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

Recently on IPCV ...

- wavelets: signal representation that is localised in space and frequency
- in contrast to Laplacian pyramids: no redundancy
- simplest wavelet: Haar wavelet
- fast wavelet transform (FWT): similar to Laplacian pyramid
- most important application: image compression

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

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

Introduction

- Large digital images require much disk space or bandwidth. → compression
- 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
  - exploits not only redundancy, but also 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)

Lecture 8: Image Compression

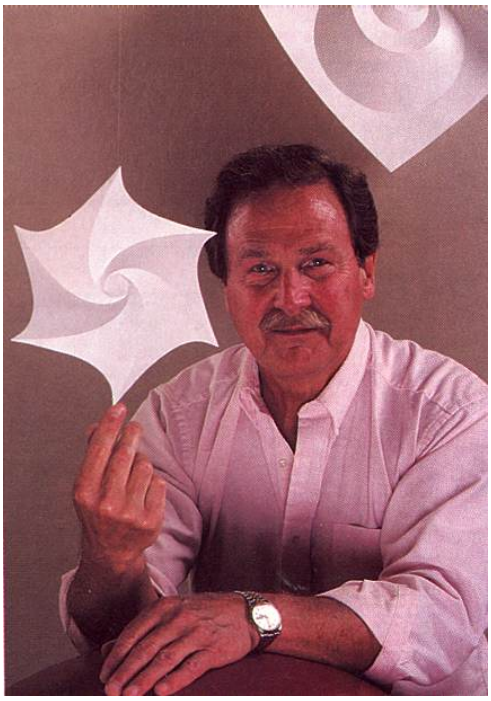
Contents

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

Huffman Coding

- lossless compression technique
- assigns shorter codes to more frequent symbols (in our case: grey values)
- widely used in many applications:
  fax machines, transmission over computer networks, high-definition television, ...
- 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

Huffman Coding (2)



David A. Huffman (1925–1999) introduced Huffman coding in a term paper as a graduate student. He worked as professor at 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: `https://www.huffmancoding.com/my-uncle/scientific-american`.

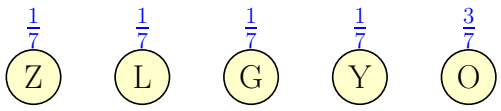
Huffman Coding (3)

Example of Huffman Coding

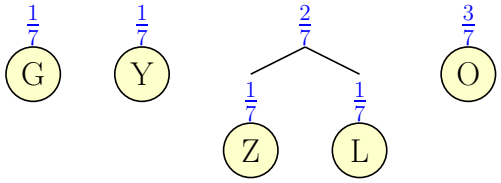
- Assume we want to encode the word ZOOLOGY.
- Its letters occur with the probabilities

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

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



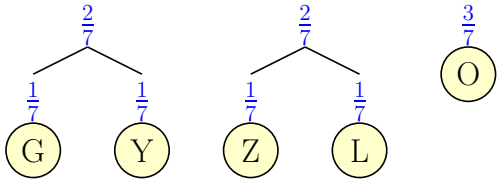
- Sum up the two with the lowest probability, and sort the result into the list again.



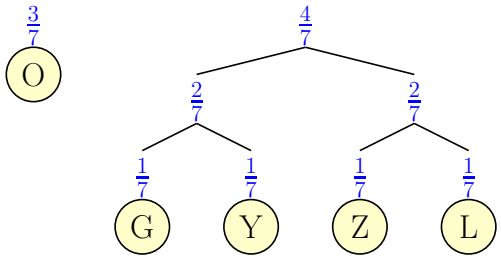
M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

Huffman Coding (4)

- Proceed as in the previous step.

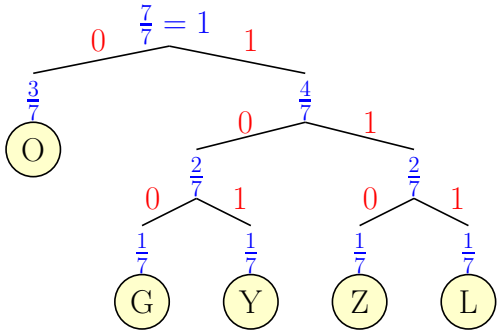


- One more time:



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.
- Thus, the word ZOOLOGY is encoded as Code(ZOOLOGY) = 110001110100101.
  This requires 15 bits.
- A conventional encoding of the five letters Z, O, L, G, Y requires 3 bits per letter.
  This results in 21 bits for the word ZOOLOGY.

Huffman Coding (6)

Huffman Decoding

- During transmission, the Huffman tree must be 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 has shown 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 average code length per letter.
  In practice, this bound is usually not reached.

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

Huffman Coding (8)

- The entropy can be seen as a measure of information content:
  A larger optimal code length 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 theoretical optimum given by the entropy.
- Approaches that aim at approximating 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 $[0, 1)$.

Outline

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

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

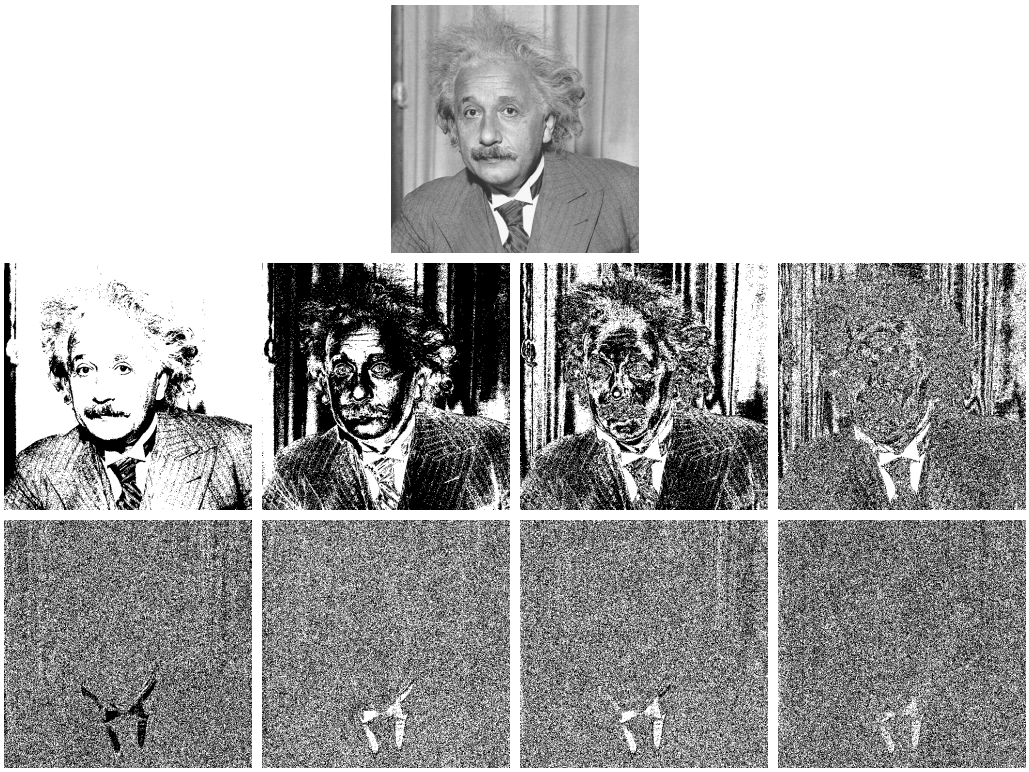
Modification for Greyscale Images

- 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.

$$\begin{aligned} 148 &= 1 \cdot 128 + 0 \cdot 64 + 0 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 0 \cdot 1 \\ &= (10010100)_2 \,. \end{aligned}$$

    The left bit 1 is called the *most significant bit,*
    and the right bit 0 is the *least significant bit.*
  - Replace the bytewise coded image by eight binary images *(bitplanes).*
    They 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:
  - The number 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.

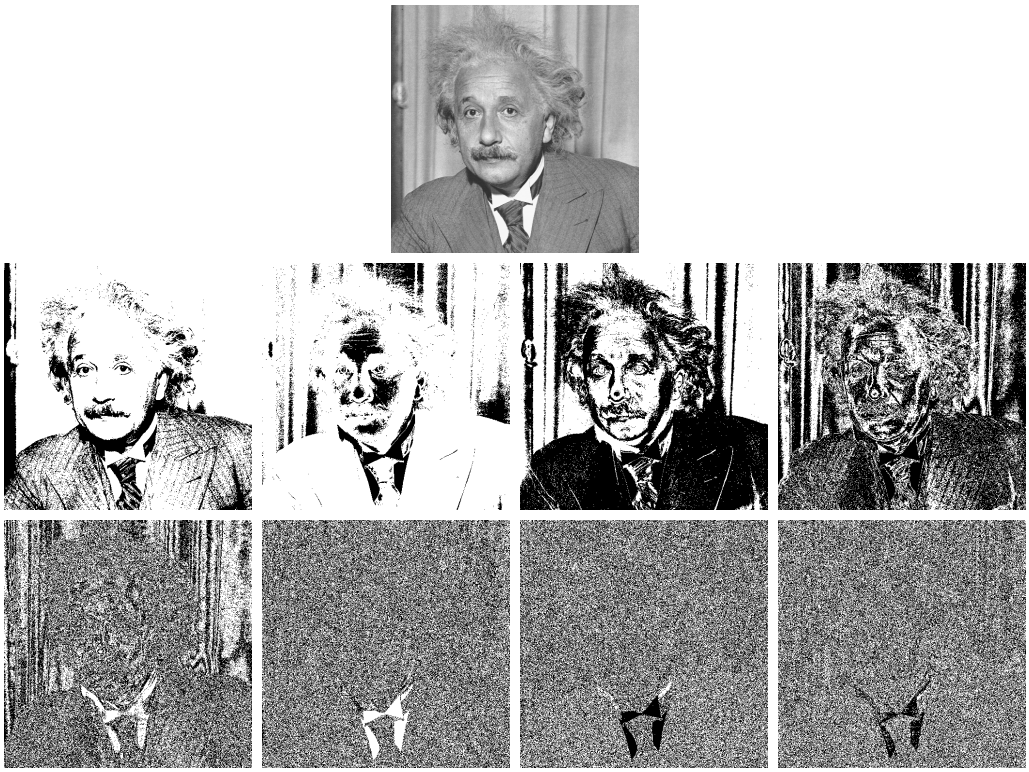
Run Length Encoding (5)

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, Gray code repeats the sequence 01 by mirroring (while conventional binary code repeats it periodically).
- For the second last digit, Gray code repeats 0011 by mirroring.
- For the third last digit, Gray code 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

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

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).
  This *codec* (coder and decoder) is often referred to as "the" JPEG mode.
- encoding of colour images consists of six main 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

The JPEG Image Compression Standard (2)

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 Chroma 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.

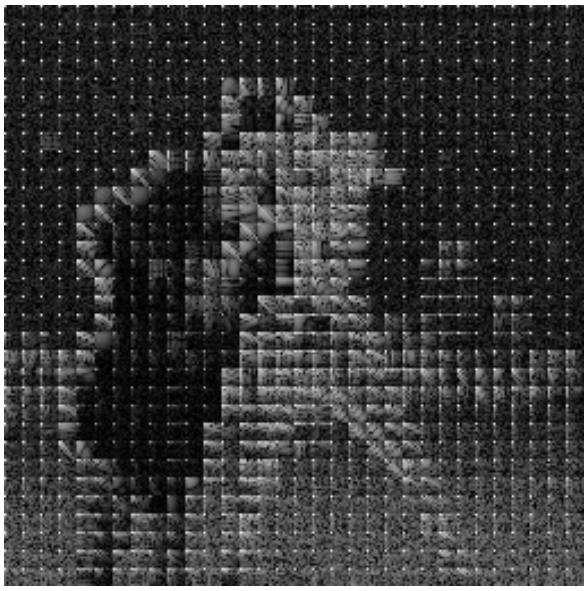
M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

The JPEG Image Compression Standard (4)

Step 3: DCT on 8×8 Pixel Blocks

- cf. Lecture 6 for the DCT
- This step is applied to all YCbCr channels.
- Decompose the image into blocks of 8×8 pixels.
  This gives reasonable homogeneity within each block and allows parallel processing.
- Compute the 64 DCT coefficients in each block.
- 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

- This is the central step for data reduction within JPEG:
  - The human visual system is not very sensitive to errors in high frequencies.
  - Thus, higher frequencies can be quantised in a coarser way.
- The quantisation interval width is specified by entries of a *quantisation matrix.* It is a 8×8 matrix that allows to quantise all 64 DCT coefficients differently.
- Example of a quantisation matrix:

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

- Divide every DCT coefficient by corresponding entry in the quantisation matrix. Round the result to the next integer.

The JPEG Image Compression Standard (7)

- 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.
  - Compute for a desired compression quality and store in the JPEG header. Different quality levels require different quantisation matrices.

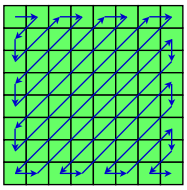
The JPEG Image Compression Standard (8)

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. Note the artefacts near edges. Authors: A. Bruhn and J. Weickert.

The JPEG Image Compression Standard (9)

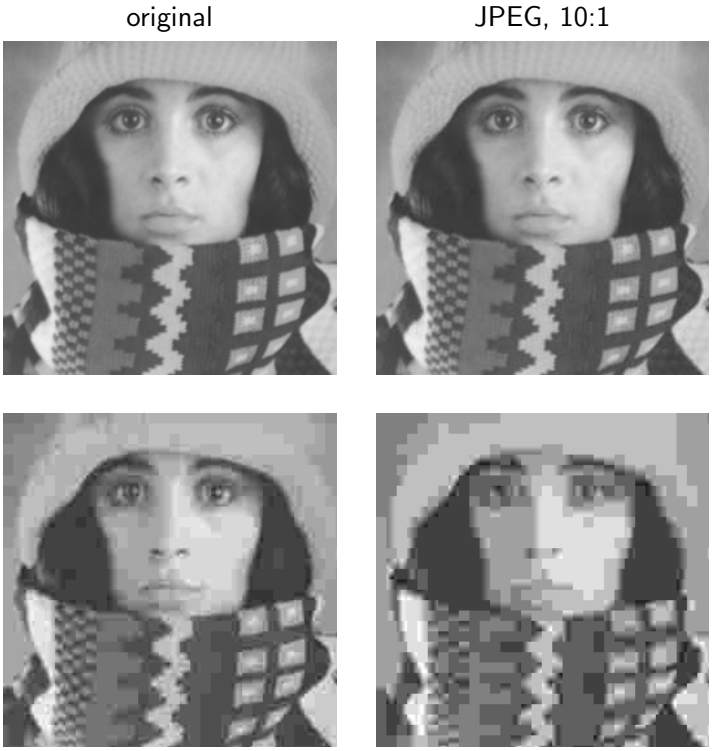
Step 5: Reordering of the DCT Coefficients

- 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.



Step 6: Entropy Coding

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



Quality deterioration under JPEG for increasing compression rates. Authors: C. Schmaltz et al.

Lecture 8: Image Compression

Contents

Alternative Codecs

JPEG 2000

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

Inpainting-based Codecs

- ongoing research, pioneered by our group
- transform-free class of codecs, i.e. no need for DCT or DWT
- store only a small, carefully optimised subset of all pixels
- reconstruct the missing data by sophisticated interpolation (inpainting)
- more details in Lecture 18 and our specialised classes:
  - Differential Equations in Image Processing and Computer Vision
  - Image Compression

Deep-Learning Codecs

- Usually a generalisation of transform-based compression.
- Learn an optimal transform for a given set of data.

original

JPEG, MSE $= 111.01$

JPEG 2000, MSE $= 70.79$

inpainting, MSE $= 42.77$

Comparison of different codecs at a compression rate of 57:1. Authors: C. Schmaltz et al.

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, JPEG 2000 or inpainting-based codecs can be better.

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: `https://en.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: `https://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)
- D. S. Taubman, M. W. Marcellin: *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer, Boston, 2002.
  (scientific background behind JPEG 2000)
- C. Schmaltz, P. Peter, M. Mainberger, F. Ebel, J. Weickert, A. Bruhn: Understanding, optimising, and extending data compression with anisotropic diffusion. *International Journal of Computer Vision*, Vol. 108, No. 3, 222–240, July 2014.
  (example of a paper on inpainting-based compression)