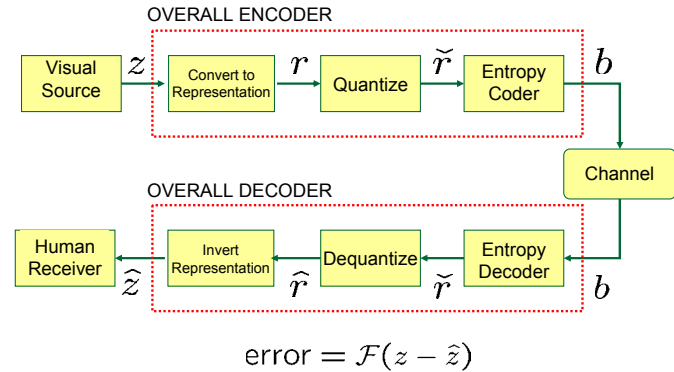


ECE6258 Lecture 20

Unitary Transforms

A General Coding Structure



Unitary Transforms

- Sort samples of $x[n_1, n_2]$ in an $M \times N$ image (or block of an image) into a column vector of length MN .
- Compute transform coefficients

$$y = Ax$$

where A is a matrix of size $MN \times MN$.

- The transform is unitary iff

$$A^{-1} = A^T = A^H$$

- If A is real-valued, i.e., $A = A^*$, the transform is "orthogonal".

Energy conservation with unitary transforms

- For any unitary transform $y = Ax$, we obtain

$$\|y\|^2 = y^H y = x^H A^H A x = \|x\|^2$$

- Interpretation: every unitary transform is simply a rotation of the coordinate system.
- Vector lengths ("energies") are conserved.

Energy distribution for unitary transforms

- Energy is conserved, but often will be unevenly distributed among coefficients.
- Autocorrelation matrix

$$\mathbf{R}_{yy} = E[\mathbf{y}\mathbf{y}^H] = E[\mathbf{A}\mathbf{x}\mathbf{x}^H\mathbf{A}^H] = \mathbf{A}\mathbf{R}_{xx}\mathbf{A}^H$$

- Mean squared values of the coefficients λ_i lie on the diagonal of \mathbf{R}_{yy} .

$$E[\lambda_i^2] = [\mathbf{R}_{yy}]_{i,i} = [\mathbf{A}\mathbf{R}_{xx}\mathbf{A}^H]_{i,i}$$

Eigenmatrix of the autocorrelation matrix

- Definition:** eigenmatrix Φ of autocorrelation matrix \mathbf{R}_{xx} .
 - Φ is unitary.
 - The columns of Φ form an orthonormal set of eigenvectors of \mathbf{R}_{xx} .

$$\mathbf{R}_{xx}\Phi = \Phi\Lambda$$

$$\Lambda = \begin{bmatrix} \lambda_0 & & & 0 \\ & \lambda_1 & & \\ & & \dots & \\ 0 & & & \lambda_{MN-1} \end{bmatrix}$$

diagonal matrix of eigenvalues

- \mathbf{R}_{xx} is symmetric nonnegative definite, hence $\lambda_i \geq 0$ for all i .
- \mathbf{R}_{xx} is a normal matrix, i.e., $\mathbf{R}_{xx}^H\mathbf{R}_{xx} = \mathbf{R}_{xx}\mathbf{R}_{xx}^H$, so a unitary eigenmatrix exists.

Karhunen-Loeve transform

- Unitary transform with matrix

$$\mathbf{A} = \Phi^H$$

where the columns of Φ are ordered according to decreasing eigenvalues.

- Transform coefficients are pairwise uncorrelated

$$\mathbf{R}_{yy} = \mathbf{A}\mathbf{R}_{xx}\mathbf{A}^H = \Phi^H\mathbf{R}_{xx}\Phi = \Phi^H\Phi\Lambda = \Lambda$$

- Energy concentration property:
 - No other unitary transform packs as much energy into the first J coefficients, for all J .
 - Mean squared approximation error by choosing only first J coefficients is minimized.

Basis images and eigenimages

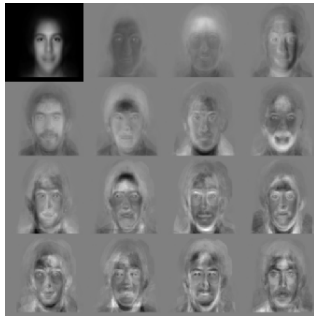
- For any unitary transform, the inverse transform

$$\mathbf{x} = \mathbf{A}^H\mathbf{y}$$

can be interpreted in terms of the superposition of "basis images" of size MN .

- If the transform is a KL transform, the basis images, which are the eigenvectors of the autocorrelation matrix \mathbf{R}_{xx} , are called "eigenimages".
- If the energy concentration property works well, only a limited number of eigenimages is needed to approximate a set of images with small error.

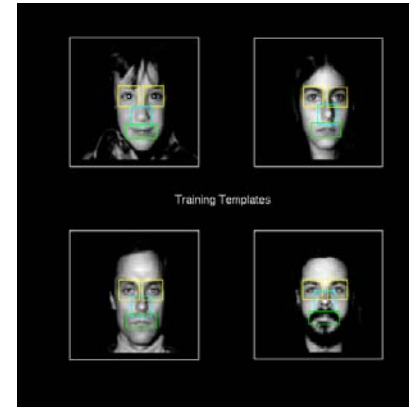
Eigenfaces



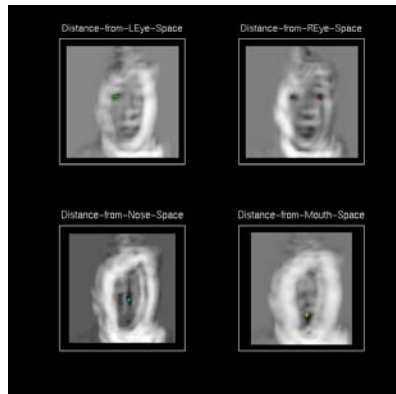
Source: MIT Media Lab

- The first 16 eigenfaces from a training set of 20 faces.
- Can be used for face recognition by nearest neighbor search in 16-d "face space".
- Can be used to generate faces by adjusting 16 coefficients.

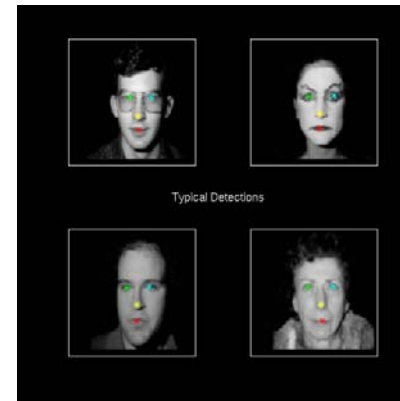
Finding eigeneyes, eigennooses, eigenmouths



Using eigeneyes, eigennooses, eigenmouths



Results



Source: MIT Media Lab

Computing eigenimages from a training set

- How can we work with an $MN \times MN$ autocorrelation matrix?
 - Use training set $\Gamma_1, \Gamma_2, \dots, \Gamma_L$.
 - define training set matrix $\mathbf{S} = [\Gamma_1, \Gamma_2, \dots, \Gamma_L]$.
 - Calculate

$$R_{\Gamma\Gamma} = \frac{1}{L} \sum_{i=1}^L \Gamma_i \Gamma_i^H = \frac{1}{L} \mathbf{S} \mathbf{S}^H$$

Problem 1: Training set size should be $L \gg MN$

If $L < MN$, autocorrelation matrix is rank-deficient.

Problem 2: Finding eigenvectors of an $MN \times MN$ matrix.

- Can we find a small set of the most important eigenimages from a small training set $L \ll MN$?

Sirovich and Kirby method

- Instead of eigenvectors of $\mathbf{S} \mathbf{S}^H$, consider the eigenvectors of $\mathbf{S}^H \mathbf{S}$, i.e.,

$$\mathbf{S}^H \mathbf{S} \mathbf{v}_i = \mu_i \mathbf{v}_i$$

Premultiply both sides by \mathbf{S}

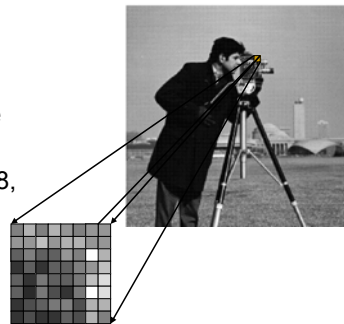
$$\mathbf{S} \mathbf{S}^H \mathbf{S} \mathbf{v}_i = \mu_i \mathbf{S} \mathbf{v}_i$$

By inspection, we find that $\mathbf{S} \mathbf{v}_i$ are eigenvectors of $\mathbf{S} \mathbf{S}^H$.

- For $L \ll MN$ this saves computation, by
 - Computing the $L \times L$ matrix $\mathbf{S}^H \mathbf{S}$.
 - Computing L eigenvectors \mathbf{v}_i of $\mathbf{S}^H \mathbf{S}$.
 - Computing eigenimages corresponding to the $L_0 \leq L$ largest eigenvalues using $\mathbf{S} \mathbf{v}_i$.

Block-wise image processing

- Subdivide image into small blocks
- Process each block independently from the others
- Typical block sizes: 8×8 , 16×16 ,



Separable blockwise transforms

- Image block written as a square matrix

$$\mathbf{X} = \mathbf{A}^T \cdot \mathbf{x} \cdot \mathbf{A}$$

$\begin{matrix} \nearrow & & \nwarrow \\ N \times N & & N \times N \\ \text{coefficients} & & \end{matrix}$

- This can only be done if the transform is separable in n_1 and n_2 .
- Inverse transform

$$\mathbf{x} = \mathbf{A}^* \cdot \mathbf{X} \cdot \mathbf{A}^H$$

Haar transform

- Haar transform matrix for sizes N=2,4,8

$$\mathbf{Hr}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\mathbf{Hr}_4 = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & -\sqrt{2} \end{bmatrix}$$

$$\mathbf{Hr}_8 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & \sqrt{2} & 0 & -2 & 0 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & -2 & 0 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & -2 & 0 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & -2 \end{bmatrix}$$

- Can be computed by taking sums and differences.
- Fast algorithms by recursively applying \mathbf{Hr}_2 .

Hadamard transform

- Transform matrices can be recursively generated

$$\mathbf{Hd}_2 = \mathbf{Hr}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\mathbf{Hd}_4 = \mathbf{Hd}_2 \otimes \mathbf{Hd}_2$$

$$\mathbf{Hd}_8 = \mathbf{Hd}_4 \otimes \mathbf{Hd}_2 = \mathbf{Hd}_2 \otimes \mathbf{Hd}_2 \otimes \mathbf{Hd}_2$$

- Example

$$\mathbf{Hd}_8 = \frac{1}{\sqrt{8}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

Note that Hadamard coefficients need reordering to concentrate energy

Hadamard basis functions

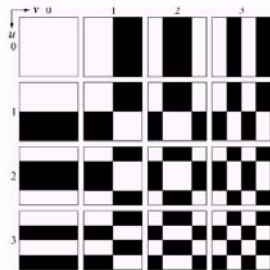


FIGURE 8.29 Walsh-Hadamard basis functions for $N = 4$. The origin of each block is at its top left.

Source: Gonzalez and Woods
Source: Gonzalez and Woods

Discrete Fourier transform

- Only a minor change in the normalization to make the DFT unitary.

$$X[k_1, k_2] = \frac{1}{\sqrt{N}} x[n_1, n_2] e^{-j \frac{2\pi n_1 k_1}{N_1}} e^{-j \frac{2\pi n_2 k_2}{N_2}}$$

Discrete cosine transform (DCT)

$$X_c[k_1, k_2] = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} 4x[n_1, n_2] \cos\left(\frac{\pi(2n_1+1)k_1}{2N_1}\right) \cos\left(\frac{\pi(2n_2+1)k_2}{2N_2}\right)$$

- (not normalized)
- Component of most of the current image and video compression standards
- More about the DCT next time...

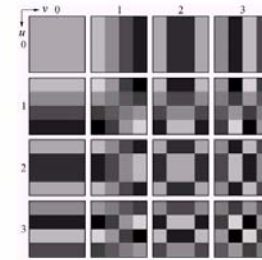
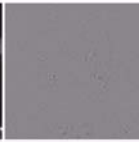


FIGURE 8.30 Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

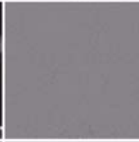
Source: Gonzalez and Woods

Example of 50% coefficient truncation

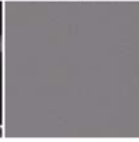
Fourier



Hadamard



cosine



Source: Gonzalez and Woods