Các phép biến đổi unitary

Unitary Transforms

Unitary Transforms

- Sort samples *f(x,y)* in 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*** $^{-1}$=***A****, transform is „orthonormal“

Energy conservation with unitary transforms

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

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

- 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

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

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

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

Eigenmatrix of the autocorrelation matrix

Definition: eigenmatrix Φ of autocorrelation matrix R_{ff}

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

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

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

is a diagonal matrix of eigenvalues.

- 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

- 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.

Optimum 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

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

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

- Lagrangian cost function to enforce unit-length basis vectors

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

- Differentiating L with respect to a_j yields neccessary condition

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

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*.

Computing eigenimages from a training set

- How to measure *MNxMN* autocorrelation matrix?
  - ❑ Use training set $\vec{\Gamma}_1, \vec{\Gamma}_2, \ldots, \vec{\Gamma}_L$
  - ❑ Define training set matrix $S = \left(\vec{\Gamma}_1, \vec{\Gamma}_2, \ldots, \vec{\Gamma}_L\right)$ and calculate

$$R_{\Gamma\Gamma} = \frac{1}{L}\sum_{l=1}^{L} \vec{\Gamma}_l \vec{\Gamma}_l^H = \frac{1}{L} SS^H$$

- **Problem 1:** Training set size should be $L >> MN$

  If $L < MN$, autocorrelation matrix R_{ff} is rank - deficient

- **Problem 2:** Finding eigenvectors of an *MNxMN* matrix.

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

Sirovich and Kirby method

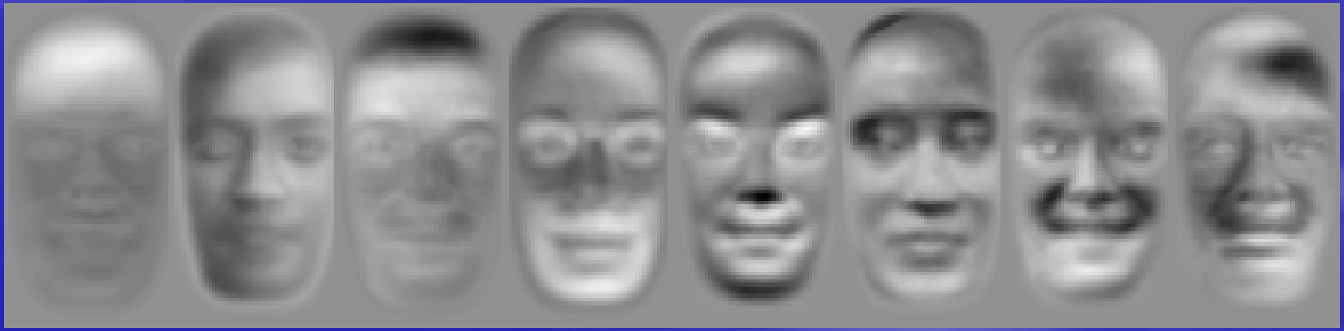
- Instead of eigenvectors of SS^H, consider the eigenvectors of S^HS, i.e.,

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

- Premultiply both sides by S: $SS^H S\vec{v}_i = \mu_i S\vec{v}_i$
- By inspection, we find that $S\vec{v}_i$ are eigenvectors of SS^H
- For this gives rise to great computational savings, by
  - Computing the LxL matrix S^HS
  - Computing L eigenvectors $S\vec{v}_i$ of S^HS
  - Computing eigenimages corresponding to the $L_0 \le L$ largest eigenvalues according as $S\vec{v}_i$

Example: eigenfaces

- The first 8 eigenfaces obtained from a training set of 500 frontal views of human faces.

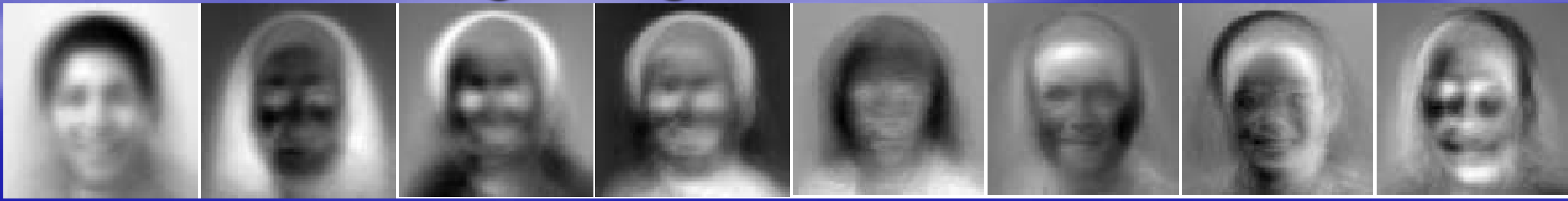


- Can be used for face recognition by nearest neighbor search in 8-d „face space.“
- Can be used to generate faces by adjusting 8 coefficients.

Gender recognition using eigenfaces

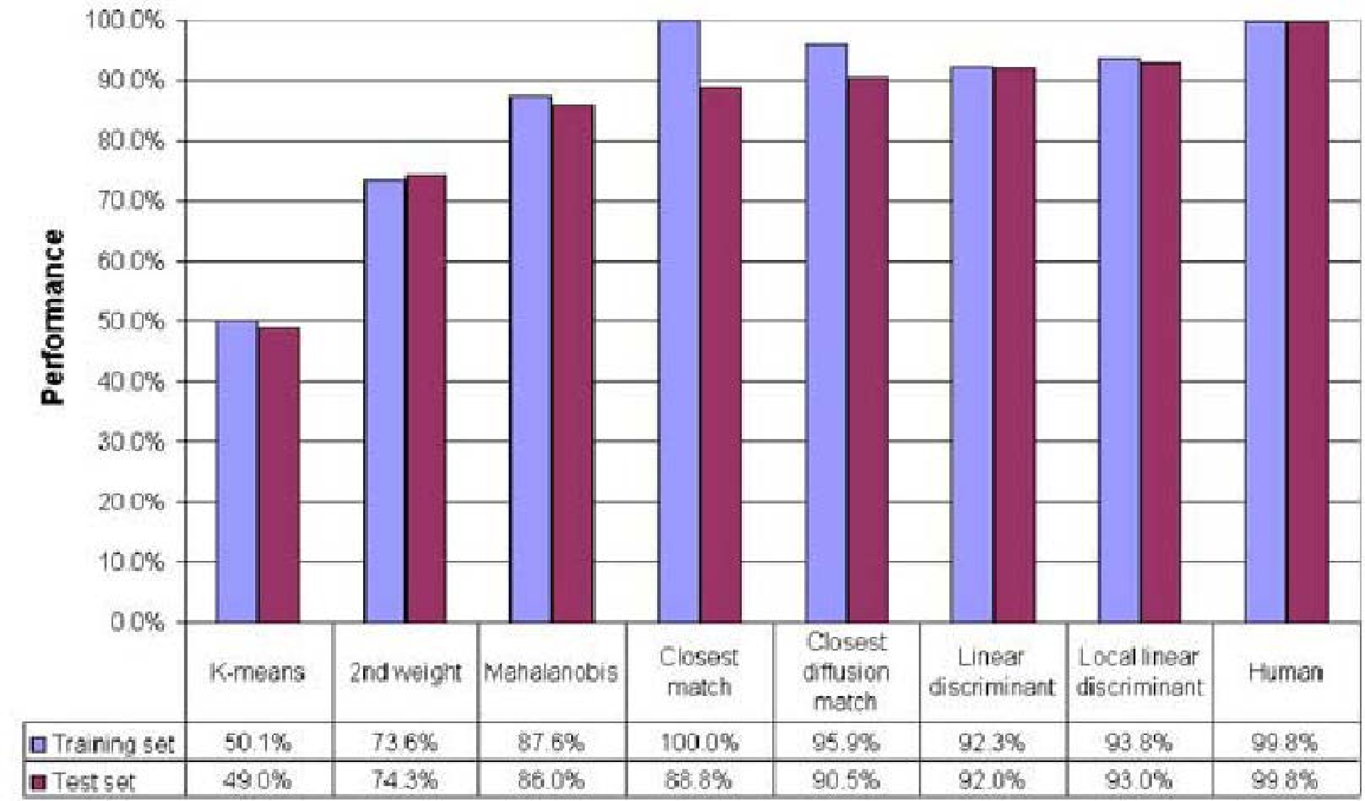
- Task: Male or female?

- Eigenimages from a data base of 20 and 20 female training images



Gender recognition using eigenfaces (cont.)

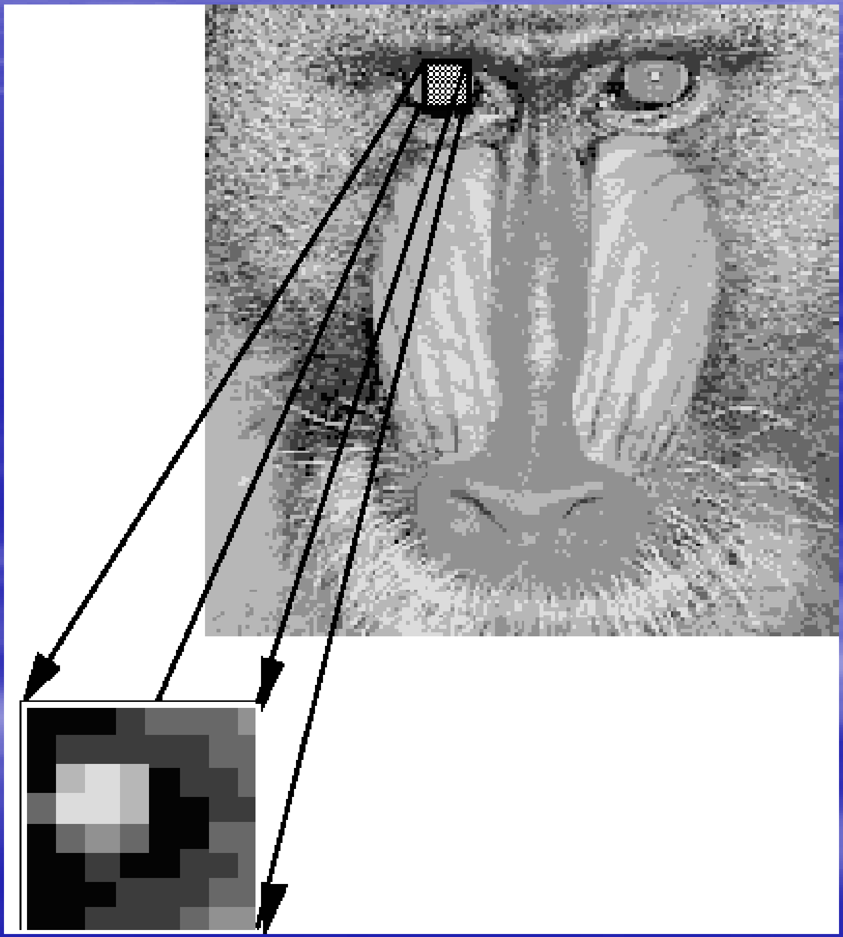
- Recognition accuracy using 8 eigenimages



	K-means	2nd weight	Mahalanobis	Closest match	Closest diffusion match	Linear discriminant	Local linear discriminant	Human
□ Training set	50.1%	73.6%	87.6%	100.0%	95.9%	92.3%	93.8%	99.8%
■ Test set	49.0%	74.3%	86.0%	88.8%	90.5%	92.0%	93.0%	99.8%

Block-wise image processing

- Subdivide image into small blocks
- Process each block independently from the others
- Typical blocksizes: 8x8, 16x16



Separable blockwise transforms

- Image block written as a square matrix **f**

$$\mathbf{c} = \mathbf{A}^T.\mathbf{f}.\mathbf{A}$$

(*NxN* coefficients)

- This can only be done, if transform is separable in *x* and *y*, i.e.,

(*N*^{2}x*N*^{2})

$$\mathbf{A}^T = \mathbf{A} \otimes \mathbf{A}$$

- Inverse transform

$$\mathbf{f} = \mathbf{A}^*.\mathbf{c}.\mathbf{A}^H$$

Haar transform

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

$$Hr_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

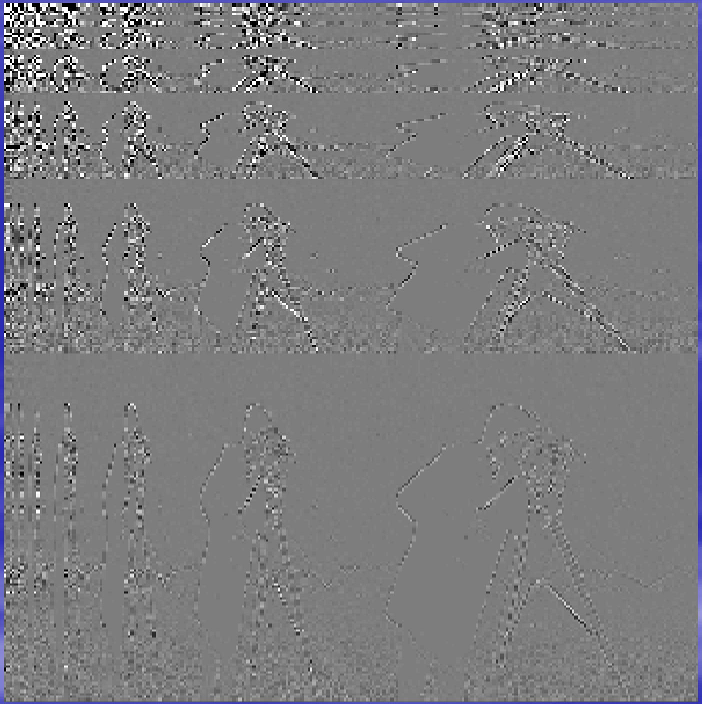
$$Hr_4 = \frac{1}{\sqrt{4}}\begin{pmatrix} 1 & 1 & \sqrt{2} & 0 \\ 1 & 1 & -\sqrt{2} & 0 \\ 1 & -1 & 0 & \sqrt{2} \\ 1 & -1 & 0 & -\sqrt{2} \end{pmatrix}$$

$$Hr_8 = \frac{1}{\sqrt{8}}\begin{pmatrix} 1 & 1 & \sqrt{2} & 0 & 2 & 0 & 0 & 0 \\ 1 & 1 & \sqrt{2} & 0 & -2 & 0 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & 2 & 0 & 0 \\ 1 & 1 & -\sqrt{2} & 0 & 0 & -2 & 0 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & 2 & 0 \\ 1 & -1 & 0 & \sqrt{2} & 0 & 0 & -2 & 0 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & -\sqrt{2} & 0 & 0 & 0 & -2 \end{pmatrix}$$

- Can be computed by taking sums and differences
- Fast algorithms by recursively applying **Hr$_2$**

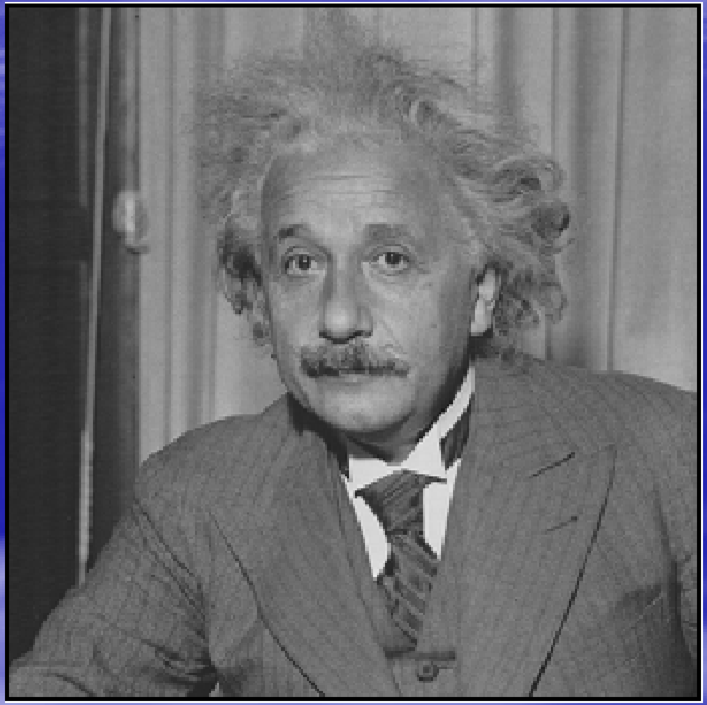
Haar transform example

Original *Cameraman 256x256*

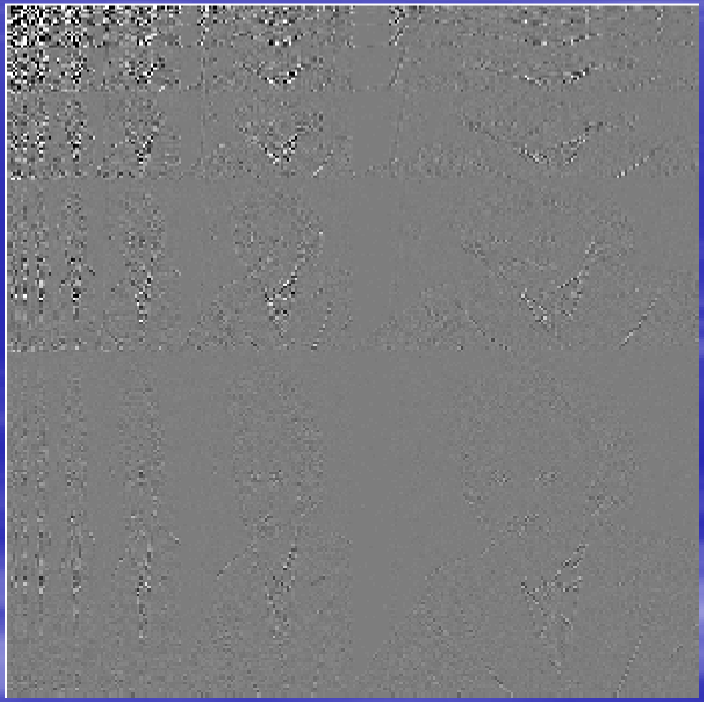


256x256 Haar transform of Cameraman

Haar transform example



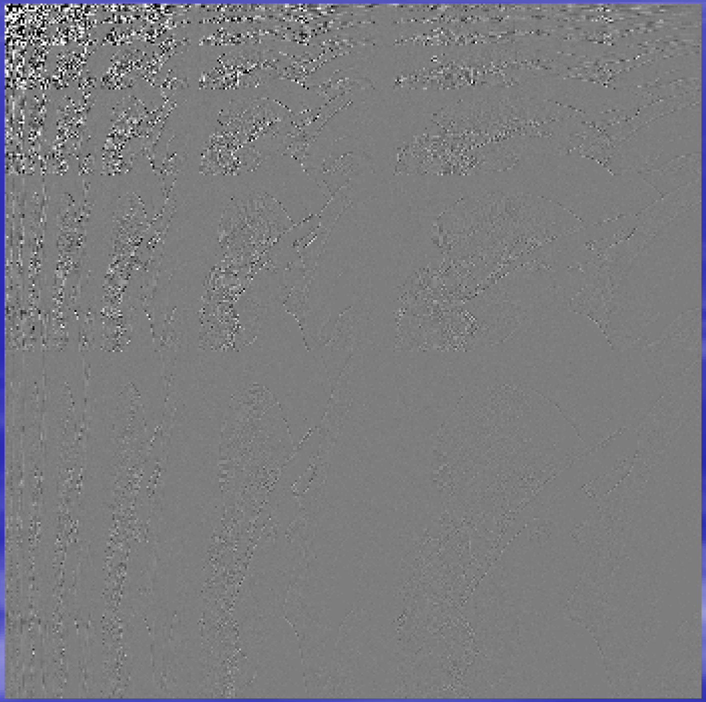
Original *Einstein*
256x256



256x256 Haar transform
of *Einstein*

Haar transform example

Original *Lena*
512x512



512x512 Haar transform
of *Lena*

Hadamard transform

- Transform matrices can be recursively generated

$$Hd_2 = Hr_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$Hd_4 = Hd_2 \otimes Hd_2$$

$$Hd_8 = Hd_4 \otimes Hd_2 = Hd_2 \otimes Hd_2 \otimes Hd_2$$

$$\ldots$$

- Example

$$Hd_8 = \frac{1}{\sqrt{8}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}$$

Note that Hadamard Coefficients need reordering to concentrate energy

Discrete Fourier transform

- For DFT of order *N*, define $W_N = \exp\left\{\frac{-j2\pi}{N}\right\}$
- Transform matrix

$$\mathrm{DTF}_N = \frac{1}{\sqrt{N}}\begin{pmatrix} W_N^0 & W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & W_N^2 & \cdots & W_N^{N-1} \\ W_N^0 & W_N^2 & & \cdots & W_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ W_N^0 & W_N^{N-1} & W_N^{2(N-1)} & \cdots & W_N^{(N-1)(N-1)} \end{pmatrix}$$

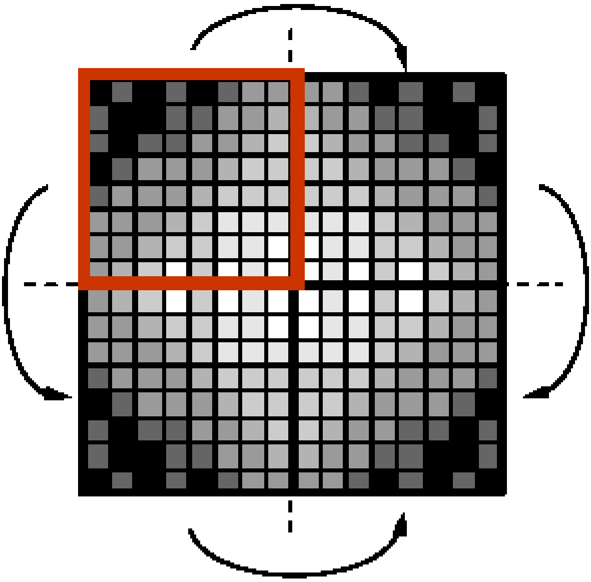
- Definition for general *N*, fast algorithms for $N=2^m$
- DFT coefficients are complex (even if input image is real)
- Inverse transform $\mathbf{DFT_N^{-1} = DTF^H = DTF^*}$

Discrete cosine transform

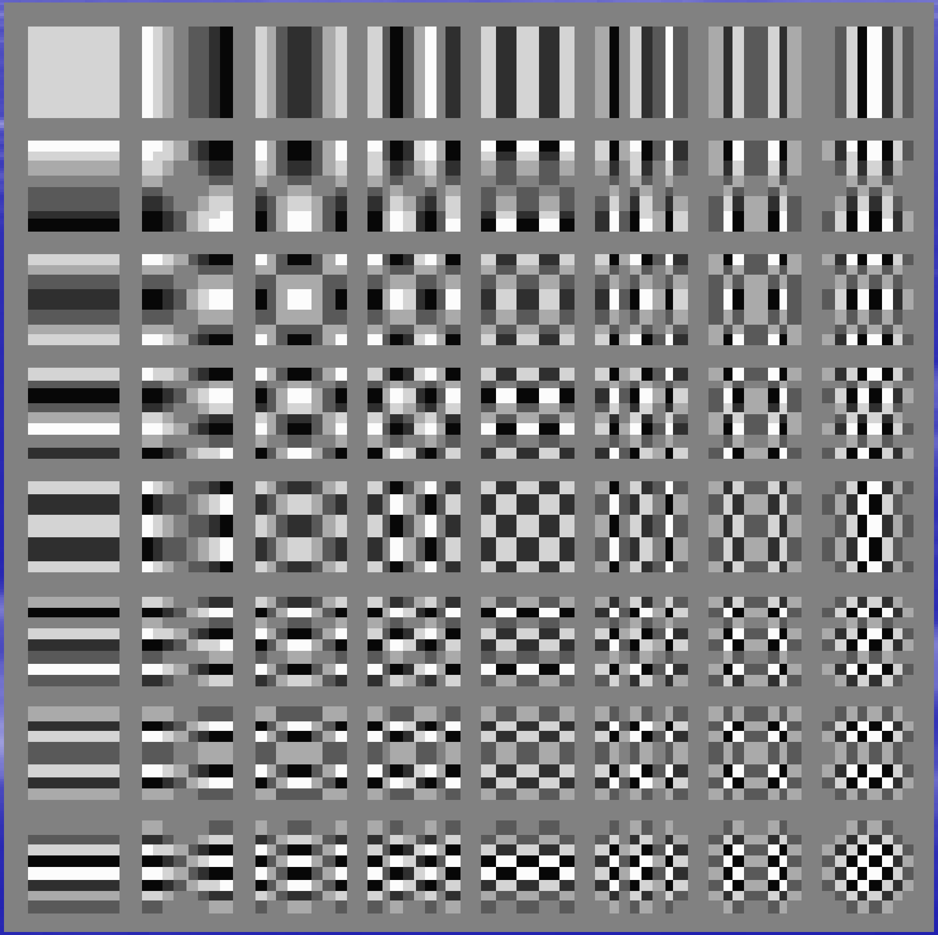
- Transform matrix

$$DCT_N = \left\{ a_{k,n} \,\middle|\, a_{k,n} = \begin{cases} \frac{1}{N} & 0 \le n \le N-1 \wedge k = 0 \\ \sqrt{\frac{2}{N}} \cos\left(\frac{\pi(2n+1)k}{2N}\right) & 0 \le n \le N-1 \wedge 1 \le k \le N-1 \end{cases} \right\}$$

- Can be interpreted as DFT of a mirror-extended image block (shown here for 2-d DCT)
- Transform is used in many coding standards (JPEG, MPEG)

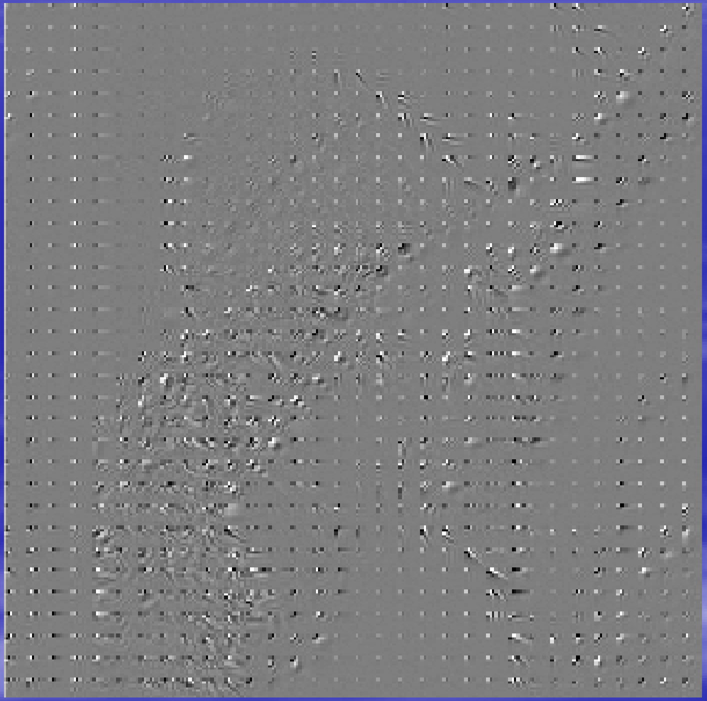


Basis images of an 8x8 DCT



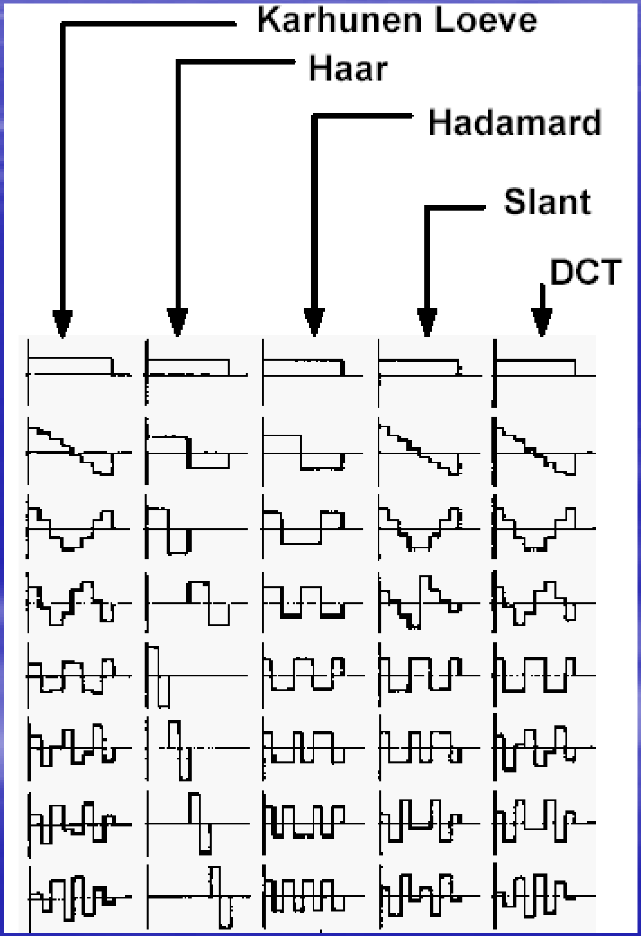
Blockwise DCT example

Original *Lena*
256x256



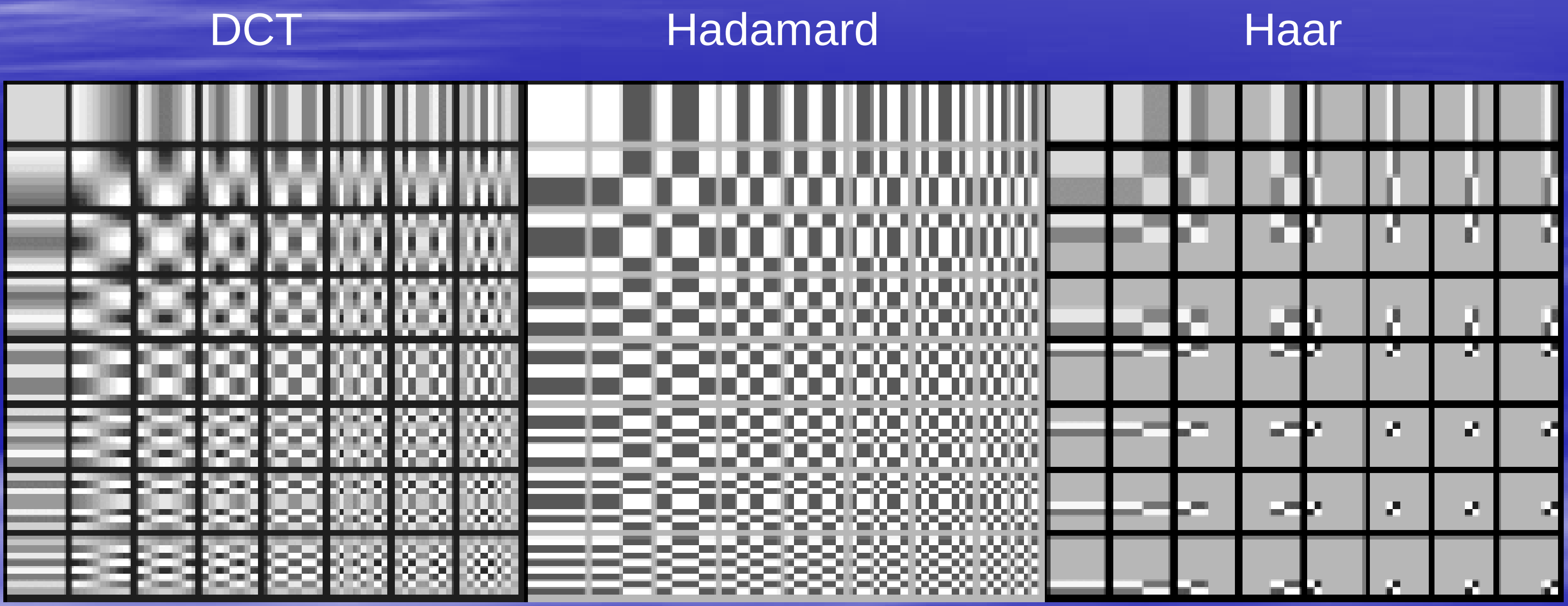
Blockwise 8x8 DCT
of *Lena*

Comparison of block transforms



Comparison of 1-D basis functions for block size N=8

Comparison of block transforms (cont.)

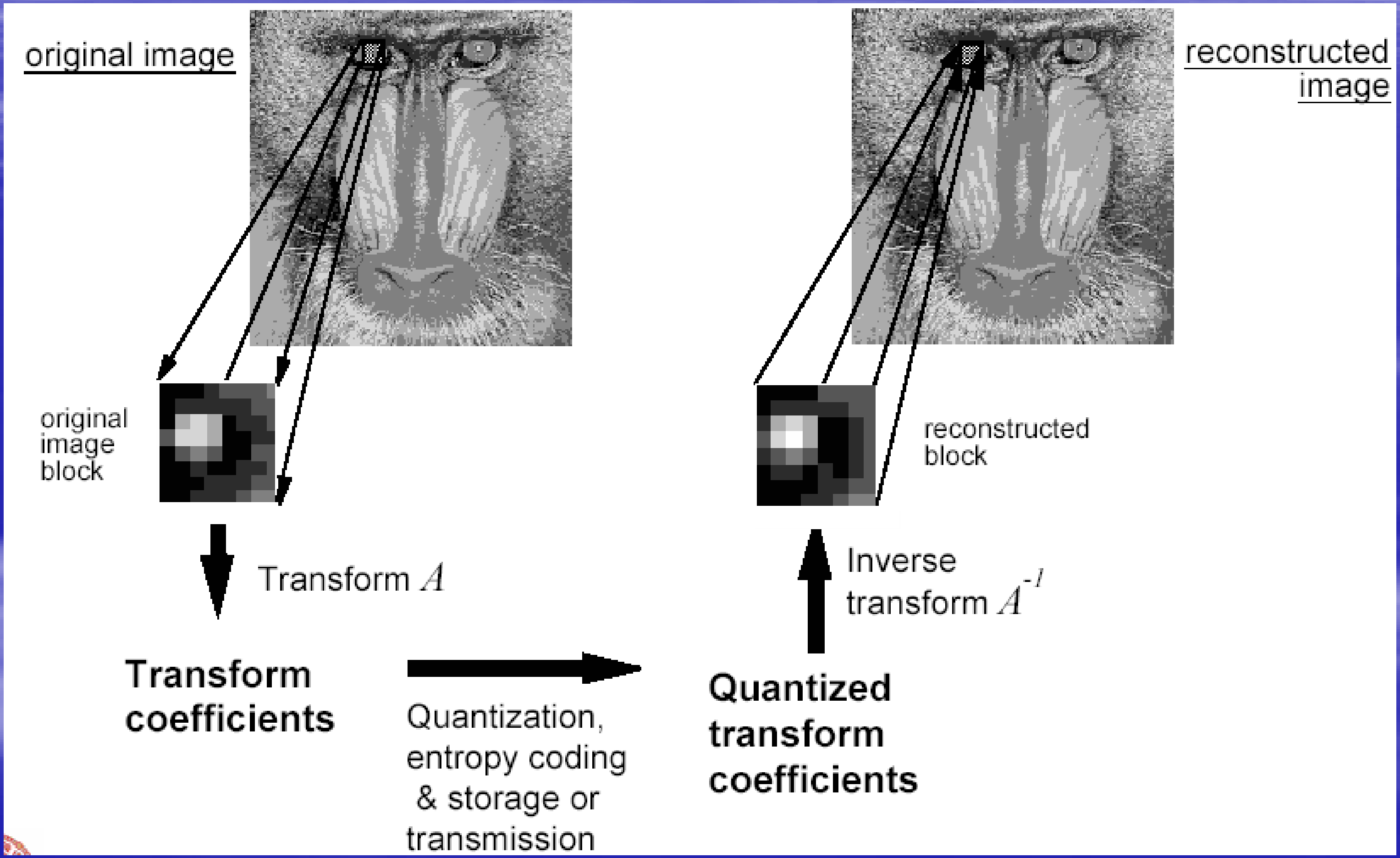


DCT

Hadamard

Haar

Transform Coding


original image
original image block
Transform A
Transform coefficients
Quantization, entropy coding & storage or transmission
Quantized transform coefficients
Inverse transform A^{-1}
reconstructed block
reconstructed image

Transform Coding (cont.)

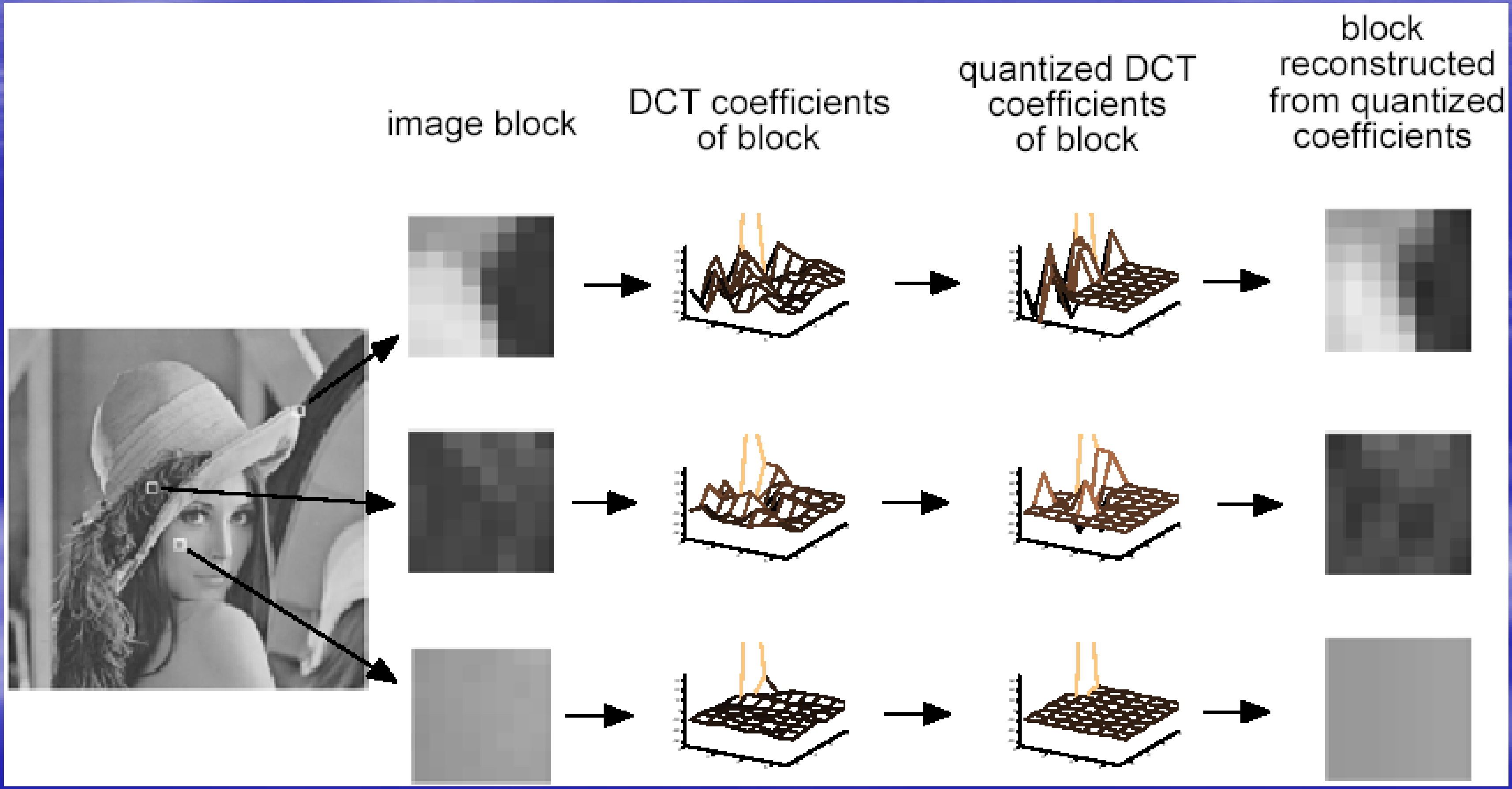
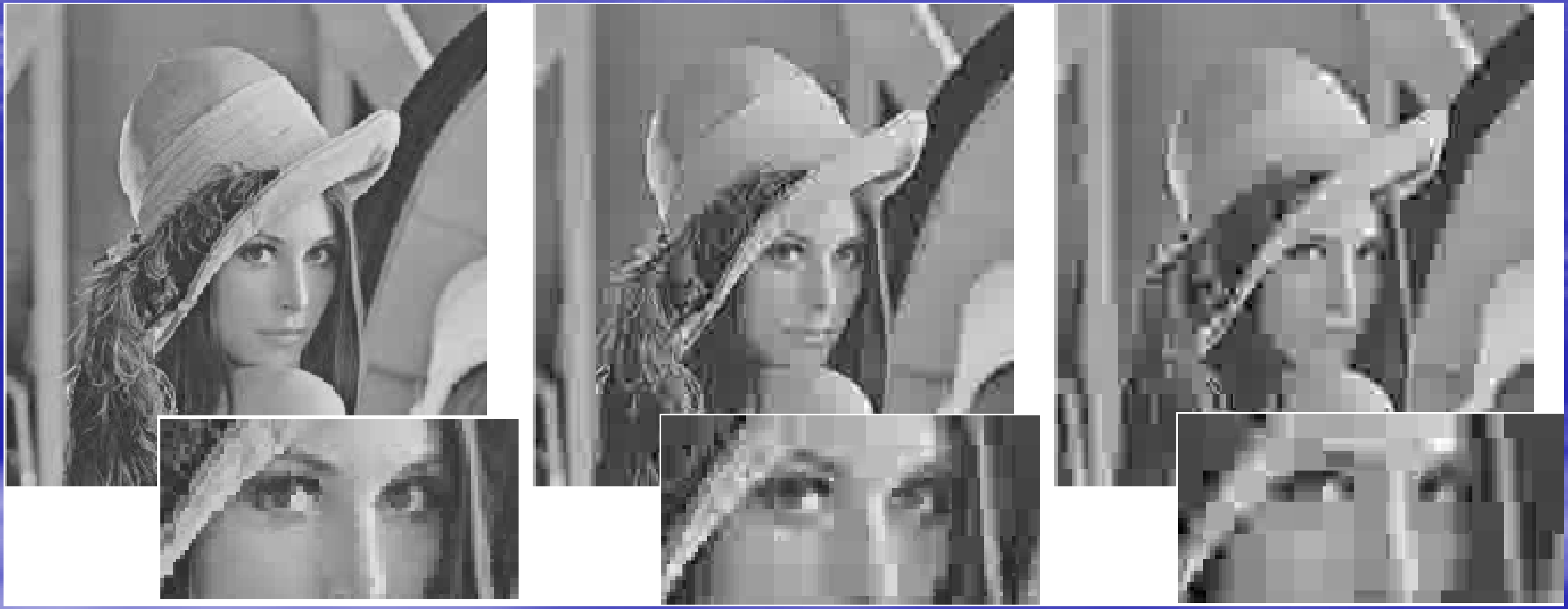


image block
DCT coefficients of block
quantized DCT coefficients of block
block reconstructed from quantized coefficients

DCT coding artifacts

DCT coding with increasingly coarse quantization, block size 8x8



quantizer stepsize
for AC coefficients: 25

quantizer stepsize
for AC coefficients: 100

quantizer stepsize
for AC coefficients: 200