Computer Vision

Spring 2006 15-385,-685

Instructor: S. Narasimhan

Wean 5403
T-R 3:00pm – 4:20pm

Lecture #20

Announcements

- Homework 5 due today.
- Homework 6 will be released this evening.

    Required for Graduate students.

    Extra-credit for undergrads.

- No class this Thursday (April 20).

Principal Components Analysis on Images

Lecture #20

Appearance-based Recognition

- Directly represent appearance (image brightness), not geometry.
- Why?

  Avoids modeling geometry, complex interactions
  between geometry, lighting and reflectance.

- Why not?

  Too many possible appearances!

  m "visual degrees of freedom" (eg., pose, lighting, etc)
  R discrete samples for each DOF

  How to discretely sample the DOFs?

  How to PREDICT/SYNTHESIS/MATCH with novel views?

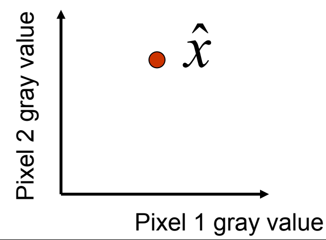
Appearance-based Recognition

- Example:
  - Visual DOFs: Object type P, Lighting Direction L, Pose R
- Set of R * P * L possible images:

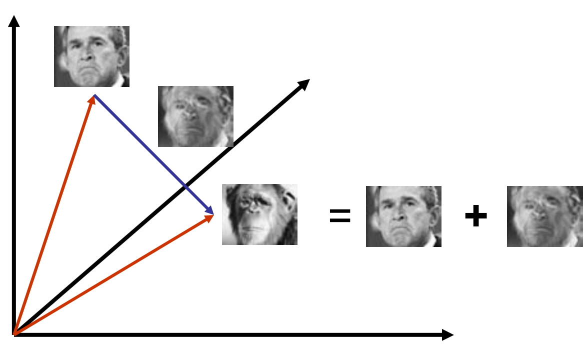
$$\chi = \{\hat{x}_{RL}^{P}\}$$

- Image as a point in high dimensional space:

$\hat{x}$ is an image of N pixels and
A point in N-dimensional space



The Space of Faces



- An image is a point in a high dimensional space
  - An N x M image is a point in R^{NM}
  - We can define vectors in this space as we did in the 2D case

[Thanks to Chuck Dyer, Steve Seitz, Nishino]

Key Idea

- Images in the possible set $\chi = \{\hat{x}_{RL}^{P}\}$ are highly correlated.
- So, compress them to a low-dimensional subspace that captures key appearance characteristics of the visual DOFs.

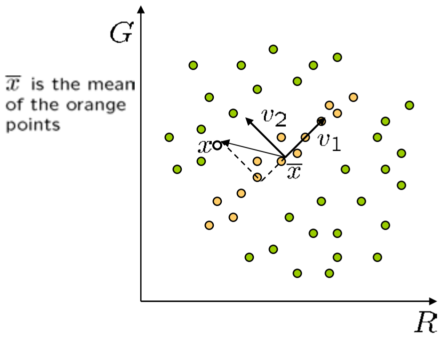
- EIGENFACES: [Turk and Pentland]

USE PCA!

Eigenfaces

Eigenfaces look somewhat like generic faces.

Linear Subspaces



convert **x** into $\mathbf{v_1}$, $\mathbf{v_2}$ coordinates

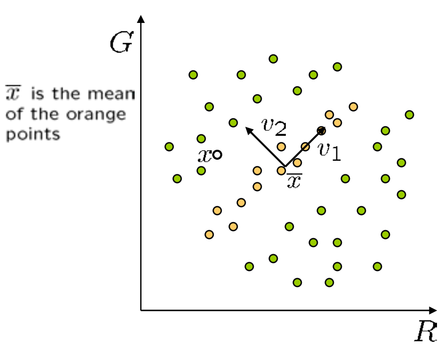
$$\mathbf{x} \rightarrow ((\mathbf{x} - \overline{x}) \cdot \mathbf{v_1}, (\mathbf{x} - \overline{x}) \cdot \mathbf{v_2})$$

What does the $\mathbf{v_2}$ coordinate measure?
- distance to line
- use it for classification—near 0 for orange pts

What does the $\mathbf{v_1}$ coordinate measure?
- position along line
- use it to specify which orange point it is

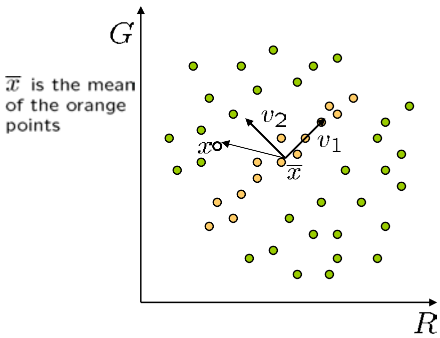
- Classification can be expensive
  - Must either search (e.g., nearest neighbors) or store large probability density functions.
- Suppose the data points are arranged as above
  - Idea—fit a line, classifier measures distance to line

Dimensionality Reduction



- Dimensionality reduction
  - We can represent the orange points with *only* their $\mathbf{v_1}$ coordinates
    - since $\mathbf{v_2}$ coordinates are all essentially 0
  - This makes it much cheaper to store and compare points
  - A bigger deal for higher dimensional problems

Linear Subspaces



Consider the variation along direction **v** among all of the orange points:

$$var(\mathbf{v}) = \sum_{\text{orange point } \mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\|^{2}$$

What unit vector **v** minimizes *var*?

$$\mathbf{v}_2 = min_{\mathbf{v}} \ \{var(\mathbf{v})\}$$

What unit vector **v** maximizes *var*?

$$\mathbf{v}_1 = max_{\mathbf{v}} \ \{var(\mathbf{v})\}$$

$$\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\| \\
&= \sum_{\mathbf{x}} \mathbf{v}^{\mathbf{T}}(\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}}\mathbf{v} \\
&= \mathbf{v}^{\mathbf{T}} \left[\sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}}\right] \mathbf{v} \\
&= \mathbf{v}^{\mathbf{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}}
\end{aligned}$$

Solution: $\mathbf{v_1}$ is eigenvector of **A** with *largest* eigenvalue
$\mathbf{v_2}$ is eigenvector of **A** with *smallest* eigenvalue

Higher Dimensions

- Suppose each data point is N-dimensional
  - Same procedure applies:

$$\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\| \\
&= \mathbf{v}^{\mathbf{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x} - \overline{\mathbf{x}})(\mathbf{x} - \overline{\mathbf{x}})^{\mathbf{T}}
\end{aligned}$$

  - The eigenvectors of **A** define a new coordinate system
    - eigenvector with largest eigenvalue captures the most variation among training vectors **x**
    - eigenvector with smallest eigenvalue has least variation
  - We can compress the data by only using the top few eigenvectors
    - corresponds to choosing a “linear subspace”
      - represent points on a line, plane, or “hyper-plane”
    - these eigenvectors are known as the ***principal components***

Problem: Size of Covariance Matrix A

- Suppose each data point is N-dimensional (N pixels)
  - The size of covariance matrix A is $N^2 \times N^2$
  - The number of eigenfaces is N
  - Example: For N = 256 x 256 pixels,
    Size of A will be 65536 x 65536 !
    Number of eigenvectors will be 65536 !

  Typically, only 20-30 eigenvectors suffice. So, this method is very inefficient!

Efficient Computation of Eigenvectors

If B is MxN and M<<N then $A=B^TB$ is NxN >> MxM

- M → number of images, N → number of pixels
- use BB^T instead, eigenvector of BB^T is easily converted to that of B^TB

$(BB^T)\, y = e\, y$

$\Rightarrow\ B^T(BB^T)\, y = e\, (B^Ty)$

$\Rightarrow\ (B^TB)(B^Ty) = e\, (B^Ty)$

$\Rightarrow$ B^Ty is the eigenvector of B^TB

Eigenfaces – summary in words

- Eigenfaces are

  the eigenvectors of
  the covariance matrix of
  the probability distribution of
  the vector space of
  human faces

- Eigenfaces are the 'standardized face ingredients' derived from the statistical analysis of many pictures of human faces

- A human face may be considered to be a combination of these standardized faces

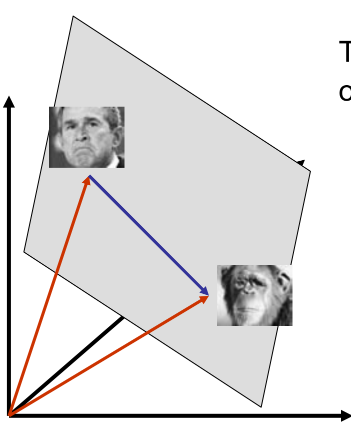
Generating Eigenfaces – in words

1. Large set of images of human faces is taken.
2. The images are normalized to line up the eyes, mouths and other features.
3. The eigenvectors of the covariance matrix of the face image vectors are then extracted.
4. These eigenvectors are called eigenfaces.

Eigenfaces for Face Recognition

- When properly weighted, eigenfaces can be summed together to create an approximate gray-scale rendering of a human face.
- Remarkably few eigenvector terms are needed to give a fair likeness of most people's faces.
- Hence eigenfaces provide a means of applying data compression to faces for identification purposes.

Dimensionality Reduction



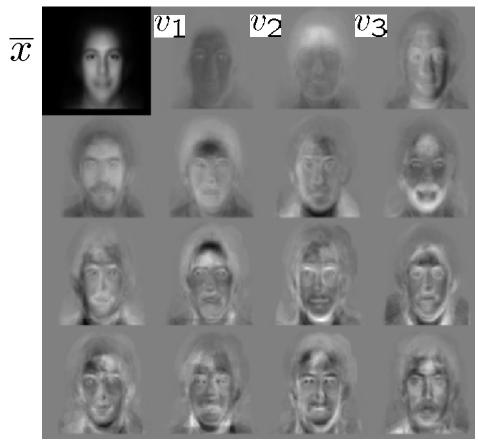
The set of faces is a “subspace” of the set of images

- Suppose it is K dimensional
- We can find the best subspace using PCA
- This is like fitting a “hyper-plane” to the set of faces
  - spanned by vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_K}$

Any face: $\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_k\mathbf{v_k}$

Eigenfaces

- PCA extracts the eigenvectors of **A**
  - Gives a set of vectors $\mathbf{v_1}$, $\mathbf{v_2}$, $\mathbf{v_3}$, ...
  - Each one of these vectors is a direction in face space
    - what do these look like?

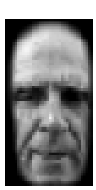


Projecting onto the Eigenfaces

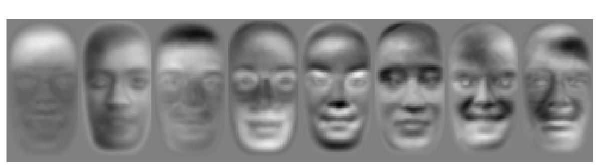
- The eigenfaces $\mathbf{v_1}$, ..., $\mathbf{v_K}$ span the space of faces
  - A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_1}_{a_1},\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_2}_{a_2},\ldots,\ \underbrace{(\mathbf{x}-\overline{\mathbf{x}})\cdot\mathbf{v}_K}_{a_K})$$

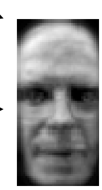
$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots + a_K\mathbf{v}_K$$



$\mathbf{x}$



$a_1\mathbf{v}_1$ $a_2\mathbf{v}_2$ $a_3\mathbf{v}_3$ $a_4\mathbf{v}_4$ $a_5\mathbf{v}_5$ $a_6\mathbf{v}_6$ $a_7\mathbf{v}_7$ $a_8\mathbf{v}_8$



Is this a face or not?

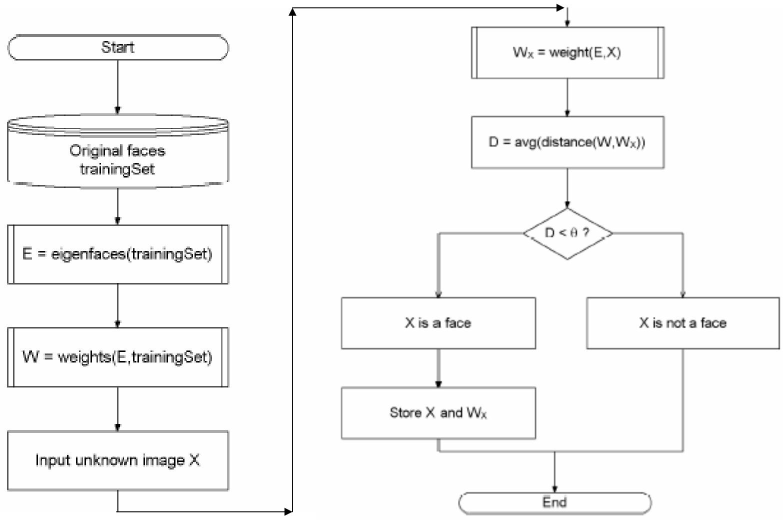


Figure 1: High-level functioning principle of the eigenface-based facial recognition algorithm

Recognition with Eigenfaces

- Algorithm

  1. Process the image database (set of images with labels)
     - Run PCA—compute eigenfaces
     - Calculate the K coefficients for each image
  2. Given a new image (to be recognized) **x**, calculate K coefficients

$$\mathbf{x} \rightarrow (a_1, a_2, \ldots, a_K)$$

  3. Detect if x is a face

$$\|\mathbf{x} - (\overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K})\| < \text{threshold}$$

  4. If it is a face, who is it?
     - Find closest labeled face in database
       - nearest-neighbor in K-dimensional space

Key Property of Eigenspace Representation

Given

- 2 images $\hat{x}_1, \hat{x}_2$ that are used to construct the Eigenspace
- $\hat{g}_1$ is the eigenspace projection of image $\hat{x}_1$
- $\hat{g}_2$ is the eigenspace projection of image $\hat{x}_2$

Then,

$$\boxed{\| \hat{g}_2 - \hat{g}_1 \| \approx \| \hat{x}_2 - \hat{x}_1 \|}$$

That is, distance in Eigenspace is approximately equal to the correlation between two images.

Face Recognition Algorithm

- Consider 2 3x3 images:

$$\begin{bmatrix} 0 & 0 & 0 \\ 10 & 10 & 10 \\ 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 10 & 0 \\ 0 & 10 & 0 \\ 0 & 10 & 0 \end{bmatrix}$$

$$I_1 \qquad\qquad\qquad I_2$$

$$\Rightarrow I_1 = [0\ 0\ 0\ 10\ 10\ 10\ 0\ 0\ 0]^T$$
$$I_2 = [0\ 10\ 0\ 0\ 10\ 0\ 0\ 10\ 0]^T$$

- Say $M' = 1$ and
$$E_1 = [5\ 0\ 5\ 10\ 5\ 10\ 5\ 0\ 5]^T$$

(M' is the number of eigenfaces used)

- Compute Average Image, A

$$A = (I_1 + I_2)/2$$

$$= \left[\frac{0+0}{2} \quad \frac{0+10}{2} \quad \cdots \quad \frac{0+0}{2}\right]$$

$$= \left[0 \; 5 \; 0 \; 5 \; 10 \; 5 \; 0 \; 5 \; 0\right]^T$$

- Project I_1 to 1-D "face space"

$$W_1 = [w_{11}]$$

where

$$w_{11} = E_1^T \cdot (I_1 - A)$$

$$I_1 - A = \left[0\text{-}0 \quad 0\text{-}5 \quad \cdots \quad 0\text{-}0\right]^T$$

$$= \left[0 \; -5 \; 0 \; 5 \; 0 \; 5 \; 0 \; -5 \; 0\right]^T$$

$$\Rightarrow w_{11} = 5 \cdot 0 + 0 \cdot -5 + \cdots + 5 \cdot 0 = 0$$

$$\therefore W_1 = [0]$$

- Project I_2 to 1-D face space

$$W_2 = [-100]$$

- Determine if I_1 or I_2 is closest to test image $I_{test} = \begin{bmatrix} 0 & 7 & 3 \\ 0 & 10 & 10 \\ 0 & 10 & 0 \end{bmatrix}$

$\Rightarrow$ Project I_{test} to face space

$$W_{test} = [w_{t1}]$$

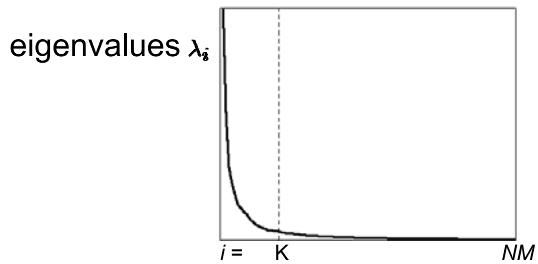
$$w_{t1} = E_1^T \cdot (I_{test} - A)$$

$$= \left[5 \; 0 \; 5 \; 10 \; 5 \; 10 \; 5 \; 0 \; 5\right] \begin{bmatrix} 0 \\ 2 \\ 3 \\ -5 \\ 0 \\ 5 \\ 0 \\ 5 \\ 0 \end{bmatrix}$$

$$= 15$$

$\Rightarrow [15]$ closer to $[0]$ than $[-100]$

Choosing the Dimension K



- How many eigenfaces to use?
- Look at the decay of the eigenvalues
  - the eigenvalue tells you the amount of variance "in the direction" of that eigenface
  - ignore eigenfaces with low variance

Papers

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991a. URL `http://www.cs.ucsb.edu/~mturk/Papers/jcn.pdf`. (URL accessed on November 27, 2002).

M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *Proc. of Computer Vision and Pattern Recognition*, pages 586–591. IEEE, June 1991b. URL `http://www.cs.wisc.edu/~dyer/cs540/handouts/mturk-CVPR91.pdf`. (URL accessed on November 27, 2002).

Limits of PCA

- Attempts to fit a *hyperplane* to the data
  - can be interpreted as fitting a Gaussian, where A is the covariance matrix
  - this is not a good model for some data
- If you know the model in advance, don't use PCA
  - regression techniques to fit parameters of a model
- Several alternatives/improvements to PCA have been developed
  - LLE: http://www.cs.toronto.edu/~roweis/lle/
  - isomap: http://isomap.stanford.edu/
  - kernel PCA: http://www.cs.ucsd.edu/classes/fa01/cse291/kernelPCA_article.pdf
  - For a survey of such methods applied to object recognition
    - Moghaddam, B., "Principal Manifolds and Probabilistic Subspaces for Visual Recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, June 2002 (Vol 24, Issue 6, pps 780-788) http://www.merl.com/papers/TR2002-13/

More Problems: Outliers

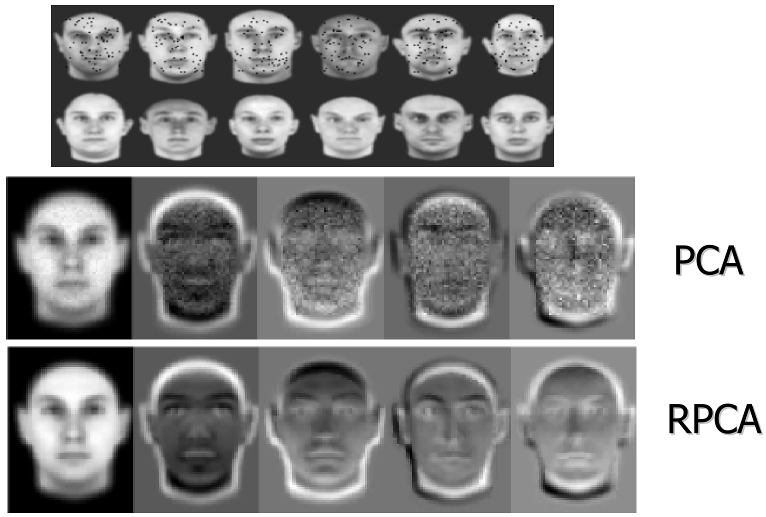
Sample Outliers

Intra-sample outliers

Need to explicitly reject outliers before or during computing PCA.

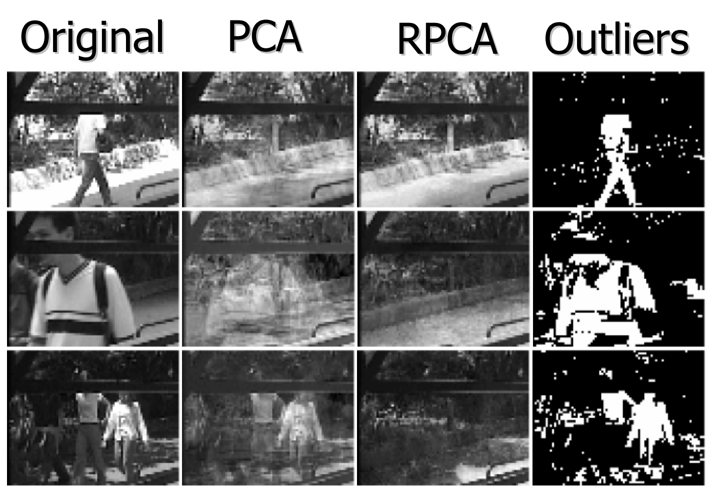
[De la Torre and Black]

Robustness to Intra-sample outliers



RPCA: Robust PCA, [De la Torre and Black]

Robustness to Sample Outliers



Finding outliers = Tracking moving objects

Next Week

Novel Cameras and Displays***

*** Topics change every year

Next Week

- Recent Trends in Computer Vision
  - This semester: Novel Sensors
  - Reading: notes, papers, online resources.