Lecture 8: Image Compression

Contents

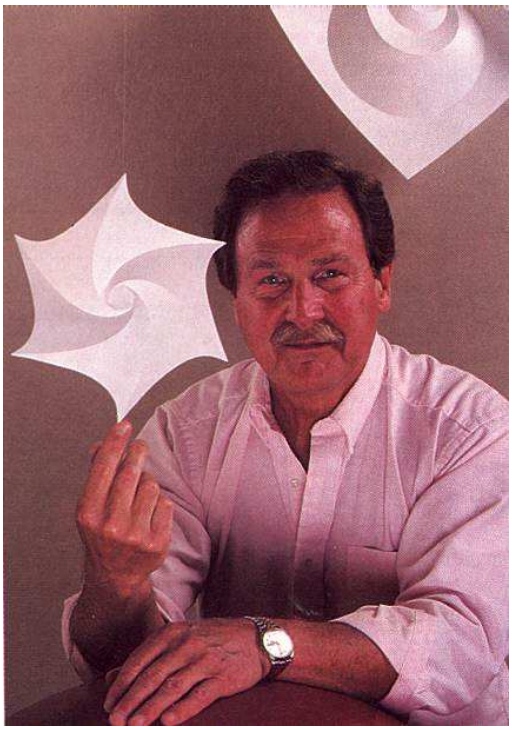
© 2006–2007 Joachim Weickert

Introduction

- Due to their size, storing images requires a lot of disk space.
- Many applications (digital photography, transmission via the internet, ...) greatly benefit from compressing images to a smaller size.
- *lossless image compression:*
  - removes redundancy, e.g. by using shorter codes for more frequent grey values or by exploiting spatial correlations
  - less efficient (compression rates up to 1:4)
  - examples: Huffman coding, run length encoding
- *lossy image compression:*
  - creates a different image that appears similar for our visual system
  - e.g. by using a coarser quantisation of high frequencies
  - high compression rates in good quality possible (beyond 1:10)
  - current standard for lossy compression: JPEG (uses discrete cosine transform)
  - emerging new standard: JPEG 2000 (uses wavelet transform)

Huffman Coding

- lossless compression technique that assigns shorter codes to more frequently used symbols (in our case: grey values).
- widely used in fax machines, modems, computer networks, and high-definition television (HDTV)
- belongs to the so-called *entropy coding* techniques using codes where the length of each codeword is proportional to the negative logarithm of its probability.
  (This is optimal from an information theoretic viewpoint.)
- greedy algorithm that generates a prefix-free code in a full binary tree structure.



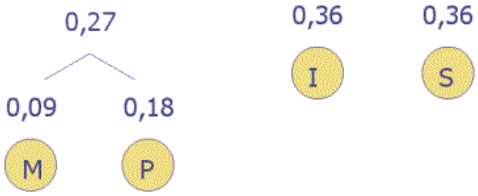
David A. Huffman (1925–1999) introduced Huffman coding in a term paper as a graduate student. He worked as professor at M.I.T and the University of California at Santa Cruz. Later on he became interested in folding paper into unusual sculptured shapes. He never tried to patent his work, and said "My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

Huffman Coding (3)

Example of Huffman Coding

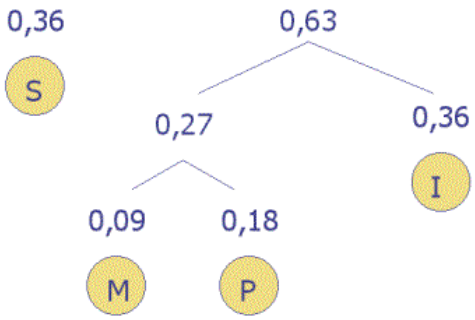
- The following example is not highly efficient, but demonstrates the principle.
  (pohlig.de/Unterricht/Inf2002/Tag44/29.2_Huffmann_Algorithmus.htm)
- Assume we want to encode the word MISSISSIPPI.
  Its letters occur with the (rounded) probabilities
  $H(M) = 0.09$, $H(P) = 0.18$, $H(I) = 0.36$, $H(S) = 0.36$.
- Order the letters according to their probability.

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

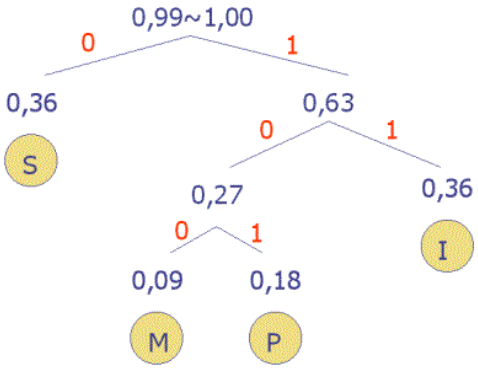


Huffman Coding (4)

- Proceed as in the previous step.



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

- This gives the following code for the letters:
  Code(M) = 100, Code(P) = 101, Code(I) = 11, Code(S) = 0.
- Hence the word MISSISSIPPI is coded as
  Code(MISSISSIPPI) = 100110011001110110111.

Huffman Decoding

- In the transmission process, the Huffman tree is transmitted in the code header.
- Proceeds through the chain. Whenever a leaf in the Huffmann tree is reached, a letter is identified.
- In our example:
  100-11-0-0-11-0-0-11-101-101-11
  yielding MISSISSIPPI.

Run Length Encoding (RLE)

Basic Idea

- simple lossless compression technique that exploits the spatial context in signals
- The grey values are written in a single array, e.g. by proceeding from top left to bottom right.
- If a grey value occurs in multiple consecutive positions, one stores the single data value and count.
- highly useful for simple images such as the binary data of fax machines

Modification

- For typical grey value images, small grey value fluctuations spoil the performance.
- Remedy: Apply RLE for each of the 8 bitplanes of a bytewise coded image *(bitplane coding)*. The "coarse" bitplanes hardly fluctuate.
- If one codes the grey values such that changes by 1 change only a single bit (gray code), bitplane coding becomes even more powerful.

The Discrete Cosine Transform (DCT)

- orthogonal transform closely related to the discrete Fourier transform (DFT, Lecture 4)
- in contrast to the DFT, the DCT allows computations with real-valued instead of complex-valued coefficients
- efficient implementations in software and hardware possible, similar to FFT
- transforms image data in a representation that is particularly useful for lossy image compression:
  - The magnitude of the coefficients decreases rapidly with increasing frequency.
  - Thus, only a few coefficients are responsible for the visual quality.
  - The others can be neglected or quantised in a coarse manner.

Definitions

- *discrete cosine transformation (DCT)* of a 1-D signal $f = (f_0, ..., f_{M-1})^\top$:

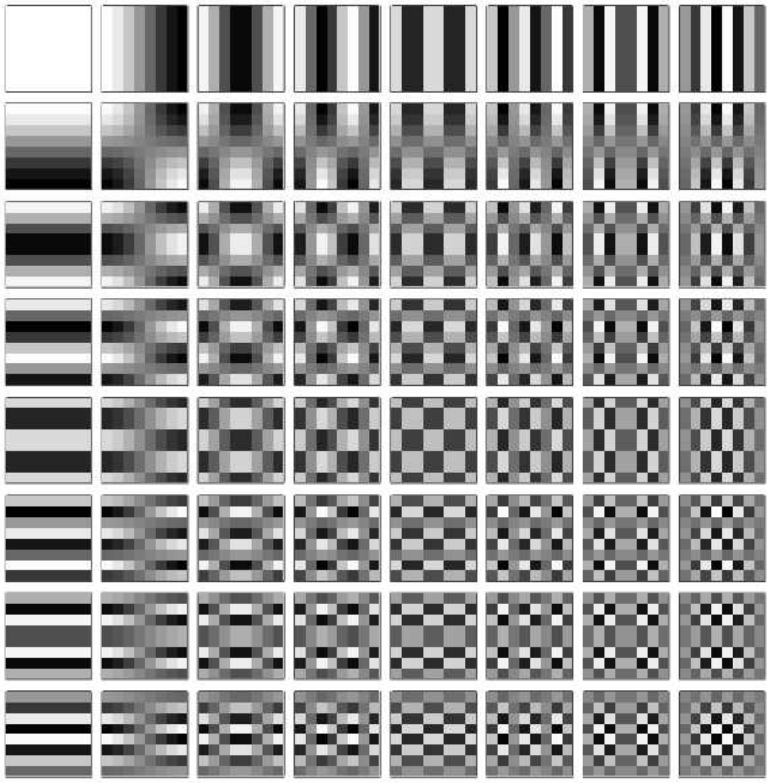
$$\tilde{f}_p := c_p \sum_{m=0}^{M-1} f_m \cos\left(\frac{(2m+1)\,p\pi}{2M}\right) \qquad (p = 0, ..., M-1)$$

  with $c_0 := \sqrt{\frac{1}{M}}$ and $c_p := \sqrt{\frac{2}{M}}$ for $p > 0$.

- *inverse discrete cosine transformation (IDCT)* of a 1-D signal $\tilde{f} = (\tilde{f}_0, ..., \tilde{f}_{M-1})^\top$:

$$f_m := \sum_{p=0}^{M-1} c_p\, \tilde{f}_p \cos\left(\frac{(2m+1)\,p\pi}{2M}\right) \qquad (m = 0, ..., M-1)$$

- In higher dimensions, the DFT is defined such that it can be separated into 1-D DCTs.



The DCT base for an image with 8×8 pixels. Each of the 64 basis images consists of cosine functions along both axes. Source: `http://oku.edu.mie-u.ac.jp/~okumura/compression/dctbase.png`.

The JPEG Image Compression Standard

- has been introduced by the **J**oint **P**hotographic **E**xperts **G**roup
- allows four compression modes, including also lossless compression
- We focus on the most widely used one: sequential DCT-based compression (lossy).
- coding of colour images consists of six steps:
  1. transformation from RGB to YCbCr
  2. subsampling of the croma channels
  3. DCT
  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)

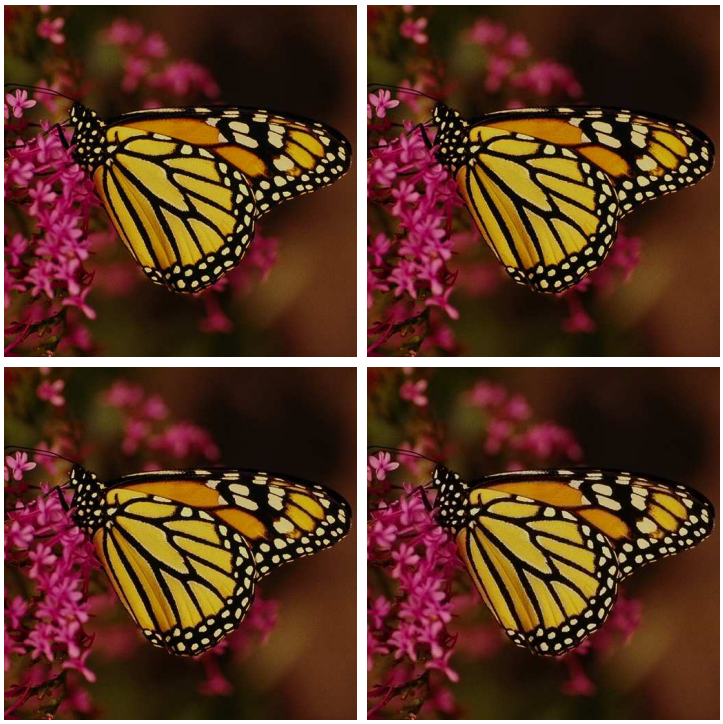
Step 1: Transformation from RGB to YCbCr

- cf. Lecture 7
- YCbCr is a colour representation that separates the luminosity Y from the croma channels Cb and Cr.
- 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 croma channels Cb and Cr than the luma channel Y.
- Subsampling the croma channels Cb and Cr by a factor 2 in each direction results in a lossy compression of good quality.

The JPEG Image Compression Standard (3)



(a) Top left: Original image, 512×512 pixels. **(b) Top right:** Reconstruction from the YCbCr space, when the channels Cb and Cr have been subsampled by a factor 2 in each direction. **(c) Bottom left:** Factor 4. **(d) Bottom right:** Factor 8. Author: J. Weickert (2007).

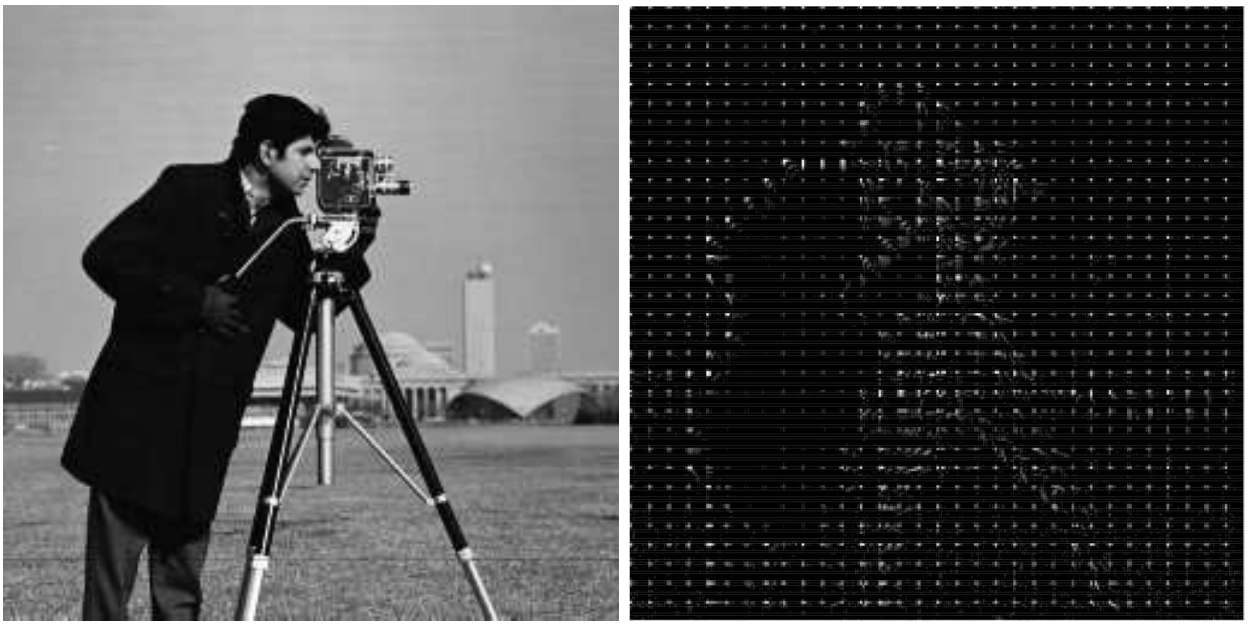
M I A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27

The JPEG Image Compression Standard (4)

Step 3: DCT

- This step is applied to all YCbCr channels.
- Shift the unsigned image values from the interval $[0, 2^b - 1]$ to $[-2^{b-1}, 2^{b-1} - 1]$ (yields a smaller first DCT coefficient).
- Decompose the image into blocks of 8×8 pixels (yields a reasonable homogeneity within each block).
- Compute the 64 DCT coefficients in each block independently.
- This step is lossless.
- However, most of the DCT coefficients are small. This forms the basis for compression.

The JPEG Image Compression Standard (5)



(a) Left: Original image, 256×256 pixels. **(b) Right:** Magnitude of the DCT coefficients computed in each block of size 8×8 pixels. Source: `http://visl.technion/ac.il`.

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

The JPEG Image Compression Standard (6)

(a) Left: Original image, 256×256 pixels. **(b) Right:** Result after performing the DCT, keeping only the 10 lowest frequencies in each 8×8 block, and applying the inverse DCT. Source: `http://visl.technion/ac.il`.

The JPEG Image Compression Standard (7)

Step 4: Quantisation of the DCT Coefficients

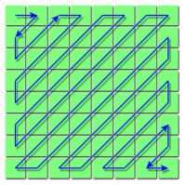
- central step for data reduction
- Divide every DCT coefficient by a corresponding entry of a 8×8 quantisation matrix, and round it to the next integer.
- The selection of the quantisation matrix is responsible for the visual quality of the compression. This matrix is computed for a desired compression quality and stored in the JPEG header.
  Higher frequencies can be quantised in a coarser way, since the eye is less sensitive to errors in high frequencies.
  Thus, the quantisation matrix has larger values for high frequencies.
- 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 JPEG Image Compression Standard (8)

Step 5: Reordering of the DCT Entries

- The quantised DCT coefficients are ordered in a zigzag way from low to high frequencies. This makes the subsequent entropy coding step more efficient, since the zeroes at the end of the chain are not coded.



- The first coefficients in each block represent the average grey values.
  Since they usually do not vary too much over the image, the differences to their neighbours are coded.

Step 6: Entropy Coding

- uses shorter codes for the more frequently occuring coefficients
- one popular possibility: Huffman coding

The JPEG Image Compression Standard (9)

(a) Top left: Original image, 256×256 pixels, 8 bits-per-pixel (bpp). **(b) Top right:** JPEG compression to 0.8 bpp (compression rate 1:10) gives rather good quality. **(c) Bottom left:** At 0.4 bpp (compression rate 1:20) visible deteriorations appear. **(d) Bottom right:** Severe block artifacts at 0.2 bpp (compression rate 1:40). Author: I. Galić (2006).

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 6)
- better quality than JPEG for high compression rates
- additional possibilities (meta data, progressive image transmission)
- not very frequently used so far due to lack of free coding software

JPEG 2000 (2)

(a) Left: Original image, 256×256 pixels, 8 bits-per-pixel (bpp). **(b) Middle:** JPEG compression to 0.2 bpp. **(c) Right:** JPEG 2000 compression to 0.2 bpp. Author: I. Galić (2006).

Summary

- Lossless image compression removes redundancy.
- Huffman coding is a lossless compression technique that assigns shorter codes to more frequent grey values.
- Runlength encoding exploits the spatial coherence of the grayvalues in a lossless way.
- JPEG is the standard for lossy image compression.
- based on the discrete cosine transform (DCT), a modification of the discrete Fourier transform
- The DCT is performed on 8×8 pixel blocks, and the resulting coefficients are quantised in a way that is adapted to the visual system.
- For high compression rates, the wavelet-based JPEG 2000 compression gives better quality than JPEG.

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

Literature

- K. D. Tönnies: *Grundlagen der Bildverarbeitung*. Pearson, München, 2005.
  (Chapter 6 gives an overview of lossless and lossy compression.)
- JPEG page of Wikipedia: `http://de.wikipedia.org/wiki/JPEG`.
  (good overview of JPEG)
- N. Ahmet, T. Natarajan, K. R. Rao: Discrete cosine transform. *IEEE Transactions on Computers*, Vol. 23, pp. 90–93, Jan. 1974.
  (made the DCT popular)
- T. Strutz: *Bilddatenkompression*. Vieweg, Braunschweig, dritte Auflage, 2005.
  (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 (old) 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 P2 – Programming Work

Please download the required files from the webpage

`http://www.mia.uni-saarland.de/Teaching/ipcv07.shtml`

into your own directory. You can unpack them with the command `tar xvzf Ex02.tgz`.

Problem 1 (Colour Spaces) (8 points)

The file `YCbCr.c` implements a programme that converts an RGB image into a YCbCr image, subsamples the chromatic channels Cb and Cr and transforms the resulting image back into RGB.

(a) Supplement the routine `RGB_to_YCbCr` with the missing code, such that it transforms an RGB into a YCbCr image. The programme can then be compiled using the command

```
gcc -O2 -o YCbCr YCbCr.c -lm
```

(b) The integer subsampling factor S allows to reduce the resolution of the chromatic channels Cb and Cr by a factor of S in each dimension. Use the test image `baboon.ppm` and compute results for $S = 1, 2, 4$ and 8. Compare the obtained results visually. What can you observe ?

(c) How many bits per pixel are required to store the YCbCr images directly for $S = 2, 4$ and 8? Please recall in this context that RGB images need 24 bits per pixel (3×8 bits).

Problem 2 (Discrete Cosine Transform / JPEG) (12 points)

The programme `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 6 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 programme can then be compiled using 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 (8x8 DCT). Compare the resulting spectra. What are your findings ? What can you say about the runtime ?
(*Since no DCT coefficients are modified the original images are obtained via the backtransform).*

(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 ?

(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 the weighting matrix from JPEG before quantisation. Compare once again the obtained spectra as well as the visual quality. Which strategy yields better results ?

Assignment P2 (3)

Submission

Please remember that up to three people from the same tutorial group can work and submit their results together. For submitting the files rename the main directory `Ex02` to `Ex02_<your_name>` and use the command

```
tar czvf Ex02_<your_name>.tgz Ex02_<your_name>
```

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

- the source code for `YCbCr.c` with implemented conversion form RGB to YCbCr.
- the results for `baboon.ppm` for $S = 2, 4$ and 8.
- the source code for `dct.c` with implemented DCT and IDCT.
- the DCT spectra as well as the backtransformed images for problem 2.
- a text file README that contains answers to all questions in the problems 1 and 2 as well as information on all people working together for this assignment.

Please make sure that only your final version of the programmes and images are included. 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`, `w1` or `w2` depending on your tutorial group.

Deadline for submission: Tuesday, November 27, 10 am (before the lecture)