Visual Computing:
Unitary transforms

Prof. Marc Pollefeys

ETH

inf Informatik Computer Science

Last week

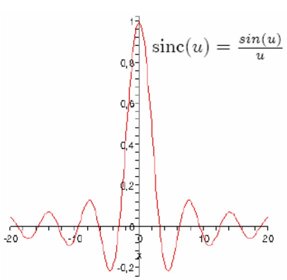
The Convolution Theorem

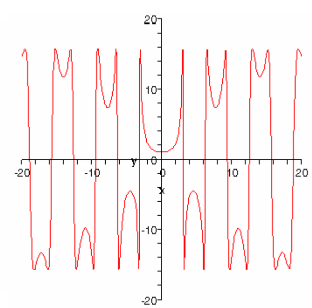
$$F.G = \mathbf{U}\left(f ** g\right)$$ (cfr. filtering)

$$F ** G = \mathbf{U}\left(f . g\right)$$ (cfr. sampling)

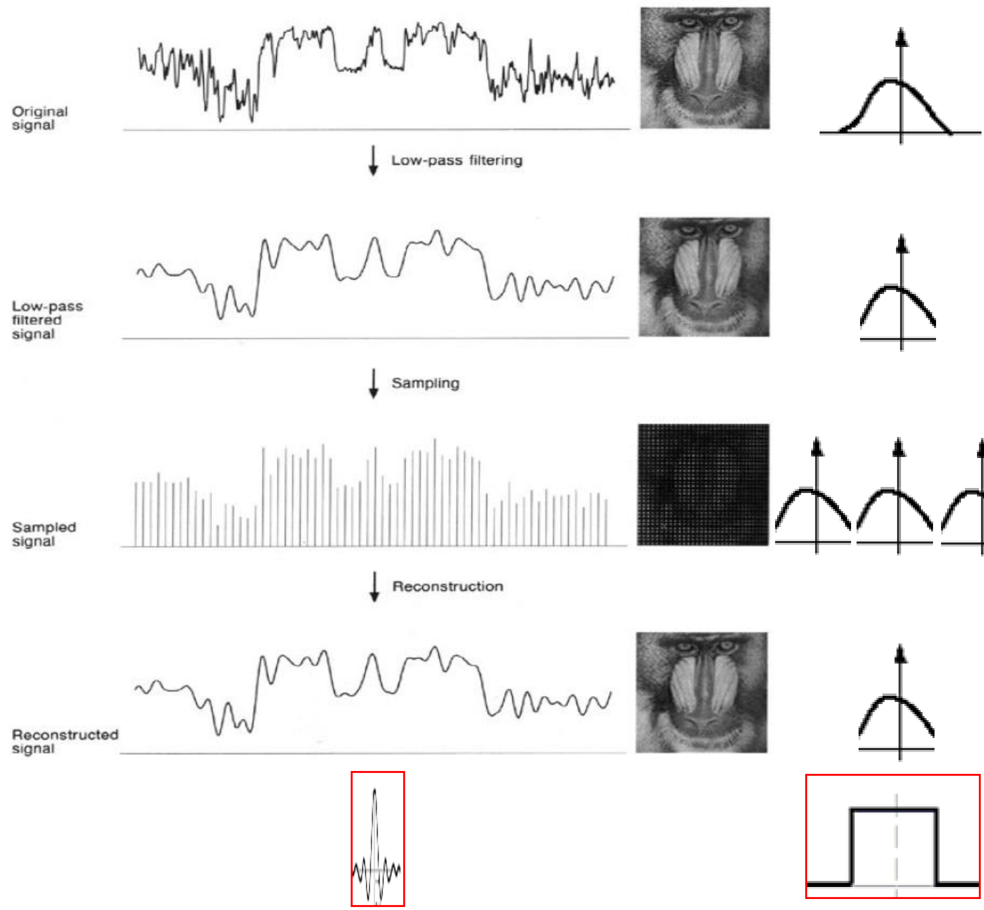
Image restoration

$$f(\mathbf{x}) \longrightarrow \boxed{h(\mathbf{x})} \longrightarrow g(\mathbf{x}) \longrightarrow \boxed{\tilde{h}(\mathbf{x})} \longrightarrow f(\mathbf{x})$$





Digital Processing Pipeline



Visual Computing:
Unitary transforms

Prof. Marc Pollefeys

ETH
inf Informatik Computer Science

A digital image can be written as a matrix

$$
\begin{array}{cc}
 & \xrightarrow{\quad x \quad} \\
\mathbf{f} = & \begin{bmatrix} f(0,0) & f(1,0) & \cdots & f(N-1,0) \\ f(0,1) & f(1,1) & \cdots & f(N-1,1) \\ \vdots & \vdots & & \vdots \\ f(0,L-1) & f(1,L-1) & \cdots & f(N-1,L-1) \end{bmatrix} \Big\downarrow y
\end{array}
$$

- The pixels $f(x,y)$ are sorted into the matrix in „natural" order, with x corresponding to the column and y to the row index. This results in $f(x,y) = f_{yx}$, where f_{yx} denotes an individual element in common matrix notation.
- For a color image, $\mathbf{f}$ might be one of the components.

(many slides from Bern Girod)

A digital image can be written as a vector

$$\vec{f} = \begin{pmatrix} f(0,0) \\ f(1,0) \\ \vdots \\ f(N-1,0) \\ f(0,1) \\ \vdots \\ f(N-1,1) \\ \vdots \\ \vdots \\ f(0,L-1) \\ \vdots \\ f(N-1,L-1) \end{pmatrix} = \begin{pmatrix} f_{00} \\ f_{01} \\ \vdots \\ f_{0,N-1} \\ f_{10} \\ \vdots \\ f_{N-1,1} \\ \vdots \\ \vdots \\ f_{L-1,0} \\ \vdots \\ f_{L-1,N-1} \end{pmatrix}$$

Column vector of length LxN.

This makes the math easier.

Linear Image Processing

- Any linear image processing algorithms can be written as

$$\vec{g} = H\vec{f}$$

Note: matrix H need not be square.

- Definition of a linear operator O[.]

$$O\left[\alpha_1 \cdot \vec{f}_1 + \alpha_2 \cdot \vec{f}_2\right] = \alpha_1 \cdot O\left[\vec{f}_1\right] + \alpha_2 \cdot O\left[\vec{f}_2\right]$$

$$\text{for all scalars } \alpha_1, \alpha_2$$

- Almost all image processing systems contain at least some linear operators.

Linear image processing problems

For the linear image processing system

$$\vec{g} = H\vec{f}$$

how does one choose H . . .

- . . . so $\vec{g}$ separates the salient features from the rest of the image signal.
- . . . so $\vec{g}$ looks better?
- . . . in order for $\vec{g}$ to be sparse?

Unitary transforms

- Sort samples $f(x,y)$ of an MxN image (or a rectangular block in the image) into colunm vector of length MN
- Compute transform coefficients

$$\vec{c} = A\vec{f}$$

where A is a matrix of size MNxMN

- The transform A is unitary, iff

$$A^{-1} = \underbrace{A^{*T} \equiv A^{H}}_{\text{Hermitian conjugate}}$$

- If A is real-valued, i.e., $A=A^*$, transform is „orthonormal“

Energy conservation with unitary transforms

- For any unitary transform $\vec{c} = A\vec{f}$ we obtain

$$\|\vec{c}\|^2 = \vec{c}^H \vec{c} = \vec{f}^H A^H A \vec{f} = \|\vec{f}\|^2$$

- Interpretation: every unitary transform is simply a rotation of the coordinate system (and, possibly, sign flips)
- Vector lengths („energies") are conserved.

Image collection

$$f_i$$

one image

$$F = [f_1 f_2 \cdots f_n]$$

Image collection

$$R_{ff} = E[f_i . f_i^H] = \frac{F.F^H}{n}$$

image collection
auto-correlation function

Energy distribution with unitary transforms

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

$$R_{cc} = E\left[\vec{c}\vec{c}^{\,H}\right] = E\left[A\vec{f} \cdot \vec{f}^{\,H} A^H\right] = A R_{ff} A^H$$

- Mean squared values („average energies") of the coefficients c_i are on the diagonal of R_{cc}

$$E\left[c_i^2\right] = \left[R_{cc}\right]_{i,i} = \left[A R_{ff} A^H\right]_{i,i}$$

ETH

Eigenmatrix of autocorrelation matrix

<u>Definition:</u> eigenmatrix Φ of autocorrelation matrix R_{ff}

- Φ is unitary
- The columns of Φ form a set of eigenvectors of R_{ff}, i.e.,

$$\boxed{R_{ff}\Phi = \Phi\Lambda}$$

Λ is a diagonal matrix of eigenvalues λ_i

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

- R_{ff} is symmetric nonnegative definite, hence $\lambda_i \geq 0$ for all i
- R_{ff} is normal matrix, i.e., $R_{ff}^H R_{ff} = R_{ff} R_{ff}^H$, hence unitary eigenmatrix exists

Karhunen-Loeve Transform

(aka PCA)

- Unitary transform with matrix

$$A = \Phi^H$$

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

- Transform coefficients are pairwise uncorrelated

$$R_{cc} = AR_{ff}A^H = \Phi^H R_{ff} \Phi = \Phi^H \Phi \Lambda = \Lambda$$

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

Optimal energy concentration by KL transform

- To show optimum energy concentration property, consider the truncated coefficient vector

$$\vec{b} = I_J \vec{c}$$

where I_J contain ones on the first J diagonal positions, else zeros.

- Energy in first J coefficients for arbitrary transform A

$$E = Tr(R_{bb}) = Tr(I_J R_{cc} I_J) = Tr(I_J A R_{ff} A^H I_J) = \sum_{k=0}^{J-1} a_k^T R_{ff} a_k^*$$

where a_k^T is the k - th row of A.

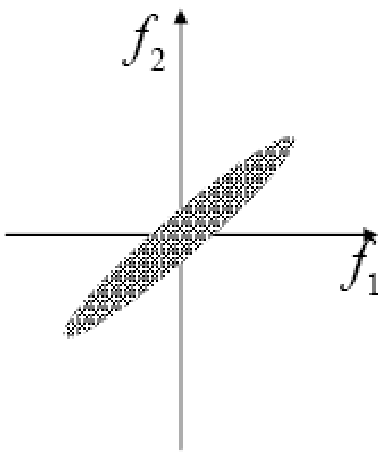
- Lagrangian cost function to enforce unit-length basis vectors

$$L = E + \sum_{k=0}^{J-1} \lambda_k (1 - a_k^T a_k^*) = \sum_{k=0}^{J-1} a_k^T R_{ff} a_k^* + \sum_{k=0}^{J-1} \lambda_k (1 - a_k^T a_k^*)$$

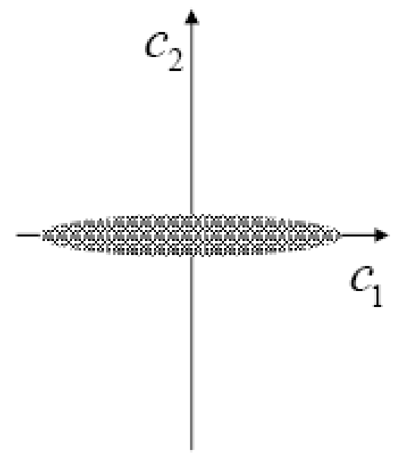
- Differentiating L with respect to a_j yields necessary condition

$$R_{ff} a_j^* = \lambda_j a_j^* \quad \text{for all } j < J$$

Illustration of energy concentration



$$\mathbf{A} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$



Strongly correlated samples,
equal energies

After KLT:
uncorrelated samples,
most of the energy in first coefficient

Basis images and eigenimages

- For any unitary transform, the inverse transform

$$\vec{f} = A^H \vec{c}$$

  can be interpreted in terms of the superposition of „basis images“ (columns of A^H) of size MN.
- If the transform is a KL transform, the basis images, which are the eigenvectors of the autocorrelation matrix R_{ff}, are called „eigenimages.“
- If energy concentration works well, only a limited number of eigenimages is needed to approximate a set of images with small error. These eigenimages form an optimal linear subspace of dimensionality J.

Eigenimages for recognition

- To recognize complex patterns (e.g., faces), large portions of an image (say of size MN) might have to be considered
- High dimensionality of "image space" means high computational burden for many recognition techniques

  Example: nearest-neigbor search requires pairwise comparison with every image in a data base
- Transform $\vec{c} = W\vec{f}$ can reduce dimensionality from MN to J by representing the image by J coefficients
- Idea: tailor a KLT to the specific set of images of the recognition task to preserve the salient features

Simple recognition

- Simple Euclidean distance (SSD) between images
- Best match wins

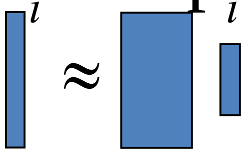
$$\arg\min_i D_i = \left\| \mathrm{I}_i - \mathrm{I} \right\|$$

- Computationally expensive, i.e. requires presented image to be correlated with every image in the database !

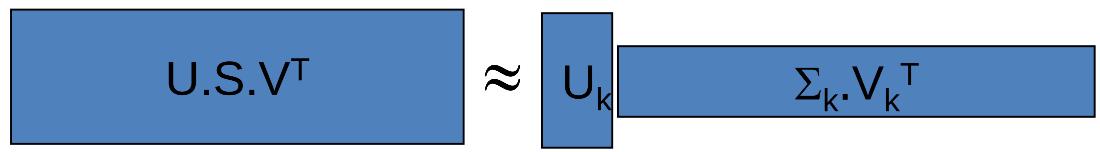
Eigenspace matching

- Consider PCA (aka KLT)

$$\hat{\mathrm{I}}_i \approx \mathbf{E}\,\mathrm{p}_i$$



Closest rank-k approximation property of SVD



- Then,

$$\mathrm{I}_i - \mathrm{I} = \hat{\mathrm{I}}_i - \hat{\mathrm{I}} \approx \mathbf{E}\left(\mathrm{p}_i - \mathrm{p}\right)$$

$$\left\|\mathrm{I}_i - \mathrm{I}\right\| \approx \left\|\mathrm{p}_i - \mathrm{p}\right\|$$

$$\hat{\mathrm{I}} = \mathrm{I} - \overline{\mathrm{I}}$$

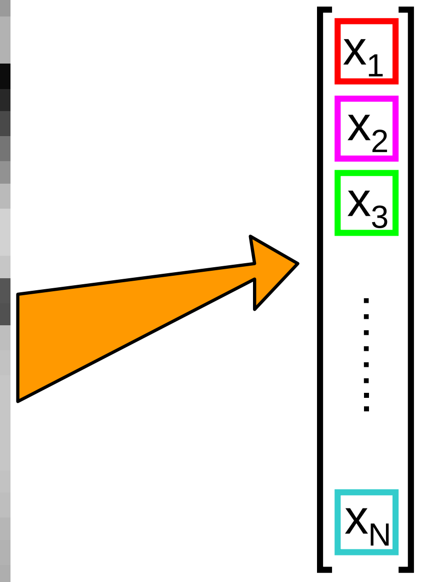
$$\mathrm{p} = \mathbf{E}^T\,\hat{\mathrm{I}}$$

approximate $\arg\min_i D_i = \left\|\mathrm{I}_i - \mathrm{I}\right\|$

with $\arg\min_i \left\|\mathrm{p}_i - \mathrm{p}\right\|$

Much cheaper to compute!

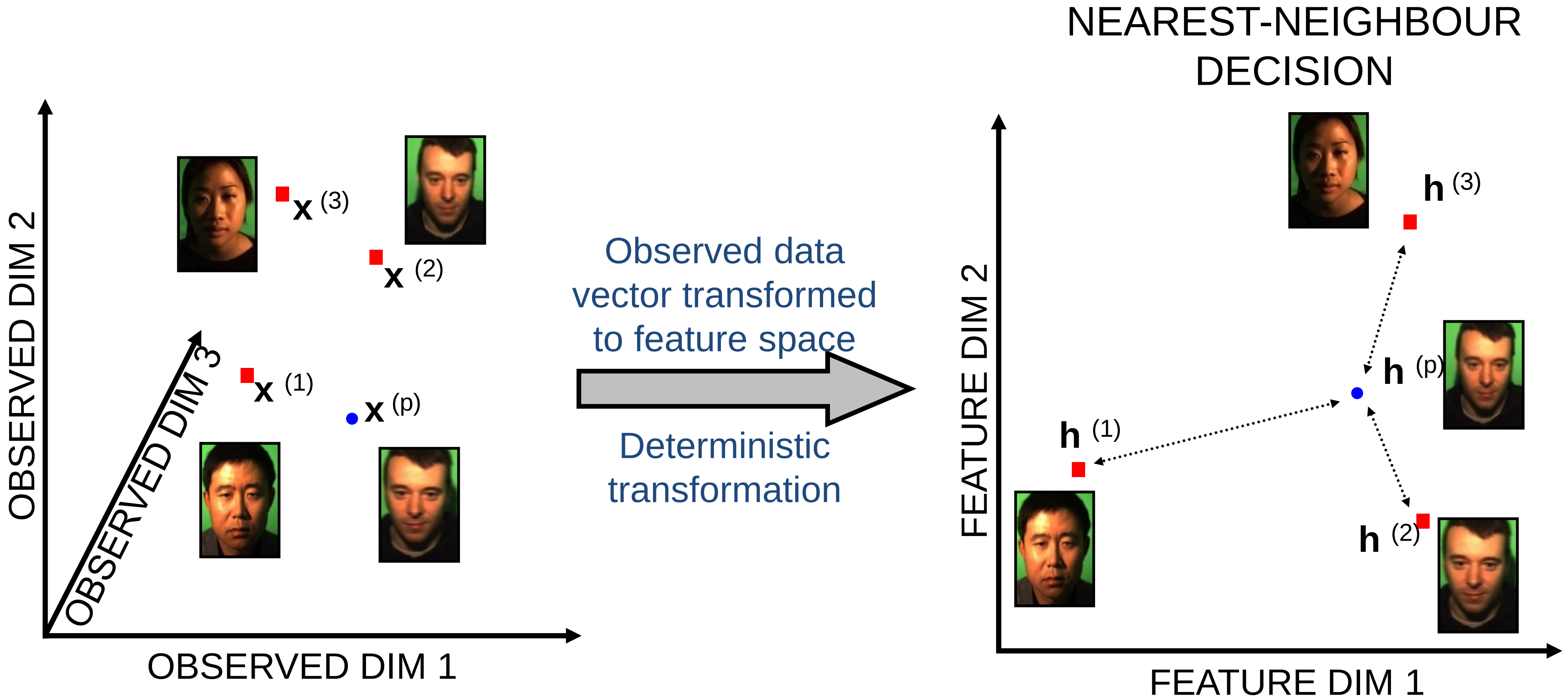
Application to faces



Concatenate face pixels into "observation vector", **x**.

(slide courtesy of Simon Prince)

Distance-Based Methods

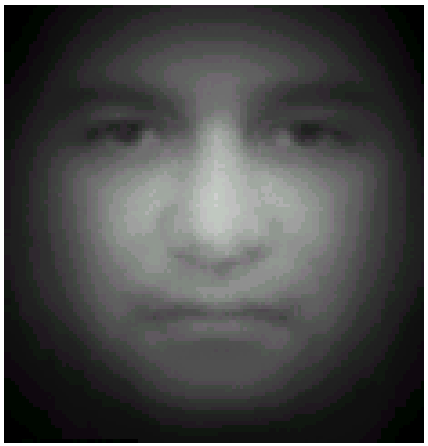


UNDERLIES: Eigenfaces, Fisherfaces, Laplacianfaces, ICA, Kernel PCA etc,
LOGIC: by projecting to a suitable space signal:noise ratio is improved

(slide courtesy of Simon Prince)

Eigenfaces

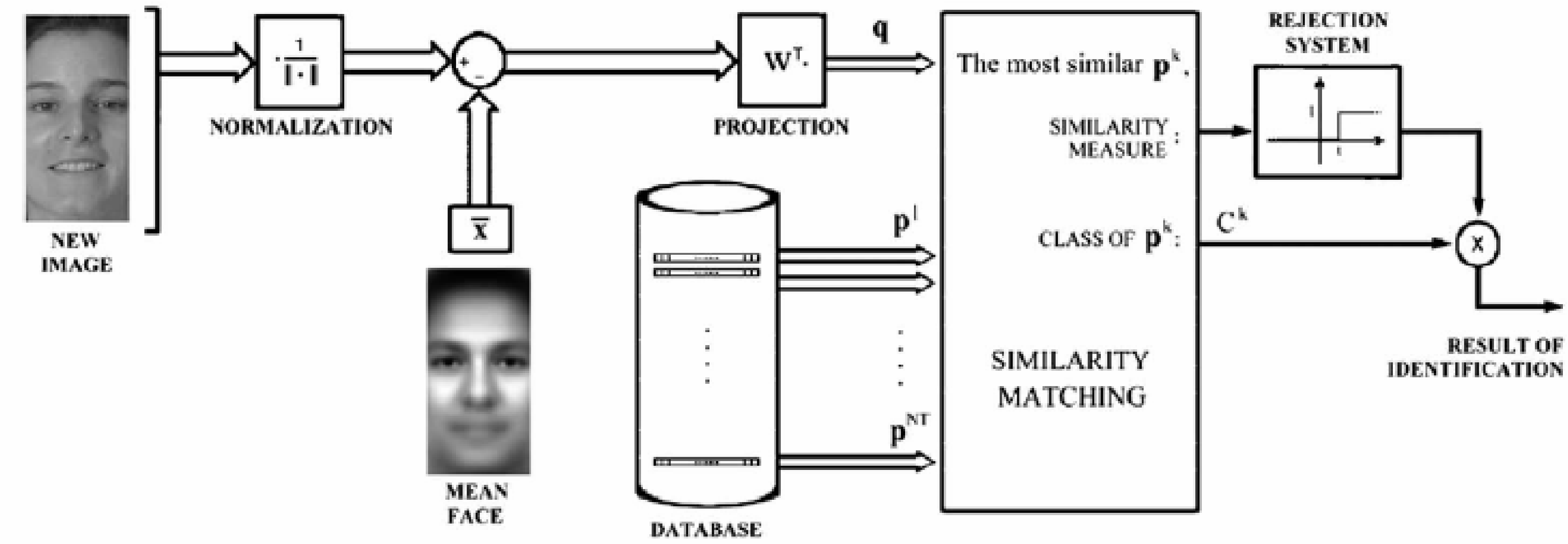
average face



plus a linear
combination of eigenfaces

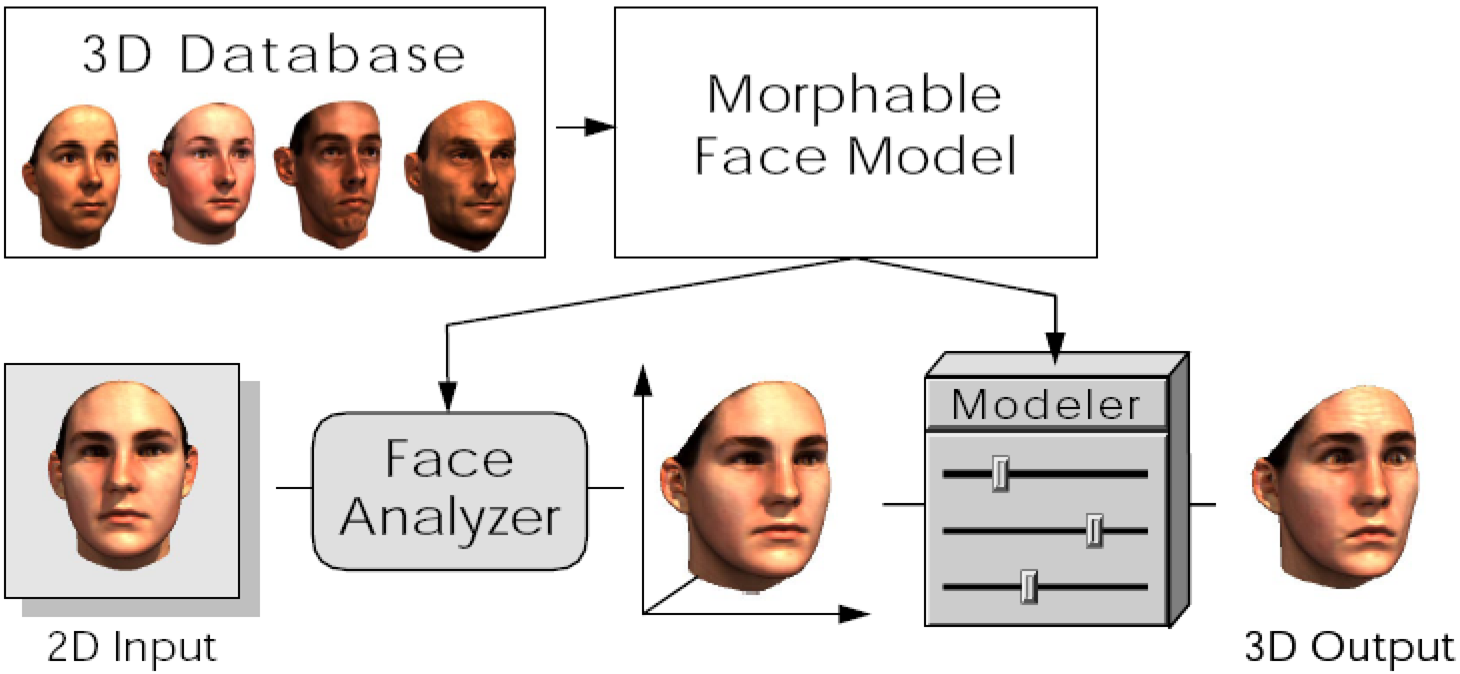
- Can be used for face recognition by nearest neighbor search in 8-d "face space"
- Can be used to generate faces by adjusting 8 coefficients

Eigenimages for recognition (cont.)



[Ruiz-del-Solar and Navarrete, 2005]

3D geometry + appearance

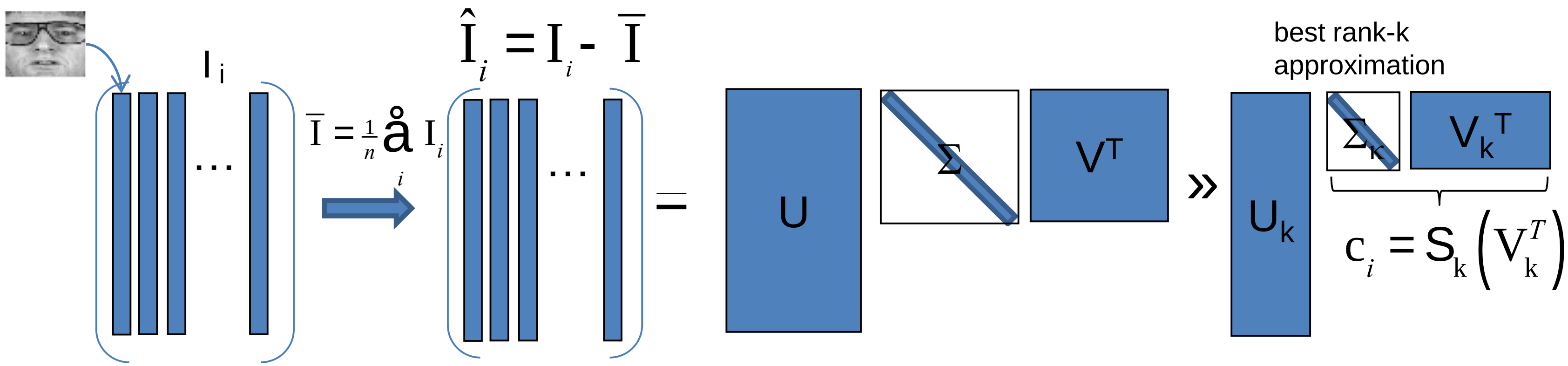


video

[Blanz and Vetter ’99]

Eigenspace: summary

- PCA (or KL transform)



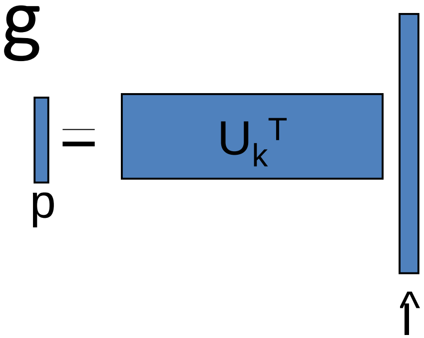
- SSD matching vs. Eigenspace matching

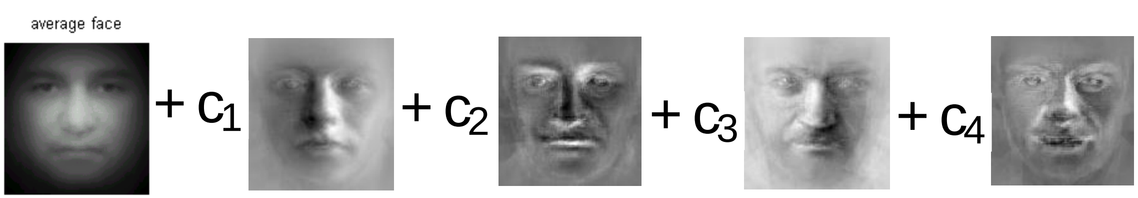
$$I_i - I = \hat{I}_i - \hat{I} \approx U_k\left(c_i - c\right)$$

with

$$c = U_k^T \hat{I}$$

$$c_i = U_k^T \hat{I}_i$$

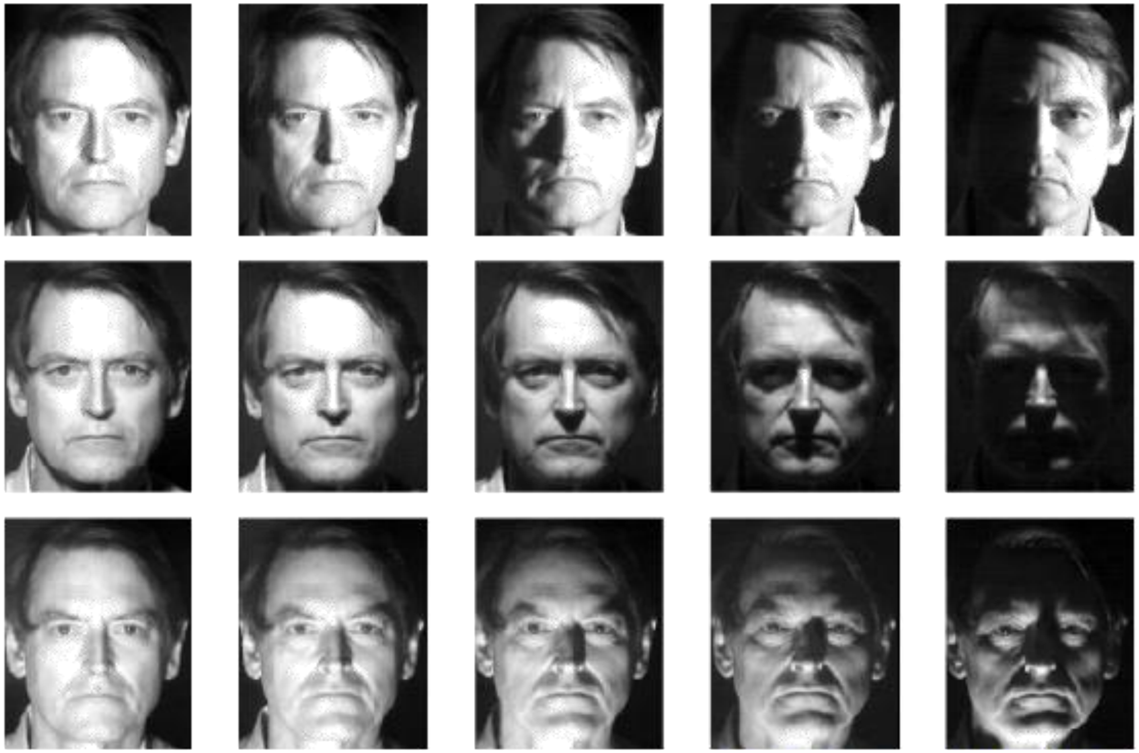


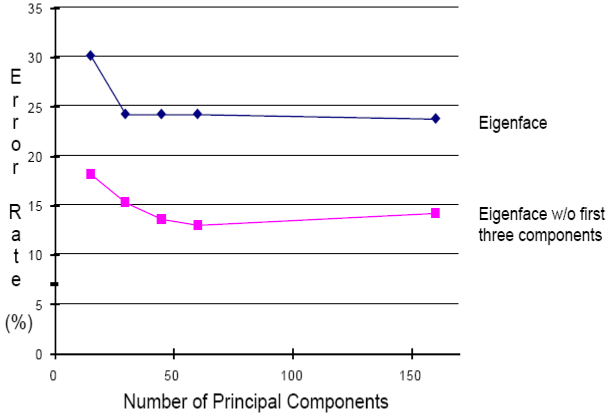


Eigenspace matching will typically work better because only main characteristics are preserved and irrelevant details are discarded

Limitations of Eigenfaces

Differences due to varying illumination can be much larger than differences between faces!

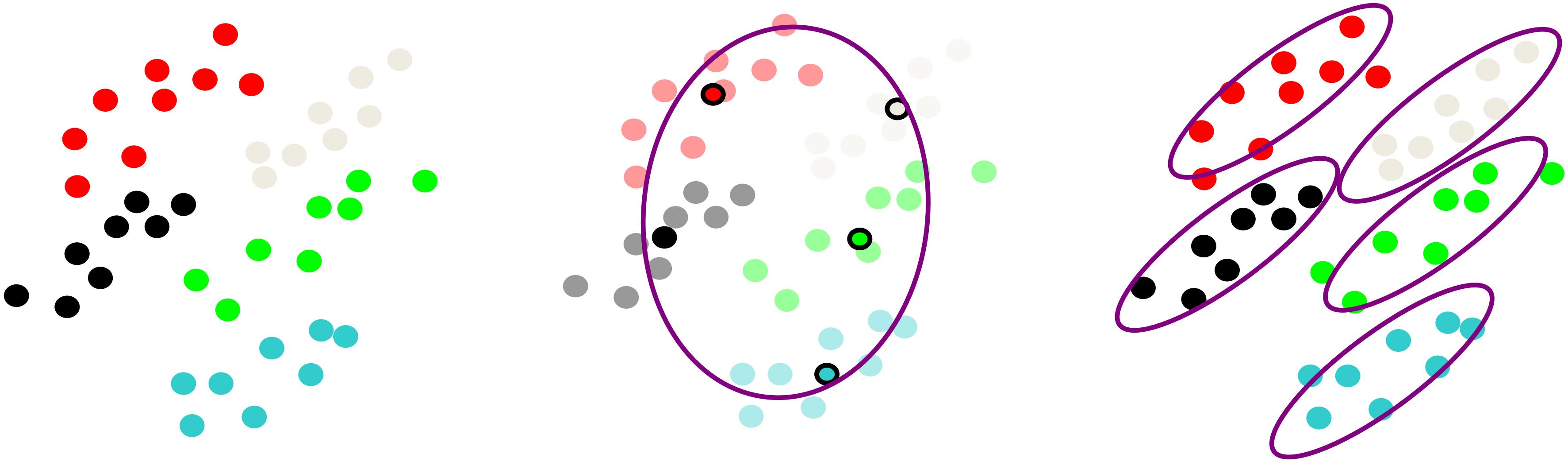




[Belhumeur, Hespanha, Kriegman, 1997]

Fisherfaces / LDA (Belhumeur et al. 1997)

Training Data | Between Individual Variance | Within-Individual Variance



KEY IDEAS:

- Find directions where ratio of between:within individual variance are maximized
- Linearly project to basis where dimension with good signal:noise ratio are maximized

ETH

(slide courtesy of Simon Prince)

Fisher linear discriminant analysis

- Eigenimage method maximizes "scatter" within the linear subspace over the entire image set – regardless of classification task

$$W_{opt} = \arg\max_{W} \left(\det \left(W R W^{H} \right) \right)$$

- Fisher linear discrimant analysis (1936): maximize between-class scatter, while minimizing within-class scatter

$$\boxed{W_{opt} = \arg\max_{W} \left(\frac{\det \left(W R_B W^{H} \right)}{\det \left(W R_W W^{H} \right)} \right)}$$

$$R_B = \sum_{i=1}^{c} N_i \left(\overrightarrow{\mu_i} - \overrightarrow{\mu} \right) \left(\overrightarrow{\mu_i} - \overrightarrow{\mu} \right)^{H}$$

Samples in class i

Mean in class i

$$R_W = \sum_{i=1}^{c} \sum_{\overrightarrow{\Gamma_l} \in Class(i)} \left(\overrightarrow{\Gamma_l} - \overrightarrow{\mu_i} \right) \left(\overrightarrow{\Gamma_l} - \overrightarrow{\mu_i} \right)^{H}$$

ETH

Fisher linear discriminant analysis (cont.)

- Solution: Generalized eigenvectors $\overrightarrow{w_i}$ corresponding to the K largest eigenvalues $\{\lambda_i \mid i = 1, 2, \ldots, K\}$, i.e.

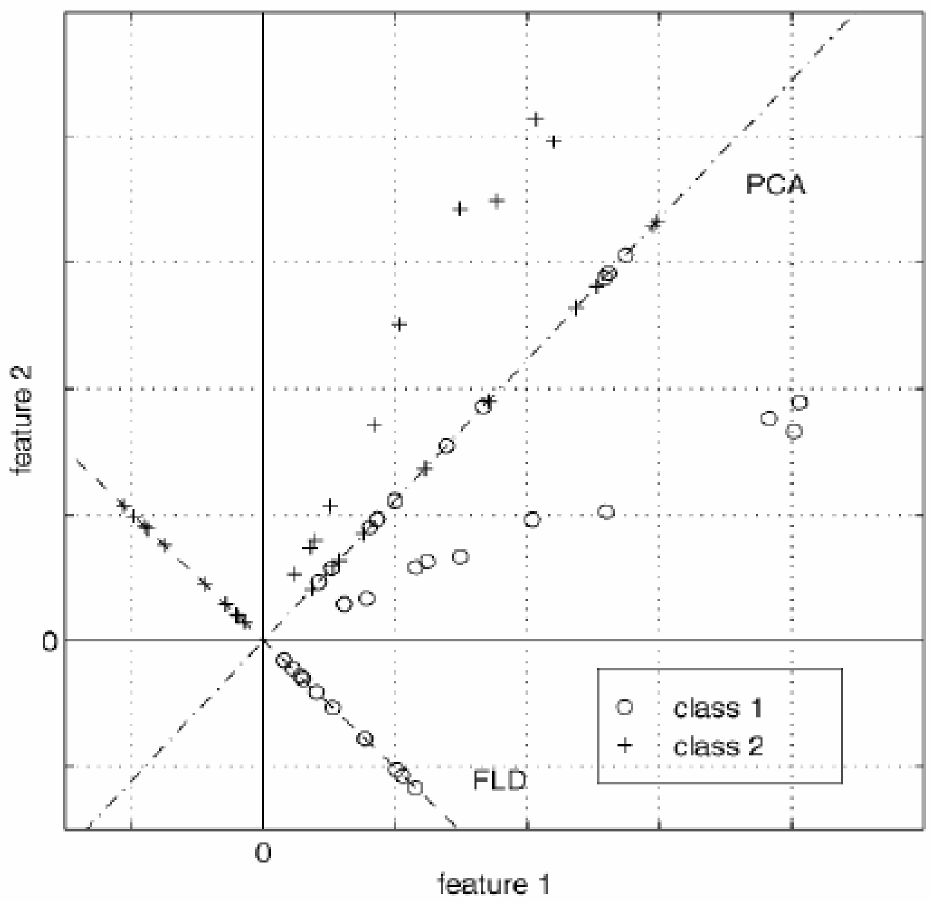
$$\boxed{R_B \overrightarrow{w_i} = \lambda_i R_W \overrightarrow{w_i} \ , \quad i = 1, 2, \ldots, K}$$

- Problem: within-class scatter matrix R_w at most of rank L-c, hence usually singular.
- Apply KLT first to reduce dimension of feature space to L-c (or less), proceed with Fisher LDA in low-dimensional space

Eigenfaces vs. Fisherfaces

<u>2d example:</u>

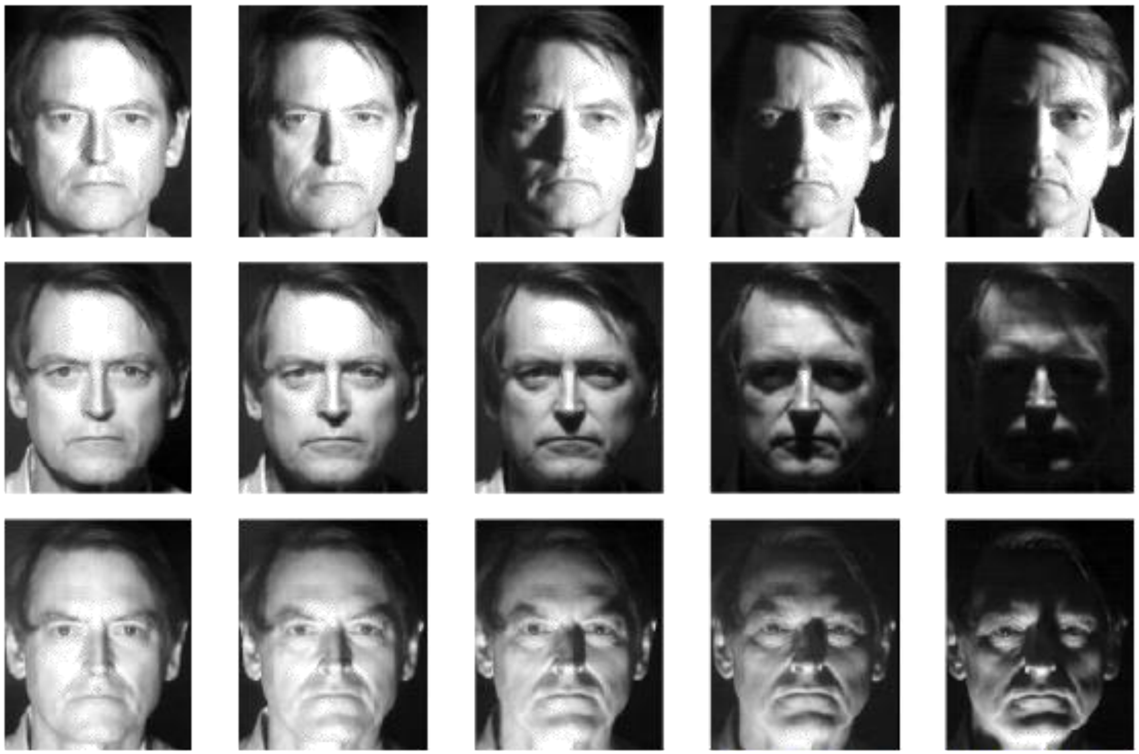
Samples for 2 classes are projected onto 1d subspace using the KLT (aka PCA) or Fisher LDA (FLD). PCA preserves maximum energy, but the 2 classes are no longer distinguishable. FLD separates the classes by choosing a better 1d subspace.

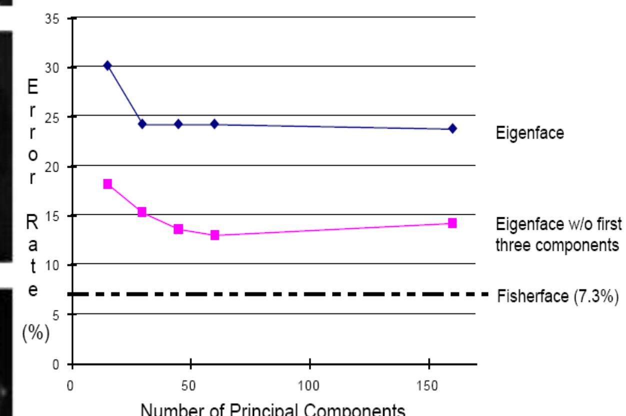


[Belhumeur, Hespanha, Kriegman, 1997]

Eigenfaces vs. Fisherfaces

Differences due to varying illumination can be much larger than differences between faces!



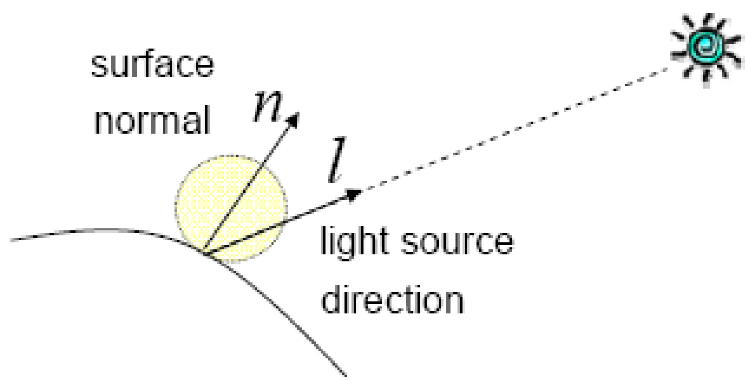


[Belhumeur, Hespanha, Kriegman, 1997]

ETH

Fisher images and varying illumination

- All images of same Lambertian surface with different illumination (without shadows) lie in a 3d linear subspace
- Single point source at infinity



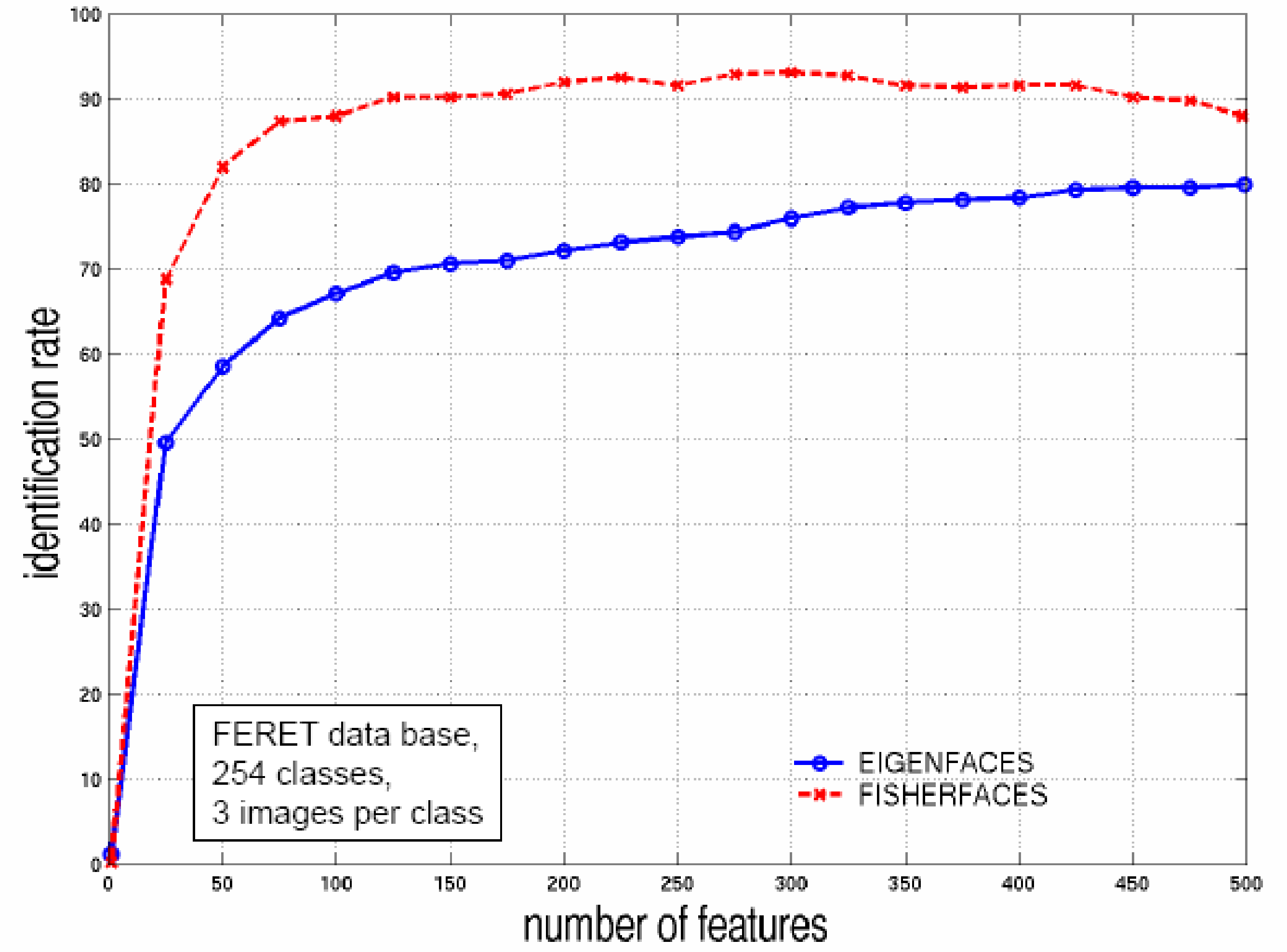
Light source intensity

$$f(x,y) = a(x,y)\left(l^T n(x,y)\right)L$$

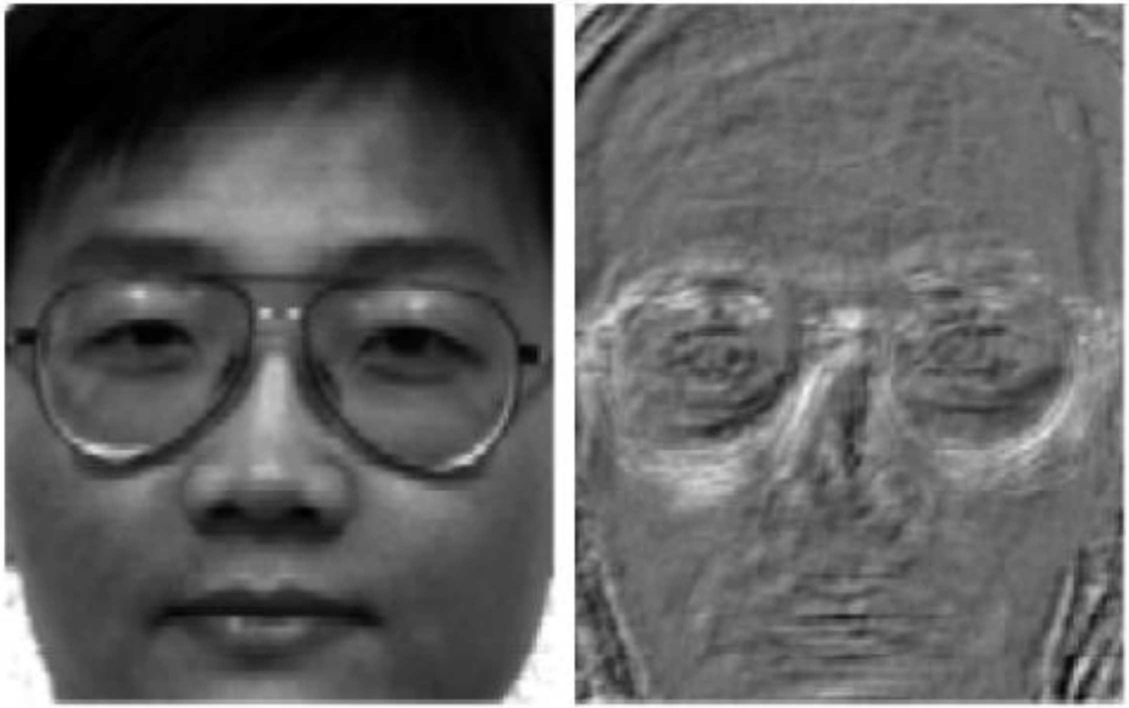
Surface albedo

- Superposition of arbitrary number of point sources at infinity still in same 3d linear subspace, due to linear superposition of each contribution to image
- Fisherimages can eliminate within-class scatter

Face recognition with Eigenfaces and Fisherfaces



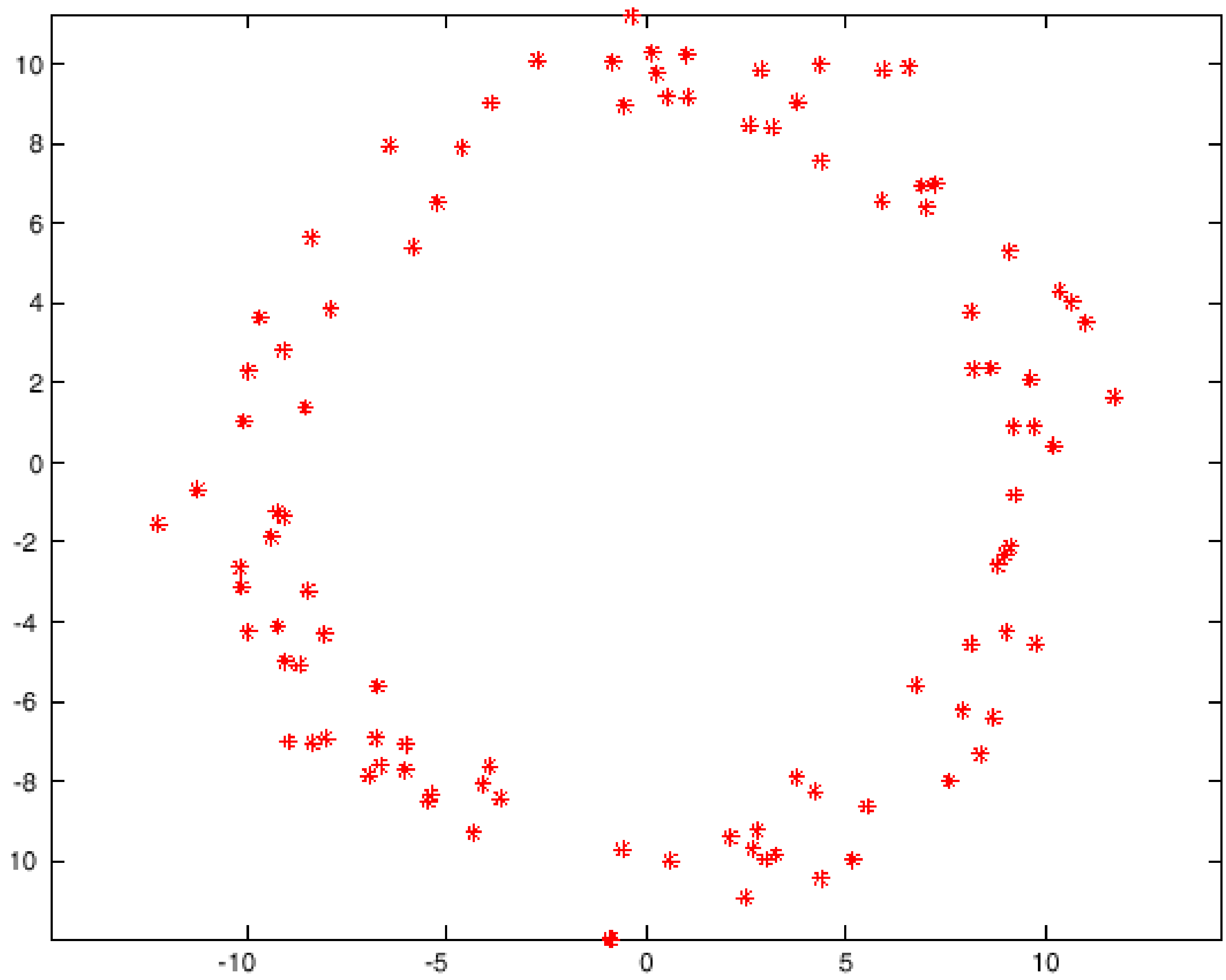
Fisher images trained to recognize glasses



Glasses Recognition		
Method	Reduced Space	Error Rate (%)
Eigenface	10	52.6
Fisherface	1	5.3

ETH

[Belhumeur, Hespanha, Kriegman, 1997]


10
8
6
4
2
0
-2
-4
-6
-8
10
-10
-5
0
5
10

Appearance manifold approach

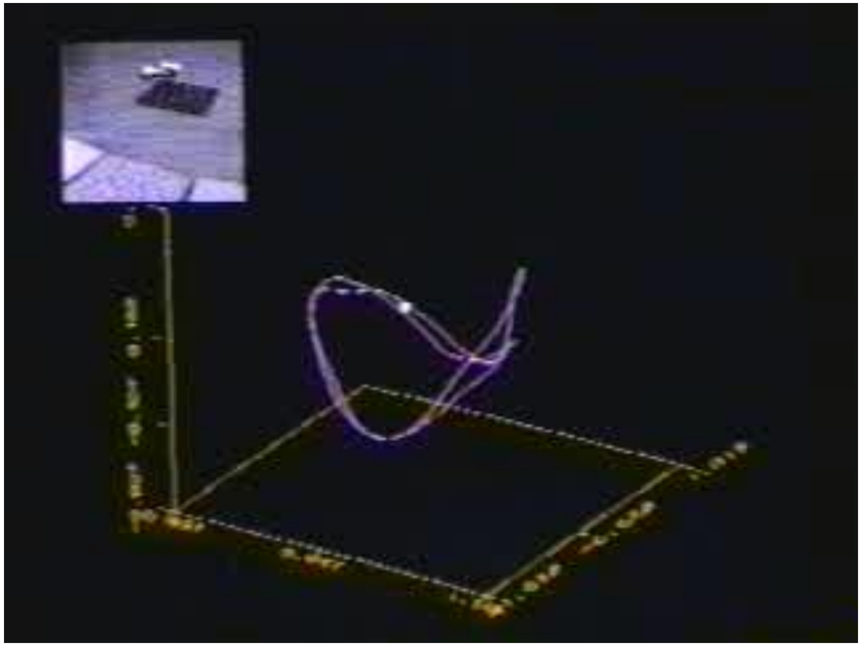
- for every object

  sample the set of viewing conditions
- use these images as feature vectors
- apply a PCA over all the images
- keep the dominant PCs
- sequence of views for 1 object represent a manifold in space of projections
- what is the nearest manifold for a given view?

[Nayar et al. ’96]

Object-pose manifold

- Appearance changes projected on PCs (1D pose changes)
- Sufficient characterization for recognition and pose estimation



[Nayar et al. ’96]

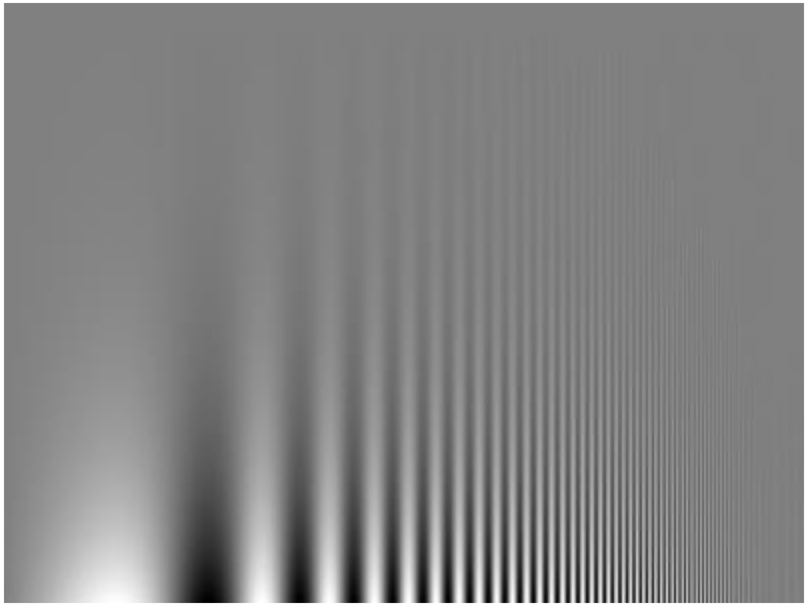
Real-time recognition system

[Nayar et al. ’96]

JPEG image compression

Lenna, 256x256 RGB
Baseline JPEG: 4572 bytes

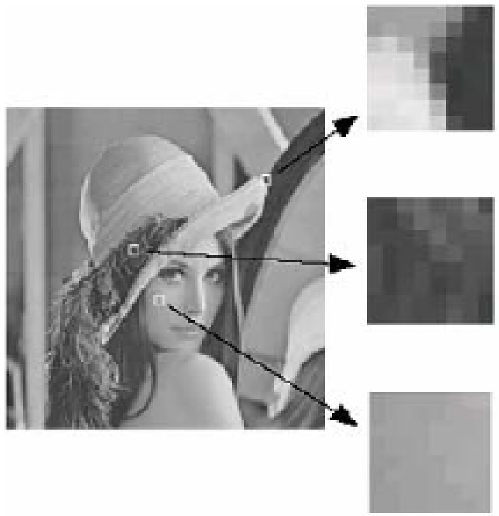
Campbell-Robson contrast sensitivity curve

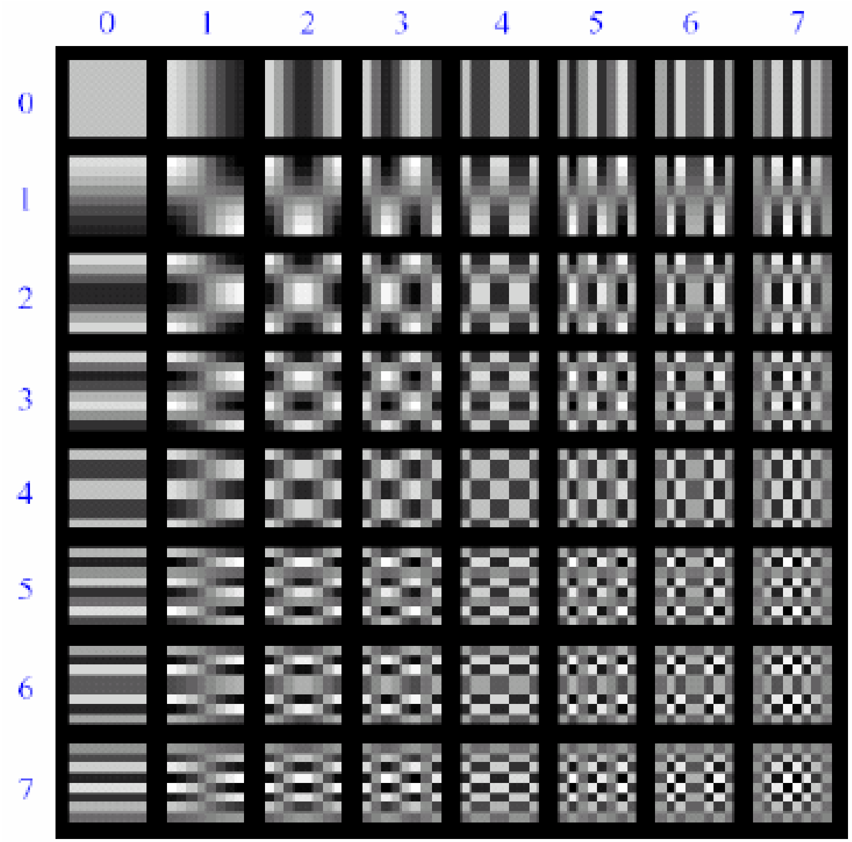


We don't resolve high frequencies too well…

… let's use this to compress images… JPEG!

Lossy Image Compression (JPEG)



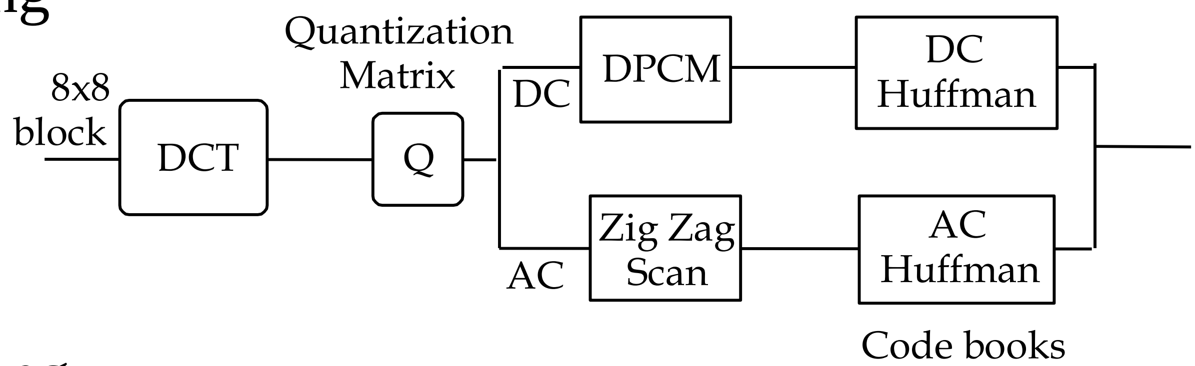


Block-based Discrete Cosine Transform (DCT)

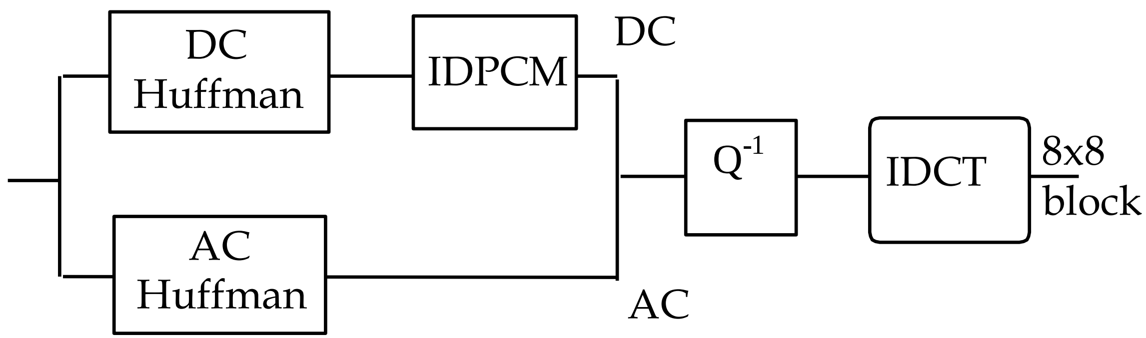
ETH

JPEG Encoding and Decoding

Encoding



Decoding



www.jpeg.org

Using DCT in JPEG

A variant of discrete Fourier transform

- Real numbers
- Fast implementation

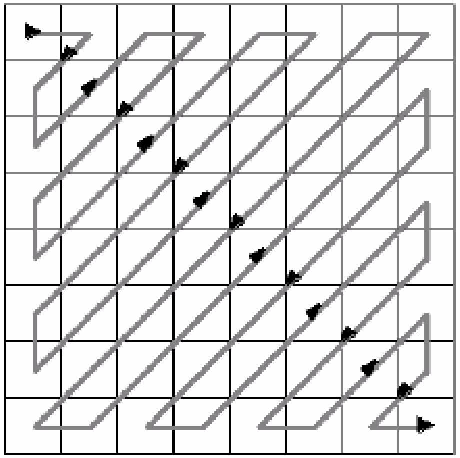
Block size

- small block
  - faster
  - correlation exists between neighboring pixels
- large block
  - better compression in smooth regions

Using DCT in JPEG

The first coefficient $B(0,0)$ is the DC component, the average intensity

The top-left coeffs represent low frequencies, the bottom right – high frequencies



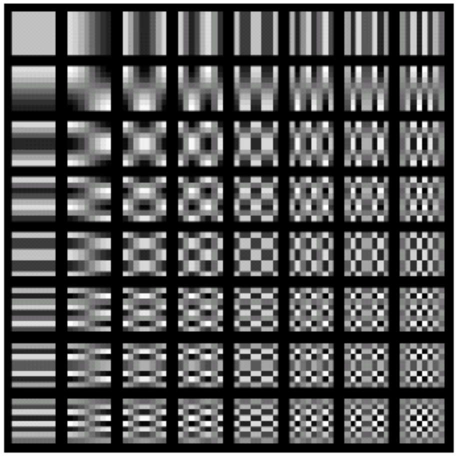
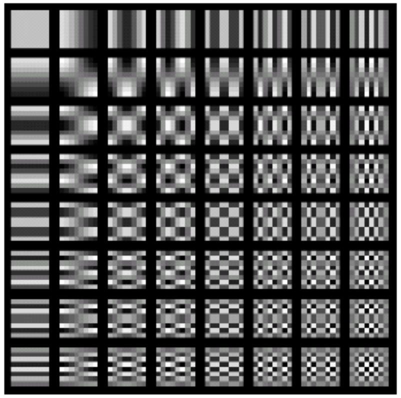


Image compression using DCT

- DCT enables image compression by concentrating most image information in the low frequencies
- Loose unimportant image info (high frequencies) by cutting $B(u,v)$ at bottom right
- The decoder computes the inverse DCT – IDCT

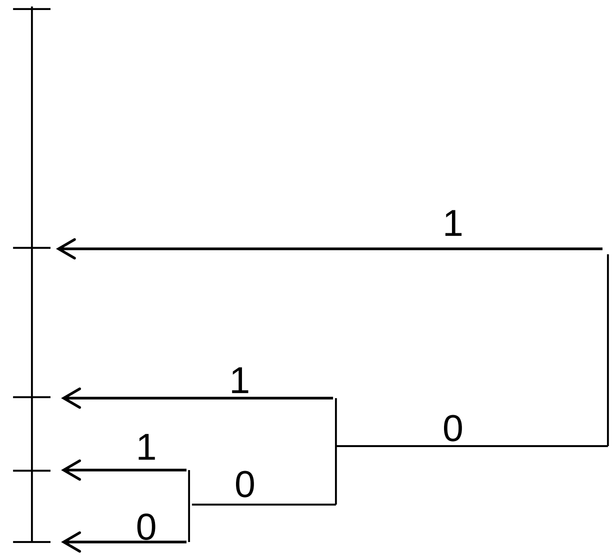
•Quantization Table

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31



Entropy Coding (Huffman code)

Symbol	Prob.	Code	Binary Fraction
Z	0.5	1	0.1
Y	0.25	01	0.01
X	0.125	001	0.001
W	0.125	000	0.000



- The code words, if regarded as a binary fractions, are pointers to the particular interval being coded.
- In Huffman code, the code words point to the base of each interval.
- The average code word length is $H = -\Sigma p(s) \log_2 p(s)$ -> optimal

JPEG compression comparison

89k

12k

Thursday:
Pyramids and wavelets