Digital Image Processing

25 January 2007

Dr. ir. Aleksandra Pizurica
Prof. Dr. Ir. Wilfried Philips

Aleksandra.Pizurica @telin.UGent.be
Tel: 09/264.3415

UNIVERSITEIT GENT
Telecommunicatie en Informatieverwerking

INSTITUUT VOOR PERMANENTE VORMING

Lossy Image Compression

A general image compression scheme

reducing spatial correlation

- Orthogonal transformation
- Predicting pixel values from the previous values

(integer) prediction errors or (real) transform coefficients

removing visually-irrelevant data

- Quantisation of the coefficients/prediction errors
- Not coding some coefficients/prediction errors

integer numbers

removing statistical redundancy

- Huffman coding
- Arithmetic coding

bit stream

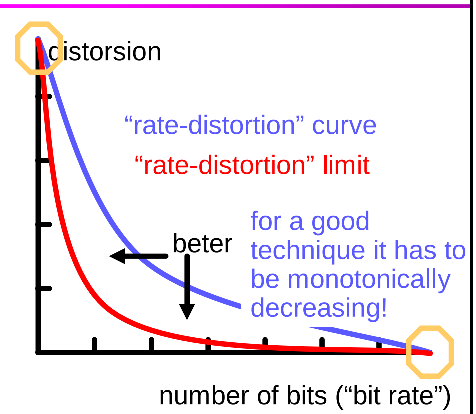
The “Rate-Distortion” (RD) curve

In lossy compression there is a trade-off between quality and the compression factor

A common numerical distortion criterion is squared error

$$\frac{1}{MN}\sum_{y=0}^{N-1}\sum_{x=0}^{M-1}\left(\tilde{f}(x,y)-f(x,y)\right)^2$$

or the PSNR, an average score given by a test panel, …

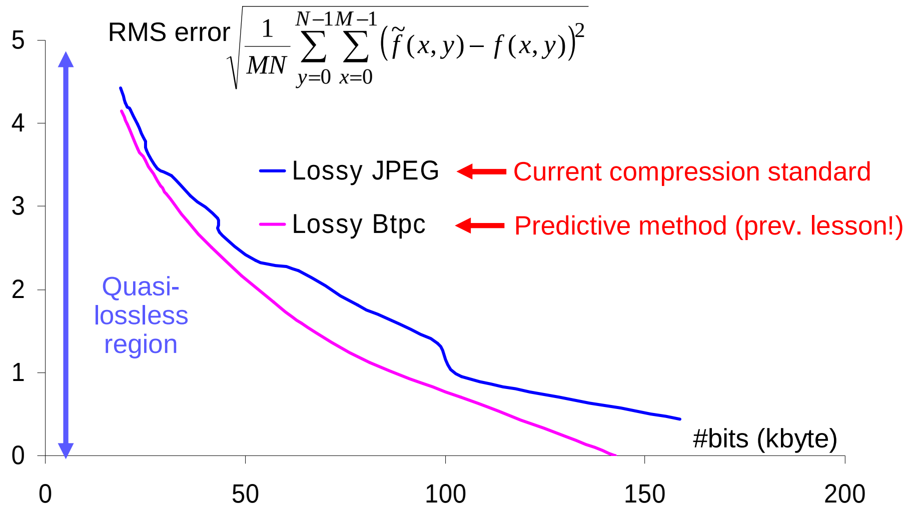


The “Rate-distortion” curve gives the corresponding values of the distrorsion and the number of bits, for different specifications of the coding parameters

Usually it is computed for a representative set of images

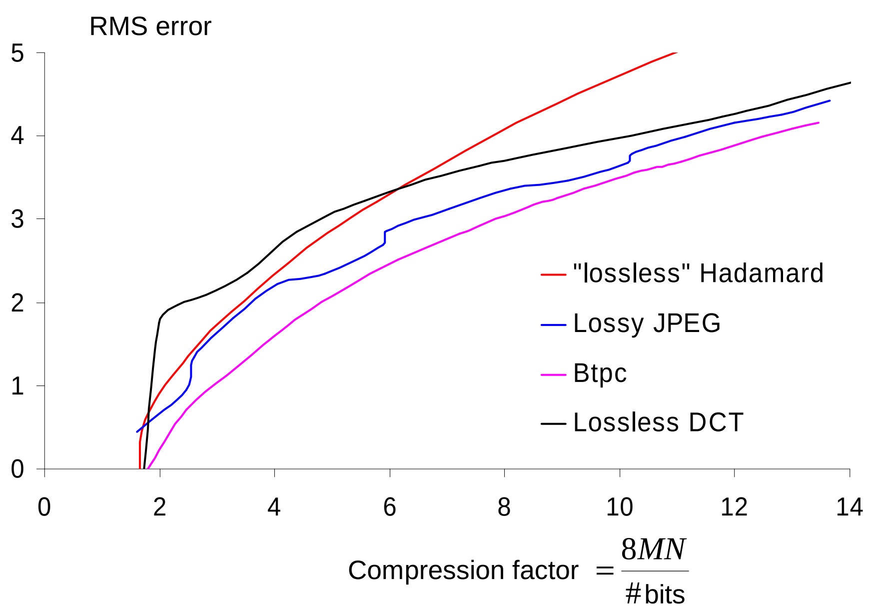
The “Rate-distortion” limit gives the smallest possible distorsion that can be achieved by any coding scheme for a given number of bits

Examples...



The current compression standard is no longer optimal in the quasi-lossless region

…Examples: compression versus distorsion



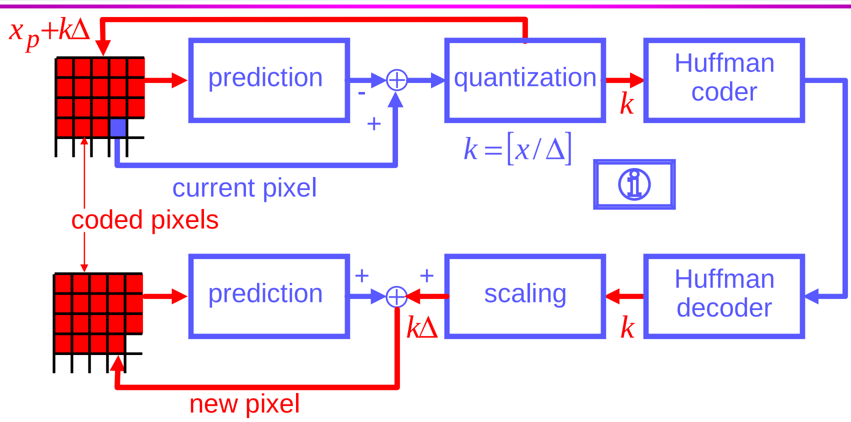
“Subjectieve” quality measures

These measures take into account the characteristics of the human eye

Example: frequency weighted mean square error

- $\sum_{k=0}^{M-1}\sum_{l=0}^{N-1}\left|F_{k,l}-\tilde{F}_{k,l}\right|^2 W_{k,l}$ ($F_{k,l}$, $\tilde{F}_{k,l}$ are 2D-DFTs)
- The weighting function $W_{k,l}$ takes into account spatial and frequency sensitivity of the eye
- $W_{k,l}$ depends on the viewing distance! ⇒ only for a fixed viewing distance!
- Interesting for optimizing coding schemes based on transform coding

Lossy prediction techniques



The coder makes prediction based on the coded pixels instead of on the basis of original pixels

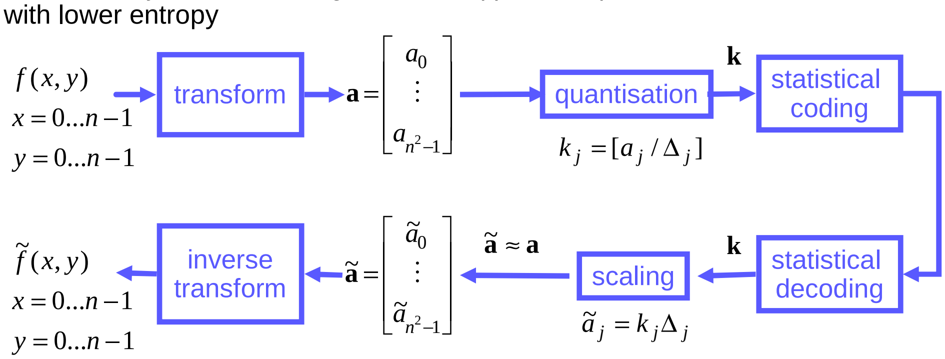
⇒ less accurate prediction in the coder
⇒ no error propagation
} ⇒ netto beter prediction

Quantization: example

Quantisation (Δ=2)			De-quantization (scaling with Δ=2)		
-2	$\rightarrow$	-1	-1	$\rightarrow$	-2
-1	$\rightarrow$	0	0	$\rightarrow$	0
0	$\rightarrow$	0	0	$\rightarrow$	0
1	$\rightarrow$	1	1	$\rightarrow$	2
2	$\rightarrow$	1	1	$\rightarrow$	2
3	$\rightarrow$	2	2	$\rightarrow$	4
4	$\rightarrow$	2	2	$\rightarrow$	4
5	$\rightarrow$	3	3	$\rightarrow$	6
6	$\rightarrow$	3	3	$\rightarrow$	6

Transform coding

Goal: linearly transform image data into (quantized) coefficients with lower entropy



Advantage of the orthogonality of the basis functions: energy preserved, squared error easy to estimate:

$$d=\sum_{x=0}^{n-1}\sum_{y=0}^{n-1}\left(\tilde{f}(x,y)-f(x,y)\right)^2=\sum_{i=0}^{n^2-1}(\tilde{a}_i-a_i)^2\leq n^2\Delta^2/4 \text{ als } \Delta_j=\Delta$$

$0\leq|\tilde{a}_j-a_j|\leq\Delta_j/2$

The best orthogonal transform

The best transform is the one that yields the best rate-distortion (RD) curve:

- For a given compression factor distortion is as small as possible
- For a given distortion the compression factor is as high as possible

The best transform in RD sense is the Karhunen-Loève transform

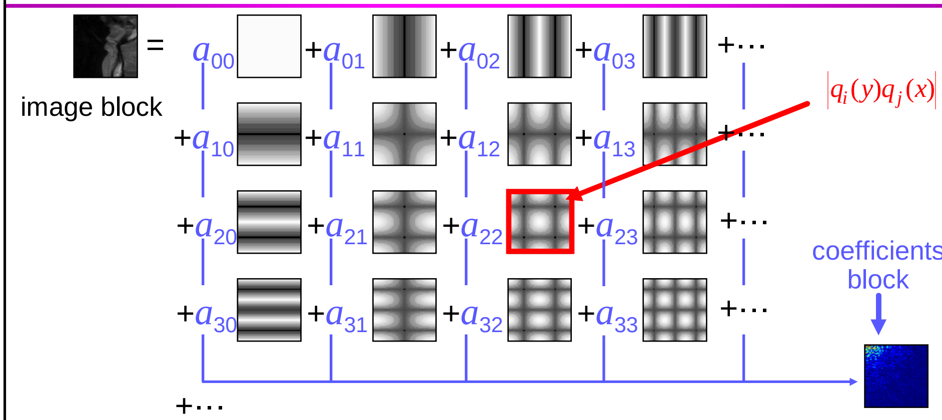
⇒This transform is image dependent and not-separable!

Based on experiments the Discrete-CosineTransform (DCT) is almost optimal

⇒ This transform is image independent
and separable (so faster to compute)!

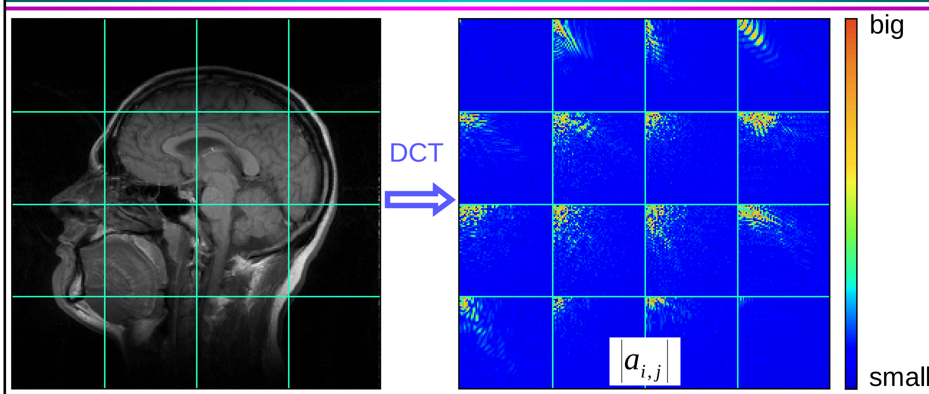
DCT based image compression

The discrete cosine transform (DCT)



Basis functions: $q_i(y)q_j(x)$; $\quad q_i(x)=\sqrt{\frac{c_i}{n}}\cos\left(\frac{i(x+1/2)\pi}{n}\right)$ with $c_i = \begin{cases} 1 \text{ for } i=0 \\ 2 \text{ for } i\neq 0 \end{cases}$

Blokcodering met de DCT



There are a lot of small coefficients; big ones are rare

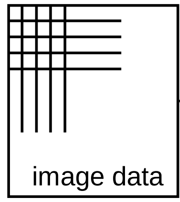
- ⇒ The histogram of the quantized coefficients is strongly non-uniform
- ⇒ideal for statistical coding

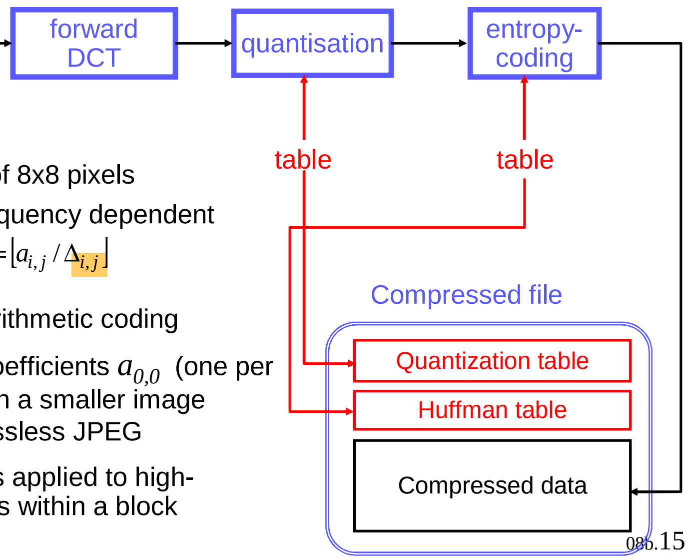
Experiments show that block sizes n=8 to n=32 yield the best rate-distortion curve

The DCT is separable and requires hence 4n computations per pixel ⇒ n=8 is thus 4 times faster than n=32 ⇒ usually n=8 is chosen

The JPEG-standard

JPEG = Joint Photographic Experts Group





Division into blocks of 8x8 pixels

Quantisation with frequency dependent accuracy : $k_{i,j} = \lfloor a_{i,j} / \Delta_{i,j} \rfloor$

Huffman coding or arithmetic coding

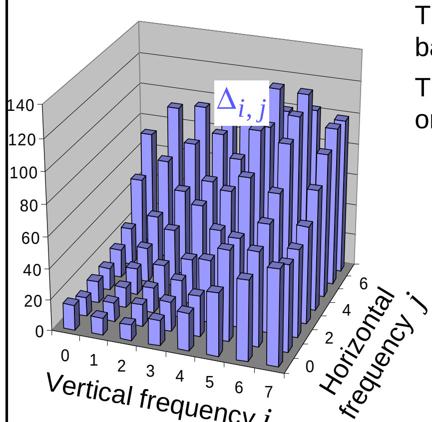
The quantized DC-coefficients $a_{0,0}$ (one per block) are gathered in a smaller image which is coded by lossless JPEG

“Runlength” coding is applied to high-frequency coefficients within a block

Psychovisually adapted quantization

The quantization step is frequency-dependent

High frequency coefficients are quantized more coarsely because eye is less sensitive to high spatial frequencies



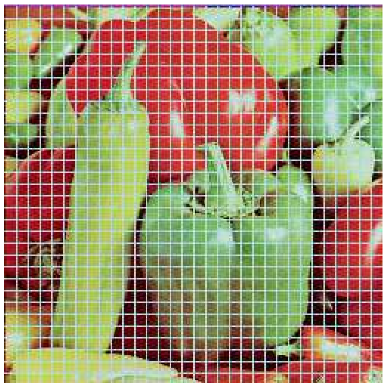
The optimal quantization table is derived based on experiments with test persons

The percieved spatial frequency depends on the viewing distance

- ⇒ this implies implicitely a fixed viewing distance
- ⇒ frequency dependent quantization is not a good option when increasing image is desired (e.g., in medical applications)

Coding of DC-coefficients

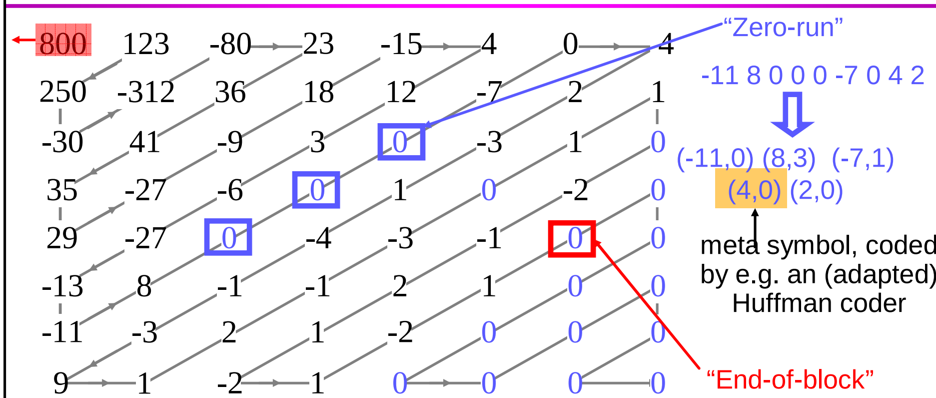
256x256 image, divided in 8x8 blocks



DC-coefficients of each block, grouped in 32x32 image (here shown 8x enlarged)

Neighboring DC-coefficients are still pretty much similar
⇒From these coefficients a DC image is formed and compressed with LJPEG

Zig-zag scan and run-length coding



The quantized HF coefficients disappear quickly with increasing frequency

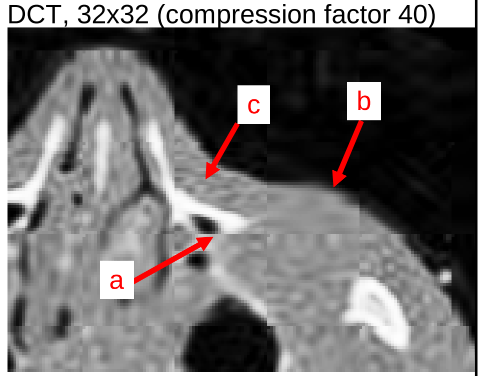
- Because of typical concentration of image energy at low frequencies
- Because of stronger quantization of the coefficients at high frequencies

To make use of this "zero-runs" and "end-of-block markers" are introduced

Errors in highly compressed images

Original

DCT, 32x32 (compression factor 40)



At the borders of the blocks the error changes discontinuously

⇒ Block distortion (a)

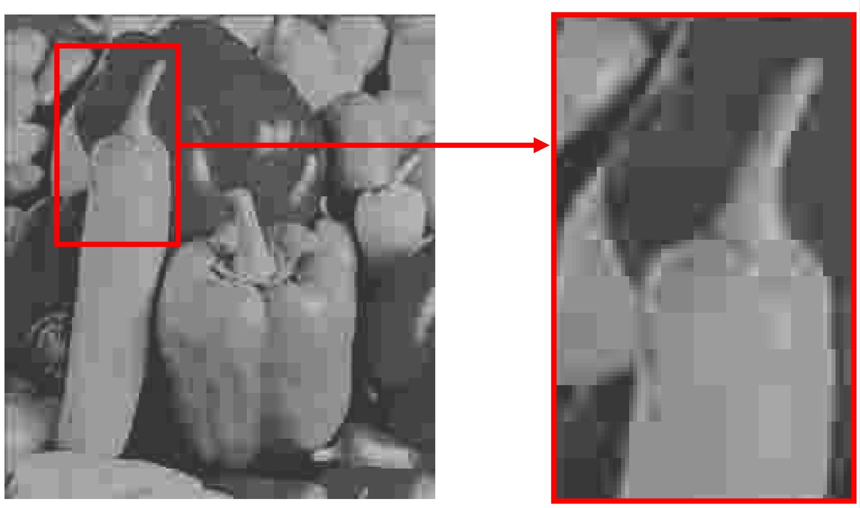
The coefficients at highest frequencies are by quantization set to zero, as if an ideal low-pass filter was applied

⇒ low-pass filtering effect: the image becomes blurred (b)

⇒ Gibbs-effect: “wrinkles” appear in the image (c)

JPEG: Problems at high compression

Block distortion: the borders of the blocks become visible



Compression factor 27 (0.3 bpp)

Subband coding and wavelet based image compression

Subband coding – wavelet coding

Principle:

- transform image using (bi-) orthogonal discrete wavelet transform (DWT)
- quantize the wavelet coefficients
- Make use of the dependencies between the wavelet coefficients using zero tree coding or significance coding
- Use a Huffman coder or an arithmetic coder to reduce the remaining statistical redundancy

The JPEG-2000 standard

- Extension of the “old” JPEG standard
- Wavelet representation + significance coding
- Most important advantages
  - Beter quality for very low and very high compression
  - Very flexible: “scalability” in resolution, quality, spatial location and color component

Scalability in quality

Time: 3.90 s

256x256x8

Total transmission time over 33.6 modem
- without compression: 20 s
- with compression: 13 s

Progressive decompression:
- very good image after 2 s
- Useful for navigation in big image sets

Remark: compression technique in this demo does not use wavelets!

	256x256x8	512x512x12
ISDN (128 kbit/s):	4 s	24 s
33.6 modem:	20 s	2.2 min.

Scalability

Scalability means that by truncating the compressed file we get an image version the quality of which improves when we truncate less

Important here is the order in which data packets are stored in the compressed file

- Scalability in accuracy: the most significant bits of all wavelet coefficiets in all subbands are stored first, then less significant bits follow
- Scalability in resolution: low and medium frequency wavelet bands are first saved with maximum quality before taking high frequency coefficients in the reconstruction
- Scalability in spatial location: all coefficients of a “region of interest” (e.g. the central part of the image) are saved first
- Scalability in color: the coefficients of the luminance component are saved first and the chrominance components follow after

Scalability: remarks

It is possible to switch from one type of scalability to another during compression: e.g. first resolution then quality

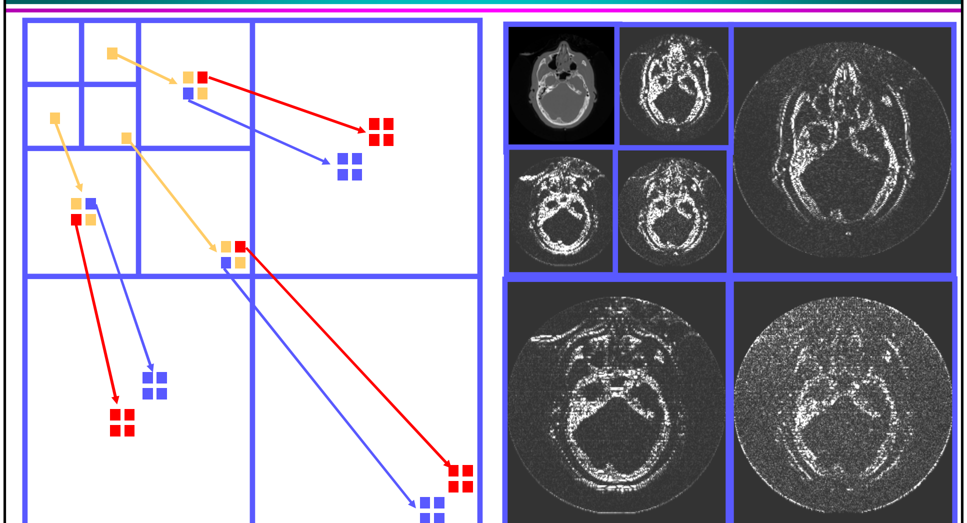
The JPEG-2000 data stream consists of packets; in the header of these packets it is noted which information they contain

⇒By reordering the packets it is possible to get a different type of scalability without decoding and coding again

Applications of scalability:

- Progressive transmission over channels with low bandwidth: while the bits are ariving the decoder calculates better and better versions of the compressed image
- When printing on a greyscale printer the chrominance packets can be omitted
- Generating “thumbnails” on a web page by truncating high quality images

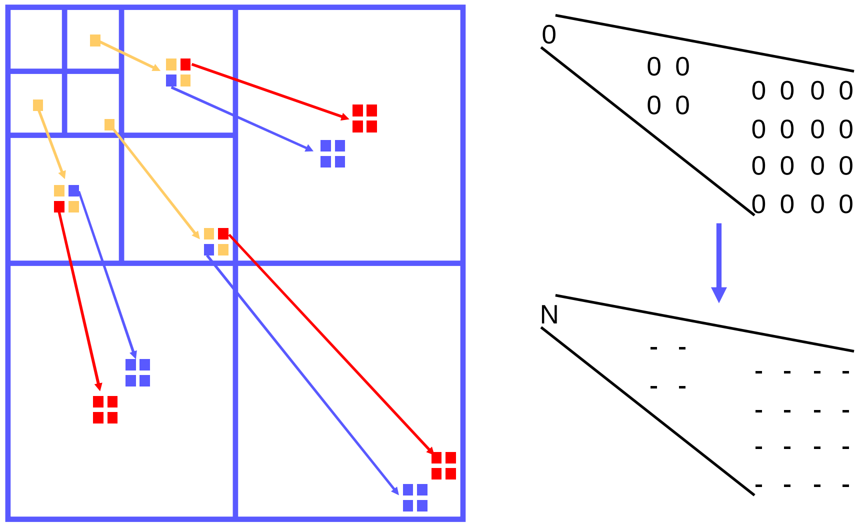
Zero-tree coding...



The coefficients in a tree

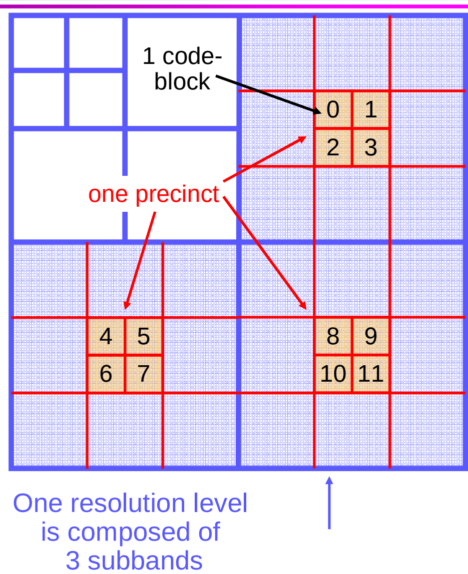
- correspond with the same position in the image
- And are result of the same type of filtering applied to a low-pass (and subsampled) version of the image

... Zero-tree coding



The “children” of a coefficient with value zero are often also zero
⇒zero trees are replaced by a single symbol

Significance coding (JPEG2000) …



Each resolution level is divided in “precincts” (regions):

- each region contains the coef-ficients from one spatial surrounding
- and from all 3 subbands of the resolution level

Each region is split in a similar way into code blocks

Code blocks are coded independently of one another

- method: context dependent encoding of “bit planes”, starting with the most significant
- bit plane *k* = image formed by *k*-th bit of the absolute values of each coefficient in a code book

… Significance coding (JPEG2000)

1	0	1	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	0	0	1	0
1	1	0	1	0	0	0	0…
1	1	1	1	1	0	1	0
0	1	1	1	0	1	1	0
0	0	1	0	1	0	0	0
			…				

Each bit plane is subdivided into stripes of height 4

Order of arithmetic coding

- Most significant bit planes first
- Inside of the bit plane: stripe per stripe
- Inside a stripe: column per column

Sign bit is separately encoded and not described as bit plane

The compressed data are grouped into layers and packets:

- a packet consists of a (variable) number of bit planes of a number of code blocks from the same one region

  ⇒ contains data of a given spatial location and resolution
- a layer consists of a packet of each region at each resolution level; a layer does not necessarily contain all the bit planes

  ⇒ contains data that increase the quality of the whole image

Bit plane coding

Each bit plane is separately encoded, but coding is adaptive and makes use of context information of more significant bit planes

Definition: a given wavelet coefficient becomes "significant" (in the sense of "significance coding") when at least one of its already coded bits is 1

The coding of a bit plane conists of 3 steps:

- step 1, significance propagation: coding the bits with value 2^k of the coefficients that are not significant yet, but have at least one neighbor that is already significant
- step 2, magnitude refinement : coding of the bits of the coefficients that became significant in the previous bit plane
- step 3, cleanup pass : coding all the remaining bits

Remark: when the coefficient become significant for the first time, its sign is coded immediately

Bit plane coding: example

Step	Coefficients: 10	1	3	-7	
3. clean up	1+	0	0	0	← bit plane 3: value 2^3
1. significance		0			← bit plane 2: value e 2^2
2. refinement	0				
3. clean up			0	1-	
1. significance		0	1+		← bit plane 1: value 2^1
2. refinement	1			1	
3. clean up					← no remaining coefficients ⇒ no work for clean up
1. significance		1+			
2. refinement	0		1	1	← bit plane 0: value 2^0
3. clean up					

sign bit

10=+1010 -7=-0111

Remark: for the first bit plane the steps 1 and 2 are not performed; there are no significant coefficients yet at that point

Philosophy behind bit plane coding

Philosophy of bit plane coding: "sorting" the bits for an optimal adaptive operation of the arithmetic coder

- in step 1 , "1"-bits are very likely: significant coefficients = large coefficients, and these are spatially grouped
- in step 2, "1"-bits are equally likely as "0"-bits
- in step 3, "0"-bits are likely, typically the smallest coefficients coded in this step

⇒In each of the 3 steps we have a different statistics

⇒ Arithmetic coder works better when encoding bits of a given type after one another instead of mixing all the types

Remarks

Not discussed aspects

- Run-legth coding is also used in some steps
- The image is first divided into tiles; these are subimages that are coded independently of one another ⇒ extra high level scalability

Why significance coding and not zero-tree coding in JPEG 2000?

- In zero-tree coding the subbands are jointly coded
  - ⇒ less possibilities for scalability
  - ⇒ and more complex implementation!

Wavelet coding versus JPEG...


JPEG 2000, factor 40

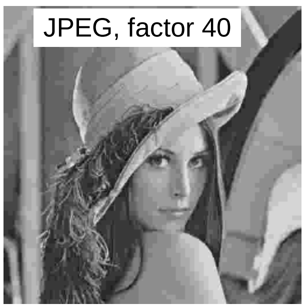

JPEG, factor 40

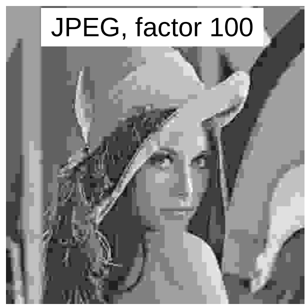

SPIHT, factor 40

SPIHT= Set Partitioning in Hierarchical Trees = wavelets+zero tree coding
SPIHT is much better than JPEG at low bit rates
The new JPEG-standard “JPEG-2000” contains a wavelet algorithm

Original: 256 kbyte, 8 bit/pixel
Compressed: 6500 byte, 0.2 bit/pixel

…Wavelet coding versus JPEG...

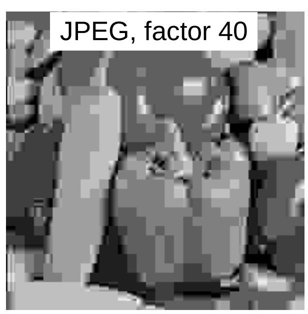




JPEG, factor 200

cannot be realised

... Wavelet coding versus JPEG



Original: 64 kbyte, 8 bit/pixel

Compressed: 1600 byte, 0.2 bit/pixel

SPIHT and JPEG-2000 never cause block distrortion

Small images compress obviously worse than big images

⇒it is not the resolution of the image but its content that that finally determines the number of bits for a given quality

Bibliography

Technical overview of JPEG-2000

- http://www.rii.ricoh.com/%7Egormish/pdf/dcc2000_jpeg2000_note.pdf

JPEG-2000 software implementation

- http://www.ece.uvic.ca/~mdadams/jasper/

(JasPer Project - an open-source initiative to provide a free software-based reference implementation of the JPEG-2000 codec).

Commands

Conversion to jpeg-2000

- factor 40: **jasper --input lena512x512.bmp --output-format jp2 --output lena.jp2 -O rate=0.025**
- factor 100: **jasper --input lena512x512.bmp --output-format jp2 --output lena.jp2 -O rate=0.01**

Conversion from jpeg-2000

- **jasper --input lena.jp2 --output-format pnm --output lena.pnm**

Conversion to jpeg

- factor 40: **convert -quality 10 lena512x512.bmp lena.jpg**
- factor 100: **convert -quality 3 lena512x512.bmp lena.jpg**