Multimedia storage

Why do we need compression?

- Crucial for a number of multimedia applications
  - Efficient transmission over communication channel
  - Efficient storage on storage devices

Motivation

- Image size
  - Color 12 megapixel image 4000×3000px = ~35Mb
  - Channel speed 1Mbit/s = transfer time: ~4,6 min
- Audio size
  - 3 minute song, 44100 16-bit samples per second ~= 15Mb
  - Channel speed 1Mbit/s = transfer time: ~2 min
- Video size
  - 1920 x 1080 color video, 25 FPS, 120 minutes = ~ 1Tb
  - Channel speed 1Mbit/s = transfer time: ~80 days

Compression of sensory data

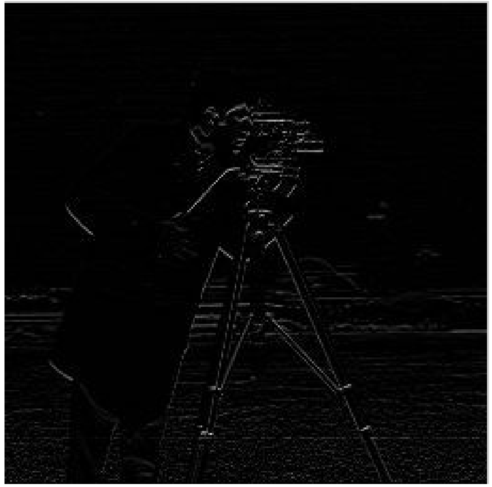
- Redundancy of sensory data
  - Correlation across space and time
  - Human perceptual sensitivity
- Lossless compression
  - Minimize size without loosing information
- Lossy compression
  - Minimize size with controlled loss of information
  - Discard data that is not noticed by a human

Spatial correlation

Neighborhood pixels are frequently similar (image, video)

$I(x,y)$

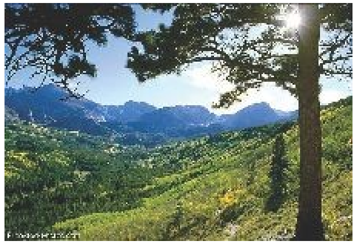
$I(x,y) - I(x,y+1)$

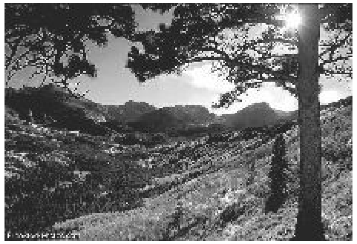


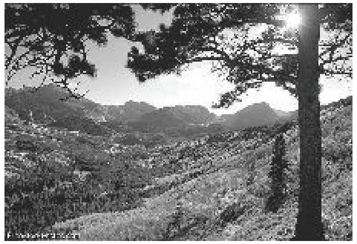
Spectrum correlation

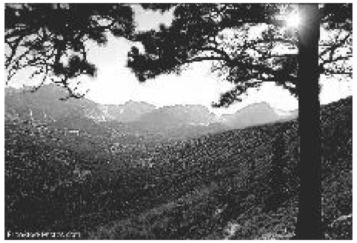
Correlation across spectrum

(Similar intensity of colors in colors in all RGB channels)




Red


Green


Blue

Temporal correlation

Two frames in a video stream are very similar

$I_t(x, y)$

Frame t

$I_{t+1}(x, y)$

Frame t+1

$\Delta_t = I_t(x, y) - I_{t+1}(x, y)$

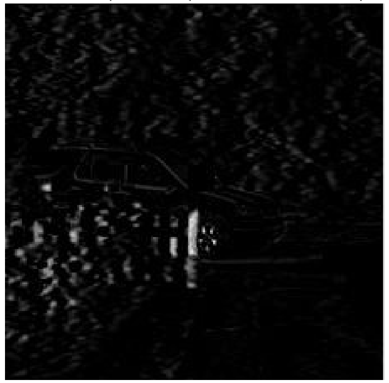
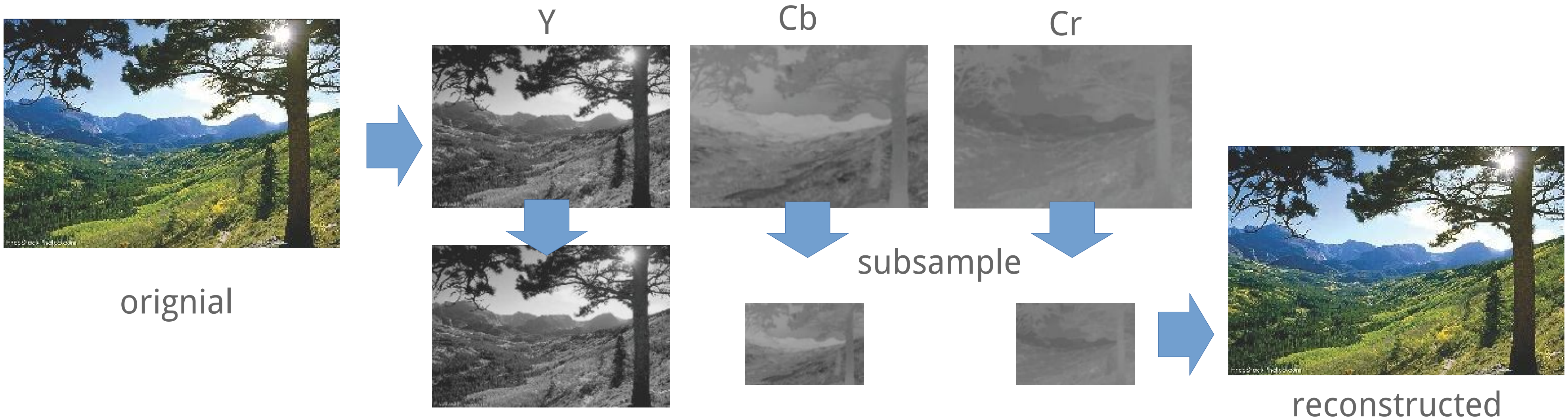


Image difference

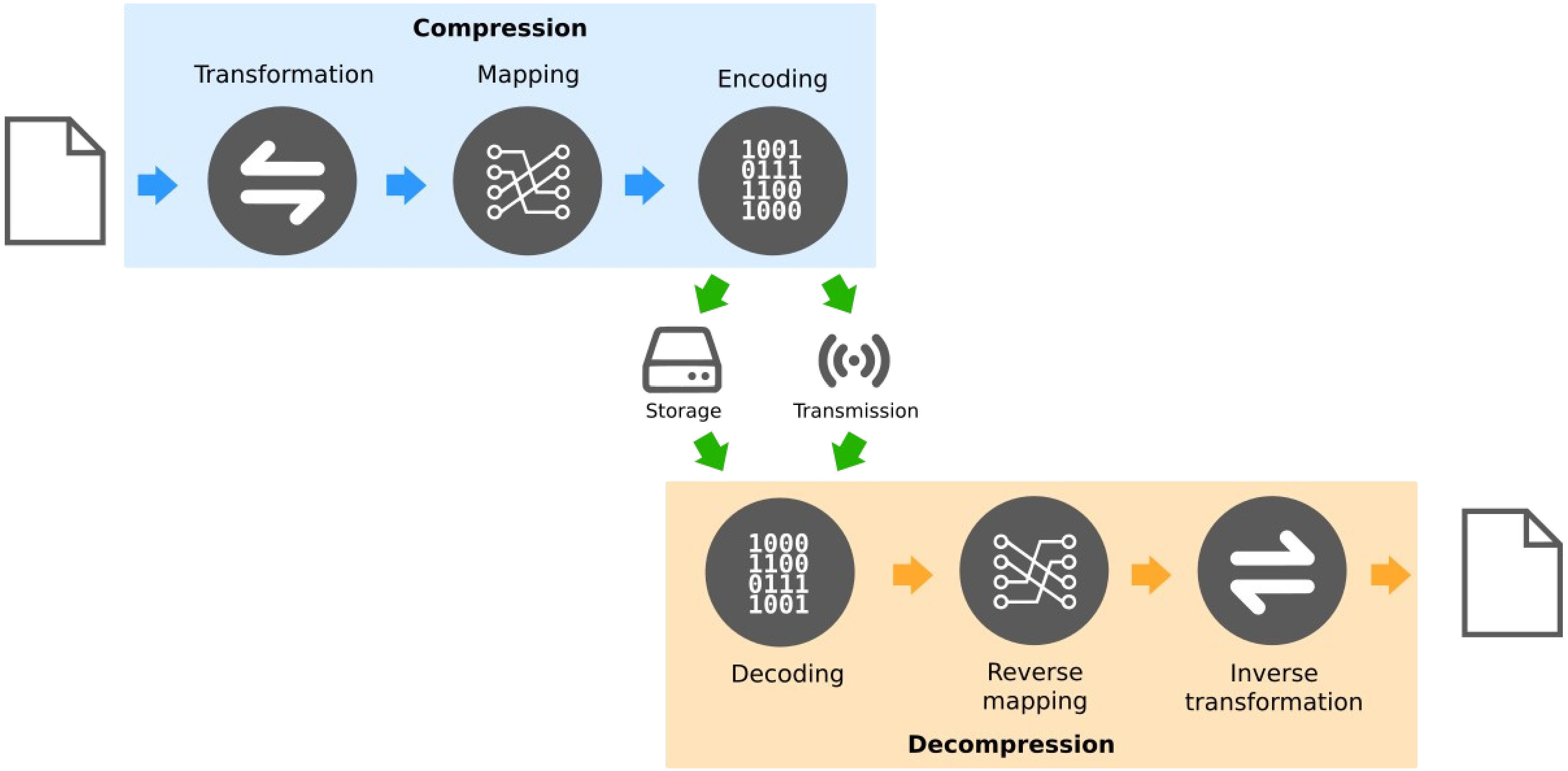
Perceptual sensitivity

Human perceptual system is differently sensitive to different types of information

(e.g. human vision is more sensitive to changes in intensity than in chroma)



Compression process



Lossless vs. lossy

- Lossless compression
  - Decoded information is equal to the original image (numerically/quantitatively and perceptually/qualitatively)
- Lossy compression
  - Decoded information should be perceptually equal to the original image (to a required level)
  - Achieves higher compression levels, removes redundant information
- Choice is application dependent
  - Biomedical signals have to be stored without loss
  - Entertainment industry can save space and bandwidth with lossy compression

Transforming signal/data

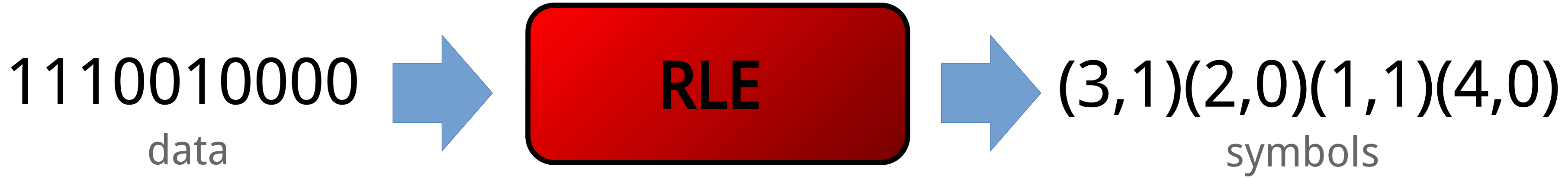
- Mapping signal/data to a form that is easier to encode
    - Reducing correlation in data
    - Reducing redundancy
    - Change statistical properties
- Typical transformations
    - Differential (predictive) coding
    - Bijective transformation to frequency space (e.g. DCT)
    - Convert colors from RGB to YCbCr (separate intensity and chroma)

From data to symbols

- Data to symbols for efficient encoding
- Split signal to smaller chunks (partitioning)
  - Split image into blocks – each block is a symbol
  - Encode each block separately
- Code sequences, repetitions (run-length coding)
  - Absolute correlation in subsequences of data
  - Blocks are sequences of repeated data

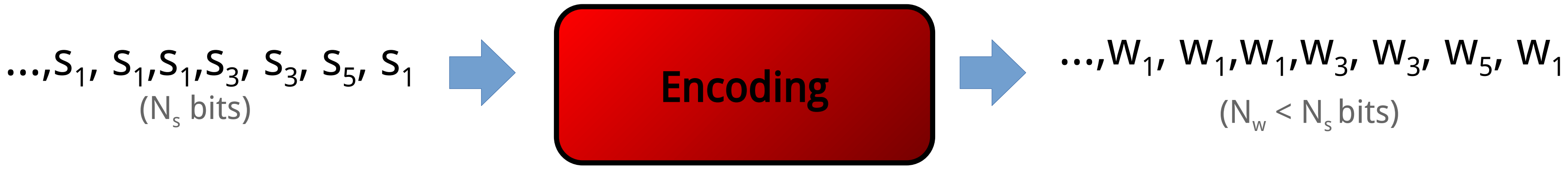
Run-length encoding - RLE

- Sequence of same numbers write as symbol (count, value)
  - Count – how many times the symbol is repeated
  - Value – the value of single symbol
- Sequence of data is transformed into sequence of symbols



Lossless encoding

- Transformation and mapping to symbols are preprocessing
- Most lossless data compression occurs in symbol encoding
  - Binary stream where each symbols is assigned a code word
  - Code dictionaries can be formed on-line or off-line
  - Entropy encoders minimize expected number of bytes to represent sequences of symbols


$\ldots, s_1, s_1, s_1, s_3, s_3, s_5, s_1$
(N_s bits)
Encoding
$\ldots, w_1, w_1, w_1, w_3, w_3, w_5, w_1$
($N_w < N_s$ bits)

Lossless symbol coding

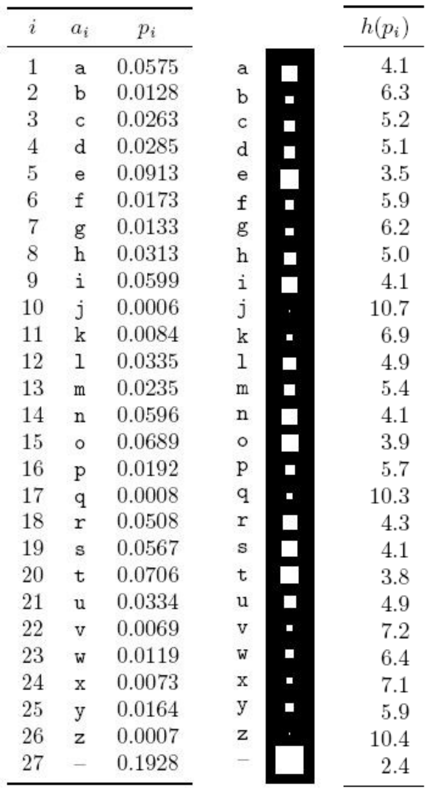
- Minimize size (number of bits) without losing information
- **Statistical methods** (Huffman, arithmetic)
  - More frequent symbols are assigned shorter code words (variable length coding - VLC)
  - Statistics of symbol occurrence in data stream can be fixed or computed on-the-fly
- **Dictionary methods** (LZ77, LZW, LZMA)
  - No statistics about symbol frequency
  - Dynamically generate code tables (dictionary) to encode sequences of various length
  - Store dictionary and references

Information entropy

- How many bits required for each symbol
  - 4 bit – 16 symbols
  - 5 bit – 32 symbols
- Information entropy measures how many bits are needed to encode a single character based on probability of its appearance in a sequence

$$h(p_i) = -log_2(p_i)$$

$$H(X) = -\sum_{i=1}^{N} p_i \ \log_2 p_i$$



i	a_i	p_i	$h(p_i)$
1	a	0.0575	4.1
2	b	0.0128	6.3
3	c	0.0263	5.2
4	d	0.0285	5.1
5	e	0.0913	3.5
6	f	0.0173	5.9
7	g	0.0133	6.2
8	h	0.0313	5.0
9	i	0.0599	4.1
10	j	0.0006	10.7
11	k	0.0084	6.9
12	l	0.0335	4.9
13	m	0.0235	5.4
14	n	0.0596	4.1
15	o	0.0689	3.9
16	p	0.0192	5.7
17	q	0.0008	10.3
18	r	0.0508	4.3
19	s	0.0567	4.1
20	t	0.0706	3.8
21	u	0.0334	4.9
22	v	0.0069	7.2
23	w	0.0119	6.4
24	x	0.0073	7.1
25	y	0.0164	5.9
26	z	0.0007	10.4
27	–	0.1928	2.4

$$H(X) = -log_2(1/27) = 4.75 bit/ch$$

$$H(X) = -\sum_{i=1}^{N} p_i log_2(p_i) = 4.1 bit/ch$$

Entropy coding

- Known set of symbols S of size N, each of them has a probability of occurring in stream $S = \{(s_1, p_1), (s_2, p_2), \cdots, (s_N, p_N)\}$
- Average length of word $\overline{w} = \sum_{i=1}^{N} p_i |w_i|$
- Determine code words that minimize the average length of word and satisfy requirements:
  - Regular – different symbols are assigned different words
  - Unique – message can be understood in only one way
  - Instantaneous – each word can be decoded as soon as it is read

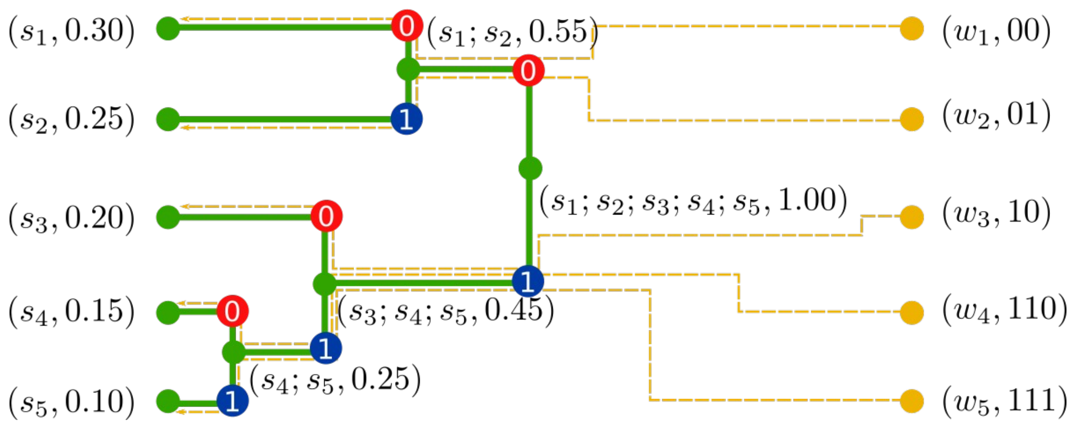
Huffman encoding/decoding

- Type of optimal prefix coding
- Encoding
  - Establish symbol statistics
  - Generate word codes
  - Map symbols to words
- Decoding
  - Binary search tree
  - Lookup table

Determining words by Huffman coding

- Sort symbols by decreasing probability of occurrence
- Merge two symbols with lowest probability and combine their probabilities
- Repeat until all symbols have words assigned

$S = \{(s_1, 0.30), (s_2, 0.25), (s_3, 0.20), (s_4, 0.15), (s_5, 0.10)\}$



Naive encoding (3bit codes) – 15 bit

$s_1, s_3, s_1, s_2, s_2 \rightarrow 010001010011011$

Huffman encoding – 10 bit

$w_1, w_3, w_1, w_2, w_2 \rightarrow 0010000101$

Huffman algorithm properties

- Each symbols separately requires a non-zero integer number of bits
  - Lower bound 1bit/symbol
  - Optimal when encoding each symbol separately
- Encoding based on statistical properties of the data-stream
  - Deviation from the model makes compression sub-optimal
  - On-line updates (adaptive) are computationally expensive
- Improved algorithms
  - Limit longest code words
  - Avoid one-symbol one-word mapping (arithmetic coding)

Dictionary coding

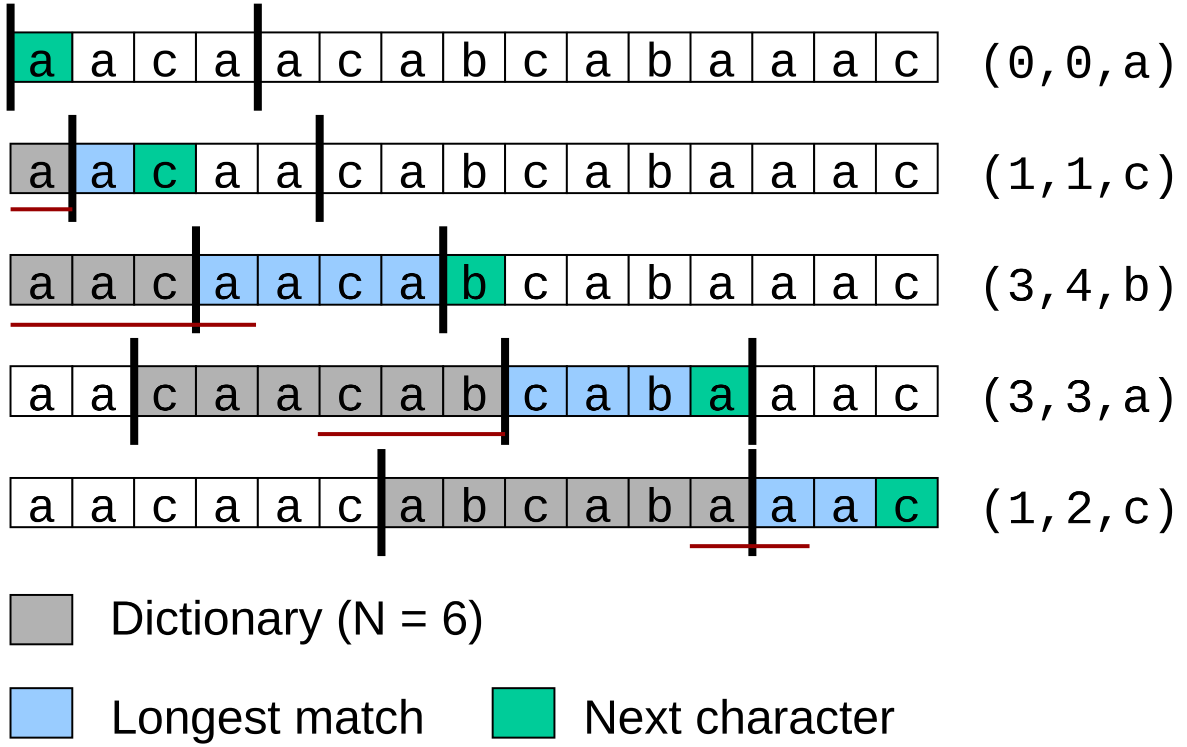
- Statistical coding requires to know symbol distribution in advance
- Dictionary coding schemes
  - Dynamic dictionary generation for variable length sequences of symbols
  - Code words with fixed length
  - Decoder reads message and reconstructs dictionary on-the-fly
  - Not suitable for short sequences (resulting bitstream can even increase in size)

LZ77 coding

- Lempel and Ziv (1977)
- Sliding-window compression
  - Maintain a window buffer of past N symbols
  - Can only decompress from the beginning
- Replace repeated occurrences of data with references to a single copy
  - Match encoded as a triplet command (offset, length, next)
  - Length can also include symbols that will be written by the command

LZ77 algorithm

```
while (stream not empty) do
begin
   find longest prefix p of remaining stream starting in window
   i = position of p in window
   j = length of p
   X = first char after p in view
   output triplet (i,j,X)
   move j+1 chars
end
```



a a c a a c a b c a b a a a c (0,0,a)
a a c a a c a b c a b a a a c (1,1,c)
a a c a a c a b c a b a a a c (3,4,b)
a a c a a c a b c a b a a a c (3,3,a)
a a c a a c a b c a b a a a c (1,2,c)
Dictionary (N = 6)
Longest match
Next character

Improving LZ77

- LZ78 encoding
  - Explicit dictionary
  - Supports partial decompression
- LZW encoding (Lempel-Ziv-Welch)
  - Patented by Unisys (expired)
  - Dictionary pre-initialized with all possible symbols
  - When a match is not found, the current symbol is assumed to be the first symbol of an existing string in the dictionary

DEFLATE encoding

- Combination of LZ77 and Huffman encoding
  - LZ77 – duplicate string elimination
  - Huffman – replace symbols with prefix codes
- Configurable encoding
  - Set time that can be spend for searching sub-strings

Compression in multimedia

- Image compression
  - PNG compression
  - JPEG compression
- Video compression
  - MPEG video codecs
- Sound compression
  - FLAC compression
  - MPEG-1 Layer 3 compression

Encoding comparison

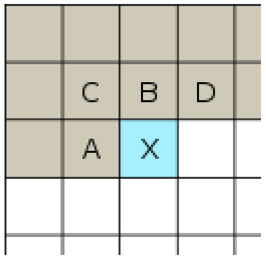
- Compression efficiency
  - What is the ratio between encoded and raw data size
- Delay
  - Time for compression/decompression
  - Amount of data processed at once (buffer)
- Implementation Complexity
- Robustness
  - Corruption of encoded data stream
- Scalability
  - Support for multiple profiles

PNG format

- Portable Network Graphics – 1996
- Lossless data compression
- Substitute for GIF format (patents, limitations)
  - Alpha channel (transparency)
  - RGB and indexed modes
- Two stage compression
  - Predictive filtering – encode difference to predicted value
  - DEFLATE compression

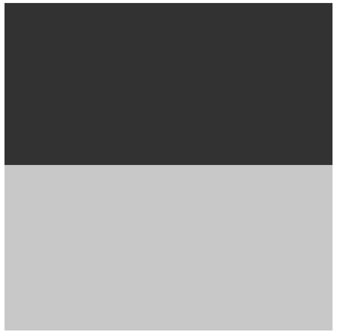
Predictive filtering

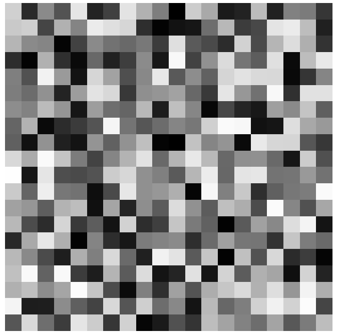
- Predicting pixel values based on value of previous pixel, encoding the difference
- Different filter types, chosen for each scan-line using heuristic algorithm
  - Unaltered value
  - Use value of A
  - Use value of B
  - Use mean value of A and B (round down)
  - Use A + B - C

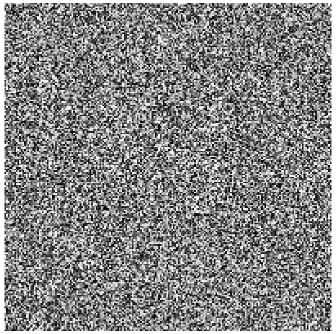


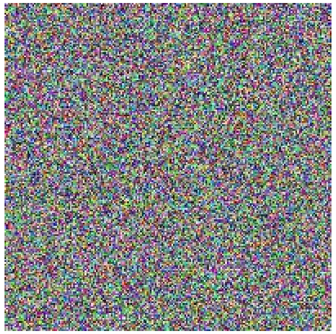
PNG examples

200 x 200 px RGB image – raw size: 120kB


2 grays – 616 B


10 px chunks – 2,3 kB


Random gray – 105 kB


Random RGB – 120, 5 kB

Vector graphics – 22,1 kB

ViCoS
sualgnitive
ystemslab

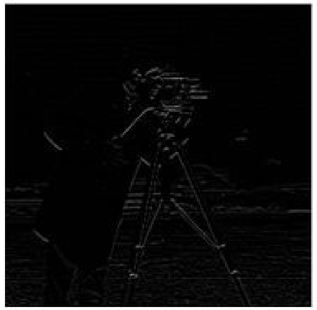
Photograph – 79,9 kB

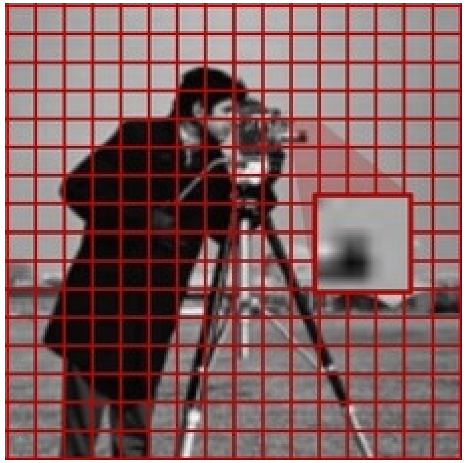
Lossy compression in images

- Exploit spatial correlation
  - Local pixels are usually very similar
  - Divide image in small regions
  - Encode region in a way that some information is discarded

$I(x, y)$

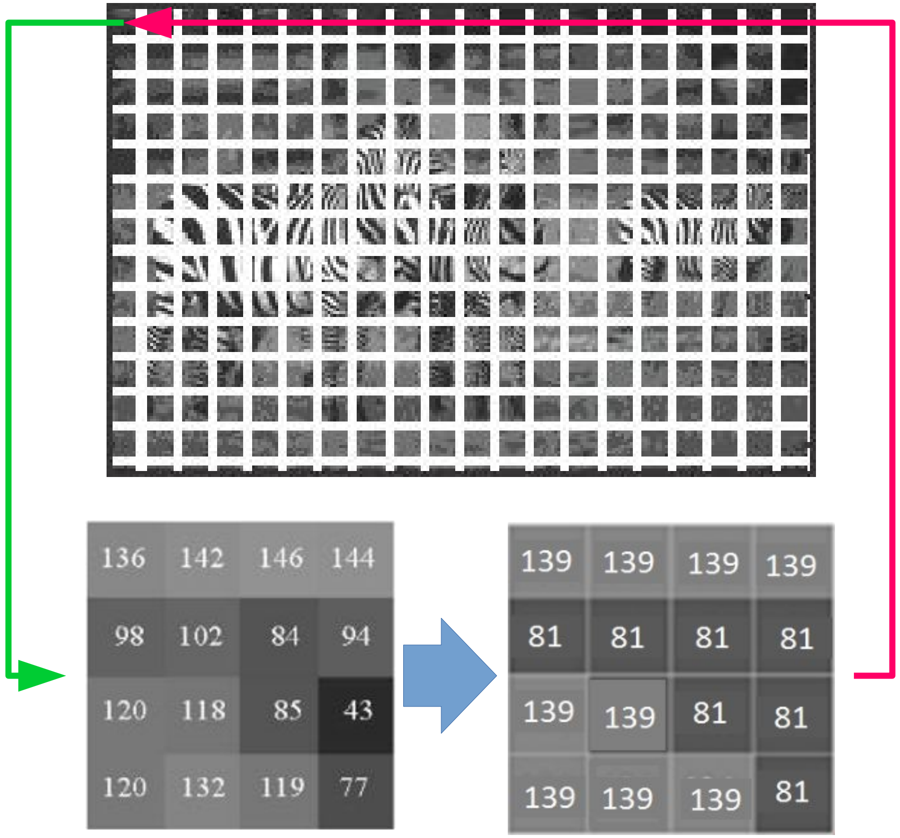
$I(x, y) - I(x, y + 1)$





Block truncation coding (BTC)

- Divide image into square blocks
- Encode each block with only two intensities
  - Mean value and standard deviation of the block should not change
- Simple, but used in real life
  - Mars Pathfinder (1997)

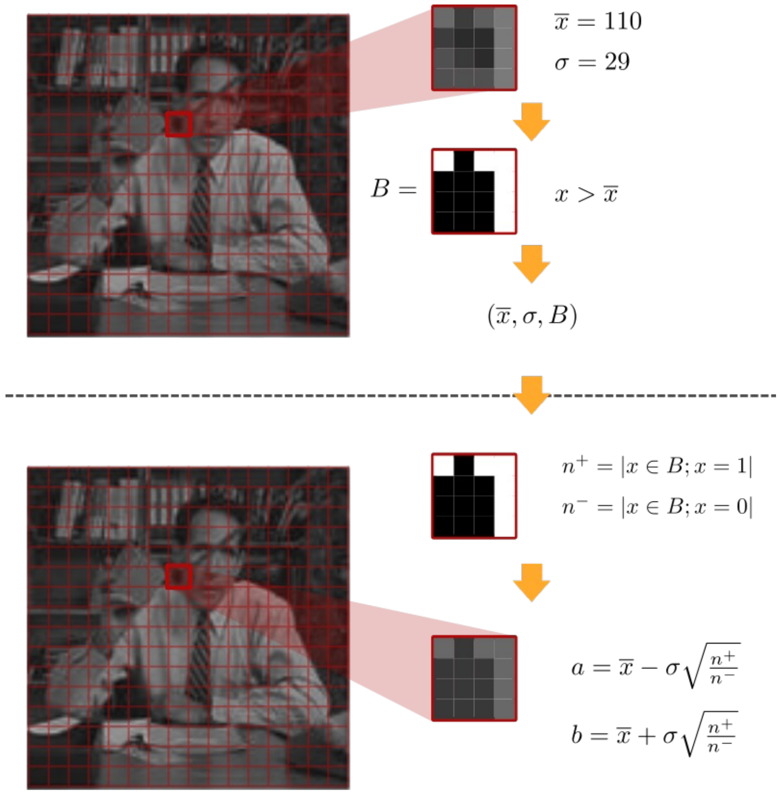


136	142	146	144
98	102	84	94
120	118	85	43
120	132	119	77

139	139	139	139
81	81	81	81
139	139	81	81
139	139	139	81

BTC compression / decompression

- Compression
  - Divide image into n x n blocks
  - Threshold each block with mean value of values in the block
  - Store thresholded value (1bit per pixel), mean value (8bit) and standard deviation (8bit) – 4 bytes in total
- Decompression
  - Use mean and standard deviation to compute two values
  - Assign higher values to pixels above the threshold, lower to the ones below



$\overline{x} = 110$
$\sigma = 29$
$B =$
$x > \overline{x}$
$(\overline{x}, \sigma, B)$
$n^{+} = |x \in B; x = 1|$
$n^{-} = |x \in B; x = 0|$
$a = \overline{x} - \sigma\sqrt{\frac{n^{+}}{n^{-}}}$
$b = \overline{x} + \sigma\sqrt{\frac{n^{+}}{n^{-}}}$

Properties of BTC encoding

- Compression ratio (8bit values, 4x4 region) – 4:1
- Very simple method
  - Noticeable errors due to loss of information
  - Frequent edges between blocks
  - Artifacts around edges and in parts of image with low contrast where the values are slowly changing from one to another

The JPEG standard

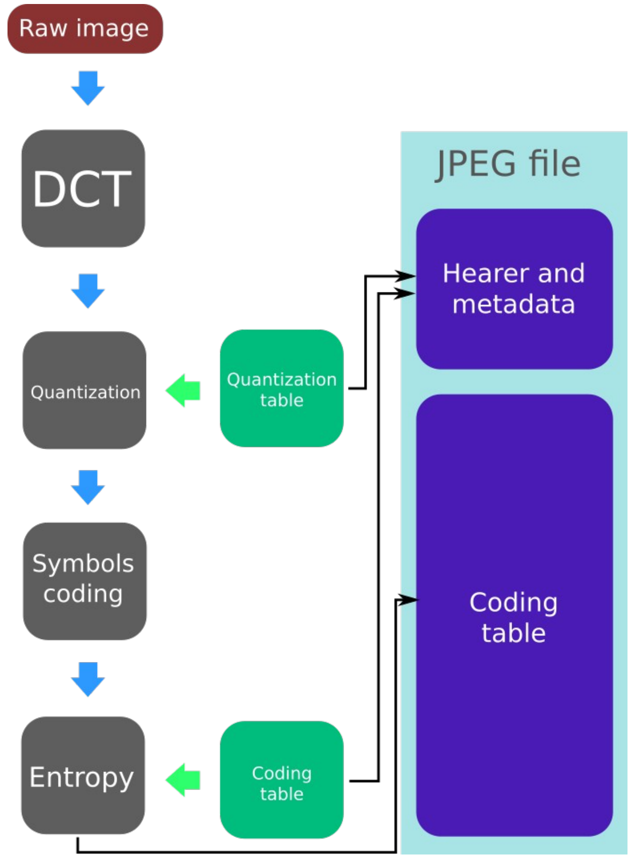
- Joint Photographic Experts Group (1992)
- Most popular standard for image compression
- Defines lossy and lossless compression
  - Lossy – DCT, quantization, RLE, Huffman
  - Lossless – predictive coding, Huffman
- Take into account human perception system
  - Discard information that is least relevant to human perception

The JPEG lossy compression

- Sequential and progressive coding
- Low computational requirements
- Suitable for all types of images, works better with photographs
  - Allows compromise between transfer speed and quality
  - Color depth: 8-12 bits
- File formats
  - Raw image data + metadata
  - JPEG File Interchange Format (JFIF) – multi-platform
  - Exchangeable Image file Format (EXIF) – digital cameras

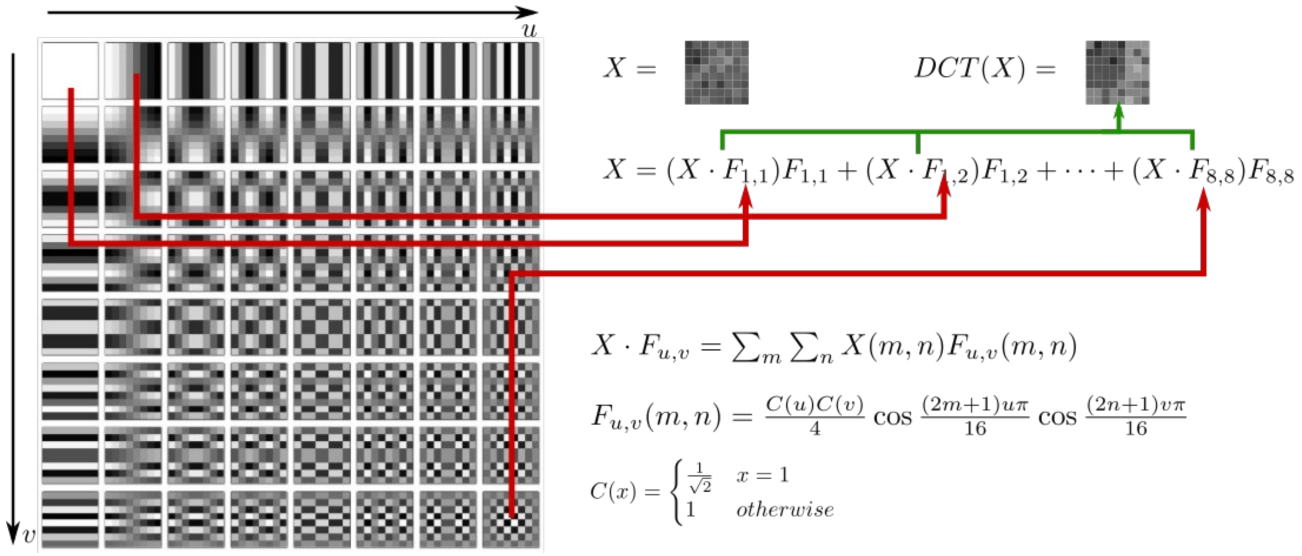
JPEG lossy encoder

- Signal transformation (DCT)
  - Image divided in 8 x 8 blocks
  - Each block described using DCT coefficients
  - If the image size is not divisible by 8 add lines/columns
- Quantization of DCT coefficients
- Mapping to symbols
  - Encode difference between DC coefficients of sequential blocks
- Encode using Huffman or arithmetic encoding



Discrete Cosine Transformation - DCT

- Apply 2D DCT to each block
  - Projection (dot product) to 64 basis functions
  - Function representation in frequency domain



$$X = (X \cdot F_{1,1})F_{1,1} + (X \cdot F_{1,2})F_{1,2} + \cdots + (X \cdot F_{8,8})F_{8,8}$$

$$X \cdot F_{u,v} = \sum_m \sum_n X(m,n) F_{u,v}(m,n)$$

$$F_{u,v}(m,n) = \frac{C(u)C(v)}{4} \cos\frac{(2m+1)u\pi}{16} \cos\frac{(2n+1)v\pi}{16}$$

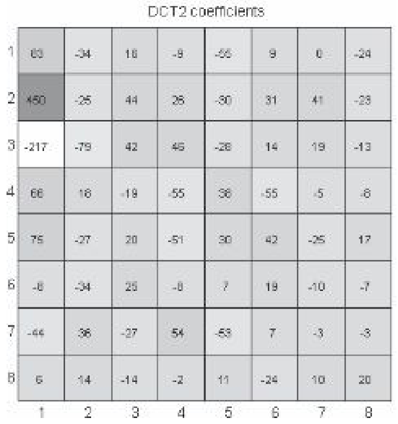
$$C(x) = \begin{cases} \frac{1}{\sqrt{2}} & x = 1 \\ 1 & otherwise \end{cases}$$

First component is direct current (DC) offset and represents the mean value in the block. Remaining components are alternating - AC

DCT coefficient quantization

- DCT coefficients are divided by quantization table and rounded
  - Defined using psychophysical tests (but not defined in standard)
  - Table quantizes higher frequencies more coarsely
- Controlled loss of information
  - Higher information is lost
  - Quality parameter scales the quantization matrix

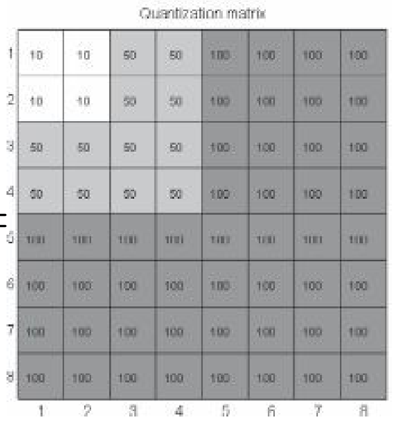
$X =$



DCT2 coefficients

	1	2	3	4	5	6	7	8
1	83	-34	16	-8	-55	9	0	-24
2	450	-25	44	26	-30	31	41	-23
3	-217	-79	42	46	-28	14	19	-13
4	66	16	-19	-55	36	-55	-5	-6
5	75	-27	20	-51	30	42	-25	17
6	-6	-34	25	-8	7	19	-10	-7
7	-44	36	-27	54	-53	7	-3	-3
8	6	14	-14	-2	11	-24	10	20

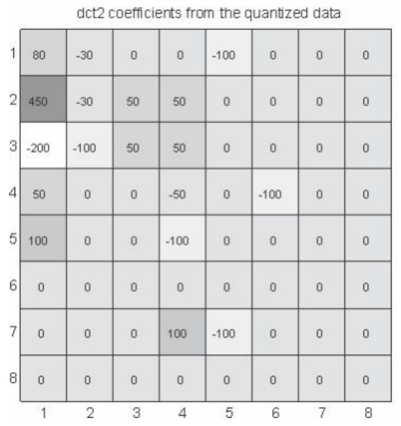
$K =$



Quantization matrix

	1	2	3	4	5	6	7	8
1	10	10	50	50	100	100	100	100
2	10	10	50	50	100	100	100	100
3	50	50	50	50	100	100	100	100
4	50	50	50	50	100	100	100	100
5	100	100	100	100	100	100	100	100
6	100	100	100	100	100	100	100	100
7	100	100	100	100	100	100	100	100
8	100	100	100	100	100	100	100	100

$\lfloor X/K \rfloor \cdot K =$

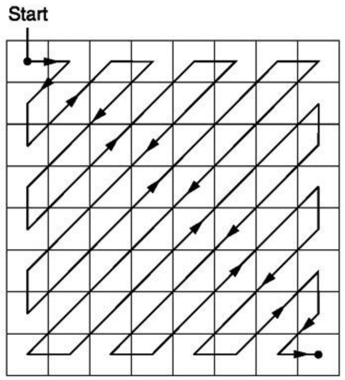


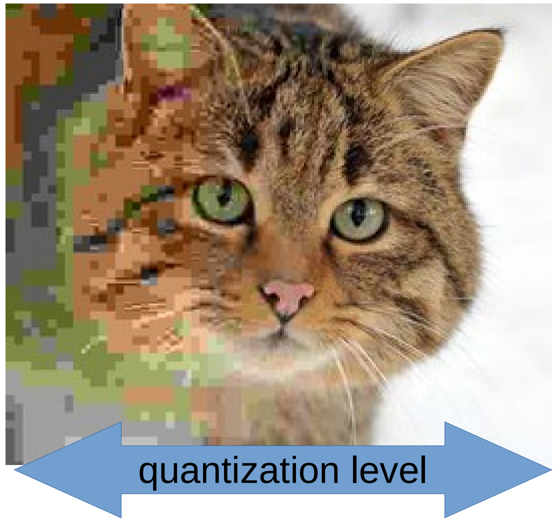
dct2 coefficients from the quantized data

	1	2	3	4	5	6	7	8
1	80	-30	0	0	-100	0	0	0
2	450	-30	50	50	0	0	0	0
3	-200	-100	50	50	0	0	0	0
4	50	0	0	-50	0	-100	0	0
5	100	0	0	-100	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	100	-100	0	0	0
8	0	0	0	0	0	0	0	0

DCT coefficients as symbols

- Separate encoding for DC (1 value) and AC components (63 values)
- Encoding DC component
  - Difference between sequential blocks
- Encoding AC coefficients
  - Ordered from lower to higher frequencies
  - Sequence encoded as a RLE sequence (contains sequences of zeros)
- Writing symbols
  - DC and AC symbols encoded using Huffman or arithmetic encoding
  - Huffman code maps are predefined or calculated on-the-fly





JPEG compression example

640x480 RGB image – raw size: 900kB

- Quality 100: 200kB, error: 0.55
- Quality 80: 47.7kB, error: 1.63
- Quality 60: 32.3kB, error: 2.14
- Quality 40: 25.0kB, error: 2.61
- Quality 20: 16.6kB, error: 3.34
- Quality 0: 5.6kB, error: 9.46

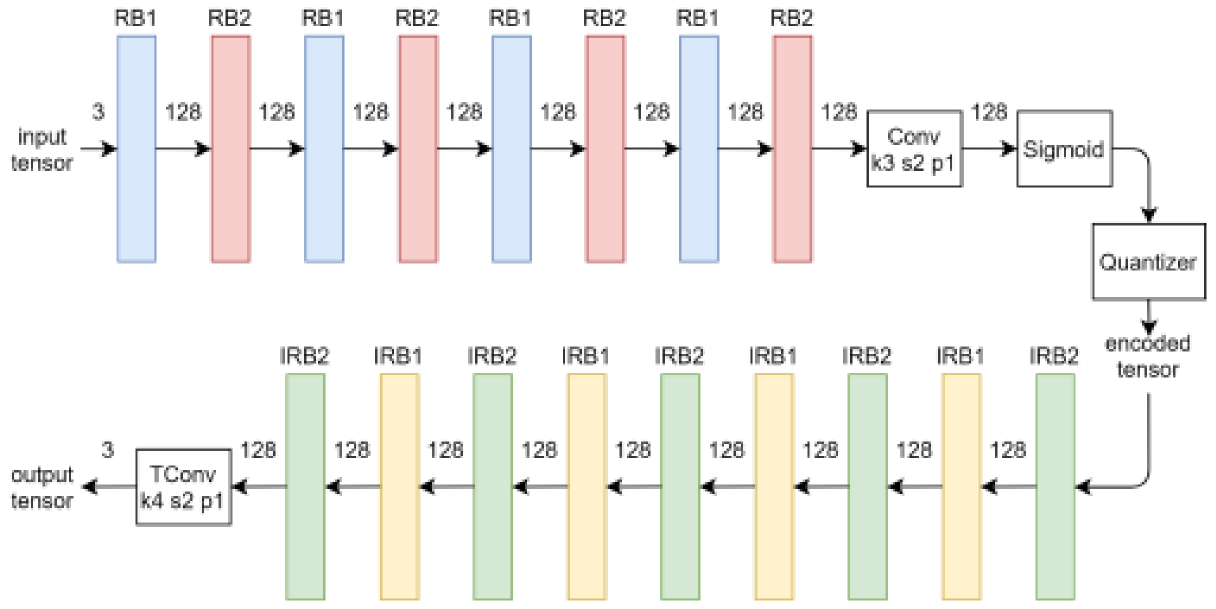
Compressing color images

- Human eye is less sensitive to changes in chroma
  - Separate RGB to intensity and chroma (YCrCb)
  - Down-sample chroma (Cr,Cb) with factor 2
  - Encode each channel separately (different quantization table for Y and Cr/Cb)
  - Using different quantization tables for chroma and intensity

Deep learning for image compression

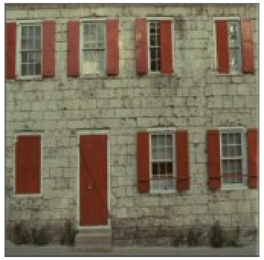
- Auto-encoder architecture
  - Reduce spatial resolution
  - Increase channels
  - Learn to decode quantized latent representation
- Unsupervised training
  - Learn to reconstruct image
  - Train on real world images

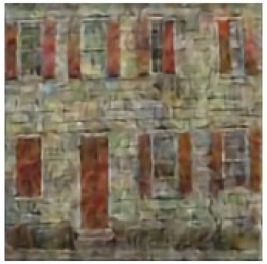
Compression architecture



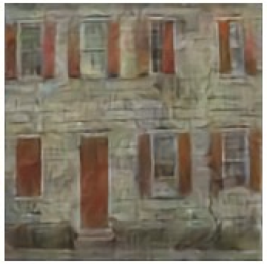
Jan Pelicon, “Kompresija video posnetkov z nevronskimi mrežami”, 2020

Comparison

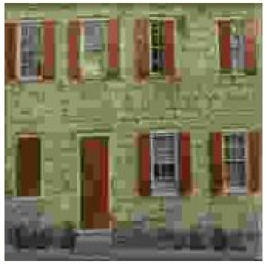




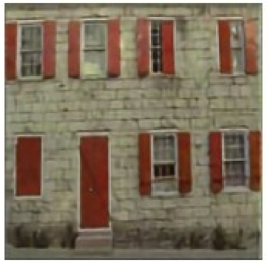
$n = 2,\ bpp = 0.150,$
$SS\text{-}SSIM = 0.57,$
$AEC_{noise} = 0$



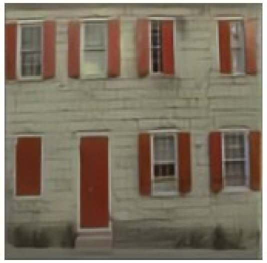
$n = 2,\ bpp = 0.152,$
$SS\text{-}SSIM = 0.58,$
$AEC_{noise} = \sigma^2$



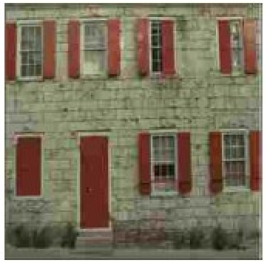
$Q = 3,\ bpp = 0.142,$
$SS\text{-}SSIM = 0.43,$
JPEG 4:2:0



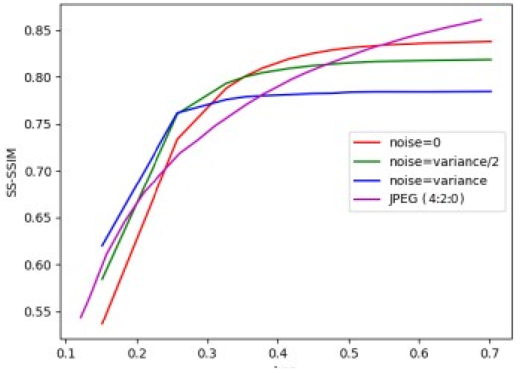
$n = 16,\ bpp = 0.511,$
$SS\text{-}SSIM = 0.77,$
$AEC_{noise} = 0$

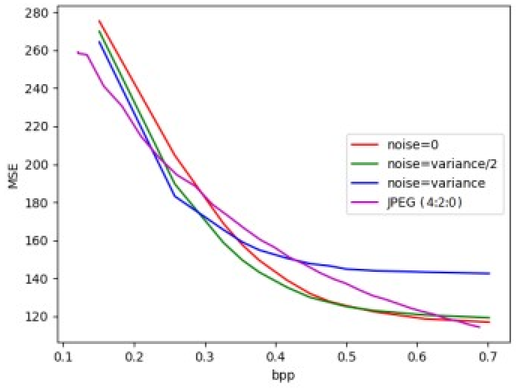


$n = 16,\ bpp = 0.503,$
$SS\text{-}SSIM = 0.70,$
$AEC_{noise} = \sigma^2$



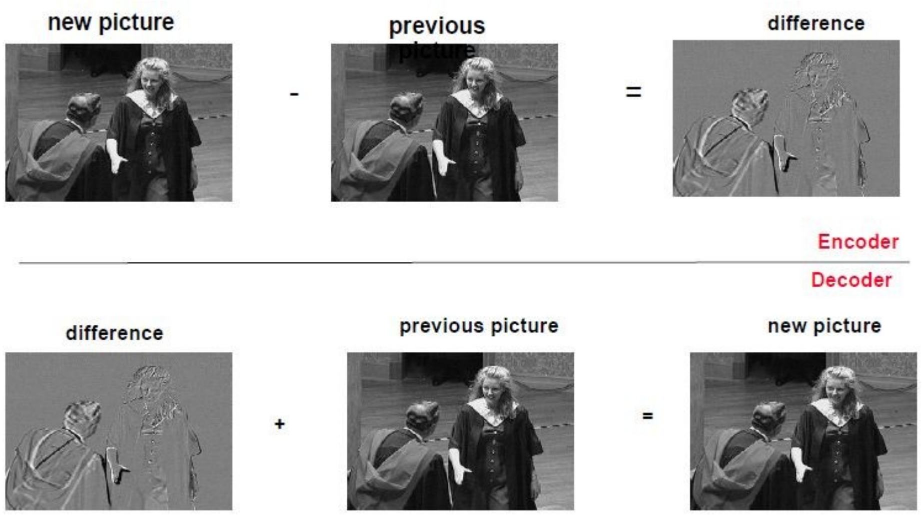
$Q = 14,\ bpp = 0.498,$
$SS\text{-}SSIM = 0.70,$
JPEG 4:2:0





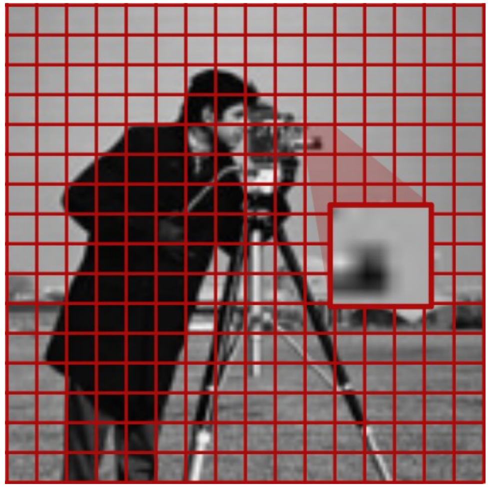
Video compression opportunities

- Minimize spatial and temporal redundancy
  - If differences are small we can compress them well
  - If we know previous image and differences, we can compute new image



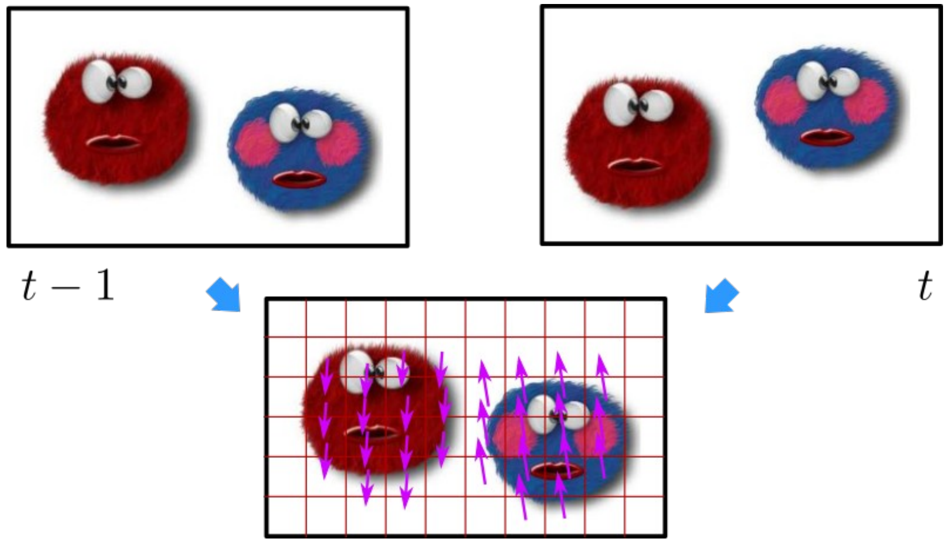
Encoding spatial correlation

- Processing 8x8 pixel blocks
  - Encoding using 2D DCT
  - Fast hardware implementation possible
- Motion JPEG
  - Simple to implement, well supported
  - Tolerance to rapid motion
  - Interframe encoding (each frame encoded separately)
  - Low compression ratios (e.g. 1:20, excellent quality 1:10)



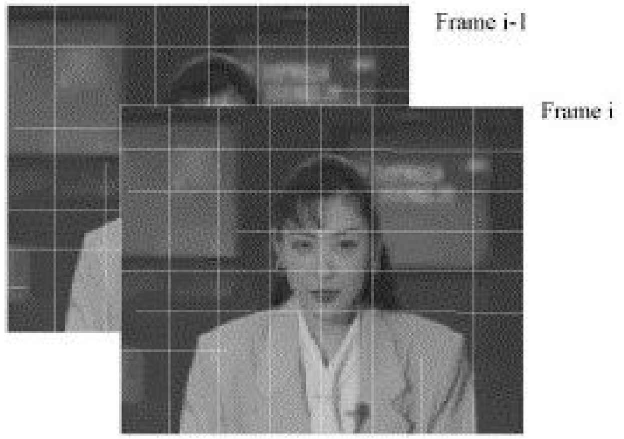
Encoding temporal correlation

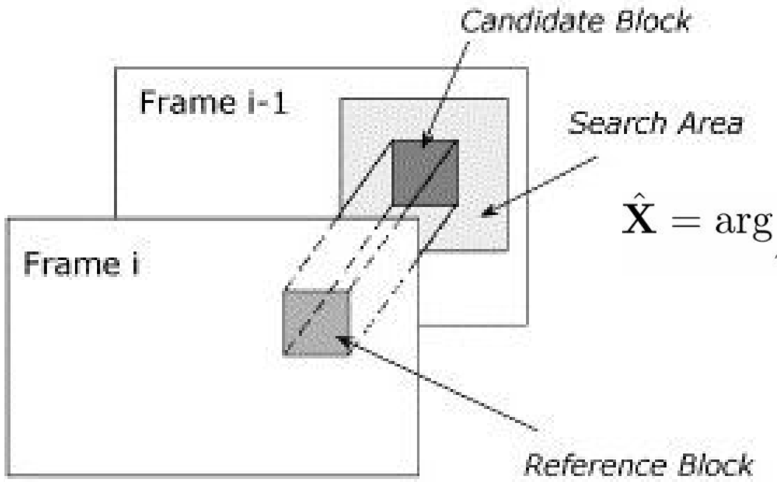
- Determining translation vector for each block
  - Translation for each block
  - Grouping blocks together (macroblocks)
- Frame reconstruction
  - Move blocks from previous frame
  - Visual difference encoded separately



Temporal correlation with blocks

- Determining block translation
  - Block matching method
  - Assume uniform motion within block
  - Searching for translation v, that minimizes distance between blocks





$$\hat{\mathbf{X}} = \arg\min_{v} \frac{1}{nm} \sum_{i,j} (x_{ij} - \hat{x}_{ij})^2$$

Video codec selection

- Compression level
- Losing information during compression
- Transfer speed (bit-rate) vs. distortion/loss
- Algorithm complexity
- Fixed / variable bit rate
- Communication channel characteristics
- Standard compatibility

Motion Picture Expert Group (MPEG)

- Working group that proposes standards for video and audio compression and transmission
  - Established in 1988 (ISO + IEC)
- Proposes MPEG-X phases with multiple parts
  - Each part covers a certain aspect of the whole specification
  - Some standards are later revisited and amended

MPEG parts

- MPEG-1: Video and audio for digital media
- MPEG-2: Generic video coding
- MPEG-4: Interactive video, audio, 3D graphics, Web
- MPEG-H: High Efficiency Coding and Media Delivery in Heterogeneous Environments
- MPEG-I: Coded Representation of Immersive Media

MPEG profiles and levels

- Standard defines wide range of applications
  - In many cases support for entire standard is unrealistic or expensive
  - Profiles and levels formalize support constraints
- Profile – subset of specification
  - Picture coding
  - Chroma format
- Level – limit parameters
  - Maximum resolution
  - Framerate
  - Bitrate

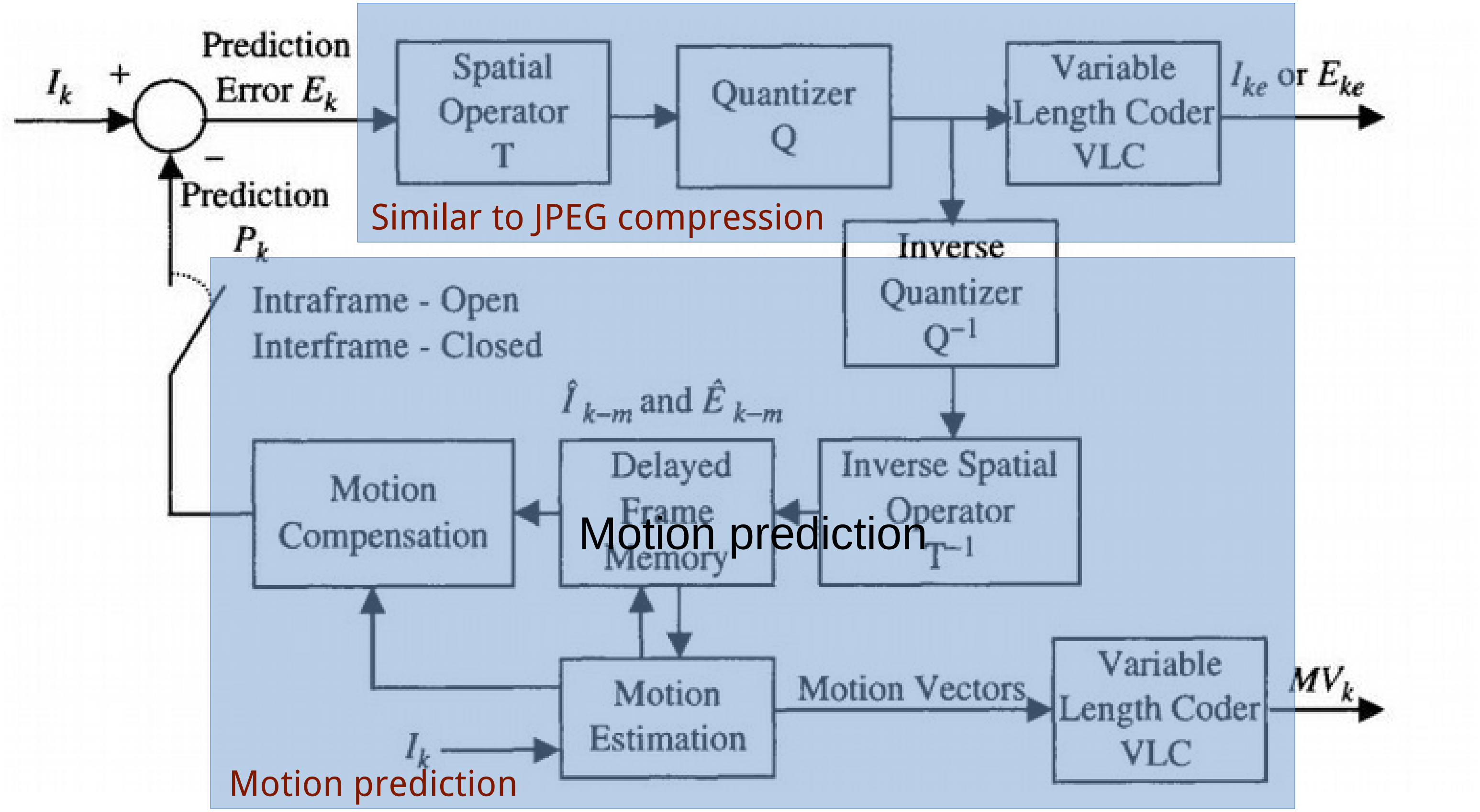
Video compression standards

Compression Standard	Publish Date	Application
H.261	1990	Video Conferencing, Video Telephony
MPEG 1 Part 2	1993	Video CD
H.262/MPEG 2 Part 2	1995	DVD Video, Blu-Ray, DVB, SVCD
H.263	1996	Video Conferencing, Videotelephony, Video on Mobile Phones(3GP)
MPEG 4 Part 2	1999	Video on the Internet
H.264/MPEG-4 AVC	2003	Blu-Ray, DVB, HD DVD
HEVC: *MPEG-H Part 2* ITU-T *H.265*	2013	Double data compression ratio compared to H.264
VVC: MPEG-I Part 3 ITU-T H.266	2020	Higher resolutions, omnidirectional, HDR

MPEG-1

- Audio-video storage and playback for VHS-quality video
  - Transfer rate ~1.5Mbit/s
  - Compression rate 26:1 (video), 6:1 (audio) without excessive loss
- Five parts: video, audio, compliance, reference implementation, system
- Asymmetric application
  - Compress once, decompress many times
  - Encoder can be complex, encoding slow
  - Decoded is simpler, decoding fast

MPEG compression overview

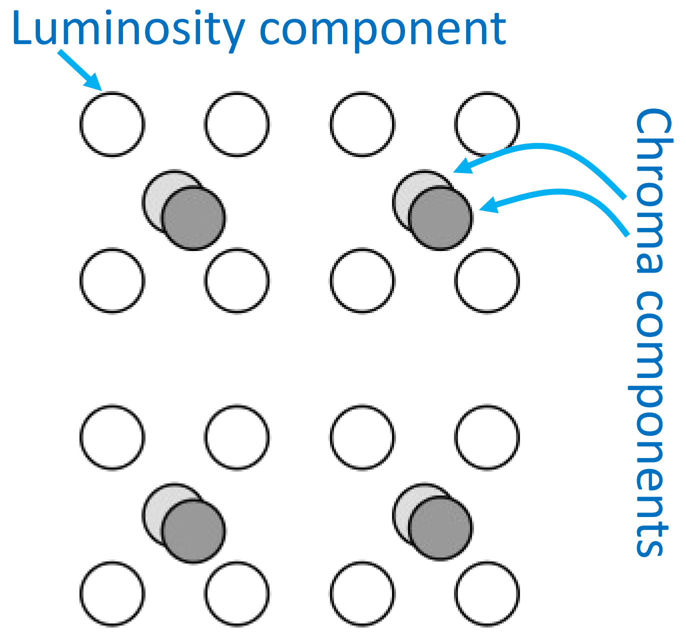


Requirements

- Normal playback with random access
- Support for video editing
- Reverse playback and fast playback
- Different resolution, frame rate
- Cheap hardware implementation of decoders

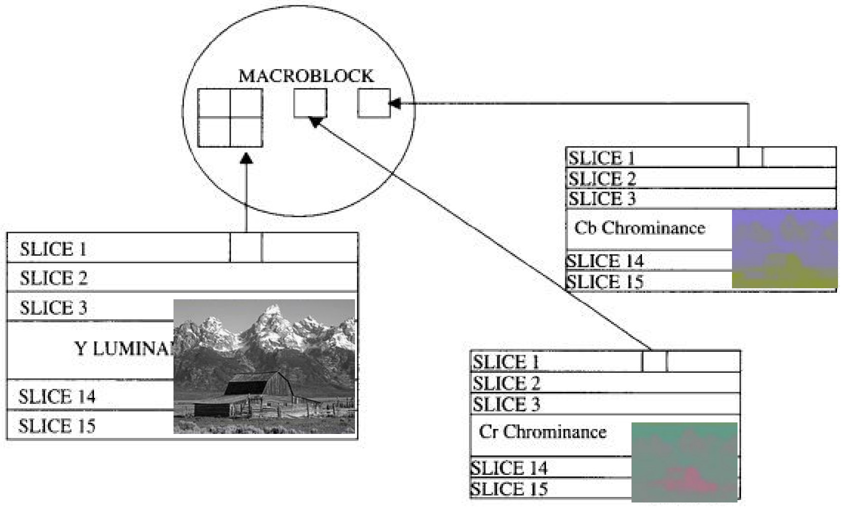
Video resolution

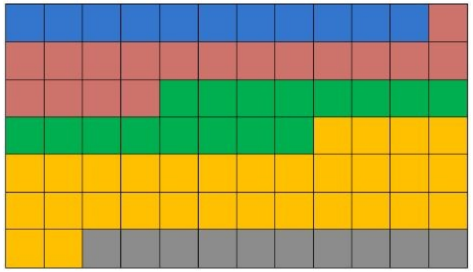
- Using YCrCb color space
  - Luminosity and two chroma channels
  - Chroma subsampling (Cr, Cb)
  - 8 bit values, 2:1:1 subsampling
- Common resolutions
  - 352x240, 352x288, 320x240
  - 352x240 (Y channel), 176x120 (Cr, Cb channels)



Macroblocks

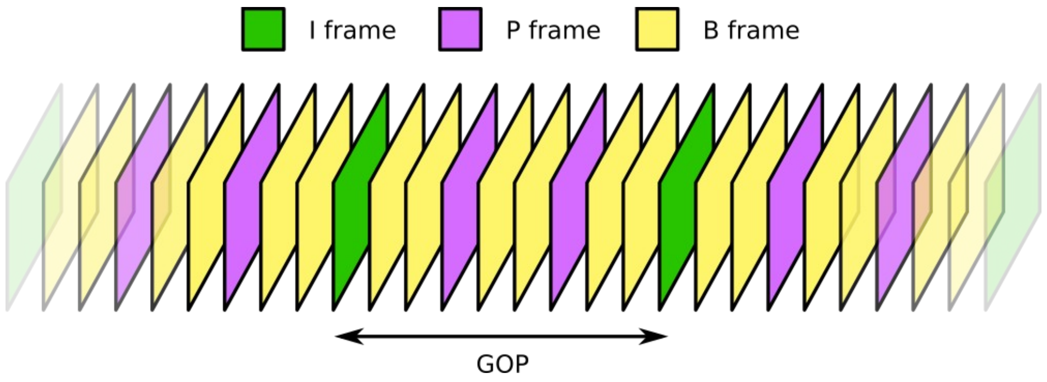
- YCrCb color space
  - Y channel - 16x16 blocks
  - Cr, Cb channels - 8x8 blocks
- Macroblock contains
  - Four 8x8 Y blocks
  - One 8x8 Cr block
  - One 8x8 Cb block
- Slice is sequence of macroblocks
  - Each slice can have different compression parameters
  - Easier error recovery (each slice encoded separately)





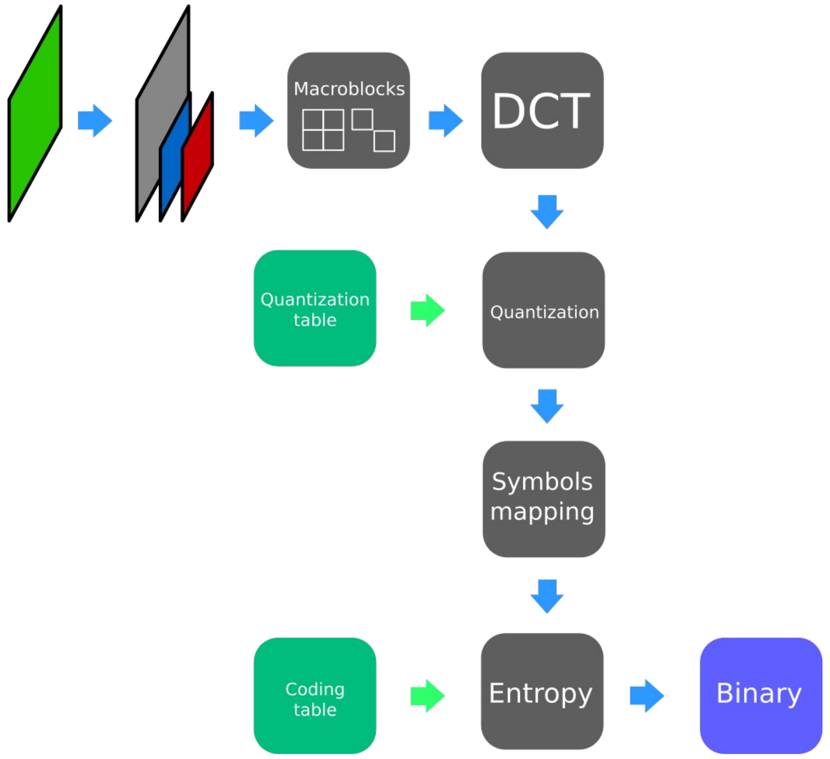
Frame types

- Group of pictures (GOP)
- Intra frame (I frame)
  - Encoded similarly than JPEG image
  - Approximately 12 frames apart
- Predictive frame (P frame)
  - Encoded with motion from previous I or P frame
- Bi-directional frames (B frames)
  - Encoded with prediction from previous and next I or P frames
- Low-resolution frame (frame D)
  - Fast video navigation
  - Only DC coefficients of DCT
  - Rarely used



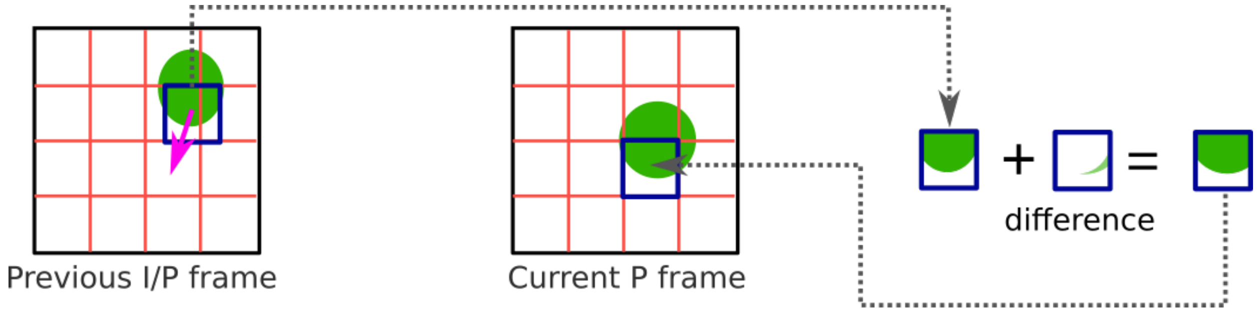
Type I frames

- Each macroblock is encoded statically
  - Chroma subsampling
  - Macroblocks
  - Similar approach than JPEG (different quantization tables)
- Enables random access to video content



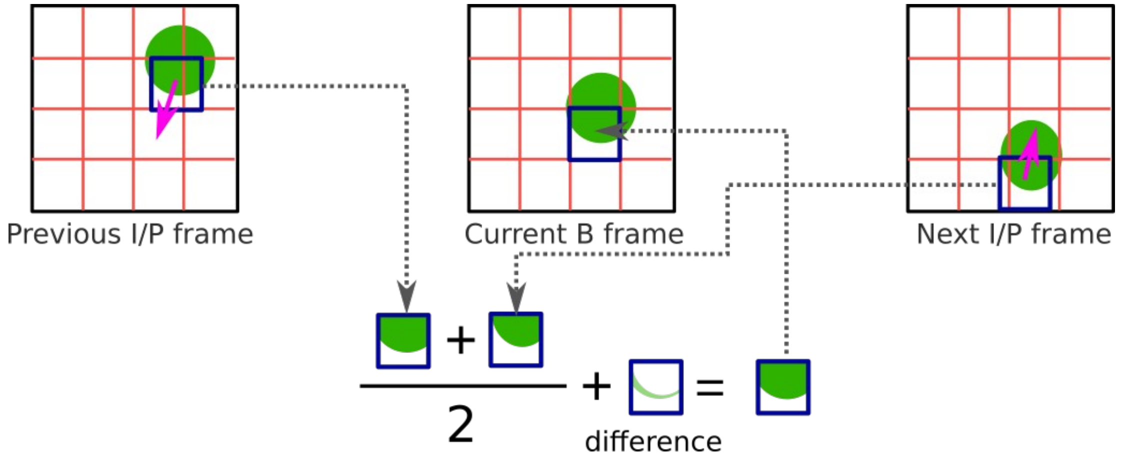
Type P frames

- Use information from previous I or P frame
  - Motion vector of each macroblock computed based on Y channel
  - Encode visual difference between the previous and current macroblock
  - If difference too big encode entire macroblock



Type B frames

- Encoded using previous and next I or P frame
  - Averaging motion-compensated images
  - Encoding only difference
  - Delayed decoding



Example of decoding a frame

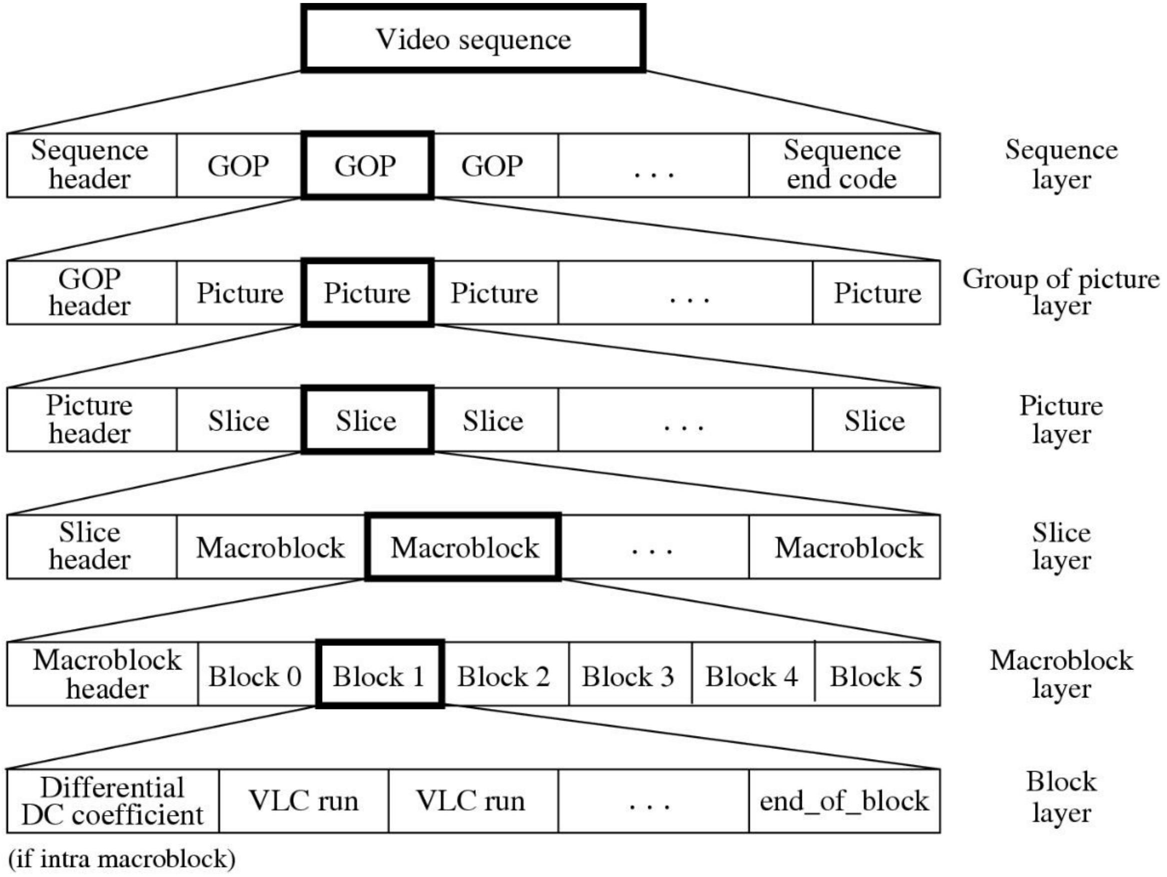
Our video stream contains frames

$$\{I_1, B_1, B_2, P_1, B_3, B_4, P_2, B_5, B_6, I_2, B_7, B_8, P_3, B_9, B_{10}, P_4, B_{11}, B_{12}, I_3\}$$

Which frames do we have to read to reconstruct the following frames

- I2 – only I2
- P2 – I1, P1, P2
- P3 – I2, P3
- B4 – I1, P1, P2, B4

Datastream hierarchy

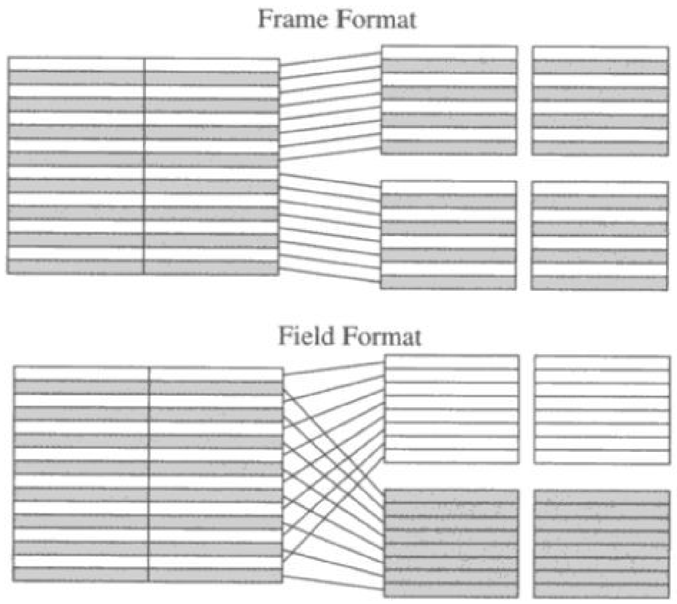


MPEG-2

- Higher resolution video (HDTV) for 4-15Mb/s channels
  - Suitable for digital broadcast TV
  - Support for interlaced video
  - Better robustness (errors)
  - Scalability (various bitrates)
- MPEG-2 decoders should also decode MPEG-1 streams
- Excellent quality compression rate 1:30
- Two interlaced video formats
  - Frame format
  - Field format

Field/frame format

- Frame format
  - Odd and even lines are encoded as single image
  - Image encoded using single header
- Field format
  - Odd and even lines encoded separately, every field encoded as separate image
  - Each field has its own header
- Interleaved format can switch between two modes for each image
- Progressive mode uses only frame format



Scalability

- Different profiles for various applications and quality levels (e.g., DVD, recordings from two cameras)
- Video separated in multiple layers
  - Base layer: information for coarse image reconstruction
  - Enhancement layers: information for improving coarse image quality
  - Separate transmission of basic (high priority) and enhancement (lower priority) layers

Spatial and frequency scalability

- Spatial scalability (image size)
  - Base layer: Images encoded with lower resolution
  - Enhancement layers: Missing high-resolution information, base layer used for prediction
- Frequency scalability (DCT coefficients)
  - Base layer: lower-frequency DCT (e.g. DC + motion vectors)
  - Enhancement layers: higher frequencies DCT (AC)

SNR and temporal scalability

- SNR scalability (signal-to-noise ratio)
  - Base layer: Heavily quantized intensity, coded in original resolution
  - Enhancement layers: residual information
- Temporal scalability
  - Base layer: images with lower framerate (e.g. I frames)
  - Enhancement layers: missing images in between, using motion prediction to reconstruct them (e.g. B frames)

MPEG-4

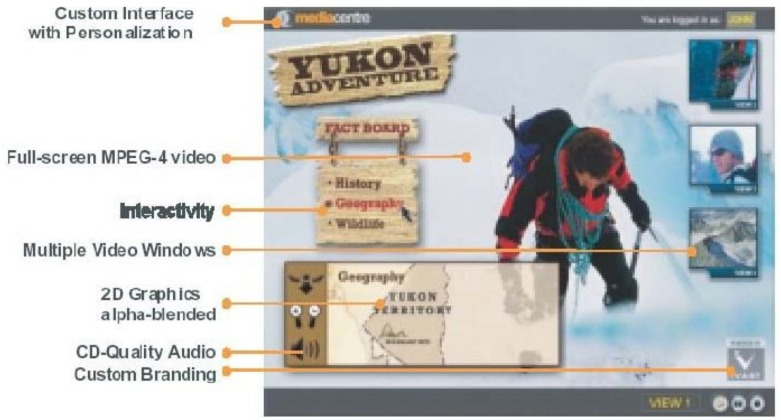
- Focus on interactive multimedia content
  - Better compression rate with higher throughput (bitrates).
  - Robust in environments with frequent errors
- Object oriented content representation – media objects
  - Interactivity on object level
  - Random access to objects

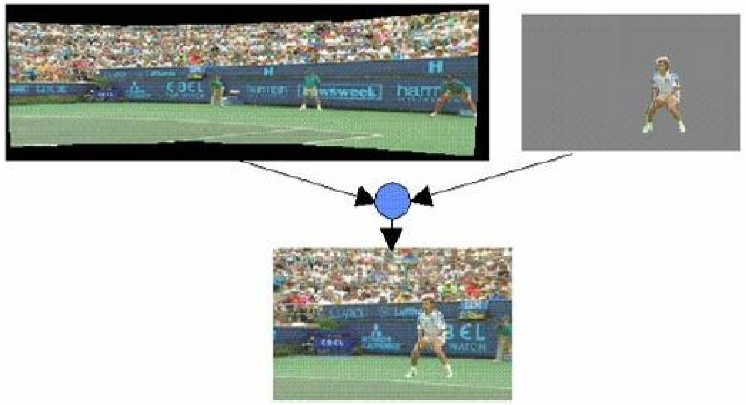
Media objects

- Different media sources
  - Real and synthetic images, sound, graphics
  - Interaction, manipulation, indexing, retrieval
- Hierarchical organization of media objects
  - Objects form dynamic/composed scenes
  - Objects separated in channels and encoded separately
  - Objects are transferred in separate streams
  - Composition occurs on the decoding device
  - Scene description language (BIFS)

Scene hierarchy

- Video-object Sequence (VS)
  - Complete visual scene
- Video Object (VO)
  - Arbitrary non-rectangular shape
- Video Object Layer (VOL)
  - Scalable coding support
- Video Object Plane (VOP)
  - Snapshot of VO at time t
  - MPEG1, 2: Entire frame is one VOP
- Group of Video Object Planes (GOV)
  - Optional group of VOPs





H.263/MPEG 4 Part 2

- 21 profiles – various applications
- Quarter Pixel motion compensation (Qpel)
  - Quarter pixel size for encoding motion vectors
  - Interpolation of blocks
- Global motion compensation (GMC)
  - Motion of scene based on affine transformation
  - Each macroblock encoded using global or local transformation
- Notable implementations: DivX and Xvid

H.264/MPEG-4 Part 10

- First version finished in 2003
  - Intended for storing high-definition video (Blu-Ray)
- Most widely used video codec
  - Used for HDTV, DVB-T
  - Included in Adobe Flash Player, Microsoft Silverlight
  - De-facto standard in HTML5
- Increased compression against MPEG-2
- Number of profiles available, from video-conference to high-resolution stereoscopic streams

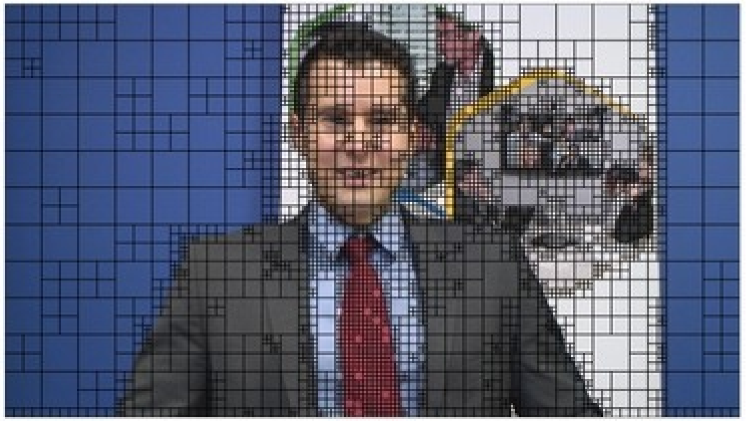
H.264 improvements

- Variable Length entropy coding of blocks
  - Context-adaptive variable length coding (CAVLC)
  - Context-adaptive binary arithmetic coding (CABAC)
- Flexible block size
- Motion compensation
  - Larger range
  - More accurate (Qpel)
- Intra-frame prediction
  - Prediction of blocks using neighboring blocks
- Signal-adaptive deblocking filters
  - Smoothing edges of blocks
  - Increases subjective quality

Excellent quality compression ratio 1:50

H.265 – High Efficiency Video Coding

- Part of MPEG-H phase (heterogeneous environments)
- Boost video streaming
  - Various block sizes, organized hierarchically (CTU)
  - Better intra- and inter-frame prediction
  - Parallel processing (WPP)
- 50% smaller streams than H.264
- Limited acceptance
  - Patents
  - Higher computational requirements (10x)



H.266 - Versatile Video Coding

- Part of MPEG-I phase
- Resolutions from 4K to 16K
- Support for 360° videos
- High dynamic range support
- Auxiliary channels (depth, transparency)
- Variable frame rate (0 to 120 Hz)
- Encoding complexity up to ten times that of H.265

Evaluating video/image compression

- Qualitative evaluation
- Quantitative evaluation
  - Evaluation datasets (Kodak, Xiph, CDVL)
  - Evaluation measures (PSNR, SSIM)

Peak Signal-to-Noise Ratio (PSNR)

- Based on Mean Squared Error
  - Measured in dB
  - MAX = 255 for 8 bit image

$$\begin{aligned} PSNR &= 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ &= 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \\ &= 20 \cdot \log_{10}\left(MAX_I\right) - 10 \cdot \log_{10}\left(MSE\right) \end{aligned}$$

Structural Similarity Index (SSIM)

- Perceived change in structural information
  - Luminance, contrast, structure
- Neighborhood pixels have strong dependencies
- MS-SSIM – multi scale variant
- Calculated on a fixed local window (m x n)

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

PSNR and SSIM examples

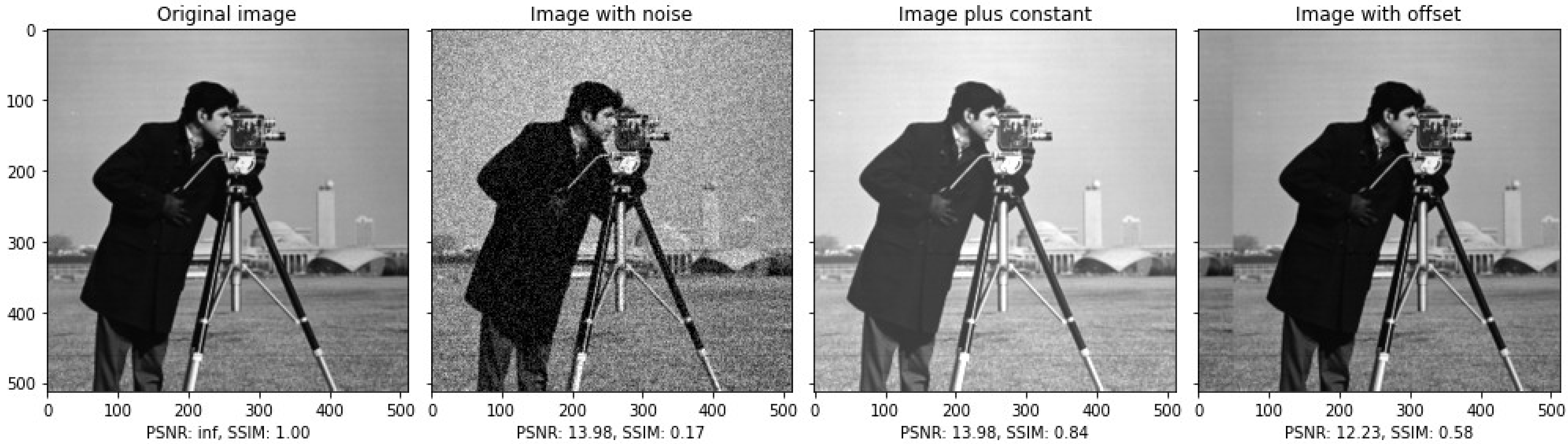
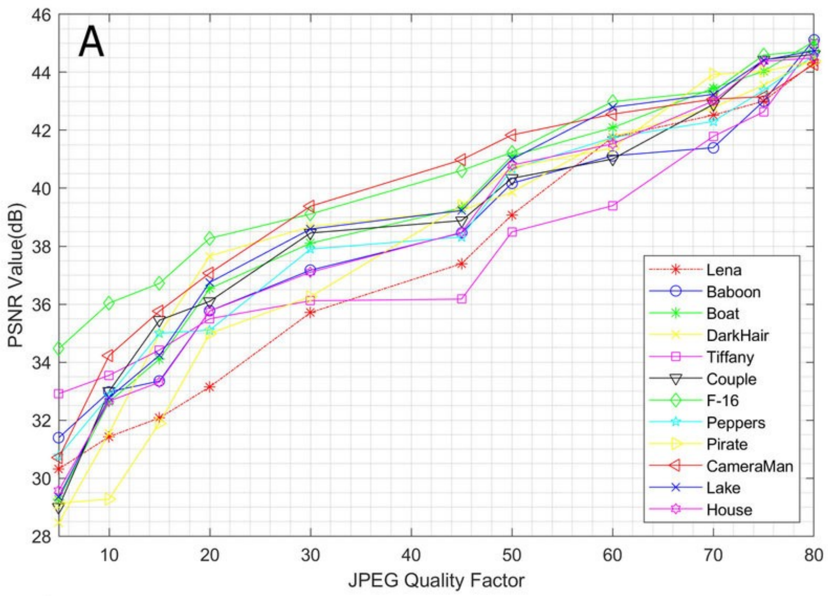
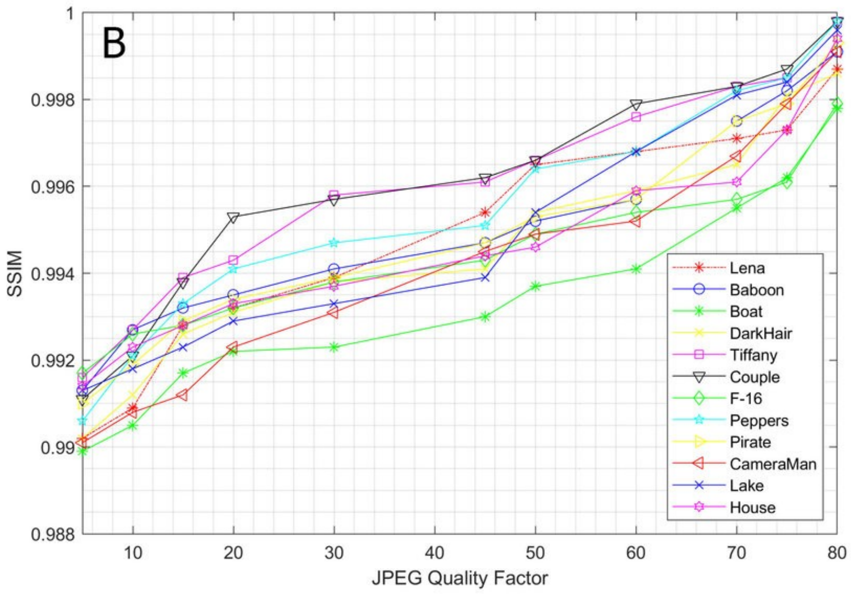
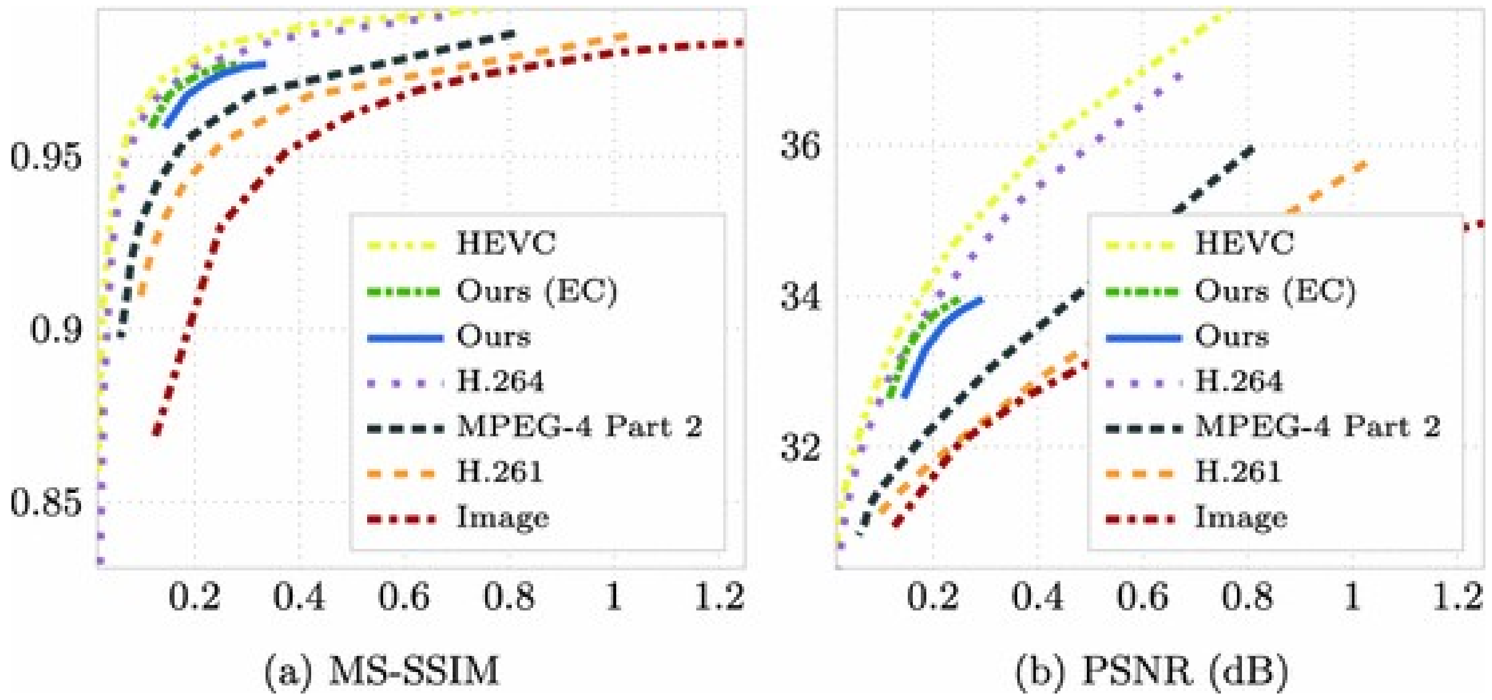


Image compression (JPEG)





Video compression comparison



(a) MS-SSIM (b) PSNR (dB)

Audio compression

- Lossless compression (~1:3)
  - Predictive coding
- Lossy compression
  - Speech compression (~1:8)
  - Sound compression (~1:6)

Lossless audio compression

- Quality does not degrade
  - Predictive coding
  - Encoding (entropy/dictionary)
- Uncompressed audio (e.g. WAV)
  - Linear pulse-code modulation
  - Large size

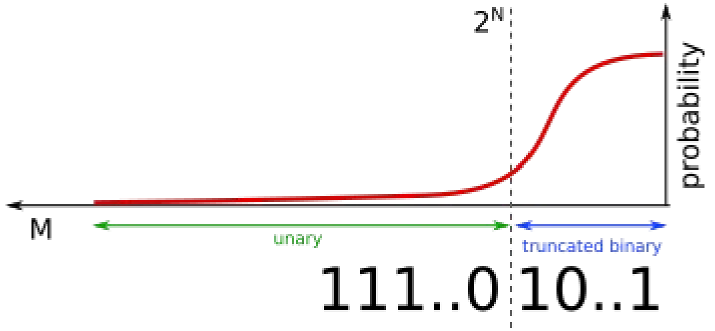
FLAC compression

- Free Lossless Audio Codec
- Reduction from 50% up to 80% (in some cases)
  - Linear prediction
  - Error coding (Golomb-Rice)
  - RLE
  - Stereo (inter-channel correlation)
- Data hierarchy
  - Before/after encoded: block, sub-block
  - Encoded: frame, sub-frame
  - Sub-frames share some encoding parameters

Prediction and residual

- Zero
  - Digital silence, constant value
  - RLE
- Verbatim
  - Zero-order predictor
  - Residual is signal itself
- Fixed Linear
  - Fitting p-order polynomial to p points
  - Efficient algorithm
- FIR Linear
  - Linear combination of previous samples
  - Slower, diminishing returns

- Golomb codes
  - Split value in two parts (divide by M)
  - Quotient – unary coding
  - Remainder – (truncated) binary coding
  - Efficient if small values dominate distribution
  - Rice coding – M is 2N



Speech compression

- Speech is formed by air traveling from lungs to mouth
  - Excitation signal formed by vocal folds opening and closing
- During speech the vocal tract shape is changing
  - Mouth opening and closing, tongue moving
  - This gives the excitation signal its spectral shape
- Speech coding approaches
  - Non-linear quantization
  - Multi-channel encoding (bands with different resolution)
  - Modeling speech (Source-filter model)

Vocoders

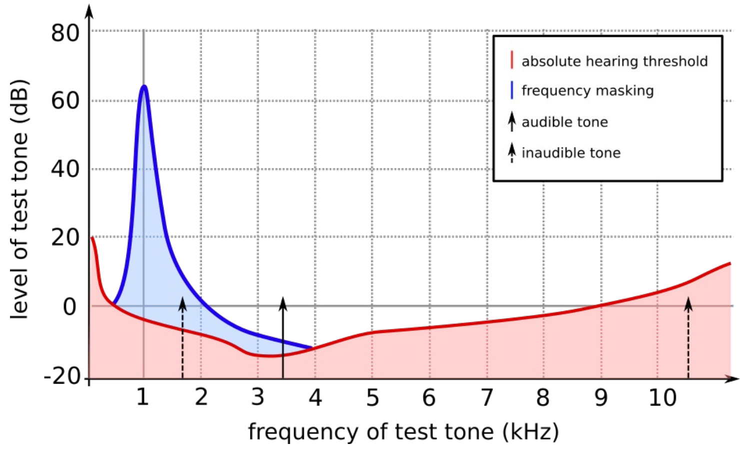
- Vocoder – encoder for speech signal
- LPC-10 vocoder
  - Linear prediction based on past model coefficients
  - Two modes (voiced - periodic waveform, unvoiced – noise generator)
  - Gain, pitch estimation
  - Sound processed in frames (30-50 frames per second)
  - Intended bandwidth: 2.4 kbps (GPS)
- CELP vocoder
  - Analysis by synthesis (optimizing resulting signal perceptual error in closed loop)
  - Short-term prediction (LPC analysis) + Long-term prediction (codebooks)
  - Intended bandwidth: 4.8 kbps
  - MPEG-4 Audio

MPEG-1 Audio Compression

- Three layers of compression
  - Downwards compatibility
  - Each more complex (encoder)
  - Quality depends on available space (bitrate)
- Main concepts
  - Frequency domain (DCT)
  - Non-uniform quantization - bands
  - Imperfections in human auditory system

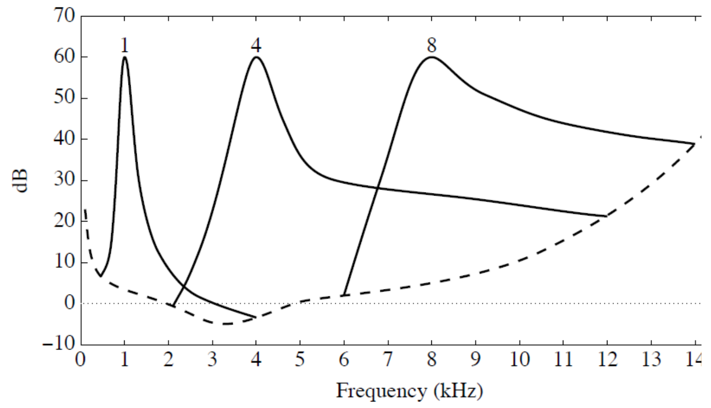
Frequency masking

- Core compression mechanism in MPEG-1
- Louder tone masks silent tones nearby
  - Silent tones are not perceived
  - Lower frequencies mask higher ones better
  - Louder the sound, more tones it masks



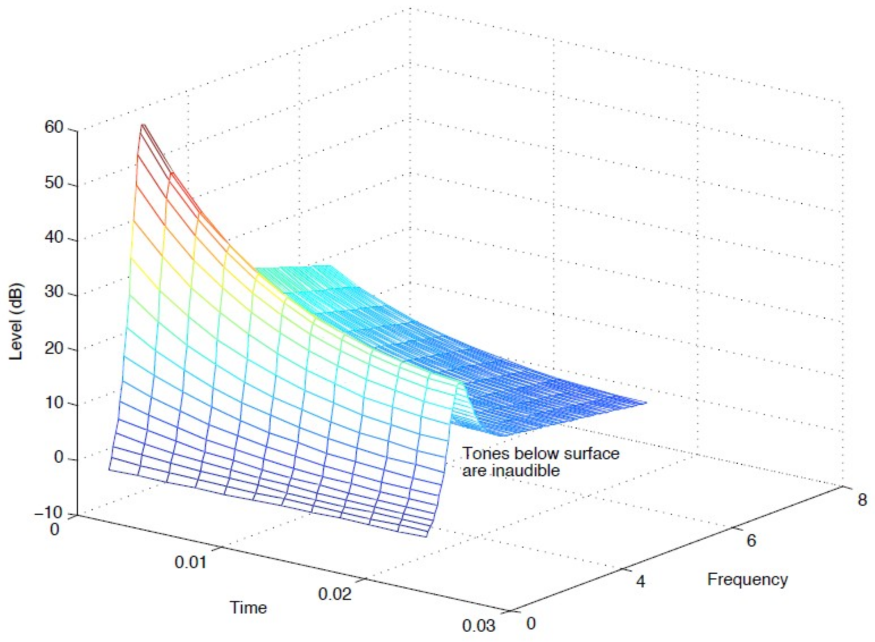
Critical bands

- Groups of hair cells respond to frequency range
  - Within the band a strong frequency overwhelms cells
  - Other frequencies are not detected
- About 24-25 critical bands
  - Sound will seem louder if it spans two bands
- Perceptual non-uniformity
  - Bandwidth constant below 500Hz (100Hz)
  - Bandwidth linearly increases above 500Hz



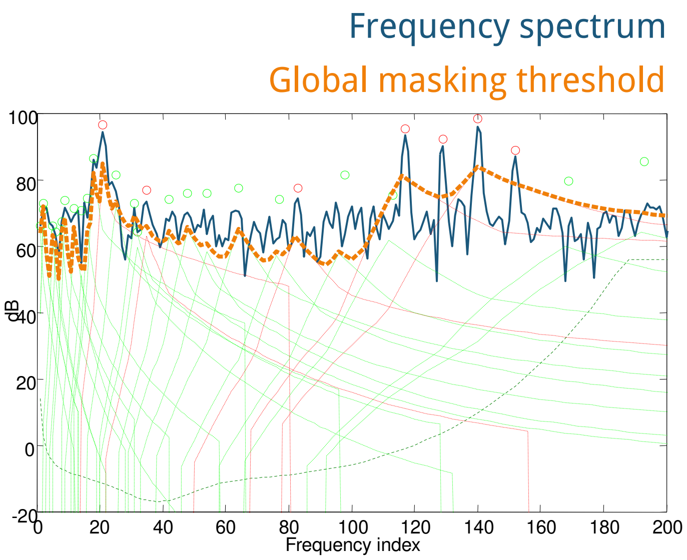
Temporal masking

- After a loud sound it takes time to hear quiet sound
  - Hair cells need a time-out
  - Duration depends on time and frequency similarity



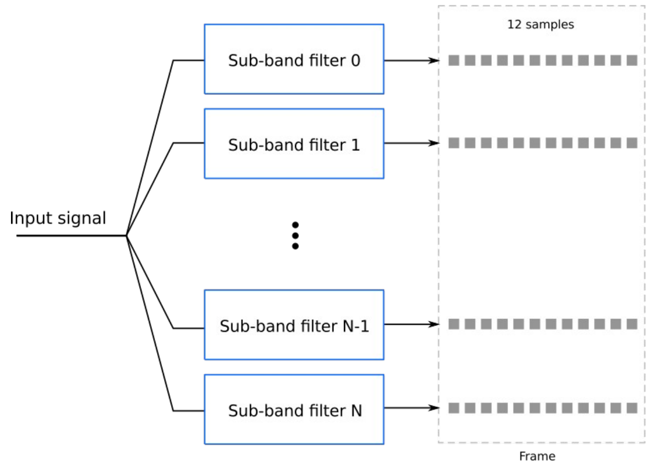
Psycho-acoustical model

- Compute masking levels
  - Frequency masking + absolute hearing threshold = global masking threshold (GMT)
- Signal-to-Mask Ratio, Mask-to-Noise Ratio
  - SMR = Signal – GMT (how much is signal louder than mask)
  - MNR = SNR – SMR (SNR based on quantization levels)
- Bit allocation
  - Quantization noise below GMR
  - Not always possible (low bitrate)
  - Distribute bits across bands based on MNR



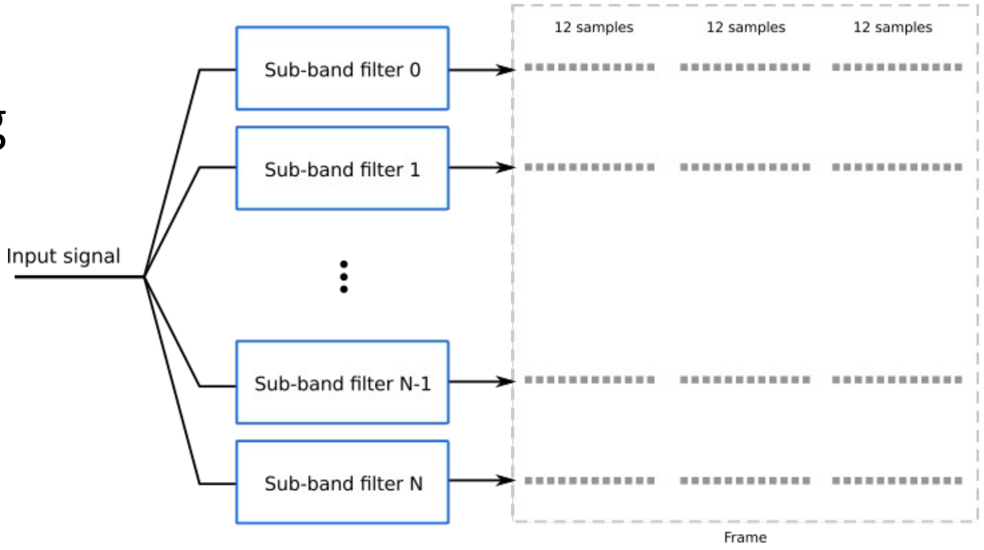
MPEG-1 Layer 1

- Intention: stored audio
- Critical band analysis
  - 32 filters, using FFT
  - Equal frequency spread per critical band
  - 12 samples in frame (scaled and quantized)
- Perceptual model
  - Only frequency masking



MPEG-1 Layer 2

- Intended use: digital audio broadcast
- Improvements
  - Use three groups together (3x12 samples)
  - Groups can share scaling factors
  - Supports frame skipping and sharing
  - Basic temporal masking



MPEG-1 Layer 3

- Intention: audio transmission over ISDN
- Improvements
  - Using MDCT instead of FFT
  - Non-uniform critical bands
  - Analysis-by-Synthesis noise allocation
  - Temporal masking
  - Stereo redundancy
  - Huffman coding

Stereo signal

- Intensity stereo coding:
  - Single summed signal and scale factors
  - Same signal, different magnitudes
- Middle/Side Stereo Mode:
  - Middle (sum of L and R) - M
  - Difference between channels - S

Decoding MPEG1 audio

- Psychoacoustic model is not required
- Quantization levels and scaling factors are used to reconstruct frequency bands
- Inverse frequency domain transform gives us waveform for decoded segment

MPEG-1 Overview

Layer	Target bitrate	Ratio	Quality @ 64 kbit	Quality @ 128 kbit	Theoretical min. delay
1	192 kbit	4:1	/	/	19 ms
2	128 kbit	6:1	2.1 – 2.6	4+	35 ms
3	64 kbit	12:1	3.6 – 3.8	4+	59 ms

Perceptual quality: 5 – perfect … 1 – very annoying

MPEG-2 Part 3 audio coding

- Backwards compatible
- Extensions
  - Multi-channel coding – 5.1 channel audio
  - Multilingual coding - 7 multilingual channels
  - Lower sampling frequencies
  - Optional Low Frequency Enhancement

MPEG-2 Part 7

- Advanced Audio Coding
- Not backwards compatible
- Increased complexity
- Up to 48 channels
- Used on DVD