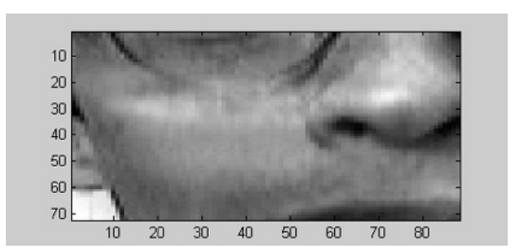

10
20
30
40
50
60
70
10 20 30 40 50 60 70 80

Mouth Space

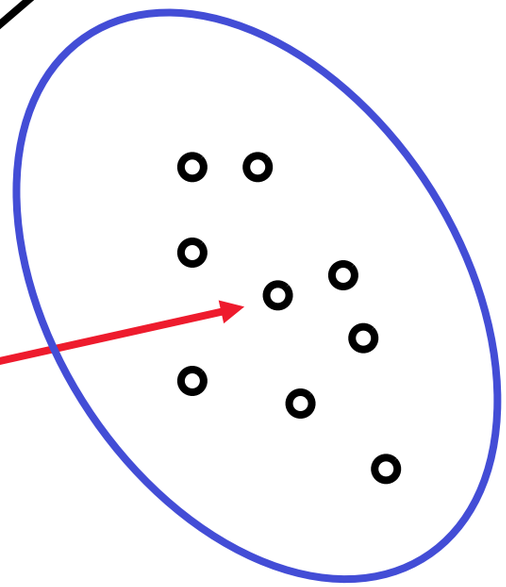




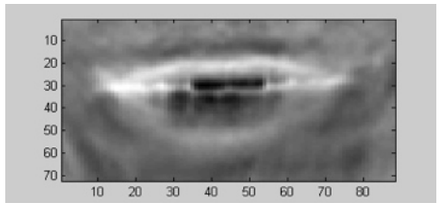




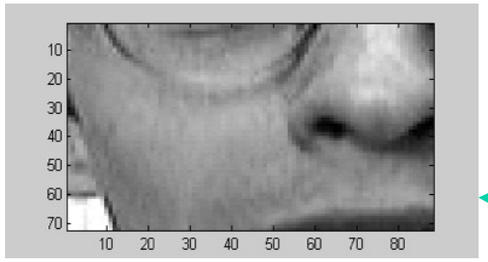
$$\begin{bmatrix} -725.1966 \\ 500.9761 \\ 872.4541 \end{bmatrix}$$

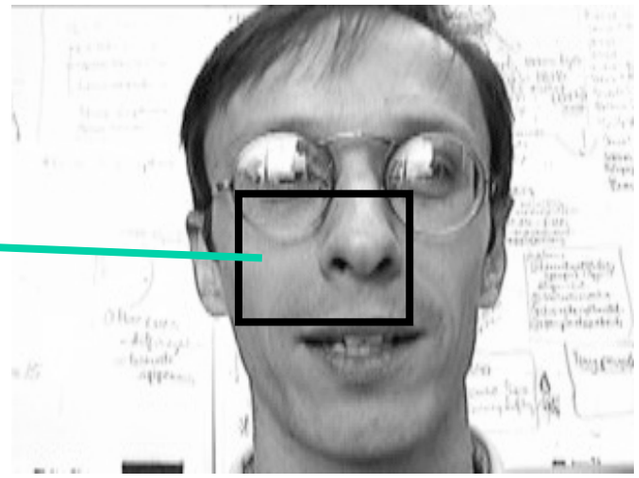


$$\begin{bmatrix} 715.2646 \\ -376.2406 \\ -331.9533 \end{bmatrix}$$



Simple Search Strategy





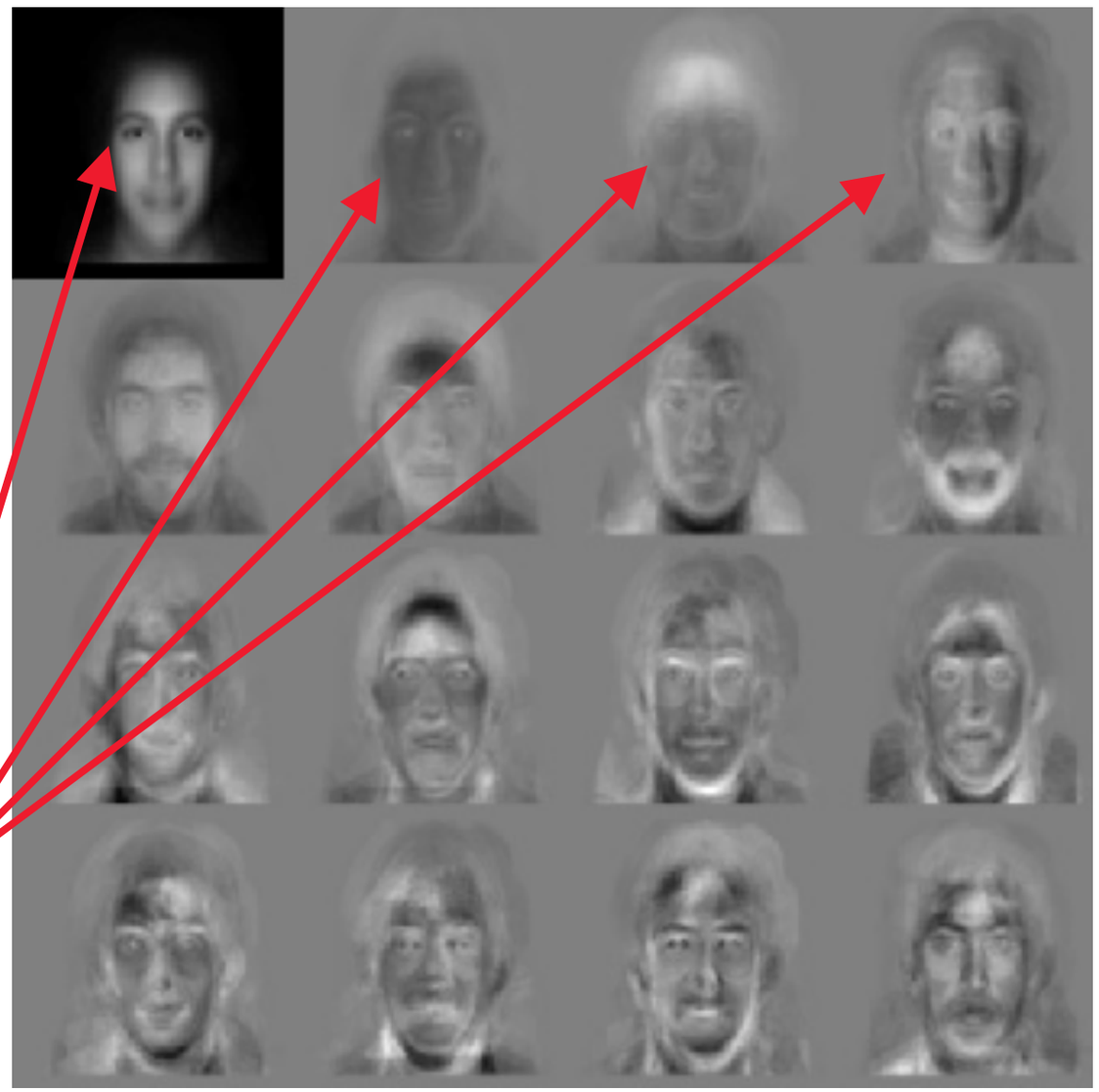
- Project each training image onto the low-dimensional subspace. Store the vectors of coefficients.
- For each image region:
  - Project it onto the low-dimensional subspace.
  - Compare this to each stored coefficient vector (cheap).
  - If the smallest distance is less than some threshold, then it is a mouth.

EigenFaces

- First popular use of PCA for object recognition was for the detection and recognition of faces

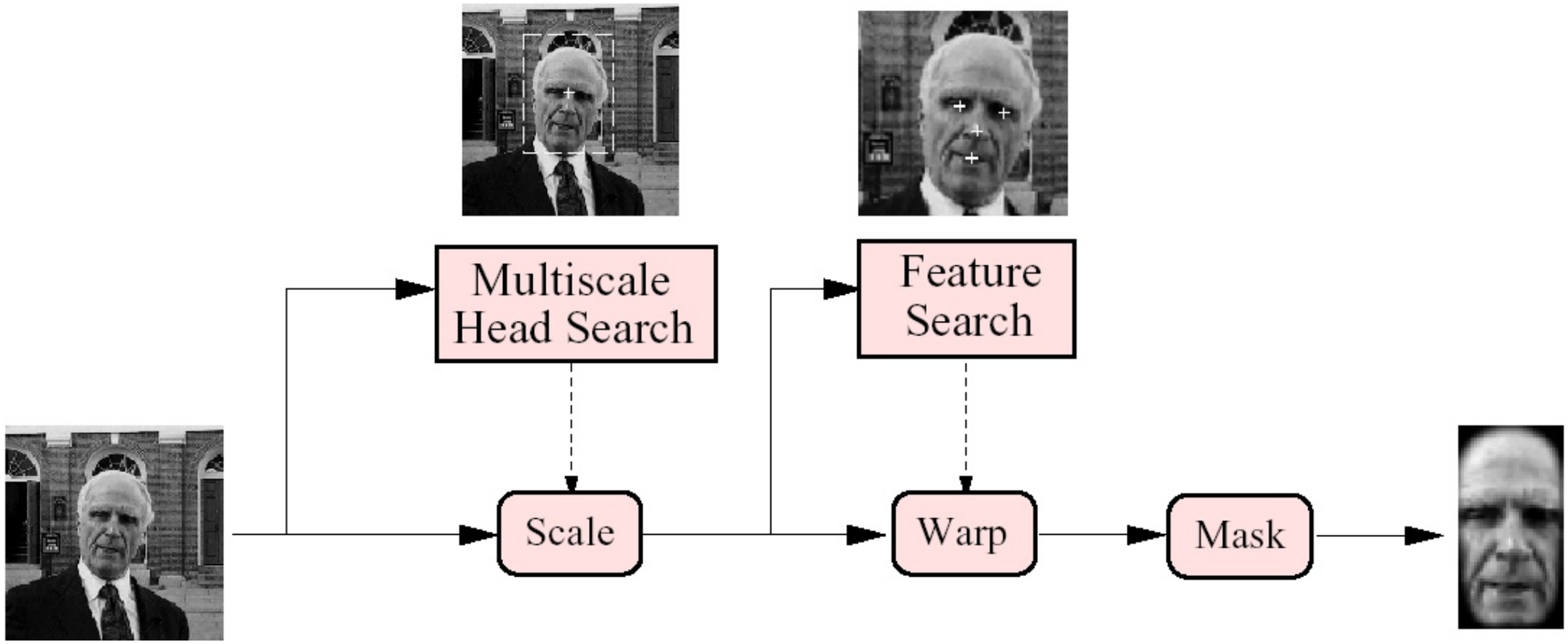
  [Turk and Pentland, 1991]

  - Collect a face ensemble
  - Normalize for contrast, scale, & orientation.
  - Remove backgrounds
  - Apply PCA & choose the first D eigen-images that account for most of the variance of the data.



Fleet & Szeliski

Face Recognition



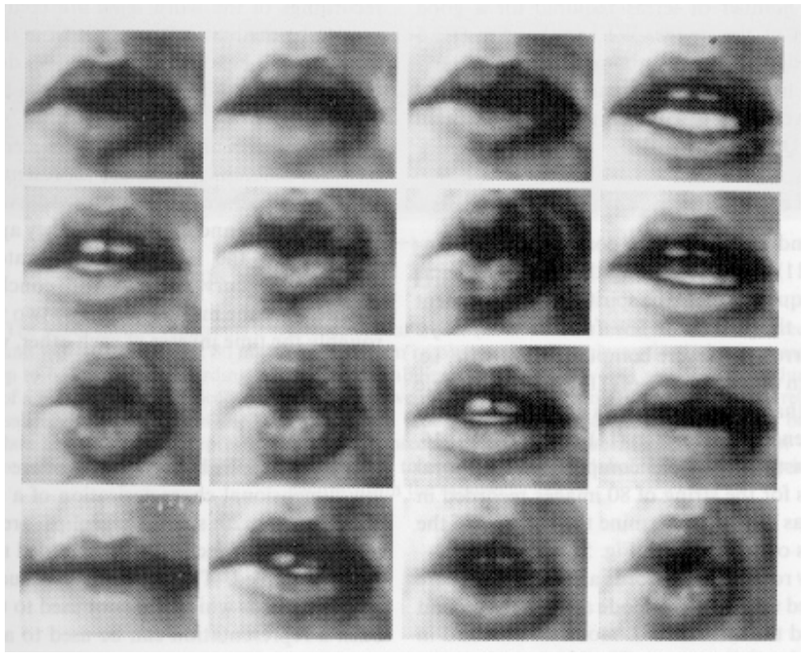
[Moghaddam, Jebara & Pentland, 2000]

Fleet & Szeliski

EigenFeatures

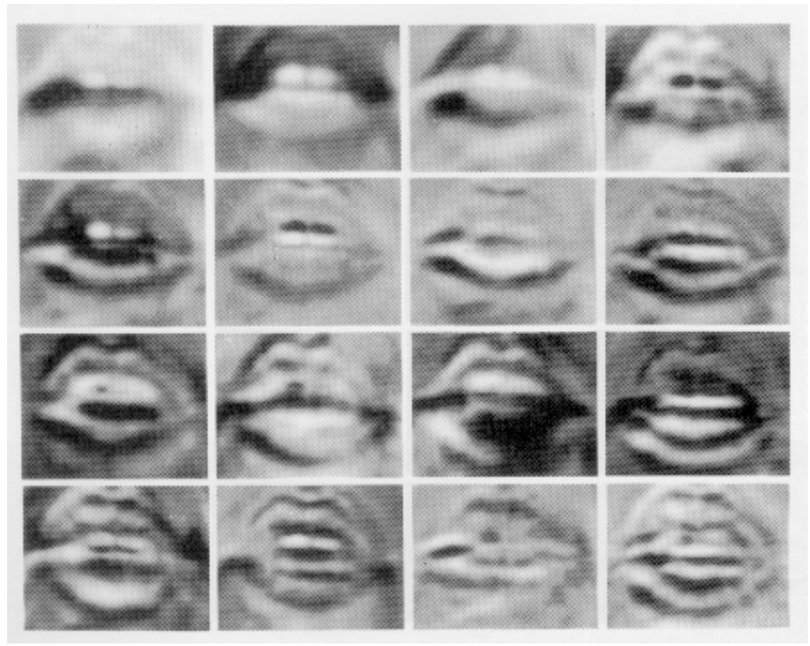
TECHNISCHE UNIVERSITÄT DARMSTADT

Data:



PCA

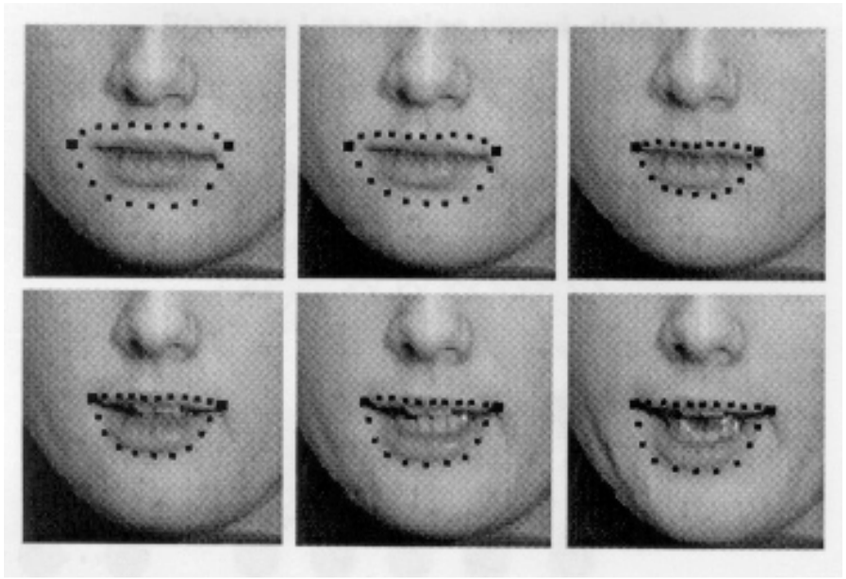
New Basis Vectors



Fleet & Szeliski

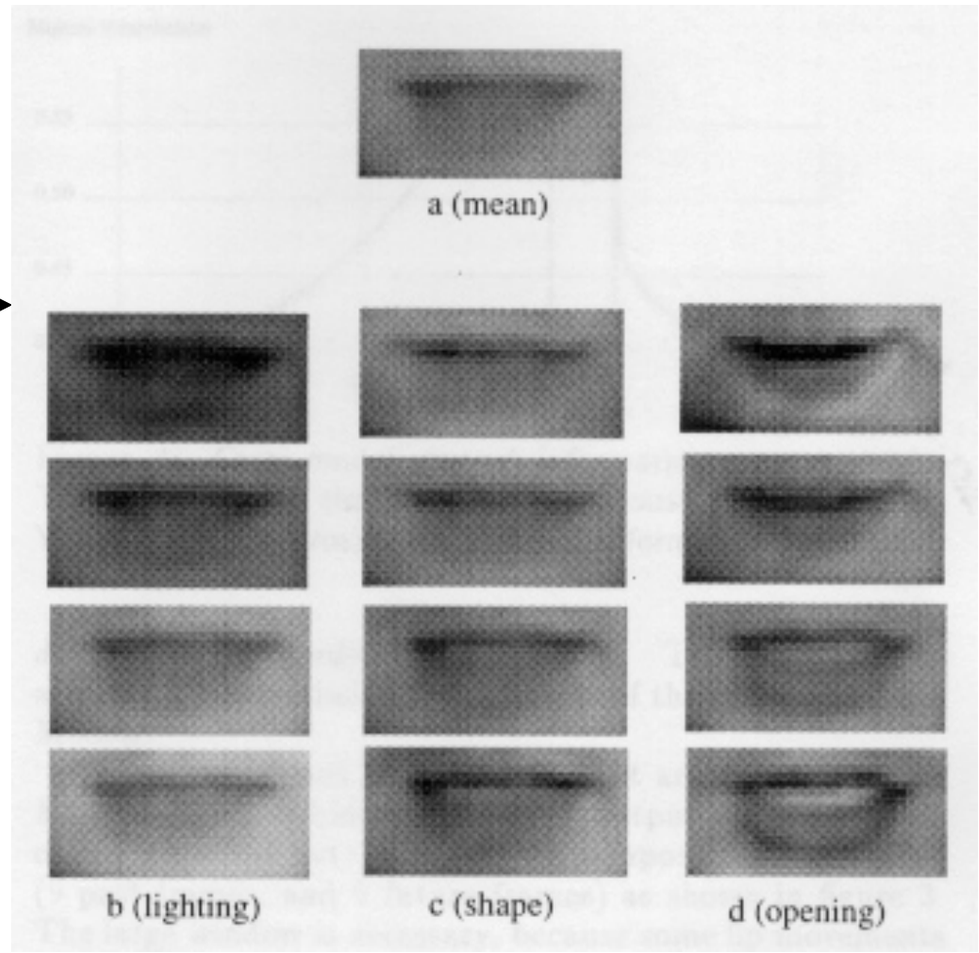
EigenFeatures

Data:



PCA

EigenLips

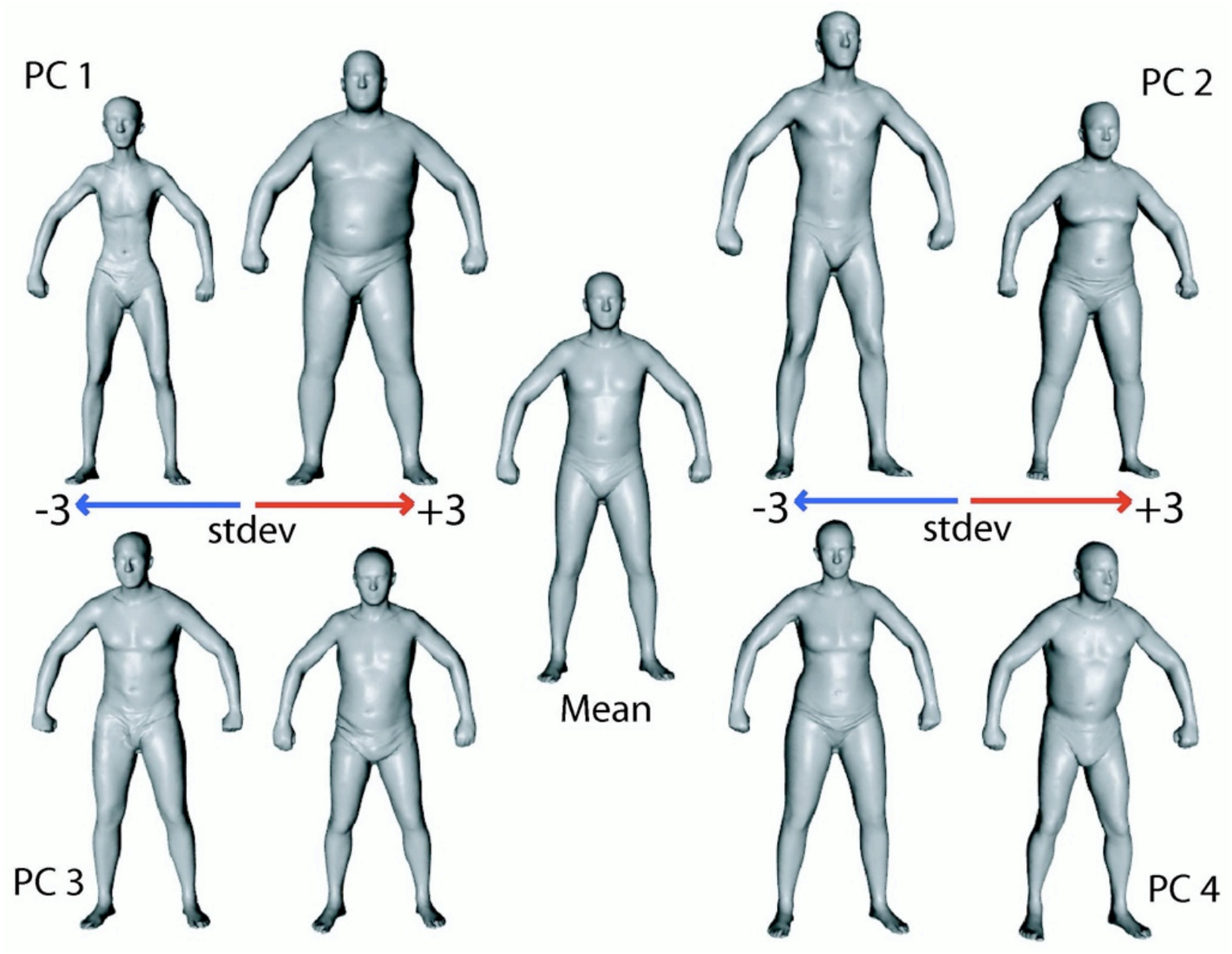


Fleet & Szeliski

PCA Model of Body Shapes

TECHNISCHE UNIVERSITÄT DARMSTADT

- PCA on a detailed triangle model of human bodies:



[Anguelov et al. 05]

Readings

TECHNISCHE UNIVERSITÄT DARMSTADT

- Edge detection, sec. 4.2
- PCA and face recognition, sec. 14.2

- For next time:
  - Feature points, sec. 4.1
  - Bag of word models, sec. 14.4.1

Announcements

TECHNISCHE UNIVERSITÄT DARMSTADT

- **No class next week!**
  - Project review meeting

- Homework assignment 1:
  - Is out...
  - Signup for the “Testat” current read only -- will go online again soon

- **Exam**
  - 21. August 2013, 10:00 - 12:00