Computer Science Basics
Lossy Compression

Fall Term 2025/2026

Emmanuel Benoist | BFH-TI

Lossy Compression

- **Last Week**
- **Lossy compression**
  Example
- **JPEG**
- **Videos**
- **Conclusion**

Last Week

Lossless compression

Compression is central in a lot of Med Inf topics

- Storage of large documents
- Transfer of large documents

Algorithms

- RLE is efficient if their are a lot of repetitions (Black and White images for instance).
- Huffman generates a tree. Is efficient if the type is known or if the tree remains small regarding the size of the document.
- LZW is efficient even if nothing is none.

ZIP files

- For lossless compression of files
- Uses the algorithm Deflate based on LZ77 and Huffman
- First compress the duplicate series of bytes (Lempel-Ziv)
- Then replaces commonly used symbols (Huffman).
- Very efficient with text files.

Lossless vs lossy compression

- Sometime, we do not need all of the information, a part of it is sufficient

Lossy compression

Lossy Compression

Irreversible compression

- Use inexact approximations
- Partial data discarding
- The exact data can not be regenerated

Reduce data size

- For storing
- handling
- transmitting data

Compression is much more efficient

- Lossy compressed files are much more smaller than lossless ones.

Compression of multimedia data

- Audio, Video, Images

Example

Same image: different sizes

- 31MB Without compression tiff (raw data)
- 14MB Tiff-file zipped (lossless compression using zip)
- 9MB PNG lossless compression
- 1,9MB High quality lossy compression using JPEG)
- 729kB Medium quality JPEG
- 403kB Low quality JPEG
- 86kB Low quality JPEG 2000

Example

Different sizes (JPEG high, medium and low quality)

1,9MB

729kB

403kB

Example

Different sizes (JPEG high, medium and low quality)

3,5 MB

1,1 kB

588 kB

Example: Zoom

Different sizes

JPEG High quality

JPEG medium quality

JPEG2000 Low Quality

JPEG

JPEG

Joint Photographic Experts Group Algorithm for encoding images

- Compression rate: 10:1 (with little visible quality loss)

Encoding algorithm

- Consist of different steps

Contains also metadata

- The system generating the image
- Location where it was taken
- Specific settings of the system

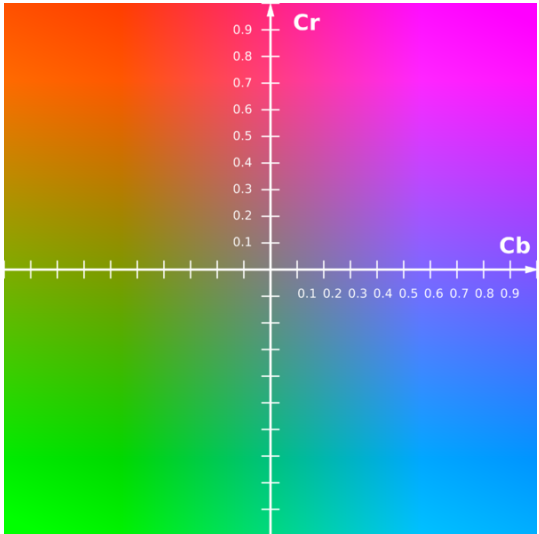
JPEG Encoding (JFIF)

Raw Data

- **Color space transformation**
- **Downsampling**
- **Block splitting**
- **Discrete cosine transform DCT**
- **Quantization**
- **Entropy coding and RLE compression**

JPEG File

Color space transformation

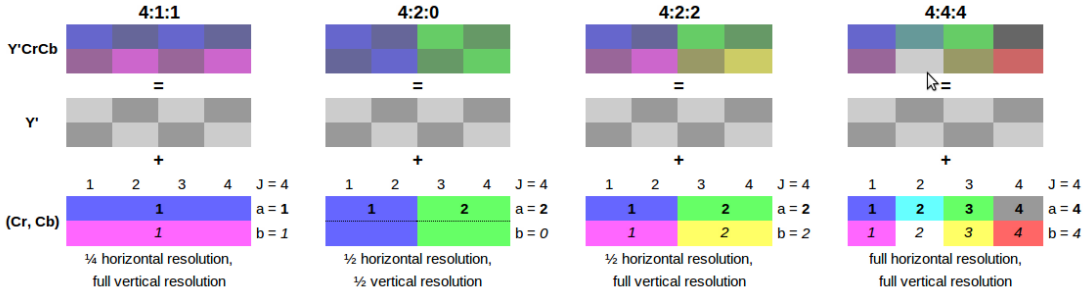


Transform RGB (Red Green Blue) image

Into Y'CrCb image

- *Luma* Y' component represents the brightness;
- *chroma* Cb and Cr components represent the color.

Downsampling



- **We can reduce the Cr and Cb images, without touching the Y' layer (Y' is more important)**
- **4 times smaller 4:1:1 or 4:2:0**
- **2 times smaller 4:2:2**
- **Size unchanged 4:4:4**

Block splitting

Create blocks of 8x8

- Complexity for larger blocks is too high

1 Block (8x8) represents:

- 64 pixels for Y'
- 64, 128, or 256 pixels for Cr and Cb

Discrete cosine transform DCT

Block is transformed using discrete cosine transform

- each block is converted to a frequency-domain representation
- A block is represented by a matrix of 64 values
- Those values can be calculated using another matrix of 64 values for cosine coefficients

Steps

- transform a byte into a value between -128 and 127
- compute the coefficient of the cosine functions that generate those values.
- The obtained matrix contains coefficients:
  - On the top-left corner, we have the DC coefficient, represents the hue for the entire block
  - on the bottom-right corner, coefficients are rather small

Discrete Cosine Transform

Initial matrix

$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}.$$

Discrete Cosine Transform

Matrix with negative values (g=Init-128)

$$
g = \begin{bmatrix}
-76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\
-65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\
-66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\
-65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\
-61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\
-49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\
-43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\
-41 & -49 & -59 & -60 & -63 & -52 & -50 & -34
\end{bmatrix}
\begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \downarrow y. \end{array}
\quad \overset{x}{\longrightarrow}
$$

Discrete Cosine Transform

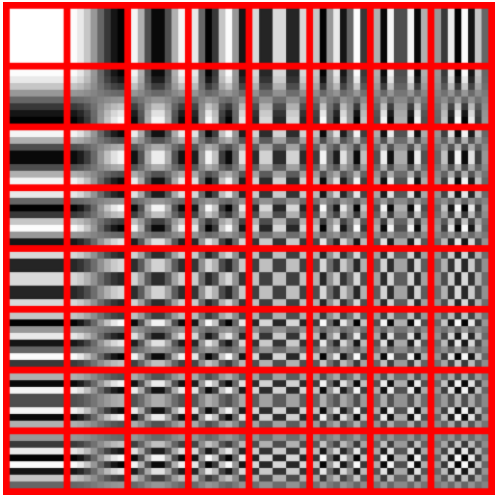
Matrix with cosine coefficients

$$
G = \begin{bmatrix}
-415.38 & -30.19 & -61.20 & 27.24 & 56.12 & -20.10 & -2.39 & 0.46 \\
4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\
-46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\
-48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\
12.12 & -6.55 & -13.20 & -3.95 & -1.87 & 1.75 & -2.79 & 3.14 \\
-7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\
-1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\
-0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68
\end{bmatrix}
\begin{matrix} \\ \\ \\ \\ \\ \\ \\ \downarrow v. \end{matrix}
$$

$u \longrightarrow$

Discrete Cosine Transform

Are the coefficients of the following functions



Quantization

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Human eye not good at distinguishing high frequency brightness variation

We reduce the information in the high frequency components

We devide each component in the matrix by a constant (also given in a matrix) and then round to the next integer.

It generates a lot of os (in the bottom-right corner)

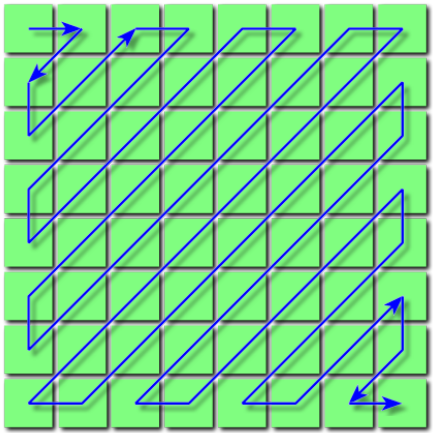
Quantization

Obtained Matrix

$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Entropy coding

The bytes are written in Zigzag



Entropy coding

$$B = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Zigzag returns:

-26,
-3, -0,
-3, -2, -6,
2, -4, 1, -3,
1, 1, 5 ,1, 2,
-1,1,-1, 2, 0, 0,
0, 0, 0, -1, -1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, ...

Run Length Encoding on 0s

Then entropic coding (Huffman for instance)

Videos

Videos compression

MPEG (Moving Picture Experts Group)

- Defined different standards. Current is MPEG-4

Based on two information

- Interframe prediction (Vectors of movement for macro blocs (16x16))
- Static image compression (DCT - Discrete Cosine Transform)

Conclusion

Conclusion

Loseless vs Lossy compression

- Loseless for programs, documents that can not be changed
- Lossy for multimedia content, where losing a small part of the information is not a problem

Image compression, JPEG

- Y'CbCr image, Downsampling of Cb Cr (because Y' is more important for the eye).
- Discrete Cosine Transform (DCT) + Quantization = most of the elements are zero
- Lose quality : impossible to reconstruct the first image.
- Compression factor is around 10:1

References

Web pages

- https://en.wikipedia.org/wiki/JPEG
- https://fr.wikipedia.org/wiki/JPEG
- https://en.wikipedia.org/wiki/YCbCr
- https://en.wikipedia.org/wiki/Lossy_compression