Subspace Methods for Visual Learning and Recognition

Horst Bischof  
Inst. f. Computer Graphics and Vision, Graz University of Technology  
Austria  
bischof@icg.tu-graz.ac.at

Aleš Leonardis  
Faculty of Computer and Information Science, University of Ljubljana  
Slovenia  
Ales.Leadonis@fri.uni-lj.si
Outline Part 1

- Motivation
- Appearance based learning and recognition
- Subspace methods for visual object recognition
- Principal Components Analysis (PCA)
- Linear Discriminant Analysis (LDA)
- Canonical Correlation Analysis (CCA)
- Independent Component Analysis (ICA)
- Non-negative Matrix Factorization (NMF)
- Kernel methods for non-linear subspaces
Outline Part 2

- Robot localization
- Robust representations and recognition
- Robust PCA recognition
- Scale invariant recognition using PCA
- Illumination insensitive recognition
- Representations for panoramic images
- Incremental building of eigenspaces
- Multiple eigenspaces for efficient representation
- Robust building of eigenspaces
- Research issues
The name of the game

- complex objects/scenes
- varying pose (3D rotation, scale)
- cluttered background/foreground
- occlusions (noise)
- varying illumination
Object Representation

• High-Level Shape Models (e.g., Generalized Cylinders)
  • Idealized Images
  • Texture Less

• Mid-Level Shape Models (e.g. CAD models, Superquadrics)
  • More Complex
  • Well-defined geometry

• Low-level Appearance Based Models (e.g. Eigenspaces)
  • Most complex
  • Complicated shapes
Problems

Segmentation:

Pose/Shape:
Illumination
Example
Learning and recognition

3D reconstruction

learning

matching

training images

scene

input image

matching
Appearance-based approaches

A renewed attention in the appearance-based approaches

Encompass combined effects of:
  • shape,
  • reflectance properties,
  • pose in the scene,
  • illumination conditions.

Acquired through an automatic learning phase.
Appearance-based approaches

A variety of successful applications:

• Human face recognition e.g. [Beymer & Poggio, Turk & Pentland]
• Visual inspection e.g. [Yoshimura & Kanade]
• Visual positioning and tracking of robot manipulators, e.g. [Nayar & Murase]
• Tracking e.g., [Black & Jepson]
• Illumination planning e.g., [Murase & Nayar]
• Image spotting e.g., [Murase & Nayar]
• Mobile robot localization e.g., [Jogan & Leonardis]
• Background modeling e.g., [Oliver, Rosario & Pentland]
Objects are represented by a large number of views:
Subspace Methods

- Images are represented as points in the N-dimensional vector space
- Set of images populate only a small fraction of the space
- Characterize subspace spanned by images
Subspace Methods

Properties of the representation:

• Optimal Reconstruction ⇒ PCA
• Optimal Separation ⇒ LDA
• Optimal Correlation ⇒ CCA
• Independent Factors ⇒ ICA
• Non-negative Factors ⇒ NMF
• Non-linear Extension ⇒ Kernel Methods
Image Matching

\[ \rho = \frac{x^T y}{\|x\| \|y\|} > \Theta \]

Normalized images

\[ \| x - y \|^2 < \Psi \]

⇒ Compress images
Eigenspace representation

- Image set (normalised, zero-mean)

\[ X = \begin{bmatrix} x_0 & x_1 & \ldots & x_{n-1} \end{bmatrix} ; \quad X \in \mathbb{R}^{m \times n} \]

- We are looking for orthonormal basis functions:

\[ U = \begin{bmatrix} u_0 & u_1 & \ldots & u_k \end{bmatrix} ; \quad k << n \]

- Individual image is a linear combination of basis functions

\[ x_i \approx \tilde{x}_i = \sum_{j=0}^{p} q_j(x_i)u_j \]

\[ \| x - y \|^2 \approx \left\| \sum_{j=1}^{k} q_j(x)u_j - \sum_{j=1}^{k} q_j(y)u_j \right\|^2 = \]

\[ \left\| \sum_{j=1}^{k} (q_j(x) - q_j(y))u_j \right\|^2 = \| q_j(x) - q_j(y) \|^2 \]
Best basis functions $\nu$?

- Optimisation problem

$$\sum_{i=0}^{n-1} \left\| \mathbf{x}_i - \sum_{j=0}^{k} q_j(x_i) \mathbf{u}_j \right\|^2 \rightarrow \min$$

- Taking the $k$ eigenvectors with the largest eigenvalues of

$$C = XX^T = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \ldots & \mathbf{x}_{n-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_0^T \\ \mathbf{x}_1^T \\ \ldots \\ \mathbf{x}_{n-1}^T \end{bmatrix}$$

- PCA or Karhunen-Loéve Transform (KLT)

$$C \mathbf{u}_i = \lambda_i \mathbf{u}_i$$
Efficient eigenspace computation

- $n \ll m$
- Compute the eigenvectors $u'_i$, $i = 0,\ldots,n-1$, of the inner product matrix

$$Q = X^T X = \begin{bmatrix} x_0^T & x_1^T & \ldots & x_{n-1}^T \end{bmatrix} \begin{bmatrix} x_0 & x_1 & \ldots & x_{n-1} \end{bmatrix} ; \quad Q \in \mathbb{R}^{n \times n}$$

- The eigenvectors of $XX^T$ can be obtained by using $XX^T x'_i = \lambda'_i x'_i$:

$$u'_i = \frac{1}{\sqrt{\lambda'_i}} X u'_i$$
Principal Component Analysis
Principal Component Analysis

\[
\begin{align*}
\text{image} &= q_1 + q_2 + q_3 + \ldots
\end{align*}
\]
Image representation with PCA
Image presentation with PCA
Properties PCA

♦ Any point $x_i$ can be projected to an appropriate point $q_i$ by:

$$q_i = U^T(x_i - \mu)$$

♦ and conversely (since $U^{-1} = U^T$)

$$Uq_i + \mu = x_i$$
Properties PCA

♦ It can be shown that the mean square error between $x_i$ and its reconstruction using only $m$ principle eigenvectors is given by the expression:

$$\sum_{j=1}^{N} \lambda_{j} - \sum_{j=1}^{m} \lambda_{j} = \sum_{j=m+1}^{N} \lambda_{j}$$

♦ PCA minimizes reconstruction error

♦ PCA maximizes variance of projection

♦ Finds a more “natural” coordinate system for the sample data.
PCA for visual recognition and pose estimation

Objects are represented as coordinates in an n-dimensional eigenspace.

An example:

3-D space with points representing individual objects or a manifold representing **parametric eigenspace** (e.g., orientation, pose, illumination).
PCA for visual recognition and pose estimation

- Calculate coefficients
- Search for the nearest point (individual or on the curve)
- Point determines object and/or pose
Calculation of coefficients

To recover $a_i$ the image is projected onto the eigenspace

$$a_i(x) = \langle x, e_i \rangle = \sum_{j=1}^{m} x^j e_{ij} \quad 1 \leq i \leq p$$

- Complete image $x_i$ is required to calculate $a_i$.
- Corresponds to Least-Squares Solution
Linear Discriminant Analysis (LDA)

- PCA minimizes projection error

- PCA is "unsupervised" no information on classes is used
- Discriminating information might be lost
Linear Discriminance Analysis (LDA)

- Maximize distance between classes
- Minimize distance within a class

Fisher Linear Discriminance

$$\rho(w) = \frac{w^T S_B w}{w^T S_W w}$$
LDA: Problem formulation

♦ n Sample images: \( \{x_1, \ldots, x_n\} \)

♦ c classes: \( \{\chi_1, \ldots, \chi_c\} \)

♦ Average of each class: 

\[
\mu_i = \frac{1}{n_i} \sum_{x_k \in \chi_i} x_k
\]

♦ Total average: 

\[
\mu = \frac{1}{n} \sum_{k=1}^{N} x_k
\]
LDA: Practice

♦ Scatter of class i:

\[ S_i = \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T \]

♦ Within class scatter:

\[ S_W = \sum_{i=1}^{c} S_i \]

♦ Between class scatter:

\[ S_B = \sum_{i=1}^{c} |\chi_i| (\mu_i - \mu)(\mu_i - \mu)^T \]

♦ Total scatter:

\[ S_T = S_W + S_B \]
LDA: Practice

- After projection:

\[ y_k = W^T x_k \]

- Between class scatter (of y's):

\[ \tilde{S}_B = W^T S_B W \]

- Within class scatter (of y's):

\[ \tilde{S}_W = W^T S_W W \]
Good separation
Maximization of
\[ \rho(w) = \frac{w^T S_B w}{w^T S_w w} \]
is given by solution of generalized eigenvalue problem
\[ S_B w = \lambda S_w w \]
For the c-class case we obtain (at most) c-1 projections as the largest eigenvalues of
\[ S_B w_i = \lambda S_w w_i \]
LDA

♦ How to calculate LDA for high-dimensional images?

♦ Problem: $S_W$ is always singular
  – Number of pixels in each image is larger than the number of images in the training set

1. Fischerfaces $\rightarrow$ Reduce dimension by PCA and then perform LDA

2. Simultaneous diagonalization of $S_W$ and $S_B$
LDA

♦ Fischerfaces (Belhumeur et.al. 1997)

♦ Reduce dimensionality to n-c with PCA

\[
U_{pca} = \arg \max_U |U^TQU| = [u_1 \ u_2 \ ... \ u_{n-c}]
\]

♦ Further reduce to c-1 with FLD

\[
W_{fld} = \arg \max_W \frac{W^T_{pca}S_BW_{pca}W}{W^T_{pca}S_SW_{pca}W} = [w_1 \ w_2 \ ... \ w_{c-1}]
\]

♦ The optimal projection becomes

\[
W_{opt} = W_{fld}^T U^T
\]
LDA

Example Fisherface of recognition Glasses/NoGlasses
(Belhumeur et.al. 1997)
LDA

- Example comparison for face recognition (Belhumeur et.al. 1997)
  ![Graph showing error rates for different recognition algorithms.](image)

- Superior performance than PCA for face recognition
- Noise sensitive
- Requires larger training set, more sensitive to different training data [Martinez&Kak2001]
Canonical Correlation Analysis (CCA)

♦ Also „supervised“ method but motivated by regression tasks, e.g. pose estimation.

♦ Canonical Correlation Analysis relates two sets of observations by determining pairs of directions that yield maximum correlation between these sets.

♦ Find a pair of directions (canonical factors) \( w_x \in \mathbb{R}^p, w_y \in \mathbb{R}^q \), so that the correlation of the projections \( c = w_x^T x \) and \( d = w_y^T y \) becomes maximal.
What is CCA?

\[ \rho = \frac{E[cd]}{\sqrt{E[c^2]E[d^2]}} = \frac{E[w_x^T x y^T w_y]}{\sqrt{E[w_x^T x x^T w_x]E[w_y^T y y^T w_y]}} \]

Canonical Correlation
\[ 0 \leq r \leq 1 \]

Between Set Covariance Matrix
What is CCA?

- Finding solutions

\[
\mathbf{w} = \begin{pmatrix} \mathbf{w}_x \\ \mathbf{w}_y \end{pmatrix}, \quad 
\mathbf{A} = \begin{pmatrix} 0 & \mathbf{C}_{xy} \\ \mathbf{C}_{yx} & 0 \end{pmatrix}, \quad 
\mathbf{B} = \begin{pmatrix} \mathbf{C}_{xx} & 0 \\ 0 & \mathbf{C}_{yy} \end{pmatrix}
\]

Rayleigh Quotient

Generalized Eigenproblem

\[
\mathbf{r} = \frac{\mathbf{w}^T \mathbf{A} \mathbf{w}}{\mathbf{w}^T \mathbf{B} \mathbf{w}}
\]

\[
\mathbf{A} \mathbf{w} = \mu \mathbf{B} \mathbf{w}
\]
CCA for images

• Same problem as for LDA

• Computationally efficient algorithm based on SVD

\[
A = C_{xx}^{-\frac{1}{2}} C_{xy} C_{yy}^{-\frac{1}{2}}
\]

\[
A = UDV^T
\]

\[
w_{xi} = C_{xx}^{\frac{1}{2}} u_i
\]

\[
w_{yi} = C_{yy}^{\frac{1}{2}} v_i
\]
Properties of CCA

• At most min(p,q,n) CCA factors

• Invariance w.r.t. affine transformations

• Orthogonality of the Canonical factors

\[ w_{xi}^T C_{xx} w_{xj} = 0 \]
\[ w_{yi}^T C_{yy} w_{yj} = 0 \]
\[ w_{xi}^T C_{xy} w_{yj} = 0 \]
CCA Example

Parametric eigenspace obtained by PCA for 2DoF in pose
CCA representation
(projections of training images onto $w_{x1}$, $w_{x2}$)
ICA is a powerful technique from signal processing (Blind Source Separation)

Can be seen as an extension of PCA

PCA takes into account only statistics up to 2\textsuperscript{nd} order

ICA finds components that are statistically independent (or as independent as possible)

Local descriptors, sparse coding
Independent Component Analysis (ICA)

♦ m scalar variables $X = (x_1 \ldots x_m)^T$

♦ They are assumed to be obtained as linear mixtures of $n$ sources $S = (s_1 \ldots s_n)^T$

$$X = AS$$

♦ Task: Given $X$ find $A, S$ (under the assumption that $S$ are independent)
ICA Example

Original Sources

Mixtures

Recovered Sources
ICA Example

ICA basis obtained from 16x16 patches of natural images (Bell&Sejnowski 96)
ICA Algorithms

1. Minimize (Maximize) function
   ♦ Complex matrix (tensor) functions

2. Adaptive Algorithms based on stochastic gradient
   ♦ Measure of independence
   ♦ Non-Gaussian, e.g. Kurtosis, Negentropy

   ♦ Fast ICA Algorithm (Hyvärinen)

   1. \[ W = \frac{W}{\sqrt{||WW^T||}} \]
      Repeat until convergence

   2. \[ W = \frac{3}{2} W - \frac{1}{2} WW^T \]
ICA Properties

♦ ICA works only for Non-Gaussian Sources

♦ Usually centering and Whiteing of data is performed

♦ We can not measure the variance of the components

\[ X = AP^{-1}PS \]

♦ ICA does not provide ordering

♦ ICA components are not orthogonal
ICA for Noise Suppression

♦ Sparse Code Shrinkage (similar to Wavelet Shrinkage, Hyvärinen 99)
Face Recognition using ICA

- PCA vs. ICA on Ferret DB (Baek et.al. 02)

PCA

ICA
Non-Negative Matrix Factorization (NMF)

♦ How can we obtain part-based representation?

♦ Local representation where parts are added

♦ E.g. learn from a set of faces the parts a face consists of, i.e. eyes, nose, mouth, etc.

♦ Non-Negative Matrix Factorization (Lee & Seung 1999) lead to part based representation
Matrix Factorization - Constraints

\[ \mathbf{V} \approx \mathbf{WH} \]

- **PCA**: \( \mathbf{W} \) are orthonormal basis vectors
  
  \[
  \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_n], \quad \mathbf{w}_i \cdot \mathbf{w}_j = \delta_{ij}
  \]

- **VQ**: \( \mathbf{H} \) are unity vectors
  
  \[
  \mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n], \quad \mathbf{h}_j^T = [0,0,1,0,\cdots,0]
  \]

- **NMF**: \( \mathbf{V}, \mathbf{W}, \mathbf{H} \) are non-negative
  
  \[
  V_{ij}, W_{ij}, H_{ij} \geq 0 \quad \forall i, j
  \]
NMF - Cost functions

- Euclidean distance between $A$ and $B$

$$\| A - B \|^2 = \sum_{ij} (A_{ij} - B_{ij})^2$$

- Divergence of $A$ from $B$ (Relative entropy)

$$D(A\|B) = \sum_{ij} (A_{ij} \log \frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij})$$
NMF - update rules

- $\|V - WH\|^2$ is non-increasing under

$$H_{a\mu} \leftarrow H_{a\mu} \frac{(W^TV)_{a\mu}}{(W^TWH)_{a\mu}} \quad W_{i\mu} \leftarrow W_{i\mu} \frac{(VH^T)_{i\mu}}{(WHH^T)_{i\mu}}$$

- $D(V \| WH)$ is non-increasing under

$$H_{a\mu} \leftarrow H_{a\mu} \frac{\sum_i W_{ia} V_{i\mu} / (WH)_{i\mu}}{\sum_k W_{ka}} \quad W_{i\mu} \leftarrow W_{i\mu} \frac{\sum_{\mu} H_{a\mu} V_{i\mu} / (WH)_{i\mu}}{\sum_\nu W_{a\nu}}$$

- We can start with random matrices for $W$ and $H$ and update each matrix iteratively until $W$ and $H$ are at a stationary point - the cost functions are invariant at this point.
Concrete example – Handwritten Digits

- Training data set for learning process

<table>
<thead>
<tr>
<th>2</th>
<th>2</th>
<th>2</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Find basis images from the training set

Training images

Basis images
Reconstruction

New image

Approximation

Basis images

Encoding

[Non-negative]

X

[Non-negative]
Face features

- **NMF**
  - Original Image
  - Basis images
  - Encoding (Coefficients)
  - Reconstructed Image

- **WQ**
  - Original Image
  - Basis images
  - Encoding (Coefficients)
  - Reconstructed Image

- **PCA**
  - Original Image
  - Basis images
  - Encoding (Coefficients)
  - Reconstructed Image
All presented methods are linear

Can we generalize to non-linear methods in a computational efficient manner?
Kernel Methods

Kernel Methods are powerful methods (introduced with Support Vector Machines) to generalize linear methods.

**BASIC IDEA:**

1. Non-linear mapping of data in high dimensional space
2. Perform linear method in high-dimensional space

Non-linear method in original space
Kernel Trick

♦ Problem: High-dimensional spaces:
  ♦ E.g. N=16x16 polynomial of degree 5 ⇒ 10^{10}

♦ Can we avoid computing the non-linear mapping directly?
  ♦ E.g. polynomial and inner products

\[
\Phi(x)\Phi(y) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)(y_1^2, \sqrt{2}y_1y_2, y_2^2) = (xy)^2 = k(x, y)
\]

♦ If algorithm can be specified in terms of dot products and non-linearity satisfies Mercer's condition we can apply the kernel trick
Example kernels

Gaussian: \[ k(x, y) = \exp\left( -\frac{\|x - y\|^2}{2\sigma^2} \right) \]

Polynomial:
\[ k(x, y) = (x \cdot y + c)^d, \quad c \geq 0 \]

Sigmoid:
\[ k(x, y) = \sigma (\kappa \cdot (x \cdot y) + \Theta), \quad \kappa, \Theta \in \mathbb{R} \]

Nonlinear separation can be achieved.
Kernel Principal Component Analysis

KPCA carries out a linear PCA in the feature space $F$

The extracted features take the nonlinear form

$$f_k(x) = \sum_{i=1}^{l} \alpha_i^k k(x_i x),$$

The $\alpha_i^k$ are the components of the $k$-th eigenvector of the matrix

$$(k(x_i x_j))_{ij}$$
Find eigenvectors $\mathbf{V}$ and eigenvalues $\lambda$ of the covariance matrix

$$C = \frac{1}{l} \sum_{i=1}^{l} \Phi(x_i) \cdot \Phi(x_i)^T.$$ 

Again, replace

$$\Phi(x) \cdot \Phi(y).$$

with

$$k(x,y).$$
Kernel PCA Toy Example

Artificial data set from three point sources, 100 point each.
De-noising in 2-dimensions

- A half circle and a square in the plane
- De-noised versions are the solid lines
Kernel-CCA

♦ Reformulation of CCA for finite sample size $n$

$X \ldots$ $p \times n$ matrix of training images

$Y \ldots$ $q \times n$ matrix of pose parameters

\[
\hat{A} = \frac{1}{n-1} \begin{pmatrix} 0 & XY^T \\ XY^T & 0 \end{pmatrix}, \quad \hat{B} = \frac{1}{n-1} \begin{pmatrix} XX^T & 0 \\ 0 & YY^T \end{pmatrix}
\]
Theorem

The component vectors $w_x^*$, $w_y^*$ of the extremum points $w^*$ of

$$r = \frac{w^T \hat{A} w}{w^T \hat{B} w}$$

lie in the span of the training data $X$, $Y$, i.e.,

$$\exists f, g : w_x^* = Xf, w_y^* = Yg$$
\[ K = X^T X \]
\[ L = Y^T Y \quad \text{n} \times \text{n} \quad \text{Inner Product (Gram) Matrix} \]

\[ r = \frac{\begin{pmatrix} f^T \\ g^T \end{pmatrix} \begin{pmatrix} 0 & KL \\ LK & 0 \end{pmatrix} \begin{pmatrix} f \\ g \end{pmatrix}}{\begin{pmatrix} f^T \\ g^T \end{pmatrix} \begin{pmatrix} KK & 0 \\ 0 & LL \end{pmatrix} \begin{pmatrix} f \\ g \end{pmatrix}} \]
Kernel-CCA

♦ Apply non-linear transformations $\phi()$, $\theta()$ to the data

$\phi(X) = <\phi(x_1), \ldots, \phi(x_n)>$  $\theta(Y) = <\theta(y_1), \ldots, \theta(y_n)>$

♦ The Kernel Trick:

$K_{ij} = \phi(x_i)^T \phi(x_j) = k_{\phi}(x_i, x_j)$  $L_{ij} = \theta(y_i)^T \theta(y_j) = k_{\theta}(y_i, y_j)$

Inner Product in Feature Space  Kernel Evaluation in Input Space
Experiments

Hippo: Rotated through 360° (1 DOF) in 2° steps.

Linear Y-Encoding: \( y_i = \) turntable position \( \alpha_i \) in degrees.

CCA-Factor Estimates for \( y_i \)

\[ w_{x1} \]
Experiments

Trigonometric Y-Encoding:
\[ y_i = \langle \sin(\alpha_i), \cos(\alpha_i) \rangle \]

CCA-Factors

\[ w_{x1} \]

\[ w_{x2} \]

Estimates for \( y_{i1}, y_{i2} \)

atan2

![Graphs showing CCA-Factors and estimates for y_i1 and y_i2.](image-url)
Experiments

♦ Using Kernel-CCA, optimal output features can be found automatically

Application of a RBF-kernel to the scalar output parameters $\alpha_i$ yielded two factors pairs with a canonical correlation of 1.
Outline Part 1

♦ Motivation
♦ Appearance based learning and recognition
♦ Subspace methods for visual object recognition
♦ Principal Components Analysis (PCA)
♦ Linear Discriminant Analysis (LDA)
♦ Canonical Correlation Analysis (CCA)
♦ Independent Component Analysis (ICA)
♦ Non-negative Matrix Factorization (NMF)
♦ Kernel methods for non-linear subspaces
Outline Part 2

♦ Robot localization
♦ Robust representations and recognition
♦ Robust recognition using PCA
♦ Scale invariant recognition using PCA
♦ Illumination insensitive recognition
♦ Representations for panoramic images
♦ Incremental building of eigenspaces
♦ Multiple eigenspaces for efficient representation
♦ Robust building of eigenspaces
♦ Research issues
Mobile Robot
Panoramic image
♦ environments are represented by a large number of views
♦ localisation = recognition
Compression with PCA

cumulative sum of λ values
Image representation with PCA
Localisation
Distance vs. similarity
Robot localisation

♦ Interpolated hyper-surface represents the memorized environment.
♦ The parameters to be retrieved are related to position and orientation.
♦ Parameters of an input image are obtained by scalar product.
Localisation

mean error: 15.970863 cm
dimension of ES: 10
Enhancing recognition and representations

- **Occlusions, varying background, outliers**
  - Robust recognition using PCA

- **Scale variance**
  - Multiresolution coefficient estimation
  - Scale invariant recognition using PCA

- **Illumination variations**
  - Illumination insensitive recognition

- **Rotated panoramic images**
  - Spinning eigenimages

- **Incremental building of eigenspaces**
- **Multiple eigenspaces for efficient representations**
- **Robust building of eigenspaces**
Occlusions
To recover $a_i$ the image is projected onto the eigenspace

$$a_i(x) = \langle x, e_i \rangle = \sum_{j=1}^{m} x_j e_{ij} \quad 1 \leq i \leq p$$

Complete image $x_i$ is required to calculate $a_i$.
Corresponds to Least-Squares Solution.
**Non-robustness**

**Drawbacks:** Prone to errors caused by

- occlusions (outliers)
- cluttered background

![Original, Occluded, Reconstruction images]
Robust method

- **Major idea:** Instead of using the standard approach we:
  - subset of data points $\rightarrow$ linear system of equations
  - Robust solution of this system of equations
  - Perform multiple hypotheses

- Hypothesize-and-test paradigm

- Competing hypotheses are subject to a **selection** procedure based on the MDL principle.
Robust algorithm

\[
\{e_1^{(1)}, \ldots, e_{k_1}^{(1)}\} \quad \{e_1^{(P)}, \ldots, e_{k_P}^{(P)}\} \quad \text{Input image}
\]

Generating Hypotheses

\[
\{(a_1, \xi_1, D_1), \ldots, (a_H, \xi_H, D_H)\}
\]

Selection

\[
\{(a_1, \xi_1, D_1), \ldots, (a_S, \xi_S, D_S)\}
\]

Family and Recovered coefficients
Selection

Three cases:

1. **One object**: Select best match \(c_{ii}\)

2. Multiple **non-overlapping** objects: Select local maximum \(c_{ii}\)

3. Multiple **overlapping** objects: MDL-criterion:

The objective function:

\[
F(h) = h^T Ch
\]

\(h^T = [h_1, h_2, \ldots, h_R]\) — set of hypotheses

Diagonal terms of \(C\) express the cost-benefit value for hypothesis \(i\)

\[
c_{ii} = K_1 |D_i| - K_2 ||\bar{\xi}_i||_D - K_3 N_i
\]

Off-diagonal terms handle overlapping hypotheses

\[
c_{ij} = -\frac{K_1 |D_i \cap D_j| + K_2 \xi_{ij}}{2}
\]
Robust recovery of coefficients

Original  Occluded  Standard  Robust
Robustness – Experimental results

Experimental testing on a standard database COIL of 1440 images (20 objects under 72 orientations).
## Recognition and pose estimation

### Pose estimation:

<table>
<thead>
<tr>
<th>Method</th>
<th>Salt &amp; Pepper [%]</th>
<th>Gaussian Noise [(\sigma)]</th>
<th>Occlusions [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0 25 50 75</td>
<td>75 150 225 300</td>
<td>15 30 45 60</td>
</tr>
<tr>
<td>Standard</td>
<td>2 3 3 48</td>
<td>3 3 4 24</td>
<td>3 25 31 45</td>
</tr>
<tr>
<td>Robust</td>
<td>2 3 3 4</td>
<td>4 5 6 10</td>
<td>3 3 16 29</td>
</tr>
</tbody>
</table>

### Recognition (50 % salt & pepper noise):

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Rate</th>
<th>Mean absolute orientation error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>46 %</td>
<td>22°</td>
</tr>
<tr>
<td>Robust</td>
<td>75 %</td>
<td>6°</td>
</tr>
</tbody>
</table>

### Recognition (50 % occlusion):

<table>
<thead>
<tr>
<th>Method</th>
<th>Recognition Rate</th>
<th>Mean absolute orientation error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>12 %</td>
<td>57°</td>
</tr>
<tr>
<td>Robust</td>
<td>66 %</td>
<td>29°</td>
</tr>
</tbody>
</table>
Robust localisation under occlusions
Robust localisation at 60% occlusion

Standard approach

Robust approach
Mean error of localisation with respect to % of occlusion
Multiresolution coefficient estimation

- **Multiresolution**
  - a well-known technique to reduce computational complexity
  - a search for the solution at the coarsest level and then a refinement through finer scales

- **Standard eigenspace method cannot be applied in an ordinary multiresolution way — it relies on the orthogonality of eigenimages.**
Standard multiresolution coefficient estimation

- Eigenimages in each resolution layer are computed from a set of templates in that layer
- Computationally costly and requires additional storage space
Robust multiresolution coefficient estimation

- Robust method requires only a single set of eigenimages obtained on the finest resolution.
- Linear system of equations: does not require orthogonality.
Multiresolution coefficient estimation

Linear System of Equations:

\[ \tilde{x}(r_j) = \sum_{i=1}^{p} a_i e_i(r_j) , \]

Convolution:

\[ (f \ast \tilde{x})(r_j) = \sum_{i=1}^{p} a_i (f \ast e_i)(r_j) , \]

Sub-sampling:

\[ \tilde{x}_\downarrow(r_j) = \sum_{i=1}^{p} a_i e_{i\downarrow}(r_j) , \]

Convolution & Sub-sampling:

\[ (f \ast \tilde{x})_\downarrow(r_j) = \sum_{i=1}^{p} a_i (f \ast e_i)_\downarrow(r_j) , \]

Same coefficients on convolved and sub-sampled eigenimages
Scaled images

Scale Sensitivity:

1. Generate multiple hypotheses at different scales.
2. **Estimate scale & coefficients simultaneously.**
Scale estimation

Minimize:

\[ E_s(\alpha) = (s(x, \alpha) - \sum_{i=1}^{p} a_i e_i)^2 \]

\( s(x, \alpha) \): image scaled by \( \alpha \).

- **Gradient descent** [Black]:
  Taylor series expansion of \( s(x, \alpha) \)
  - Small scale changes
  \( \rightarrow \) High resolution

- **Coarse exhaustive search**:
  - Computationally costly
  \( \rightarrow \) Low resolution
Numerical demonstration

Coefficient Error:

Scale Error:
Multiresolution approach

- Estimate scale & coefficients simultaneously in the pyramid
- Efficient search structure
Experimental results – test image
Experimental results

Cat

120% Scaled cat

Occluded cat

120% Scaled occluded cat

Occluded duck

120% Scaled occluded duck
Illumination insensitive recognition

- Recognition of objects under varying illumination
  - *global illumination changes*
  - highlights
  - shadows
- Dramatic effects of illumination on objects appearance
- Training set under a single ambient illumination
Illumination insensitive recognition

Our Approach

- *Global* eigenspace representation
- *Local* gradient based filters
- Efficient combination of global and local representations
- Robust coefficient recovery in eigenspaces
Eigenspaces and filtering
Filtered eigenspaces

\[ y_{r_i} = \sum_{j=1}^{n} q_j e_{j r_i} \quad 1 \leq i \leq k \]

\[ (f \ast x)(r) = \sum_{i=1}^{p} q_j (f \ast e_i)(r) \]
Gradient-based filters

Global illumination

Gradient-based filters

Steerable filters [Simoncelli]
Robust coefficient recovery

Highlights and shadows

Robust solution of linear equations

Hypothesize & Select

\[ L_{++} = a_1 + a_2 + a_3 + \ldots \]

\[ L_{++} = a_1 + a_2 + a_3 + \ldots \]

\[ L_{++} = a_1 + a_2 + a_3 + \ldots \]

\[ L_{++} = a_1 + a_2 + a_3 + \ldots \]
Experimental results

Test images

Our approach

Standard method

→ Demo
Experimental results

<table>
<thead>
<tr>
<th>obj.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>%</th>
<th>ang.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>360</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100.0</td>
<td>5.25</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>308</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>95.1</td>
<td>10.55</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>504</td>
<td>0</td>
<td>0</td>
<td>100.0</td>
<td>1.05</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>4</td>
<td>3</td>
<td>332</td>
<td>2</td>
<td>92.2</td>
<td>3.37</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td>2</td>
<td>17</td>
<td>0</td>
<td>578</td>
<td>94.4</td>
<td>3.34</td>
</tr>
<tr>
<td>avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>96.4</td>
<td>4.19</td>
</tr>
</tbody>
</table>

Robust filtered method - all eigenvectors used

<table>
<thead>
<tr>
<th>obj.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>%</th>
<th>ang.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>141</td>
<td>0</td>
<td>14</td>
<td>26</td>
<td>179</td>
<td>39.2</td>
<td>10.50</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>254</td>
<td>62</td>
<td>5</td>
<td>3</td>
<td>78.4</td>
<td>18.90</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>4</td>
<td>317</td>
<td>0</td>
<td>183</td>
<td>62.9</td>
<td>3.47</td>
</tr>
<tr>
<td>4</td>
<td>23</td>
<td>6</td>
<td>38</td>
<td>249</td>
<td>44</td>
<td>69.2</td>
<td>7.11</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>1</td>
<td>51</td>
<td>0</td>
<td>557</td>
<td>91.0</td>
<td>6.82</td>
</tr>
<tr>
<td>avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>70.3</td>
<td>8.53</td>
</tr>
</tbody>
</table>

Standard method - all eigenvectors used
Illumination invariant localisation

- Illumination variations and occlusions
Filtered eigenvectors
Experimental results

♦ Training set: straight path, uniform illumination
Experimental results

Test sets T/1/2/3 without occlusion

Test sets 4/5/6/7 with occlusion
Experimental results

♦ Comparison with standard method

Test set 2

Test set 6
Experimental results

♦ Comparison with standard method

Test set 3

Test set 7
Experimental results

♦ Comparison with standard method

Coefficient error, test sets 2 and 6

Test set 2

Test set 6
Experimental results

♦ Comparison with standard method

Coefficient error, test sets 3 and 7

Test set 3

Test set 7
### Experimental results

- **Average localisation error (in cm).**

<table>
<thead>
<tr>
<th>Set</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>7</td>
<td>48.7</td>
<td>73.8</td>
<td>2.5</td>
<td>13.5</td>
<td>57.8</td>
<td>108.0</td>
</tr>
<tr>
<td>Normalized</td>
<td>1.5</td>
<td>3.3</td>
<td>65.0</td>
<td>0.8</td>
<td>3.3</td>
<td>19.0</td>
<td>68.3</td>
</tr>
<tr>
<td>Filtered.</td>
<td>0</td>
<td>1.3</td>
<td>4.0</td>
<td>0.5</td>
<td>1.8</td>
<td>2.3</td>
<td>14.0</td>
</tr>
</tbody>
</table>
Rotated panoramic images
Unwrapping
A rotated panoramic image

- rotated/shifted $n$ times

\[ X^{mn} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \end{bmatrix} \]

- Inner product matrix $Q = X^T X$

- symmetric, Toeplitz, circulant

\[ Q = \begin{bmatrix} q_0 & q_1 & \cdots & q_{n-2} & q_{n-1} \\ q_{n-1} & q_0 & q_1 & \cdots & q_{n-2} \\ q_{n-2} & q_{n-1} & q_0 & q_1 & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots \\ q_1 & \cdots & q_{n-2} & q_{n-1} & q_0 \end{bmatrix} \]
Eigenvectors of a circulant matrix

♦ Shift theorem: the eigenvectors of a general circulant matrix are the \( N \) basis vectors from the Fourier matrix \( F = [u_0', u_1', \ldots, u_{n-1}'] \), where

\[
\mathbf{u}_i' = \left[ 1, \omega^i, \omega^{2i}, \ldots, \omega^{(n-1)i} \right]^{\top}, \quad i = 0, \ldots, n - 1
\]

♦ The eigenvalues can be calculated simply by retrieving the magnitude of the DFT of one row of \( Q \)

\[
\omega = e^{-2\pi j/n}, \quad j = \sqrt{-1}
\]

\[
\lambda_i = \sum_{l=0}^{n-1} q_l \omega^{il}
\]
From $u'_i$ to $u_i$

- The eigenvectors of $XX^T$ can be obtained by using $XX^T u'_i = \lambda'_i X u'_i$:

$$u_i = \frac{1}{\sqrt{\lambda'_i}} X u'_i$$

- **eigenvectors** $u_i$
  - same frequency as $u'_i$,
  - phase and amplitude may change
Eigenvectors
Generalisation to several locations
A set of rotated images

♦ $P$ different locations, each shifted $n$ times

$$A = X^T X = \begin{bmatrix}
Q_{00} & Q_{01} & \cdots & Q_{0,P-1} \\
Q_{10} & Q_{11} & \cdots & Q_{1,P-1} \\
\vdots & \vdots & \ddots & \vdots \\
Q_{P-1,0} & Q_{P-1,1} & \cdots & Q_{P-1,P-1}
\end{bmatrix}$$

♦ Every $Q_{ij}$ is circulant (but in general not symmetric!)
♦ Is it possible to exploit these properties?

It is still possible to compute the eigenvectors without performing the SVD decomposition of $A$. 
Rotated panoramic images
Eigenvalue problem

♦ Solution of the problem

\[ Aw' = \mu w' \]

♦ matrix blocks \( Q_{jl} \) of \( A \) are circulant matrices

♦ every circulant matrix can be diagonalised in the same basis by Fourier matrix \( F \)

♦ all the submatrices \( Q_{jk} \) have the same set of eigenvectors

\[ u'_i, \quad i = 0, \ldots, n - 1 \]

\[ w'_i = [\alpha_{i0} u'_i T, \alpha_{i1} u'_i T, \ldots, \alpha_{i,P-1} u'_i T] T \]
Derivations

♦ $A\mathbf{w}' = \mu \mathbf{w}'$ written blockwise:

$$
\sum_{l=0}^{P-1} Q_{jl}(\alpha_{il} \mathbf{u}_i') = \mu \alpha_{ij} \mathbf{u}_i', \quad j = 0, \ldots, P - 1
$$

♦ Since $\mathbf{u}_i'$ is an eigenvector of every $Q_{jl}$,

$$
\sum_{l=0}^{P-1} \alpha_{il} \lambda^i_{jl} \mathbf{u}_i' = \mu \alpha_{ij} \mathbf{u}_i', \quad j = 0, \ldots, P - 1,
$$

♦ $\lambda^i_{jk}$ is an eigenvalue of $Q_{jl}$ corresponding to $\mathbf{u}_i'$. 
This implies a new eigenvalue problem

\[ \Lambda \alpha_i = \mu \alpha_i, \]

where

\[ \Lambda = \begin{bmatrix}
\lambda_{00}^i & \lambda_{01}^i & \cdots & \lambda_{0,P-1}^i \\
\lambda_{10}^i & \lambda_{11}^i & \cdots & \lambda_{1,P-1}^i \\
\vdots & \vdots & \ddots & \vdots \\
\lambda_{P-1,0}^i & \lambda_{P-1,1}^i & \cdots & \lambda_{P-1,P-1}^i
\end{bmatrix} \]

and

\[ \alpha_i = [\alpha_{i0}, \alpha_{i1}, \ldots, \alpha_{i,P-1}]^T \]
Since $Q_{jk} = Q_{kj}^T$, it can be proved that $\Lambda$ is Hermitian and we have $P$ linearly independent eigenvectors $\alpha_i$, which provide $P$ linearly independent eigenvectors $w'_i$. Since the same procedure can be performed for every $v'_i$, we can obtain $N \cdot P$ linearly independent eigenvectors of $A$.

It is therefore possible to solve the eigen-problem using $N$ decompositions of order $P$ (as opposed to decomposition of $P \cdot N$).
Complex eigenspace of spinning images

- Real and imaginary part of one of the vectors:

- Real and imaginary part of one of the $w$ vectors:

\[ w_i = \frac{1}{\sqrt{\mu_i}} X w'_i \]
Eigenvectors
Energy distribution

- compressing efficiency of the eigenspace

\[ \frac{\sum_{i=0}^{K-1} \lambda_i}{\sum_{j=0}^{S-1} \lambda_j} \cdot 100\% \]

- 62 images, each in 50 different orientations
Timings for building the eigenspace

♦ Images of dimensions 40x68, each image was rotated/shifted 68 times, i.e., for 40 locations we got 2720 images.
♦ This is also the number of image elements (the border case when the covariance matrix is of the same size as the inner product matrix, and the complexity of the SVD method reaches its upper bound).

<table>
<thead>
<tr>
<th>locations (P)</th>
<th>$XX^T$</th>
<th>$X^TX$</th>
<th>CPLX</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2507.3</td>
<td>55.8</td>
<td>16.1</td>
</tr>
<tr>
<td>20</td>
<td>2569.6</td>
<td>429.2</td>
<td>105.3</td>
</tr>
<tr>
<td>30</td>
<td>2634.8</td>
<td>1400.3</td>
<td>312.4</td>
</tr>
<tr>
<td>40</td>
<td>3007.7</td>
<td>3252.3</td>
<td>853.2</td>
</tr>
</tbody>
</table>
Eigenspace of spinning images

- K-L expansion of a set of rotated panoramic images
- SVD on the complete covariance matrix is not necessary
- Instead, we solve a set of smaller eigen-problems
- The final eigenvectors are composed of locally varying harmonic functions (analytic functions!)
Other approaches

♦ Images stored in arbitrary orientation
♦ Images stored in a reference orientation (e.g. gyrocompass)
♦ Autocorrelation
♦ FFT power spectra
♦ Zero Phase Representation
♦ Eigenspace of spinning-images
Batch computation of PCA

Subspace Methods for Visual Learning and Recognition  H. Bischof and A. Leonardis
Incremental computation of PCA
Incremental computation of PCA – Algorithm

- Increase dimensionality
- Discard a dimension
- Preserve dimensionality
Determining the number of eigenvectors

- Increase the number of dimensions by one if the distance between the last image and its projection is high.

- Increase the number of dimensions by one if the cumulative distance between the images and their projections is high.
Incremental PCA in detail

♦ Extend $U$ with a residual $h_n$ of the new image $y$ and rotate by $R$

♦ Rotation matrix $R$ is a result of the eigenproblem

$$U' = \begin{bmatrix} U & h_n \end{bmatrix} R ; \quad R \in \mathbb{R}^{(k+1) \times (k+1)}$$

♦ $\Lambda'$ is the new eigenvalue matrix

$$D \cdot R = R \cdot \Lambda'$$
Incremental PCA in detail

- $D$ is assembled from the current eigenvalues $\Lambda$, the new image $y$ and its corresponding coefficients $a$

$$D R = R \Lambda'$$

$$D = \frac{n}{n+1} \begin{bmatrix} \Lambda & 0 \\ 0^\top & 0 \end{bmatrix} + \frac{n}{(n+1)^2} \begin{bmatrix} aa^\top & \gamma a \\ \gamma a^\top & \gamma^2 \end{bmatrix}$$

$$\gamma = h_n(y - \bar{x})$$
Comparison with batch method
Localization with incremental method
Reconstruction error through time

![Graph showing reconstruction error over timeline](image)

- **Average reconstruction error**
- **Timeline**
- **Error**

Legend:
- Green line: Incremental method
- Blue dotted line: Batch method (final value)
Multiple Eigenspaces - Motivation

♦ A single eigenspace
  - high dimensionality
  - low-dimensional structure of data is ignored
  - poor generalisation

♦ Ad-hoc partitioning of the image set is not efficient
Multiple eigenspaces – our goal

- Systematically construct multiple low-dimensional eigenspaces from a set of training images

\[ \mathcal{X} = \{x_1, x_2, \ldots, x_n | x_i \in \mathbb{R}^n \} \]

- Each image is described as a linear combination

\[ x_i = \sum_{j=1}^{m} \mathcal{I}_j^{(i)} \sum_{l=1}^{d_j} c_{jl}^{(i)} u_{jl} \]

- Design a numerically feasible and robust procedure
Eigenspace growing and selection

- TS + Initial Number of Eigenspace
- **INITIALISATION**
- **EIGENSPACE GROWING**
- **EIGENSPACE SELECTION**

Convergence:
- No
- Yes

Eigenspaces
Multiple eigenspaces - experiments
Eigenspace growing and selection
Eigenimages of individual eigenspaces
Mean images of individual eigenspaces
"Box" images in four eigenspaces
"Block" images in five eigenspaces
Multiple eigenspaces
Robust Subspace Learning

- **Subspace learning from data containing outliers:**
  - Detect outliers
  - Learn using only inliers.

[D. Skočaj, A. Leonardis, H. Bischof: A robust PCA algorithm for building representations from panoramic images, ECCV 2002]
EM algorithm for learning

Solving systems of linear equations Only in non-missing pixels

E-step: \( \forall j : x_{ij} = \sum_{p=1}^{k} u_{ip} a_{pj} \quad i = 1 \ldots m \mid x_{ij} \notin \mathcal{M} \)

M-step: \( \forall i : x_{ij} = \sum_{p=1}^{k} u_{ip} a_{pj} \quad j = 1 \ldots n \mid x_{ij} \notin \mathcal{M} \)

\[
0 = \alpha \sum_{p=1}^{k} u_{ip} (a_{p,j-1} - 2a_{pj} + a_{p,j+1}) \quad j = 1 \ldots n \mid x_{ij} \in \mathcal{M}
\]

Smoothing in missing pixels
**Energy function**

\[
E = \sum_{j=1}^{n} \sum_{i \in G_j} \left( x_{ij} - \sum_{p=1}^{k} u_{ip} a_{pj} \right)^2 + \alpha \sum_{j=1}^{n} \sum_{i \in B_j} \left( \sum_{p=1}^{k} u_{ip} a''_{pj} \right)^2
\]

- Minimization of reconstruction error in non-missing pixels (the main property of PCA).
- **Smoothing reconstructed values in missing pixels** (additional constraint to prevent over-fitting).
Robust learning algorithm

Input: Learning images containing outliers and occlusions.

1. Compute PV using SVD on the whole image set.
2. Detect outliers (pixels with large reconstruction error).
3. Compute PV from inliers using EM algorithm.
4. Repeat 1.-3. until change in outlier set is small.

Output: Principal subspace, learning images without outliers, detected outliers and occlusions.
Experimental results – synthetic data

- ground truth
- added outliers

- standard PCA 2PC
- standard PCA 8PC
- robust PCA 8PC
Experimental results – real data

- **input**
- **standard PCA**
- **robust PCA**
- **outliers**
Research issues

♦ Comparative studies (e.g., LDA versus PCA, PCA versus ICA)
♦ Robust learning of other representations (e.g. LDA, CCA)
♦ Integration of robust learning with modular eigenspaces
♦ Local versus Global subspace representations
♦ Combination of subspace representations in a hierarchical framework
Further readings

- Recognizing objects by their appearance using eigenimages (SOFSEM 2000, LNCS 1963)
- Robust recognition using eigenimages (CVIU 2000, Special Issue on Robust Methods in CV)
- Hierarchical top down enhancement of robust PCA (SSPR 2002)
- Illumination insensitive eigenspaces (ICCV 2001)
- Mobile robot localization under varying illumination (ICPR 2002)
- Eigenspace of spinning images (OMNI 2000, ICPR 2000, ICAR 2001)
- Incremental building of eigenspaces (ICRA 2002, ICPR 2002)
- Multiple eigenspaces (Pattern Recognition, In press)
- Robust building of eigenspaces (ECCV 2002)
- Generalized canonical correlation analysis (ICANN 2001)