Lecture 8: Image Compression

Contents

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30
31 32
33 34
35 36

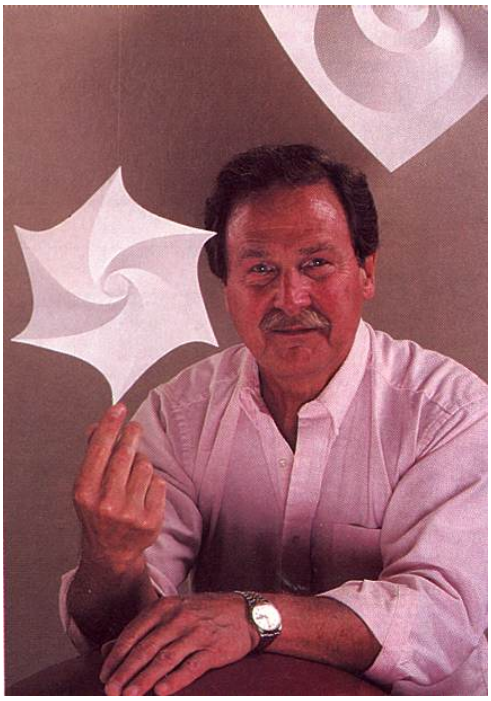
Introduction

- Due to their size, storing digital images requires much disk space. Also transmitting them needs a large bandwidth or a long time.
- Thus, it is highly desirable to compress images to a smaller size.
- Compression can exploit *redundancy* and *irrelevance.*
- *lossless image compression (verlustfreie Bildkompression):*
  - removes redundancy, e.g. by
    – using shorter codes for more frequent grey values
    – exploiting spatial correlations
  - less efficient (typically compression rates up to 4:1)
  - examples: Huffman coding, run length encoding
- *lossy image compression (verlustbehaftete Bildkompression):*
  - creates a different image that appears visually similar
  - benefits also from irrelevances for our visual system, e.g. by using a coarser quantisation of high frequencies
  - higher compression rates in good quality possible (beyond 10:1)
  - examples: JPEG (uses DCT), JPEG 2000 (uses wavelet transform)

Huffman Coding

- lossless compression technique
- assigns shorter codes to more frequent symbols (in our case: grey values).
- widely used in fax machines, transmission over computer networks, and high-definition television (HDTV)
- greedy algorithm that generates code in a binary tree structure
- does not require a separator between code words:
  prefix-free, i.e. no code word is the prefix of another one

M I
A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30
31 32
33 34
35 36



David A. Huffman (1925–1999) introduced Huffman coding in a term paper as a graduate student. He worked as professor at the MIT and the University of California at Santa Cruz. Later on he became interested in folding paper into sculptured shapes. He never tried to patent his work. He said: "My products are my students." Source: `http://www.huffmancoding.com/david/scientific.html`.

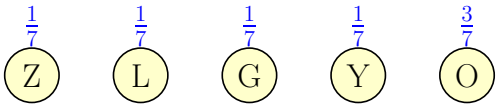
Huffman Coding (3)

Example of Huffman Coding

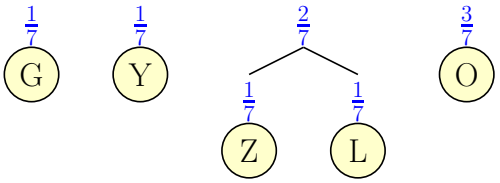
- Assume we want to encode the word ZOOLOGY.
- Its letters occur with the probabilities

$$p(Z) = \tfrac{1}{7}, \quad p(O) = \tfrac{3}{7}, \quad p(L) = \tfrac{1}{7}, \quad p(G) = \tfrac{1}{7}, \quad p(Y) = \tfrac{1}{7}.$$

- Order the letters w.r.t. increasing probability.

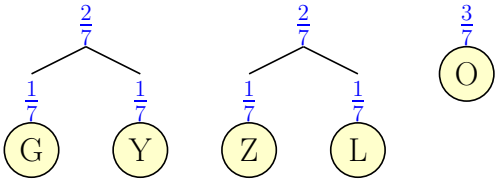


- Sum up the two with the lowest probability, and sort the result into the list again.

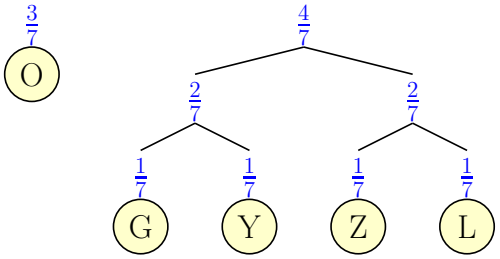


Huffman Coding (4)

- Proceed as in the previous step.

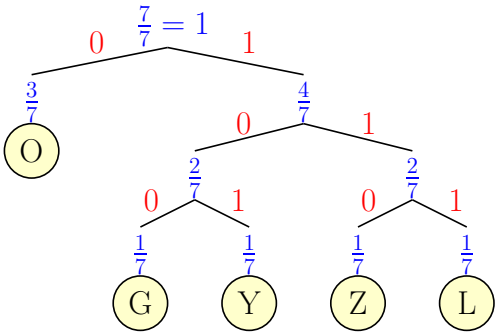


- One more time:



Huffman Coding (5)

- The last step is reached when all letters are sorted into the tree.
  Afterwards each left edge is assigned the code 0, and the right edge the code 1.



- Going from the root to the leaves gives the Huffman code for the letter:
  Code(Z) = 110, Code(O) = 0, Code(L) = 111, Code(G) = 100, Code(Y) = 101.
- Hence the word ZOOLOGY is encoded as Code(ZOOLOGY) = 110001110100101.
  This requires 15 bits.
- A conventional encoding of the five different letters Z, O, L, G, Y requires 3 bits per letter, resulting in 21 bits for the word ZOOLOGY.

Huffman Coding (6)

Huffman Decoding

- In the transmission process, the Huffman tree is transmitted in the code header.
  This overhead is uncritical, if the transmitted text is large.
- Decoding proceeds through the chain.
  Whenever a leaf in the Huffmann tree is reached, a letter is identified.
- In our example:
  110–0–0–111–0–100–101
  yielding ZOOLOGY.

Huffman Coding (7)

Information Theoretic Considerations

- Assume we want to encode some letter k that occurs with probability $p_k \in (0, 1]$. Information theory tells us that it is optimal to use $\operatorname{ld}(1/p_k) = -\operatorname{ld} p_k$ bits. Note that $\operatorname{ld} x := \log_2 x = \ln x / \ln 2$ denotes the logarithm with base 2.
- In our ZOOLOGY example:
  - The letter O should have a code length of $-\operatorname{ld} \frac{3}{7} \approx 1.22$ bits.
  - The letters Z, L, G, Y should have a code length of $-\operatorname{ld} \frac{1}{7} \approx 2.81$ bits.
- Huffman coding cannot use non-integer code lengths. It encodes
  - the letter O with 1 bit,
  - the letters Z, L, G, Y with 3 bits.
- The theoretically optimal average number of bits per letter is given by the weighted average of the optimal character lengths $-\operatorname{ld} p_k$ with weights p_k:

$$H = -\sum_k p_k \operatorname{ld} p_k.$$

H is called the *entropy (Entropie)* of the word (character string, image). It gives a lower bound on the code length. It is usually not reached in practice.

Huffman Coding (8)

- The entropy can be seen as a measure of information content: More code lenth means more information.
- In the ZOOLOGY example the entropy is given by

$$\begin{aligned} H &= -p_Z \operatorname{ld} p_Z - p_O \operatorname{ld} p_O - p_L \operatorname{ld} p_L - p_G \operatorname{ld} p_G - p_Y \operatorname{ld} p_Y \\ &= -\tfrac{1}{7} \operatorname{ld} \tfrac{1}{7} - \tfrac{3}{7} \operatorname{ld} \tfrac{3}{7} - \tfrac{1}{7} \operatorname{ld} \tfrac{1}{7} - \tfrac{1}{7} \operatorname{ld} \tfrac{1}{7} - \tfrac{1}{7} \operatorname{ld} \tfrac{1}{7} \\ &\approx 2.13. \end{aligned}$$

  Our Huffman code uses 15 bits for 7 letters, yielding ≈ 2.14 bits per character. This is very close to the optimum given by the entropy.
- Approaches that approximate the theoretically optimal code length are called *entropy coding* methods.
- Huffman coding is one example of an entropy coding method. It is the optimal letter-by-letter coding with integer code length, if the stream of letters is unrelated and the input probability distribution is known for each letter.
- Often one can do slightly better with the so-called *arithmetic encoding.* It approximates the information theoretic optimum by
  - refraining from encoding individual letters,
  - encoding an entire word by a real number in $[0, 1)$.

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36

Run Length Encoding (RLE, Lauflängenkodierung)

Basic Idea

- simple lossless compression technique that exploits the spatial context in signals
- The grey values of an image are written in a single string,
  e.g. by proceeding row-wise from top left to bottom right.
- Let a grey value occur in multiple consecutive positions.
  Then one stores only the single data value and its count.
- highly useful for simple images such as the binary data of fax machines:
  often yields large run lengths

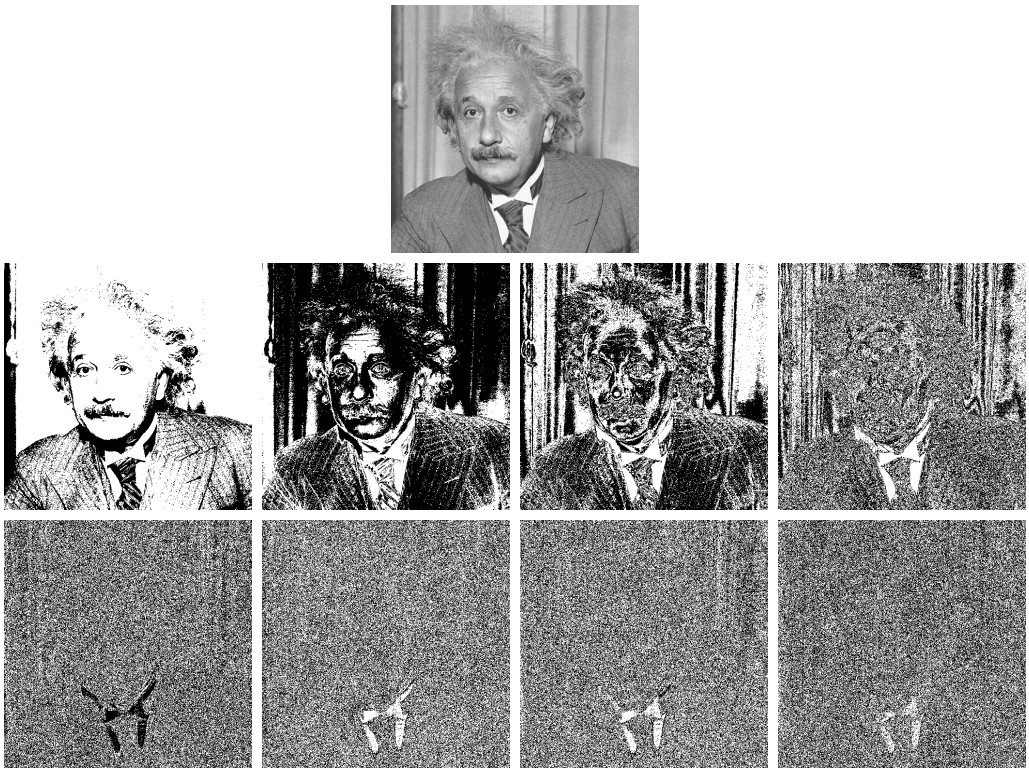
Modification for Greyscale Images

- Problem:
  - For typical greyscale images with bytewise coding, small grey value fluctuations spoil the performance of RLE.
- Remedy: *Bitplane Coding*
  - Decompose each grey value in $\{0, 1, ..., 255\}$ in its 8 bits, e.g.

$$148 \;=\; 128 + 16 + 4 \;=\; (10010100)_2$$

    where the left bit 1 is the *most significant bit,*
    and the right bit 0 the *least significant bit.*
  - Replace the bytewise coded image by its 8 binary images *(bitplanes)* that are given by the 8-bit representation.
  - The more significant bitplanes hardly fluctuate.
    They can be encoded efficiently with a separate RLE in each bitplane.

Run Length Encoding (3)



Top: Original image (512×512 pixels) with bytewise coding. **Middle left to bottom right:** Its eight bitplanes, starting from the most significant bit. Authors: M. Mainberger and J. Weickert.

Run Length Encoding (4)

Further Improvement

- Problem:
  - In the classical binary code, a grey value change by 1 can affect many bitplanes.
- Example:
  - 3 has the binary representation $(011)_2$, while 4 has $(100)_2$.
- Remedy:
  - In the so-called *Gray code*, grey value changes by 1 alter only a single bit.
  - This makes bitplane coding more efficient.

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36

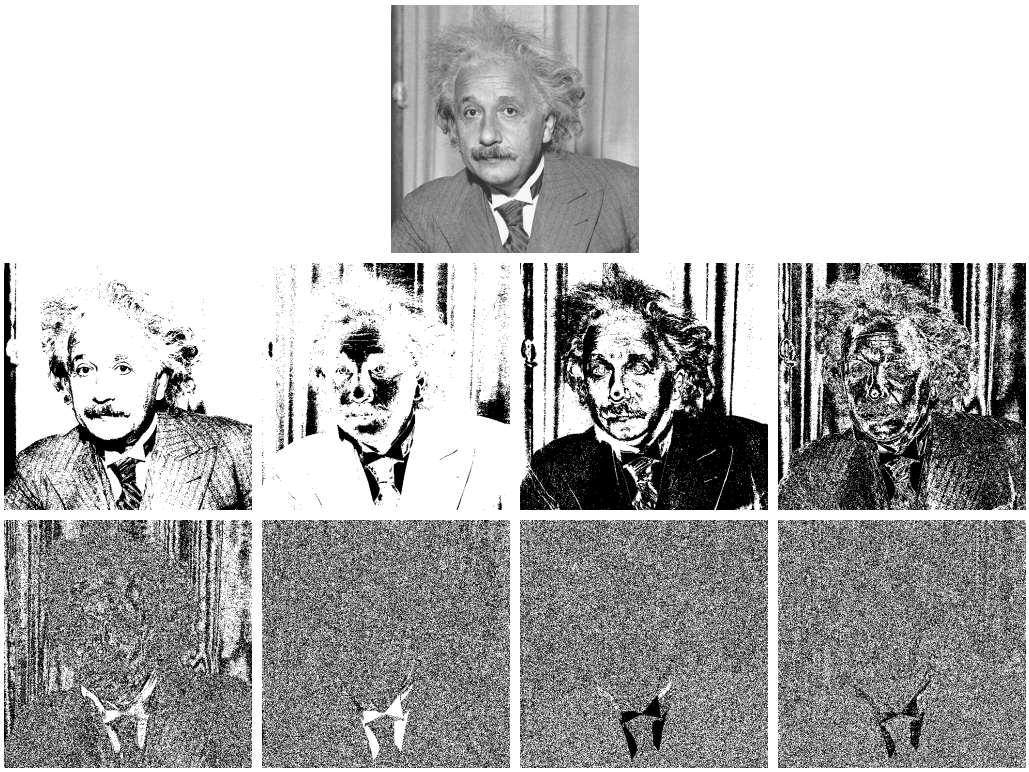
Run Length Encoding (5)

Gray Code

Number	Conventional Code	Gray Code
0	0 0 0	0 0 0
1	0 0 1	0 0 1
2	0 1 0	0 1 1
3	0 1 1	0 1 0
4	1 0 0	1 1 0
5	1 0 1	1 1 1
6	1 1 0	1 0 1
7	1 1 1	1 0 0

- For the last digit, the sequence 01 is repeated by mirroring (instead of periodically as is done for the conventional binary code).
- For the second last digit, one repeats 0011 by mirroring.
- For the third last digit, one repeats 00001111 by mirroring.
- In this way, jumps in different bitplanes occur at different locations.

Run Length Encoding (6)



Top: Original image. **Middle left to bottom right:** Its eight bitplanes using Gray code. The middle bitplanes fluctuate less than in the previous example. Authors: M. Mainberger and J. Weickert.

M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36

The JPEG Image Compression Standard

- has been introduced by the **J**oint **P**hotographic **E**xperts **G**roup in 1992
- allows four compression modes, including also lossless compression
- We focus on the most widely used one: sequential DCT-based compression (lossy). It is often referred to as "the" JPEG mode.
- coding of colour images consists of six steps:
  1. transformation from RGB to YCbCr
  2. subsampling of the chroma channels
  3. DCT on 8×8 pixel blocks
  4. quantisation of the DCT coefficients
  5. reordering of the DCT coefficients
  6. entropy coding
- decoding is done in reverse order, using the inverse DCT

Step 1: Transformation from RGB to YCbCr

- cf. Lecture 3
- YCbCr is a colour representation that separates the luma channel Y from the chroma channels C_b and C_r.
- This transformation is lossless.

Step 2: Subsampling of the Croma Channels

- The human visual system is more tolerant w.r.t. spatial imprecisions of the chroma channels C_b and C_r than the luma channel Y.
- Subsampling the chroma channels C_b and C_r by a factor 2 in each direction results in a lossy compression of good quality.

The JPEG Image Compression Standard (3)

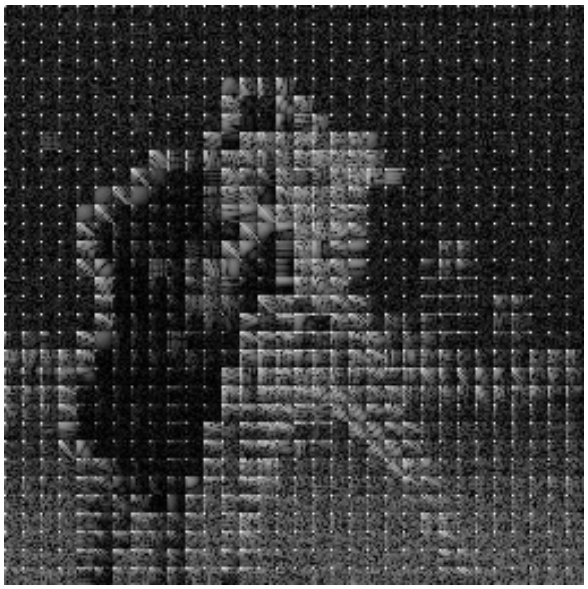
Top left: Original image, 512×512 pixels. **Top right:** Reconstruction from the YCbCr space, where the chroma channels C_b and C_r have been subsampled by a factor 2 in each direction. **Bottom left:** Factor 4. **Bottom right:** Factor 8. Author: J. Weickert.

The JPEG Image Compression Standard (4)

Step 3: DCT on 8×8 Pixel Blocks

- cf. Lecture 6
- This step is applied to all YCbCr channels.
- Shift the unsigned image values from the interval $[0, 255]$ to $[-128, 127]$ (yields a smaller first DCT coefficient).
- Decompose the image into blocks of 8×8 pixels (yields a reasonable homogeneity within each block and allows parallel processing).
- Compute the 64 DCT coefficients in each block independently.
- This step is lossless.

The JPEG Image Compression Standard (5)



Left: Original image, 256×256 pixels. **Right:** Spectrum after performing the DCT in each 8×8 block. Authors: A. Bruhn and J. Weickert.

The JPEG Image Compression Standard (6)

Step 4: Quantisation of the DCT Coefficients

- central step for data reduction:
  - The human visual system is not very sensitive to errors in high frequencies.
  - Thus, higher frequencies can be quantised in a coarser way.
- Divide every DCT coefficient by its corresponding entry in an 8×8 quantisation matrix, and round it to the next integer.
- Example of a quantisation matrix:

$$\begin{pmatrix} 10 & 15 & 25 & 37 & 51 & 66 & 82 & 100 \\ 15 & 19 & 28 & 39 & 52 & 67 & 83 & 101 \\ 25 & 28 & 35 & 45 & 58 & 72 & 88 & 105 \\ 37 & 39 & 45 & 54 & 66 & 79 & 94 & 111 \\ 51 & 52 & 58 & 66 & 76 & 89 & 103 & 119 \\ 66 & 67 & 72 & 79 & 89 & 101 & 114 & 130 \\ 82 & 83 & 88 & 94 & 103 & 114 & 127 & 142 \\ 100 & 101 & 105 & 111 & 119 & 130 & 142 & 156 \end{pmatrix}$$

- The quantisation matrix determines the visual quality of the compression:
  - Remember that we want to quantise higher frequencies in a coarser way. Therefore, the quantisation matrix uses larger values for higher frequencies.
  - It is computed for a desired compression quality and stored in the JPEG header.

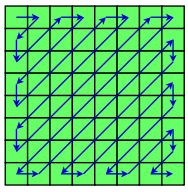
The JPEG Image Compression Standard (7)

Top left: Original image, 256×256 pixels. **Top right:** Spectrum after performing the DCT in each 8×8 block. **Bottom left:** Spectrum after quantisation as in JPEG. **Bottom right:** Reconstruction from the quantised DCT coefficients. Authors: A. Bruhn and J. Weickert.

The JPEG Image Compression Standard (8)

Step 5: Reordering of the DCT Coefficients

- Many of the quantised DCT coefficients for high frequencies are zero.
  To code them efficiently, one reorders the coefficients in a zigzag way.
  Zeroes at the end of the chain are not coded.



- The first coefficients in each block represent the average grey values.
  They usually do not vary too much over the image.
  Therefore, only the differences to their neighbouring blocks are coded.

Step 6: Entropy Coding

- uses Huffman coding of the quantised DCT coefficients
- stores run length of zeros before a non-zero coefficient and size of this coefficient

The JPEG Image Compression Standard (9)

Top left: Original image, 256×256 pixels, 8 bits-per-pixel (bpp). **Top right:** JPEG compression to 0.8 bpp (compression rate 10:1) gives rather good quality. **Bottom left:** At 0.4 bpp (compression rate 20:1) visible deteriorations appear. **Bottom right:** Severe block artifacts at 0.2 bpp (compression rate 40:1). Author: I. Galić.

JPEG 2000 (1)

JPEG 2000

- designed as the successor of JPEG
- based on the discrete wavelet transform instead of the discrete cosine transform (cf. Lecture 7)
- better quality than JPEG for high compression rates
- not very frequently used so far due to lack of free coding software

Left: Original image, 256×256 pixels, 8 bits-per-pixel (bpp). **Middle:** JPEG compression to 0.2 bpp. **Right:** JPEG 2000 compression to 0.2 bpp. Author: I. Galić.

Summary

- Lossless image compression removes redundancy. Two examples:
  - Huffman coding assigns shorter codes to more frequent grey values. It belongs to the class of entropy coding methods.
  - Runlength encoding exploits the spatial coherence of grey values.
- Lossy compression also exploits irrelevance:
  - creates another image that looks similar
  - may involve subsampling in a transformed domain or a coarser quantisation of high frequencies
- JPEG is the quasi-standard for lossy image compression:
  - Colour images are transformed from RGB to YCbCr. The chroma channels C_b, C_r are subsampled.
  - The DCT is performed on 8×8 pixel blocks. Higher frequent DCT coefficients are quantised in a coarser way.
- For high compression rates, the wavelet-based JPEG 2000 compression gives better quality than JPEG.

References

- K. D. Tönnies: *Grundlagen der Bildverarbeitung*. Pearson, München, 2005.
  (Chapter 6 gives an overview on lossless and lossy compression.)
- D. A. Huffman: A method for the construction of minimum-redundancy codes. *Proceedings of the I.R.E.*, 1098–1101, Sept. 1952.
  (introduced Huffman coding)
- JPEG page of Wikipedia: `http://de.wikipedia.org/wiki/JPEG`.
  (good overview on JPEG)
- T. Strutz: *Bilddatenkompression*. Vieweg, Braunschweig, vierte Auflage, 2009.
  (excellent book on image compression including wavelets, JPEG, MPEG)
- JPEG Specification: `http://www.w3.org/Graphics/JPEG/itu-t81.pdf`.
  (gives all details of the JPEG standard)
- W. B. Pennebaker, J. L. Mitchell: *JPEG: Still Image Data Compression Standard*. Springer, New York, 1992.
  (book on the (classical) JPEG standard)
- C. Christopoulos, A. Skodras: The JPEG2000 still image coding system: An overview. *IEEE Transactions on Consumer Electronics*, Vol. 46, No. 4, pp. 1103–1127, Nov. 2000.
  (good survey on the JPEG 2000 standard).
- D. S. Taubman, M. W. Marcellin: *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer, Boston, 2002.
  (scientific background behind JPEG 2000)

Assignment C4 – Classroom Work

Problem 1 (Discrete Wavelet Transform)

Let $n = 3$ and $N = 2^3 = 8$.

(a) Write down the specific vectors

$$\begin{gathered}\mathbf{\Phi}_{3,0}, \\ \mathbf{\Psi}_{3,0}, \\ \mathbf{\Psi}_{2,0}, \mathbf{\Psi}_{2,1}, \\ \mathbf{\Psi}_{1,0}, \mathbf{\Psi}_{1,1}, \mathbf{\Psi}_{1,2}, \mathbf{\Psi}_{1,3}\end{gathered}$$

as defined in Lecture 7.

(b) Show that these eight vectors form an orthonormal basis of $\mathbb{R}^8$ with respect to the Euclidean inner product, i.e. show that

- different vectors are orthogonal,
- their Euclidian norm is 1.

Problem 2 (Information Content)

Consider the words "SHANNON" and "ENTROPY". Which one of them contains more information?

Assignment H4 – Homework

Problem 1 (Transformations) (4+1 points)

Let $\boldsymbol{f}$ be a one-dimensional, discrete signal of length N. All the discrete transformations from the Lectures 5 to 7 are linear. Thus, they can be represented by matrices. Consider the transformations

$$\boldsymbol{g} = \boldsymbol{A}\boldsymbol{f} \qquad \text{and} \qquad \boldsymbol{f} = \boldsymbol{B}\boldsymbol{g}$$

where $\boldsymbol{g}$ is the transformed signal, $\boldsymbol{A}$ is called transformation matrix, and $\boldsymbol{B}$ is the backtransformation matrix.

(a) State explicitly how the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ look like for the following transformations with $N = 8$:
   (i) discrete Fourier transform (DFT),
   (ii) discrete cosine transform (DCT),
   (iii) discrete Haar wavelet transform (DWT).
   Simplify the matrix entries such that the structure of each matrix is revealed.
   Hint: Consider the periodicities of trigonometric functions as well as complex conjugates.

(b) Besides being inverse to each other, what is the relation between $\boldsymbol{A}$ and $\boldsymbol{B}$?

(This problem will help you to see the structural similarities between the DFT, the DCT, and the DWT.)

Problem 2 (Discrete Wavelet Transform) (2+1+2+1 points)

Consider the noisy step signal

$$\boldsymbol{f} := (1, 2, 1, 3, 5, 9, 14, 12)^\top.$$

(a) Use Haar wavelets to compute the discrete wavelet transform of $\boldsymbol{f}$.

(b) Set the four wavelet coefficients with smallest absolute values to zero.

(c) Perform the backtransformation of the modified transform in (b).

(d) The operation consisting of the steps (a)–(c) can also be used for denoising. A different method is denoising by means of the Fourier Transform. One removes the coefficients corresponding to the highest frequencies in the Fourier domain and transforms the signal back afterwards. If this method is applied to $\boldsymbol{f}$, one obtains

$$\tilde{\boldsymbol{f}} \approx (2.98,\ 0.12,\ 2.05,\ 3.03,\ 4.27,\ 9.63,\ 14.20,\ 10.72)^\top$$

State the differences between $\tilde{\boldsymbol{f}}$ and the signal obtained by using wavelets.

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30
31 32
33 34
35 36

Assignment H4 (3)

Problem 3 (Huffman Coding) (5+2 points)

(a) Consider the word "SELJALANDSFOSS".
  (i) How many bits are needed to encode this word in a conventional way?
  (ii) Determine the Huffman code for this word. Is the obtained code unique?
  (iii) How many bits are needed for the Huffman coding of this word?
  (iv) How many bits would an ideal entropy coding require?

(b) Assume you are given the following code:

$$C = 10, \quad E = 111, \quad N = 001,$$
$$O = 01, \quad R = 110, \quad U = 000$$

  (i) Draw the corresponding Huffman tree.
  (ii) Decode the word 1001011010000110110111001101111

Assignment H4 (4)

Problem 4 (Discrete Cosine Transform) (2+1+2+1 points)

Please download the archive `Ex04.tar.gz` from the webpage

`http://www.mia.uni-saarland.de/Teaching/ipcv19.shtml`

into your own directory. You can unpack them with the command `tar xvzf Ex04.tar.gz`.

The program `dct.c` computes the discrete cosine transform (DCT) of an input image. Apart from writing out the logarithmically transformed DCT spectrum, it also allows to modify the DCT coefficients in different ways before the backtransform. There are six menu options in total.

(a) Supplement the routines `DCT_2d` and `IDCT_2d` for the DCT and the inverse DCT with the missing code. The program can then be compiled with the command
```
gcc -O2 -o dct dct.c -lm
```
(b) **Menu options 1 and 2:** Use the test image `boats.pgm` and compute the DCT for the whole image as well as separately for image blocks of size 8×8 (8×8 DCT). Compare the resulting spectra. What differences do you see? What can you say about the runtime?

(c) **Menu options 3 and 4:** These options are identical to the ones in (b) apart from the fact that approximately 90 % of all frequencies are removed before the backtransform. This can be seen from the percentage of DCT coefficients which are zero. Take a look at the obtained spectra. Have the high or low frequencies been removed? Compare the backtransformed images with respect to their visual quality. Does the DCT or the 8×8 DCT give better results?

M I A
1 2
3 4
5 6
7 8
9 10
11 12
13 14
15 16
17 18
19 20
21 22
23 24
25 26
27 28
29 30
31 32
33 34
35 36

Assignment H4 (5)

(d) **Menu options 5 and 6:** These options allow you to compare two different quantisation strategies for the 8×8 DCT. While the first strategy treats all frequencies equal, the second one uses a weighting matrix from JPEG before quantisation. Compare once again the obtained spectra as well as the visual quality. Which strategy yields better results?

Assignment H4 (6)

Submission

The theoretical Problems 1, 2, and 3 should be submitted in handwritten form before the lecture. For the practical Problem 4, please submit your files as follows: Rename the main directory `Ex04` to `Ex04_<your_name>` and use the command

```
tar czvf Ex04_<your_name>.tar.gz Ex04_<your_name>
```

to pack the data. The directory that you pack and submit should contain the following files:

- the source code for `dct.c` with implemented DCT and IDCT.
- the DCT spectra as well as the backtransformed images for Problem 4.
- a text file README that contains answers to the questions in Problem 4 as well as information on all people working together for this assignment.

Please make sure that only your final version of the programs and images are included. Do **not** submit any additional files, especially no executables. Submit the file via e-mail to your tutor via the address:

`ipcv-xx@mia.uni-saarland.de`

where `xx` is either `t1`, `t2`, `t3`, `t4`, `t5`, `w1`, `w2`, `w3` or `w4`

Deadline for submission: Friday, May 10, 10 am (before the lecture)