Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

MIN-Fakultät
Fachbereich Informatik
Arbeitsbereich SAV/BV (KOGS)

Image Processing 1 (IP1)
Bildverarbeitung 1

Lecture 9 – Image Compression 2

Winter Semester 2014/15

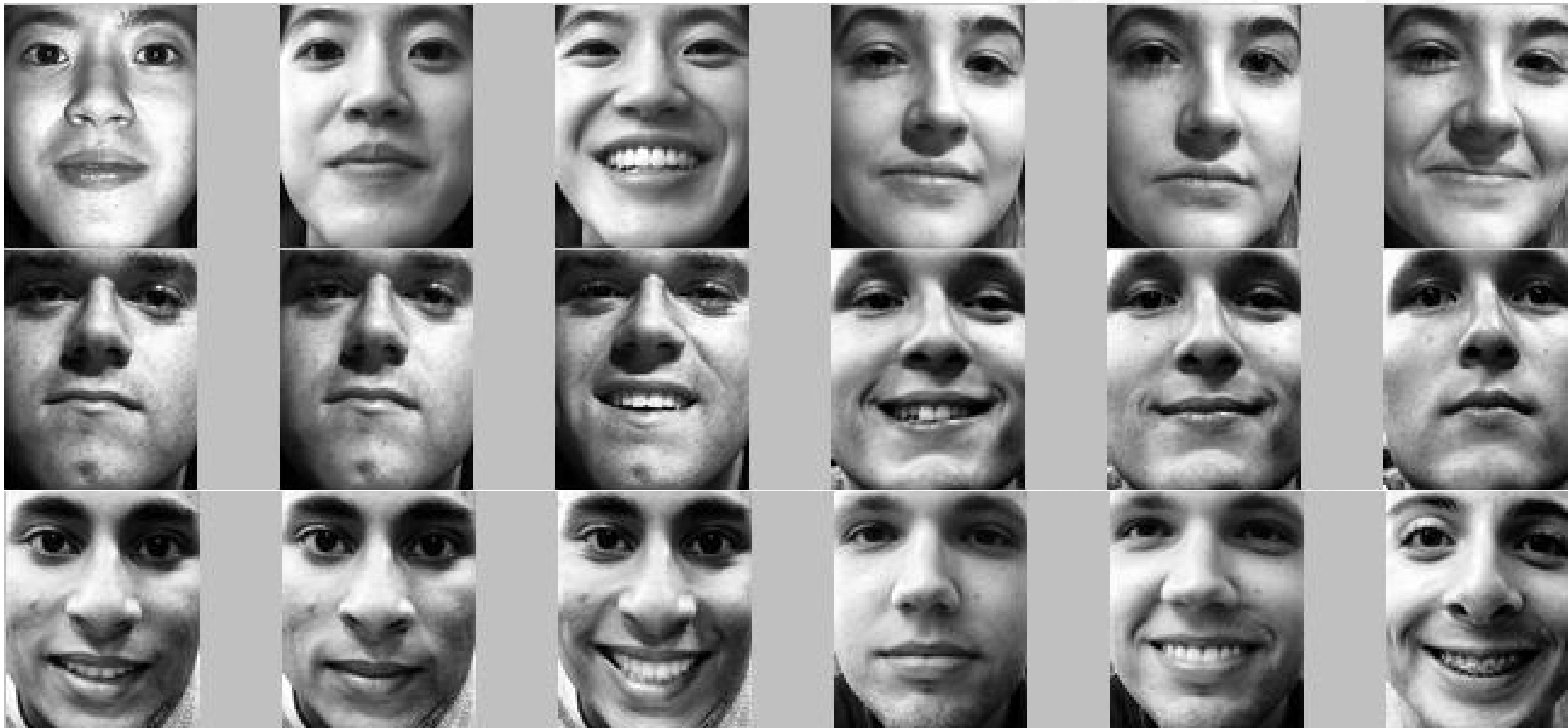
Dr. Benjamin Seppke
Prof. Siegfried Stiehl

Eigenfaces I

Turk & Pentland: Face Recognition Using Eigenfaces (1991)

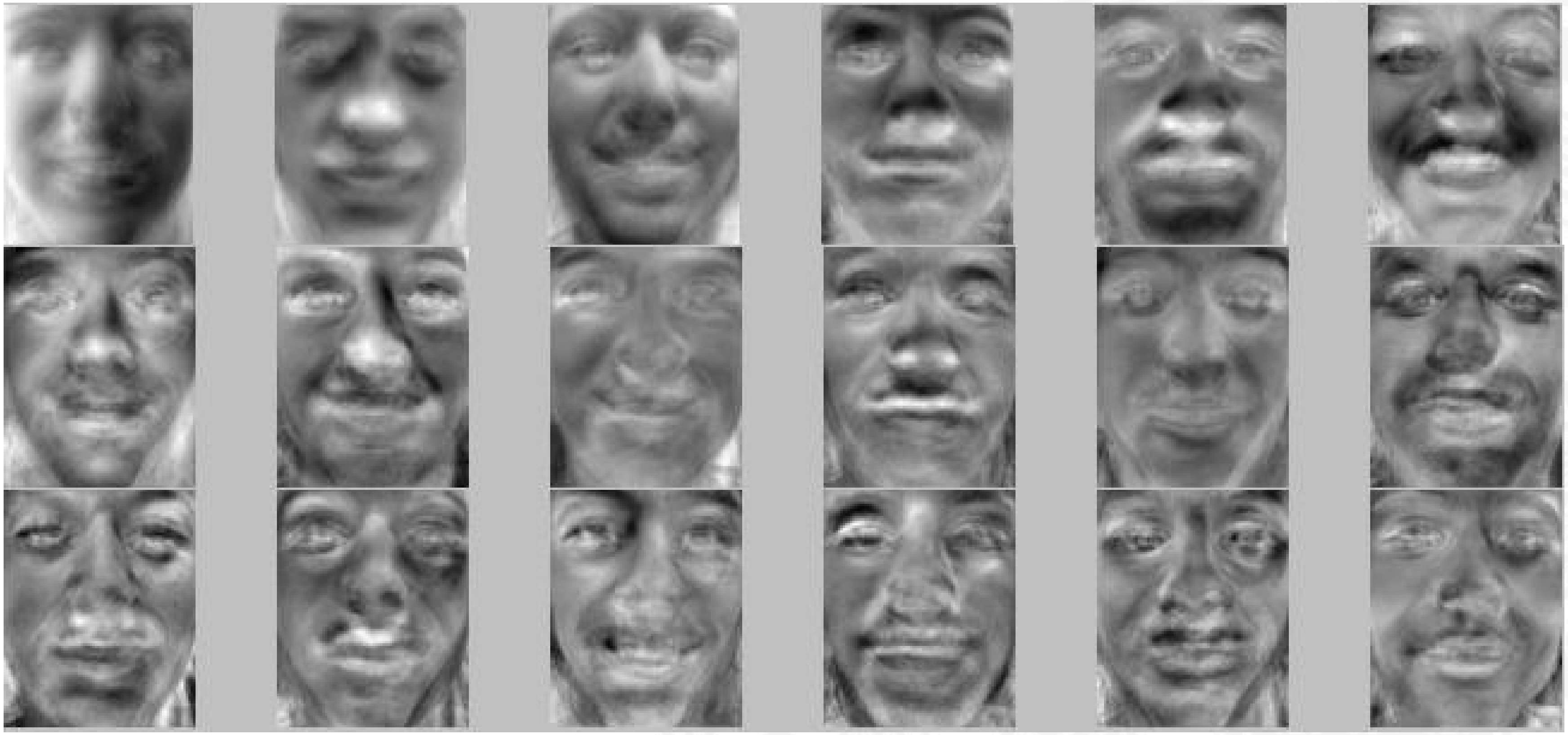
Eigenfaces = eigenvectors of covariance matrix of normalized face images

Example images of eigenface project at Rice University



Eigenfaces II

First 18 eigenfaces determined from covariance matrix of 86 face images:



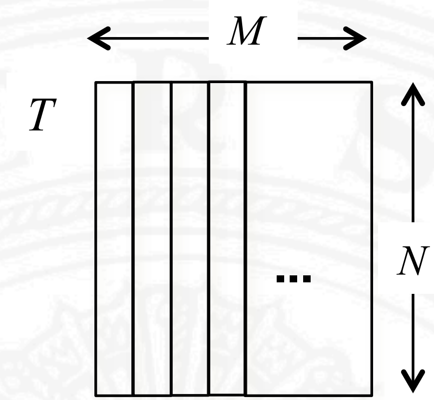
Eigenfaces III

Original images and reconstructions from 50 Eigenfaces:

Eigenvalues of Empirical Covariance Matrices

In praxis, the covariance matrix of images must be determined empirically.

1. Normalize M images to have equal format and size N
2. Form column vector of each image with N entries
3. Form matrix T of all images
4. Determine average image and subtract from all images



Empirical covariance matrix is $V = T\,T^T$ with $v_{ij} = \sum_{k=1}^{N} t_{ik} t_{jk}$

Determining the covariance matrix V of images is computationally costly:

Image size 100 x 100 → 10^8 entries!

Note that eigenvalues of $T^T T$ are related to eigenvalues of $T\,T^T$:

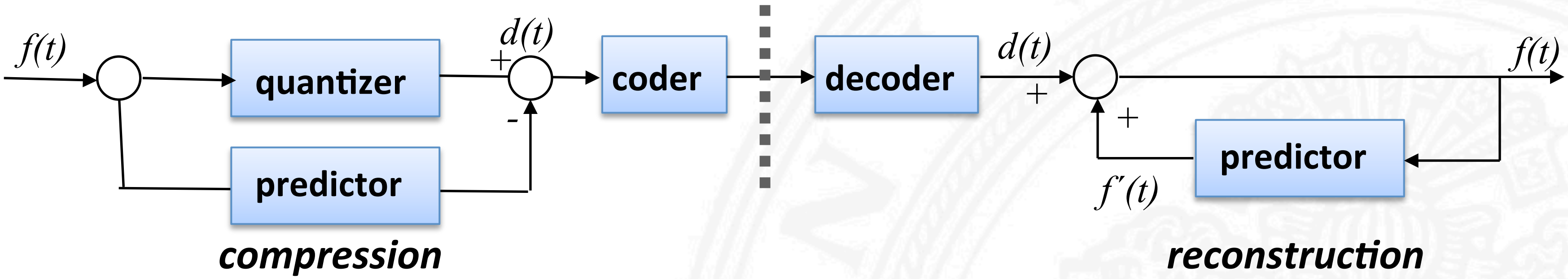
$$T^T T \vec{a}_i = \lambda_i \vec{a}_i \quad \Rightarrow \quad T\,T^T T \vec{a}_i = V\,T \vec{a}_i = \lambda_i T \vec{a}_i$$

Hence eigenvalues can be computed for $T^T T$ which is typically much smaller than V!

Lossless Predictive Coding

Principle:

- estimate $g_{mn}{'}$ from greyvalues in the neighbourhood of (mn)
- encode difference $d_{mn} = g_{mn} - g_{mn}{'}$
- transmit difference data (+ initially predictor)



Linear predictor for a neighbourhood of K pixels (1-dimensional notation):

$g_n{'} = a_1 g_{n-1} + a_2 g_{n-2} + \ldots + a_K g_{n-K}$ → difference $g_n - g_n{'}$ is encoded

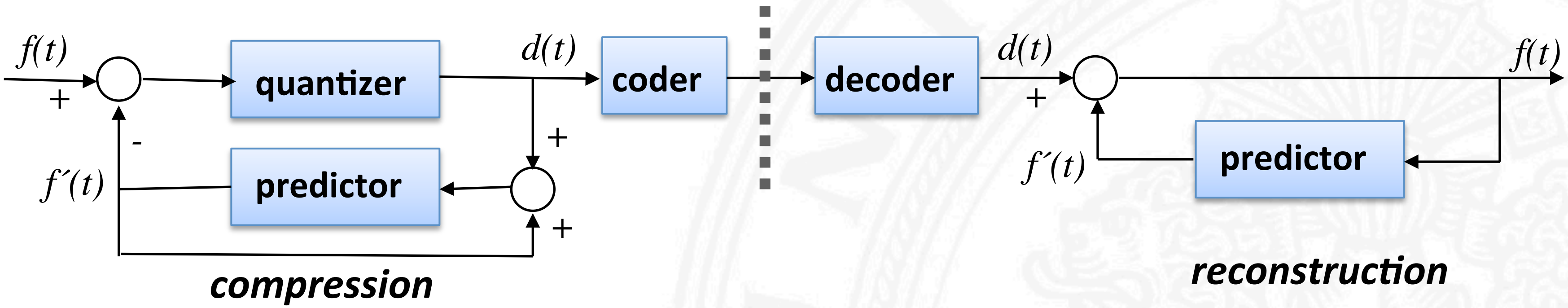
What is the effect of a first-order encoder with $K=1$?

Lossy Prediction

Principle:

- estimate g_{mn}' from greyvalues in the neighbourhood of (mn)
- encode **quantisized** difference $d_{mn} = g_{mn} - g_{mn}'$
- transmit difference data (+ initially predictor)

For a 1D signal this is known as Differential Pulse Code Modulation (DPCM):



Computation of $a_1 \dots a_K$
in a linear predictor $g_n' = a_1 g_{n-1} + a_2 g_{n-2} + \dots + a_K g_{n-K}$
by minimizing the expected reconstruction error: $E\{ [g_n' - g_n]^2 \}$

Minimizing the Prediction Error

If $\vec{g}_n$ is a zero mean stationary random process with autocorrelation C:

$$g'_n = \vec{a}^T \vec{g}_K \quad \text{where} \quad \vec{g}_K^T = \begin{pmatrix} g_{n-1} & g_{n-2} & \cdots & g_{n-K} \end{pmatrix}$$

$$C = \begin{pmatrix} E\{g_{n-1}g_{n-1}\} & \cdots & E\{g_{n-1}g_{n-K}\} \\ \vdots & \ddots & \vdots \\ E\{g_{n-K}g_{n-1}\} & \cdots & E\{g_{n-K}g_{n-K}\} \end{pmatrix} \qquad \vec{c} = \begin{pmatrix} E\{g_n g_{n-1}\} \\ \vdots \\ E\{g_n g_{n-K}\} \end{pmatrix}$$

The expectation value of the prediction error can be derived as:

$$E\left\{\left(g'_n - g_n\right)^2\right\} = E\left\{\left(\vec{a}^T \vec{g}_K - g_n\right)^2\right\} = E\left\{\left(\vec{a}^T \vec{g}_K\right)^2\right\} - 2E\left\{\left(\vec{a}^T \vec{g}_K g_n\right)^2\right\} + E\left\{\left(g_n\right)^2\right\}$$

$$= \vec{a}^T C\, \vec{a} - 2\vec{a}^T \vec{c} + E\left\{\left(g_n\right)^2\right\}$$

Minimizing the error gives:

$$\frac{\partial}{\partial \vec{a}} E\left\{\left(g'_n - g_n\right)^2\right\} = 2\vec{a}^T C - 2\vec{c} = 0 \quad \Rightarrow \quad \vec{a}^T C = \vec{c} \quad \Rightarrow \quad \vec{a}^T = 2\vec{c}\, C^{-1}$$

Example of Linear Predictor

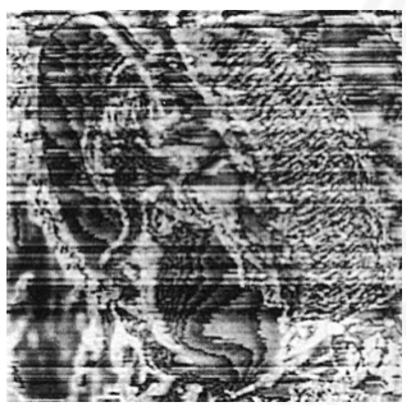
For images, a linear predictor based on 3 pixels (3rd order) is often sufficient:

$$g'_n = a_1 \vec{g}_{n-1} + a_2 \vec{g}_{n-2} + a_3 \vec{g}_{n-3}$$

Example for mapping into 2D:

n-3	n-2
n-1	n

Example:



Predictive compression with 2nd order predictor and Huffman coding, ratio 6.2

Left: Reconstructed image

Right: Difference image (right) with maximal difference of 140 greylevels

Discrete Cosine Transformation (DCT)

Discrete Cosine Transform is commonly used for image compression,
e.g. in JPEG (Joint Photographic Expert Group) Baseline System standard.

Definition of DCT:

$$G_{00} = \frac{1}{N} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} g_{mn}$$

$$G_{uv} = \frac{1}{2N^3} \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} g_{mn} \cos[(2m+1)u\pi] \cos[(2n+1)v\pi]$$

Inverse DCT:

$$g_{mn} = \frac{1}{N} G_{00} + \frac{1}{2N^3} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} G_{uv} \cos[(2m+1)u\pi] \cos[(2n+1)v\pi]$$

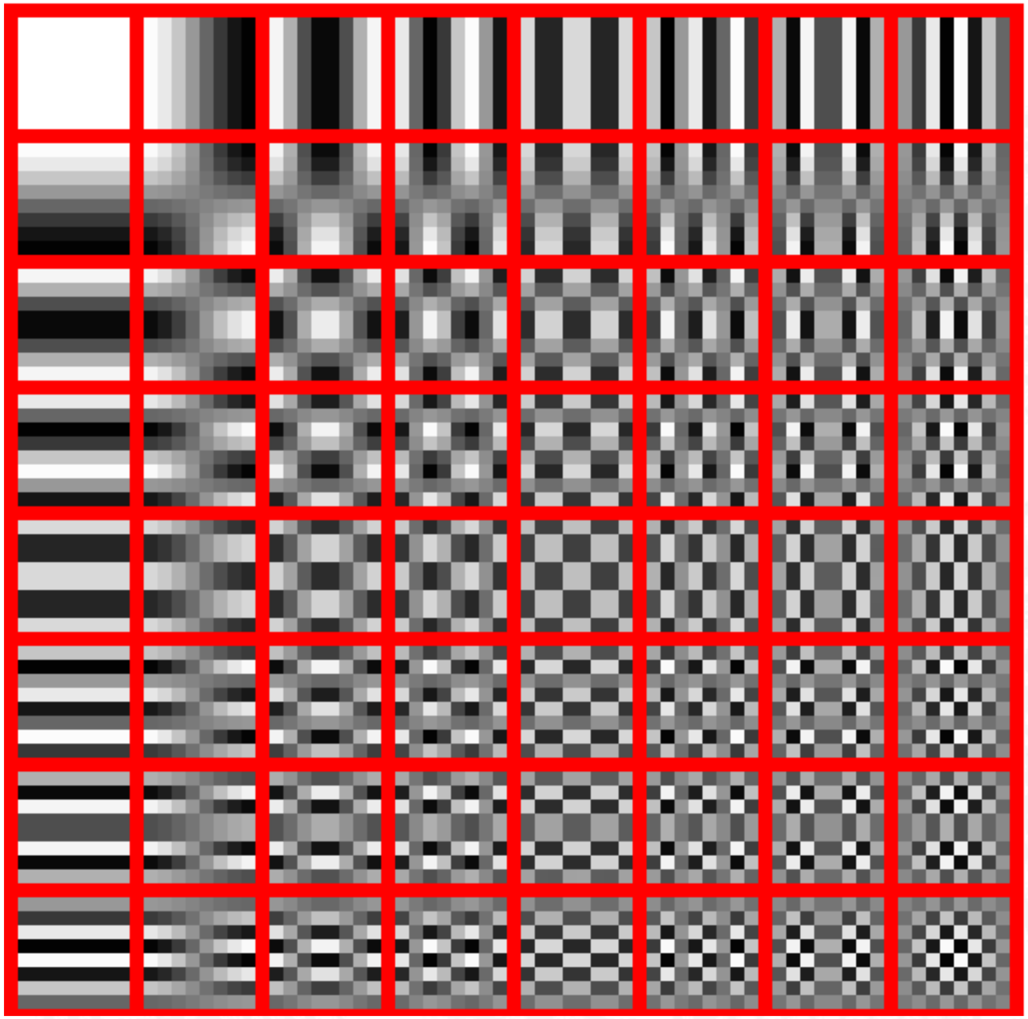
In effect, the DCT computes a Fourier Transform of a function made symmetric at N by a mirror copy.

1. Result does not contain sinus terms
2. No wrap-around errors.

DCT does not imply a compression!
But: A compression may be achieved by frequency dismissing or quantization.

Coefficients of the 2D Cosine Transformation

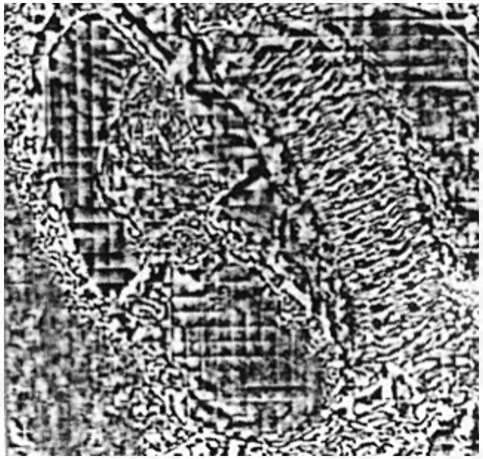
The coeffients of the 8x8-DCT-Transformation represent Cosine fractions with lengths as multiples of π:



Example: DCT Compression

Example:

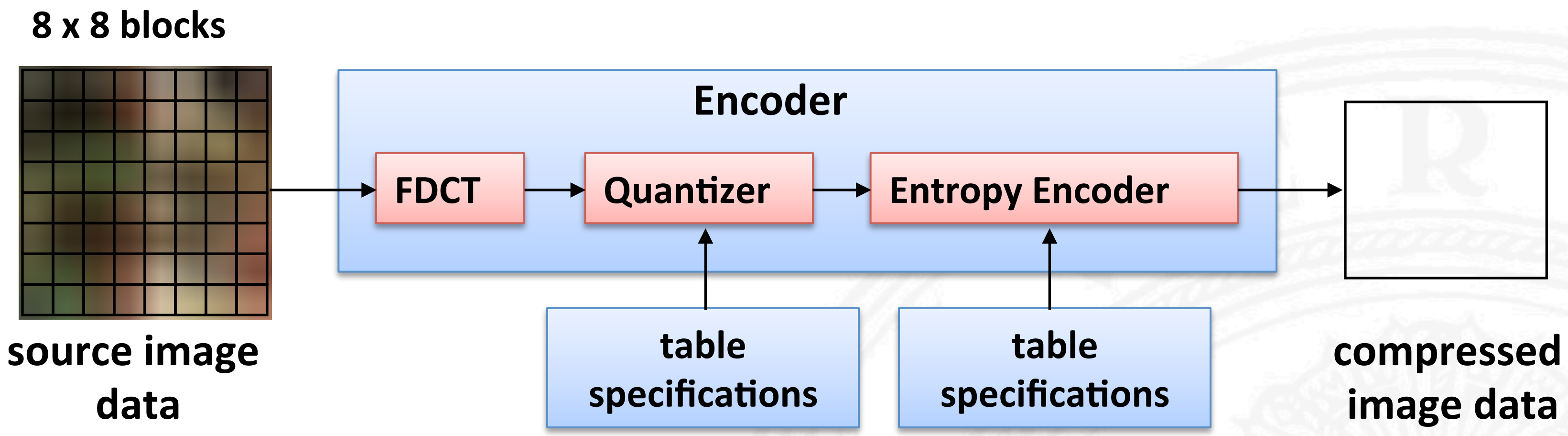
- DCT compression with ratio 1 : 5.6
- Left: Reconstructed image
- Right: Difference image (right) with maximal difference of 125 greylevels



Compare with Predictive Huffmann Coding of similar compression ratio!

Principle of Baseline JPEG

(Source: Gibson et al., Digital Compression for Multimedia, Morgan Kaufmann 98)



- transform RGB into YUV coding, subsample color information
- partition image into 8 x 8 blocks, left-to-right, top-to-bottom
- compute Discrete Cosine Transform (DCT) of each block
- quantize coefficients according to psychovisual quantization tables
- order DCT coefficients in zigzag order
- perform runlength coding of bitstream of all coefficients of a block
- perform Huffman coding for symbols formed by bit patterns of a block

YUV Color Model for JPEG

Human eyes are more sensitive to luminance (brightness) than to chrominance (color). YUV color coding allows to code chrominance with fewer bits than luminance.

CCIR-601 scheme:

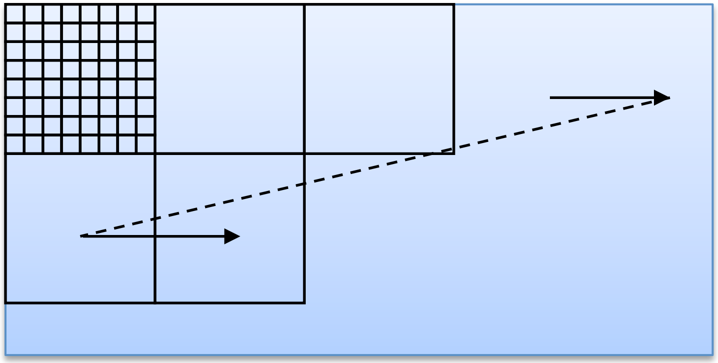
$Y =$	$0.2990\ R$	$+\ 0.5870\ G$	$+\ 0.1440\ B$	"luminance"
$Cb =$	$0.1687\ R$	$-\ 0.3313\ G$	$+\ 0.5000\ B$	"blueness“
$Cr =$	$0.5000\ R$	$-\ 0.4187\ G$	$-\ 0.0813\ B$	"redness“

In JPEG:

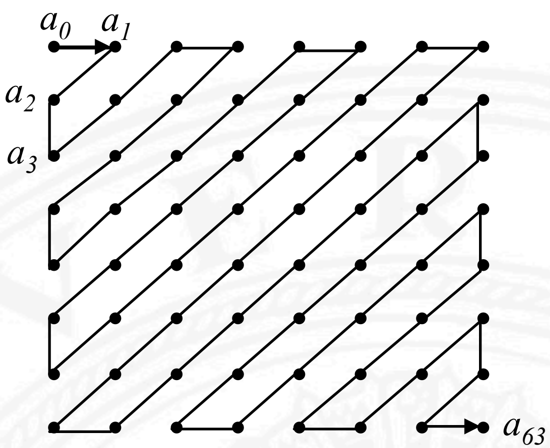
1 Cb, 1 Cr and 4 Y values for each 2 x 2 image subfield

(6 instead of 12 values)

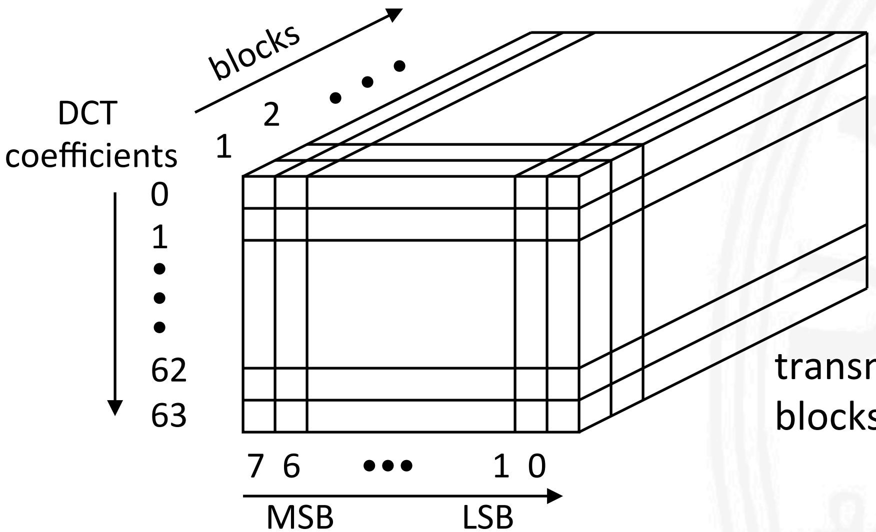
Illustrations for Baseline JPEG



partitioning the image into blocks



DCT coefficient ordering for efficient runlength coding

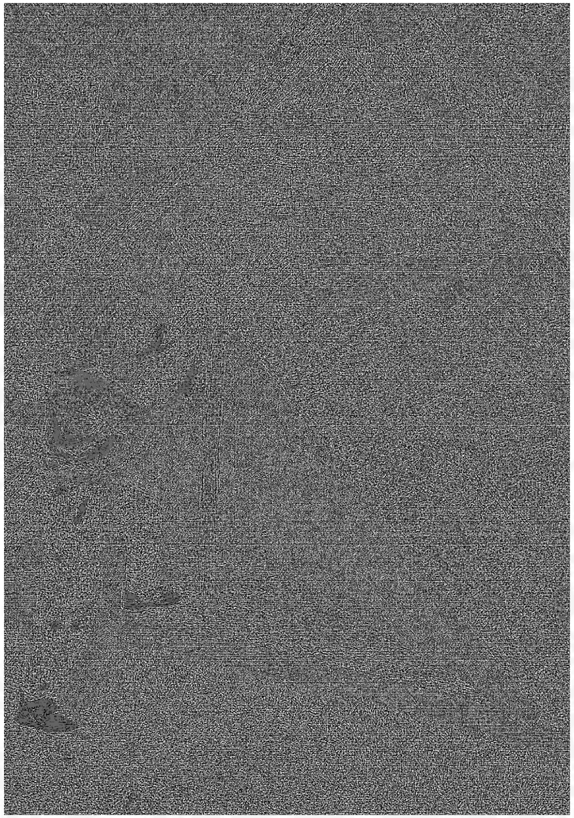


transmission sequence for blocks of image

JPEG-compressed Image

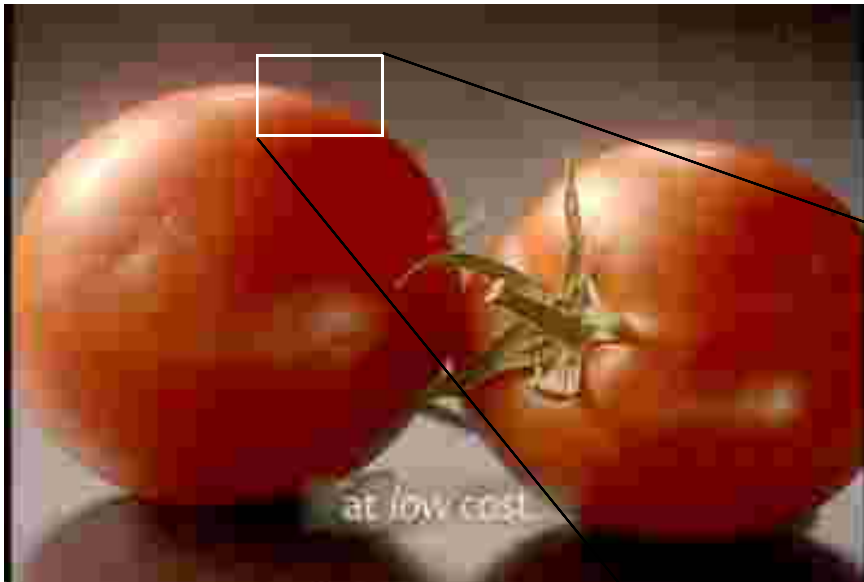
original
5.8 MB

JPEG-compressed
450 KB



difference image
standard deviation of luminance differences: 1,44

Problems with Block Structure of JPEG



JPEG encoding with compression ratio 1:70

block boundaries are visible

Progressive Encoding

Progressive encoding allows to first transmit a coarse version of the image which is then progressively refined (convenient for browsing applications).

Spectral selection

1. transmission: DCT coefficients $a_0 \dots a_{k1}$
2. transmission: DCT coefficients $a_{k1} \dots a_{k2}$

⋮

low frequency coefficients first

Successive approximation

1. transmission: bits $7 \dots n_1$
2. transmission: bits $n_1+1 \dots n_2$

⋮

most significant bits first

MPEG Compression

Original goal:
Compress a 120 Mbps video stream to be handled by a CD with 1 Mbps.

Basic procedure:
- temporal prediction to exploit redundancy between image frames
- frequency domain decomposition using the DCT
- selective reduction of precision by quantization
- variable length coding to exploit statistical redundancy
- additional special techniques to maximize efficiency

Motion compensation:
- 16 x 16 blocks luminance with 8 x 8 blocks chromaticity of the current image frame are transmitted in terms of
  - an offset to the best-fitting block in a reference frame (motion vector)
  - the compressed differences between the current and the reference block

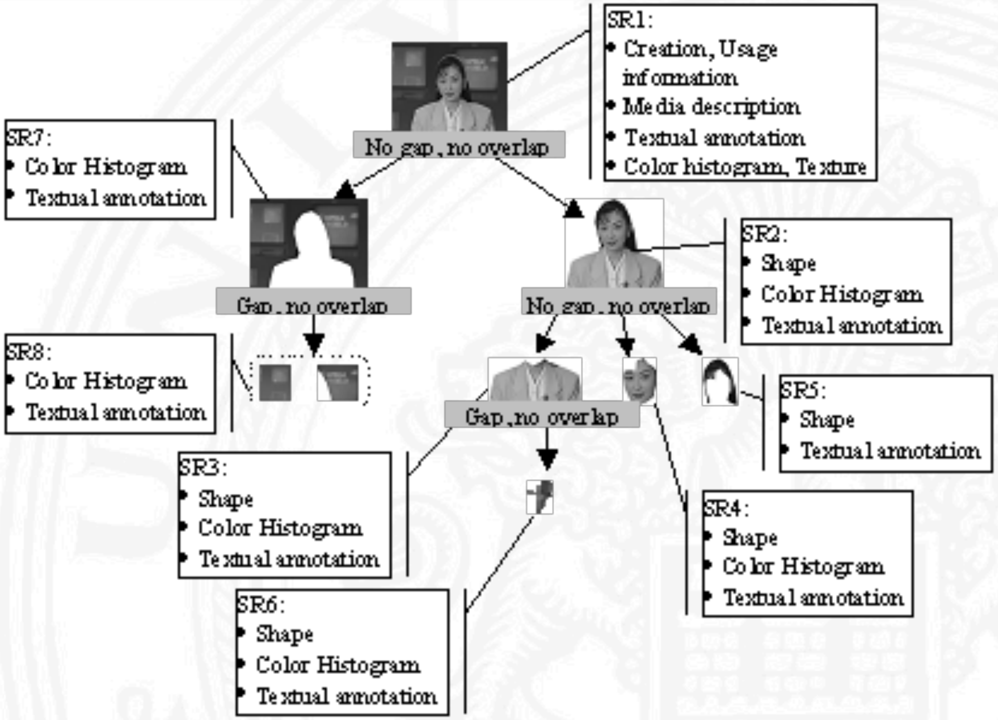
MPEG-7 Standard

MPEG-7: "Multimedia Content Description Interface"

- **introduced as standard in 2002**
- **supports multimedia content description (audio and visual)**
- **not aimed at a particular application**

Descrtiption of visual contents in terms of:

- descriptors (e.g. color, texture, shape, motion, localization, face features)
- segments
- structural information
- Description Definition Language (DDL)

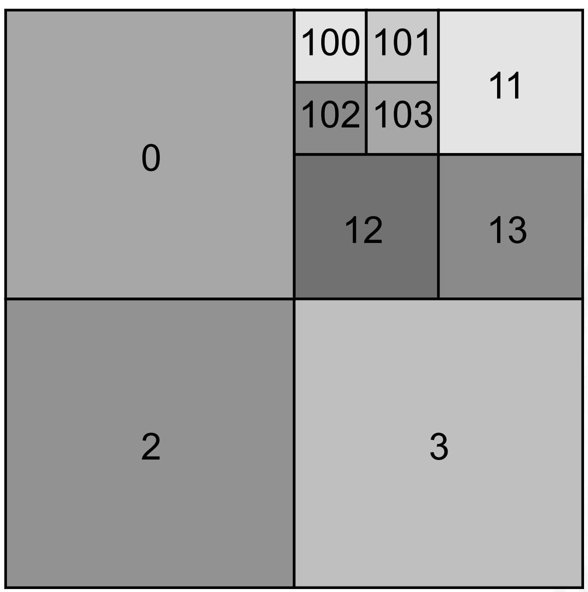


Segmentation methodology required!

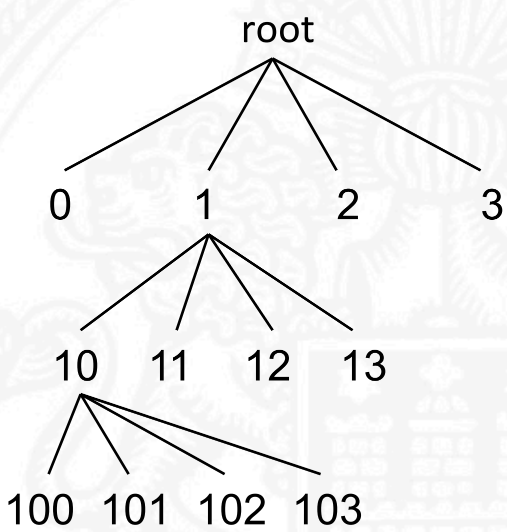
Quadtree Image Representation

Properties of quadtree:

- every node represents a squared image area, e.g. by its mean greyvalue
- every node has 4 children except leaf nodes
- children of a node represent the 4 subsquares of the parent node
- nodes can be refined if necessary



quadtree structure:



Quadtree Image Compression

A complete quadtree represents an image of $N = 2K \times 2K$ pixels with
$1 + 4 + 16 + ... + 2^{2K} \approx 1.33\ N$ nodes.

An image may be compressed by

- storing at every child node the greyvalue difference between child and parent node
- omitting subtrees with equal greyvalues

Quadtree image compression supports progressive image transmission:

- images are transmitted by increasing quadtree levels,
  i.e. images are progressively refined
- intermediate image representations provide useful information,
  e.g. for image retrieval