Recognition by Appearance

- Recognition by signal matching 2D intensity function,
  compare observed signal to stored signal.
- Matching performed in terms of energy difference:
  minimize $\sum_{all\ pixels}(f(x,y) - m(x,y))^2$.
- Good potential for simple learning — as with nearest neighbors.

- **Small set of basis vectors formed from training samples themselves.**
- **Important References**
  1. See *Eigenfaces* paper by Turk and Pentland (1991)
  2. See industrial application by Murase and Nayar (1994)
  3. See SHOSLIF paper by Swets and Weng (1996)

Good Learning Potential

1. In learning phase system is taught via many samples of what it is to recognize.
2. For example, give the system 3000 labeled 100×100 images of faces to be recognized.
3. Different variations of pose and lighting must be taught and memorized; so, perhaps only 600 persons and 5 variations each.

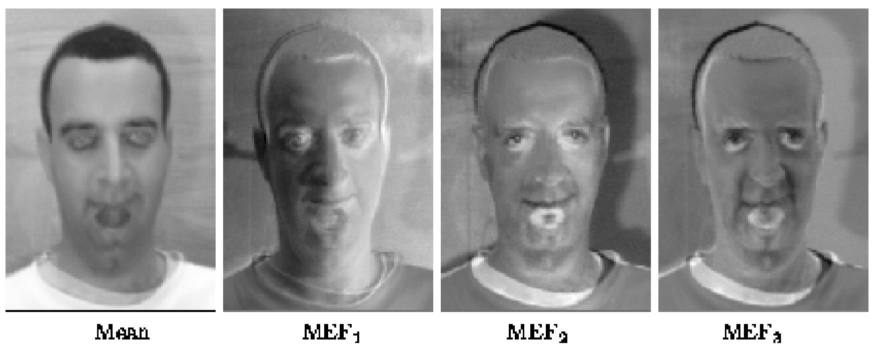
PCA: Principal Components Analysis

Small set of basis vectors formed from training samples themselves.

1. Each training sample transformed into much smaller vector.
2. For faces, it has been reported that perhaps 98% of the energy of all samples can be represented using only 20 basis vectors.
3. Matching can be performed in 20-D Euclidean space rather than say 100×100 -dimensional signal space.
4. Provided that linear combinations of 20 *eigenfaces* can represent every training image to within 98% of its original energy.
5. PCA algorithms compute eigenvectors of the scatter matrix to find the best n such basis vectors.
6. Note that we get tremendous compression using the *cos* transform in JPEG; PCA is similar, big difference is that PCA gets basis from the data itself whereas *cos* transform gets basis from mathematical theory.
7. **If** PCA algorithms can reduce dimensionality from 100×100 to 20, we need only store 20 full 100×100 *eigen images* and 3000 20-D vectors.
8. *This also provides a method of communicating faces over a limited bandwidth channel — only need to send 20 numbers!*

Eigenface idea

Six training images from one of many individuals (Database of images courtesy of Yael Moses).



Average training image and three most significant eigenvectors derived from the scatter matrix (Processed images courtesy of John Weng).

Images of the top row approximated as a linear combination of only the four eigenimages in the figure above. (Processed images courtesy of John Weng.)

Basis Idea Reviewed

Suppose that a set of orthonormal basis images **B** can be found with the following properties.

1. $\mathbf{B} = \{F_1, F_2, \ldots, F_m\}$ with m much smaller than $N = R \times C$.

2. The average quality of representing the image set using this basis is satisfactory in the following sense.
   Over all the M images I_j in the training set, we have

   $I_j^m = a_{j1}F_1 + a_{j2}F2 + \ldots + a_{jm}F_m$ and

   $\sum_{j=1}^{M} (|| \, I_j^m - I_j \, ||^2 \, / \, || \, I_j \, ||)^2 > P\%.$

I_j^m is the approximation of original image I_j using a linear combination of just the m basis images.

Eigenface Algorithm Sketch I

Offline Training Phase:
Input a set I of M labeled training images and produce a basis set B and a vector of coefficents for each image.

$I = \{I_1, I_2, \ldots, I_M\}$ is the set of training images. (input)
$B = \{F_1, F_2, \ldots, F_m\}$ is the set of basis vectors. (output)
$A_j = [a_{j1}, a_{j2}, \ldots, a_{jm}]$ is the vector of coefficients for image I_j. (output)

1. $I_{mean} = mean(I)$.

2. $\Phi = \{\Phi_i | \Phi_i = I_i - I_{mean}\}$, the set of difference images

3. Σ_Φ = the covariance matrix obtained from Φ.

4. Use the principal components method to compute eigenvectors and eigenvalues of Σ_Φ. (see text)

5. Construct the vector **B** as the basis set by selecting the most significant m eigenvectors; start from the largest eigenvalue and continue in decreasing order of the eigenvalues to select the corresponding eigenvectors.

6. Represent each training image I_j by a linear combination of the basis vectors: $I_j^m = a_{j1}F_1 + a_{j2}F_2 + \ldots + a_{jm}F_m$

Eigenface Algorithm Sketch II

Offline Training Phase:

Input a set I of M labeled training images and produce a basis set B and a vector of coeffficents for each image.

$I = \{I_1, I_2, \ldots, I_M\}$ is the set of training images. (input)
$B = \{F_1, F_2, \ldots, F_m\}$ is the set of basis vectors. (output)
$A_j = [a_{j1}, a_{j2}, \ldots, a_{jm}]$ is the vector of coefficients for image I_j. (output)

$\ldots$

Online Recogniton Phase:
Input the set of basis vectors B, the database of coefficient sets $\{A_j\}$, and a test image I_u. Output the class label of I_u.

1. Compute vector of coefficients $A_u = [a_{u1}, a_{u2}, \ldots, a_{um}]$ for I_u;

2. Find the h nearest neighbors of vector A_u in the set $\{A_j\}$;

3. Decide the class of I_u from the labels of the h nearest neighbors (possibly reject in case neighbors are far or inconsistent in labels);

Disadvantages

1. (Advantages already given.)
2. not good for recognition by parts
   —how to recognize an object when occluded by another?
3. not good for high frequency or busy signals
   —imagine alternating B/W pixels; shift image by 1 pixel
4. a lot of samples need to be provided
   —how should we train to recognize **chairs**?
5. conventional PCA algorithms work offline on all the data
   —how do we do incremental updates?