Recognition II

Introduction to Computer Vision

CSE 152

Lecture 19

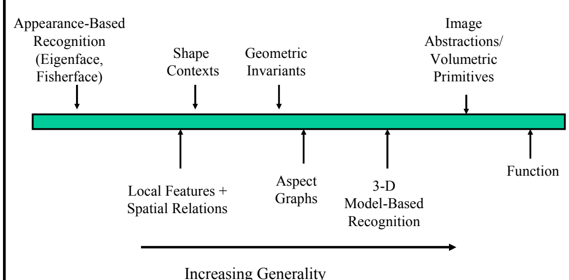
Announcements

- Assignment 5: Due Friday, 4:00
- Read: Trucco & Verri, Chapter 10 on recognition
- Final Exam: Wed, 6/9/04, 11:30-2:30, WLH 2207 (here)

Virtual Cinematography: *Making 'The Matrix' Sequels*
George Borshukov
VFX Technology Supervisor, ESC Entertainment
Friday, June 4, 2004
1:00 p.m. to 2:30 p.m.
[Pizza lunch will precede the event from noon to 1 p.m.]
Main Auditorium, San Diego Supercomputer Center

The presentation will cover the key technologies that had to be developed and deployed to create the synthetic human sequences in the Matrix sequels including Universal Capture - image-based facial animation, realistic human face rendering, and use of measured BRDF in film production. It will also feature a breakdown of The Superpunch shot (pictured above) from "*The Matrix Revolutions*" (the bullet time punch that Neo delivers to Agent Smith during the film's last face-off). This difficult, important, expensive, and challenging shot was entirely computer generated and showcased the technological developments of 3.5+ years at their best by showing a full-frame close-up of a known human actor.

A Rough Recognition Spectrum



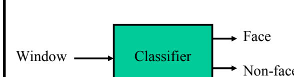
Appearance-Based Vision:

A Pattern Classification Viewpoint

1. Feature Space + Nearest Neighbor
2. Dimensionality Reduction
3. Bayesian Classification
4. Appearance Manifolds

Example: Face Detection

- Scan window over image.
- Classify window as either:
  - Face
  - Non-face



Sketch of a Pattern Recognition Architecture

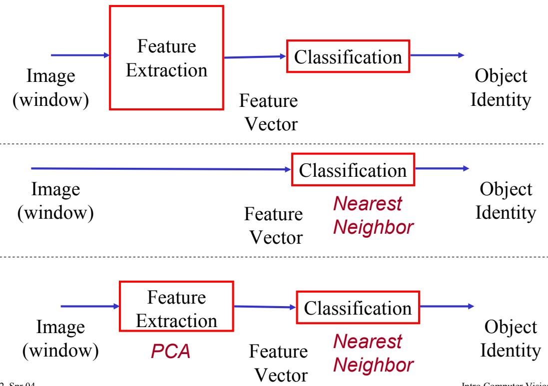
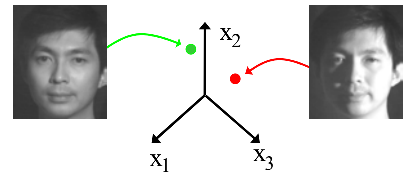


Image as a Feature Vector

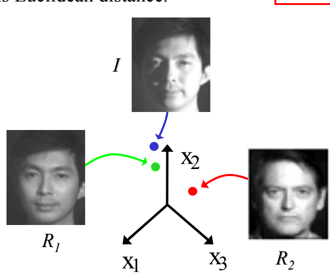


- Consider an n-pixel image (window) to be a point in an n-dimensional space, $\mathbf{x} \in \mathbf{R}^n$.
- Each pixel value is a coordinate of **x**.

Simplest Recognition Scheme

- R_j is an image.
- $c(R_j, I)$ is Euclidean distance.

⇨ **Nearest Neighbor Classifier**



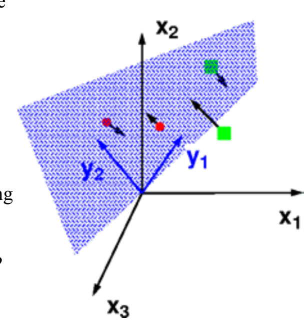
Dimensionality Redcution: Linear Projection

- An n-pixel image $x \in \mathbf{R}^n$ can be projected to a low-dimensional feature space $y \in \mathbf{R}^m$ by

$$y = Wx$$

where W is an m by n matrix.

- Recognition is performed using nearest neighbor in $\mathbf{R}^m$.
- How do we choose a good W?



Eigenfaces: Principal Component Analysis (PCA)

Assume we have a set of n feature vectors $\boldsymbol{x}_i$ $(i = 1, \ldots, n)$ in $\mathbb{R}^d$. Write

$$\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{x}_i$$

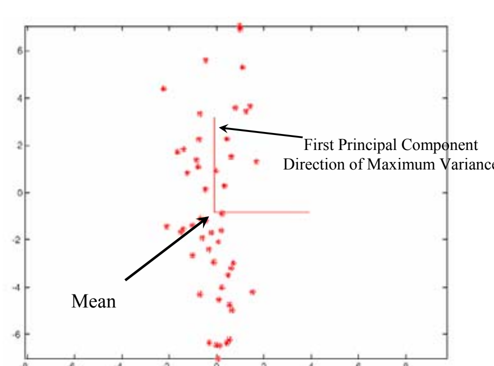
$$\Sigma = \frac{1}{n-1}\sum_i (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

The unit eigenvectors of Σ — which we write as $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_d$, where the order is given by the size of the eigenvalue and $\boldsymbol{v}_1$ has the largest eigenvalue — give a set of features with the following properties:

- They are independent.
- Projection onto the basis $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k\}$ gives the k-dimensional set of linear features that preserves the most variance.

Algorithm 22.5: *Principal components analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset.*

Some details: Use Singular value decomposition, "trick" described in appendix of text to compute basis when n<<d



Eigenfaces

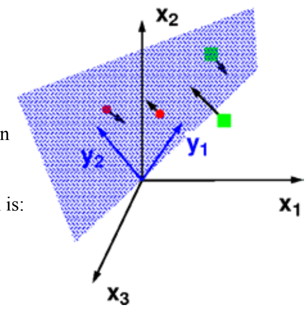
- Modeling
  1. Given a collection of n labeled training images,
  2. Compute mean image and covariance matrix.
  3. Compute k Eigenvectors (note that these are images) of covariance matrix corresponding to k largest Eigenvalues. (Or perform using SVD!!)
  4. Project the training images to the k-dimensional Eigenspace.
- Recognition
  1. Given a test image, project to Eigenspace.
  2. Perform classification to the projected training images.

Projection, and reconstruction

- An n-pixel image $\boldsymbol{x} \in \mathbf{R}^n$ can be projected to a low-dimensional feature space $\boldsymbol{y} \in \mathbf{R}^m$ by

$$\boldsymbol{y} = W\boldsymbol{x}$$

- From $\boldsymbol{y} \in \mathbf{R}^m$, the reconstruction of the point is $\mathbf{W}^T\mathbf{y}$
- The error of the reconstruction is: $||\mathbf{x}-\mathbf{W}^T\mathbf{W}\mathbf{x}||$

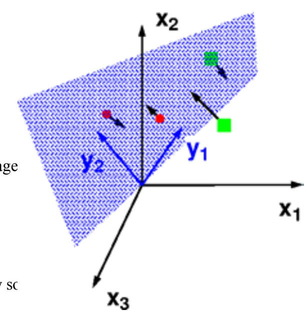


Reconstruction using Eigenfaces

- Given image on left, project to Eigenspace, then reconstruct an image (right).

Face detection using "distance to face space"

- Scan a window ω across the image, and classify the window as face/not face as follows:
- Project window to subspace, and reconstruct as described earlier.
- Compute distance between ω and reconstruction.
- Local minima of distance over all image locations less than some treshold are taken as locations of faces.
- Repeat at different scales.
- Possibly normalize windows intensity sc that |ω| = 1.



And important footnote: Don't really implement PCA this way!

Why?

1. How big is Σ?
   - n by n where n is the number of pixels in an image!!
2. You only need the first k Eigenvectors

Singular Value Decomposition

- Any m by n matrix $\boldsymbol{A}$ may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

$$[m \times n] = [m \times m][m \times n][n \times n]$$

- U: m by m, orthogonal matrix
  - Columns of **U** are the eigenvectors of AA^T
- V: n by n, orthogonal matrix,
  - columns are the eigenvectors of A^TA
- **Σ:** m by n, diagonal with non-negative entries (σ_1, σ_2, …, σ_s) with s=min(m,n) are called the called the singular values
  - **Singular values are the square roots of eigenvalues of both AA^T and A^TA *& Columns of U are corresponding Eigenvectors!!***
  - *Result of SVD algorithm:* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_s$

SVD Properties

- In Matlab [u s v] = svd(A), and you can verify that: A=u*s*v'
- r=Rank(***A***) = # of non-zero singular values.
- ***U***, ***V*** give us orthonormal bases for the subspaces of ***A***:
  - 1st *r* columns of ***U***: Column space of ***A***
  - Last *m* - *r* columns of ***U***: Left nullspace of ***A***
  - 1st *r* columns of ***V***: Row space of ***A***
  - 1st *n* - *r* columns of ***V***: Nullspace of ***A***
- *For d<= r, the first d column of U provide the best d-dimensional basis for columns of A in least squares sense.*

Performing PCA with SVD

- Singular values of A are the square roots of eigenvalues of both AA^T and A^TA & Columns of U are corresponding Eigenvectors
- And $\sum_{i=1}^{n} a_i a_i^T = [a_1 \; a_2 \; \cdots \; a_n][a_1 \; a_2 \; \cdots \; a_n]^T = AA^T$
- Covariance matrix is:

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_i - \vec{\mu})(\vec{x}_i - \vec{\mu})^T$$

- So, ignoring 1/n subtract mean image μ from each input image, create data matrix, and perform (thin) SVD on the data matrix.

Thin SVD

- Any *m* by *n* matrix ***A*** may be factored such that

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T$$

[m x n] = [m x m][m x n][n x n]

- If m>n, then one can view Σ as:

$$\begin{bmatrix} \Sigma' \\ 0 \end{bmatrix}$$

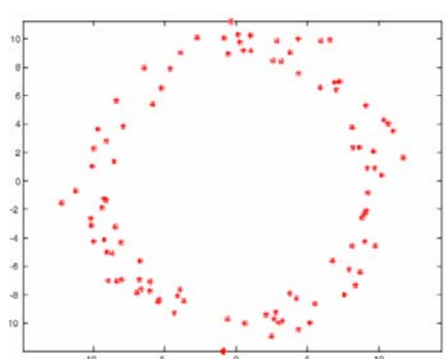
- Where Σ'=diag(σ_1, σ_2, …, σ_s) with s=min(m,n), and lower matrix is (n-m by m) of zeros.
- Alternatively, you can write:

$$\boldsymbol{A} = \boldsymbol{U}'\Sigma'\boldsymbol{V}^T$$

- In Matlab, thin SVD is:[U S V] = svds(A)

This is what you should use!!

Alternative projections



Fisherfaces: Class specific linear projection

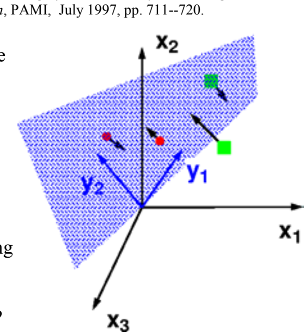
P. Belhumeur, J. Hespanha, D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, PAMI, July 1997, pp. 711--720.

- An *n*-pixel image $\boldsymbol{x} \in \mathbf{R}^n$ can be projected to a low-dimensional feature space $\boldsymbol{y} \in \mathbf{R}^m$ by

$$\boldsymbol{y} = W\boldsymbol{x}$$

where W is an *n* by *m* matrix.

- Recognition is performed using nearest neighbor in $\mathbf{R}^m$.
- How do we choose a good W?



PCA & Fisher's Linear Discriminant

- Between-class scatter

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

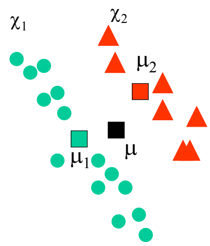
- Within-class scatter

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu_i)(x_k - \mu_i)^T$$

- Total scatter

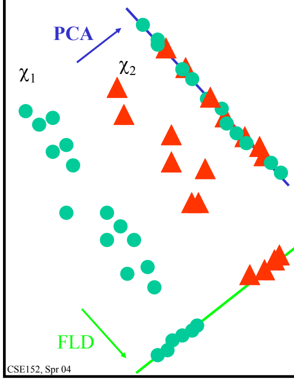
$$S_T = \sum_{i=1}^{c} \sum_{x_k \in \chi_i} (x_k - \mu)(x_k - \mu)^T = S_B + S_W$$

- Where
  - c is the number of classes
  - μ_i is the mean of class χ_i
  - $|\chi_i|$ is number of samples of χ_i.



• If the data points are projected by y=Wx and scatter of points is S, then the scatter of the projected points is W^TSW

PCA & Fisher's Linear Discriminant



- PCA (Eigenfaces)

$$W_{PCA} = \arg\max_W |W^T S_T W|$$

Maximizes projected total scatter

- Fisher's Linear Discriminant

$$W_{fld} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|}$$

Maximizes ratio of projected between-class to projected within-class scatter

Computing the Fisher Projection Matrix

$$W_{opt} = \arg\max_W \frac{|W^T S_B W|}{|W^T S_W W|} = [\mathbf{w}_1 \ \mathbf{w}_2 \ \dots \ \mathbf{w}_m] \qquad (4)$$

where $\{\mathbf{w}_i | i = 1, 2, \dots, m\}$ is the set of generalized eigenvectors of S_B and S_W corresponding to the m largest generalized eigenvalues $\{\lambda_i | i = 1, 2, \dots, m\}$, i.e.,

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \qquad i = 1, 2, \dots, m$$

• The w_i are orthonormal
• There are at most c-1 non-zero generalized Eigenvalues, so m <= c-1
• Can be computed with *eig* in Matlab

Fisherfaces

$$W = W_{fld} W_{PCA}$$

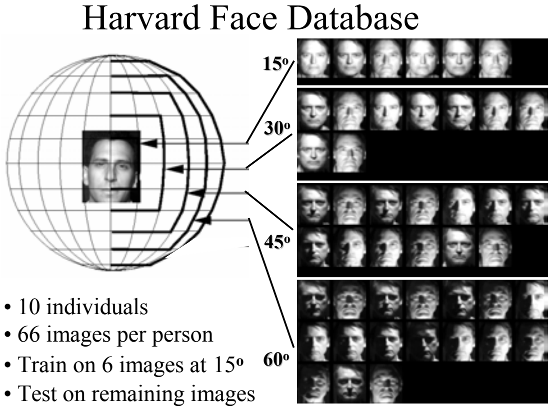
$$W_{PCA} = \arg\max_W |W^T S_T W|$$

$$W_{fld} = \arg\max_W \frac{|W^T W_{PCA}^T S_B W_{PCA} W|}{|W^T W_{PCA}^T S_W W_{PCA} W|}$$

• Since S_W is rank N-c, project training set to subspace spanned by first *N-c* principal components of the training set.
• Apply FLD to *N-c* dimensional subspace yielding *c-1* dimensional feature space.

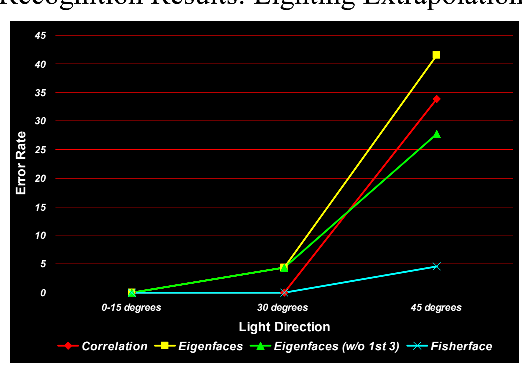
• Fisher's Linear Discriminant projects away the within-class variation (lighting, expressions) found in training set.
• Fisher's Linear Discriminant preserves the separability of the classes.

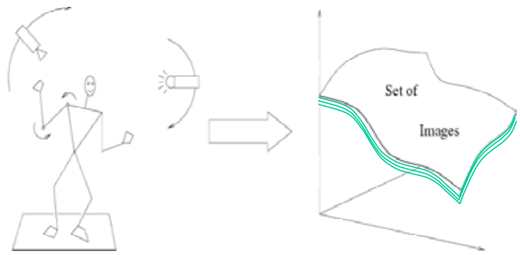
Harvard Face Database



• 10 individuals
• 66 images per person
• Train on 6 images at 15°
• Test on remaining images

Recognition Results: Lighting Extrapolation





Variability: Camera position
Illumination
Internal parameters

Within-class variations

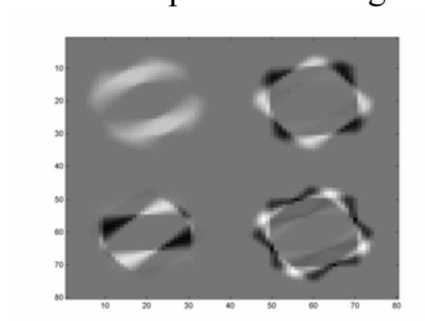
Appearance manifold approach

(Nayar et al. '96)

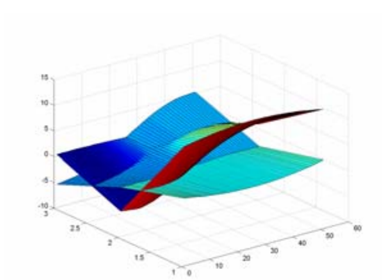
- for every object
  1. sample the set of viewing conditions
  2. Crop & scale images to standard size
  3. Use as feature vector
- apply a PCA over all the images
- keep the dominant PCs
- Set of views for one object is represented as a manifold in the projected space
- Recognition: What is nearest manifold for a given test image?

An example: input images

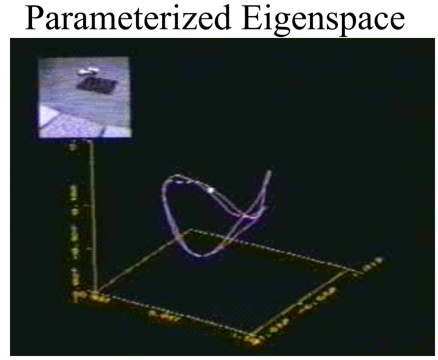
An example: basis images



An example: surfaces of first 3 coefficients



Parameterized Eigenspace



Recognition