JPEG Compression

Human vision

Light

Electromagnetic wave

Wave lenght: 400-700nm, (1nm$=10^{-9}$m), violet-red

Two perceptor types

- Cones: 6-7 million, highly sensitive to color, located in the central portion of the retina, capture fine details, each one is connected to its own nerve end
- Rods: 75-150 million, not involved in color vision, give a general overall picture of the field of view, more sensitive to brightness

Color sensitivity of the human eye

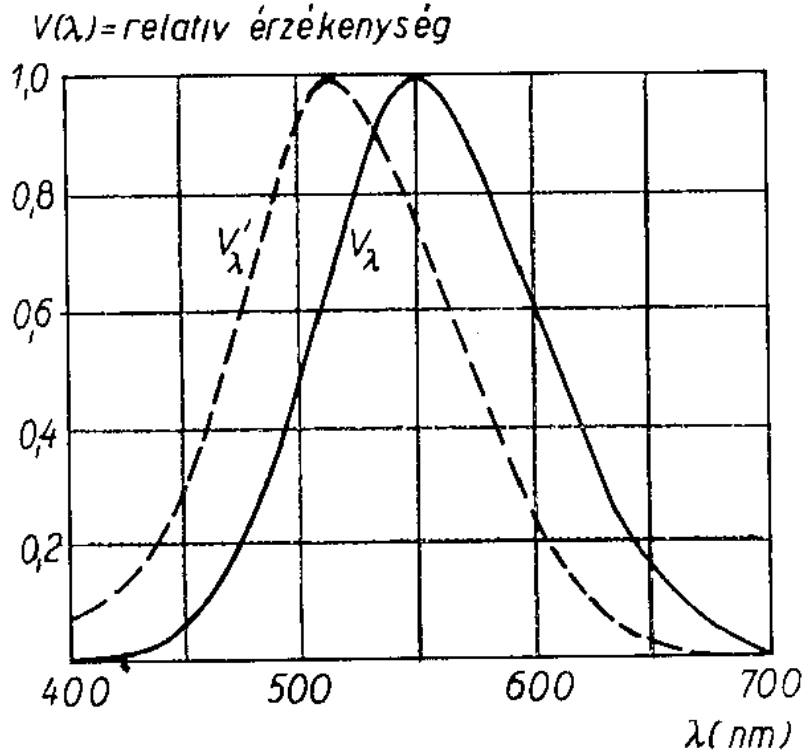


Figure: Sensitivity and wavelength

Normal light: the highest sensitivity at yellowish green ($\lambda = 555$nm).
Dim light: the highest sensitivity at blueish green ($\lambda = 510$nm)

Color sensing: Photochemical process

In cones: 3 different receptors

Most sensitive to

a) blue ($\lambda \approx 470$nm),

b) green ($\lambda \approx 560$nm),

c) red ($\lambda \approx 650$nm).

Time is needed for the receptors to recover

- Afterimage for about 0.1
- Application in film recording, 24 frame per second

Three-color theory

Young 1807, Helmholtz 1867

The RGB color space

The JPEG algorithm

JPEG: Joint Photographic Expert Group (CCIT, ISO)

Start: June 1987. First version: 1991.

Goal: compression of still images.

Expectations, requirements

a) Big compression ratio

b) Flexible, several parameters for the users to adjust

c) Good result for every continuous, natural image

d) Advanced but not to sophisticated, complicated: should work in several software, hardware configurations, environments.

The JPEG algorithm

The steps

1) Transform of color spaces: RGB$\rightarrow$YCbCr
2) Downsampling of color components
3) Taking 8×8 pixel blocks
4) Two dimensional modified discrete cosine transform for each block
5) Quantization
6) Run-lenght followed by Huffmann-encoding

Details. Step 1: transform of color spaces

The YCbCr color space: three components.

Brightness (luminance): Y. Color information in Cb, Cr (chrominance).

How to obtain Y?

By definition:

$$Y = \int_{\lambda_0}^{\lambda_1} L(\lambda) s(\lambda)\, d\lambda,$$

where $L(\lambda)$ the spectral decomposition of the light, and $s(\lambda)$ is human sensitivity as a function of wave length.

By discrete approximation: Uniform subdivision of the wavelength interval into 31 parts, each corresponds to 10nm bandwidth. Sum of 31 terms approximating the integral.

Simple model for the color components: Cb:=B-Y, Cr:=R-Y

Details. Step 1: transform of color spaces. Contd.

The reason of the color space transformation: the eye is more sensitive to brightness than to colors. They should be treated in different ways. In RGB the brightness information is hidden in the three color components.
The standard

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ Cb &= 0.168736R + 0.331264G + 0.5B + 128 \\ Cr &= 0.5R + 0.418668G + 0.081312B + 128 \end{aligned}$$

Linear transform.
The inverse transform

$$\begin{aligned} R &= Y + 1.402(Cr - 128) \\ G &= Y - 0.34414(Cb - 128) - 0.71414(Cb - 128) \\ B &= Y + 1.772(Cb - 128) \end{aligned}$$

Remark: weighted sum for Cb and Cr.

Details. Step 2: downsampling of color components

The eye is less sensitive to colors. In the color space YCbCr the color components Cb, Cr are separated from the brightness component Y. The size of the Cb, Cr matrices can be reduced

Versions of downsampling

h: horizontal, v: vertical.

- version 2h2v, creating 1 pixel, value for each 2×2 block
  total compression $\frac{1}{3} + \frac{2}{3} \cdot \frac{1}{4} = \frac{1}{2}$;
  the first term represents the Y component for which there is no compression
- version 2h1v, creating 1 pixel, value for each 1×2 block
  total compression $\frac{1}{3} + \frac{2}{3} \cdot \frac{1}{2} = \frac{2}{3}$.

Details. Step 3-4: Taking 8×8 pixel blocks, applying two dimensional modified discrete cosine transform for each block

Transform

- The size of the block: 8×8 was determined by tests. Trade of between compression and computational demand.
- Why Fourier transform: the components tend to zero. Preparation for compression.
- 8×8 two dimensional Fourier transform: linear transform, base transformation in a finite dimensional vector space. Invertible, no loss of data in this step (other then computational rounding).
- Why cosine transform: real Fourier transform.
  It corresponds to taking even extension in the 1 dimensional case. The sine transform corresponds to odd extension which causes discontinuities, and the Fourier coefficients tend slowly to 0. Not convenient for compression purpose.
- Why modified cosine transform? The consequence of discrete

Details. Step 3-4: Taking 8×8 pixel blocks, applying two dimensional modified discrete cosine transform for each block

Transform contd.

- Why modified cosine transform (MDCT)? The consequence of discrete version of even extension.

$$Tf(n,k) = \frac{1}{(2^N)^2} \sum_{j=0}^{2^N-1} \sum_{\ell=0}^{2^N-1} f(j,\ell) \cos\left(\frac{2j+n+1}{2^{N+1}}\right) \cos\left(\frac{2\ell+k+1}{2^{N+1}}\right)$$

Details. Step 5: Quantization

Dividing the 64 entries in the 8×8 MDCT matrices by the corresponding values of the quantization matrix, then rounding to nearest integer.
Two quantization matrices. One for luminance, one for chrominance.

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

Quantization matrix for the luminance component

Details. Step 5: Quantization. Contd.

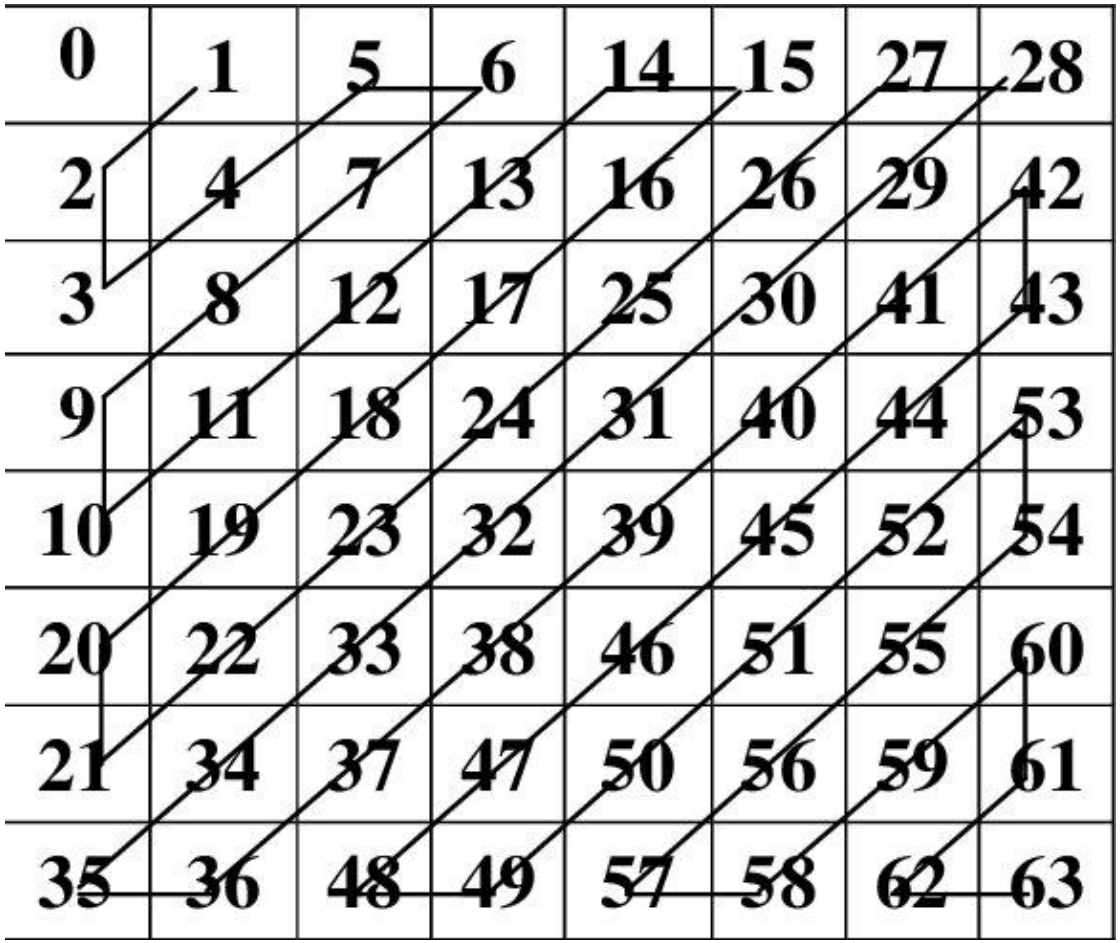
$$\begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 44 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix}$$

Quantization matrix for the chrominance components

Details. Step 6: Run length encoding, Huffman coding

Compression

- The steps 1-5: preparation for compression.
- The next preparation step for lossless encoding: zig-zag ordering. Making a 64 dimensional vector from the quantized 8×8 matrices.
- Reason: only few nonzero elements, and they are all around the upper left corner.
- Final step: Huffman coding for pairs (j, k). j : how many 0 elements followed by the actual nonzero element. k : the value of the next nonzero element. The DC component, i.e. the value in the upper left corner is coded in a different way.


0 1 5 6 14 15 27 28
2 4 7 13 16 26 29 42
3 8 12 17 25 30 41 43
9 11 18 24 31 40 44 53
10 19 23 32 39 45 52 54
20 22 33 38 46 51 55 60
21 34 37 47 50 56 59 61
35 36 48 49 57 58 62 63

Zig-zag scan