Real-time H.264 Video Coding for Telemedicine Applications

short version + practical tasks

Andreas Unterweger
Salzburg University of Applied Sciences

About the presenter I

- H.264 video codec developer at the Fraunhofer Institute for Integrated Circuits in Erlangen, Germany in 2007 and 2008
- Graduated from Salzburg University of Applied Sciences in 2008
- Worked as IPTV software engineer until 2009
- Currently researching memory management optimization in H.264 video encoders

About the presenter II

- Working at the department of Information Technology and Systems Management at the Salzburg University of Applied Sciences
- Research assistant in "Industrial information technology" group, focused on test management and generic data conversion
- Teaching Digital Technology and Microcontroller Programming laboratories as well as Applied Mathematics tutorials

About the University I

Image source: http://www.fh-salzburg.ac.at

About the University II

- 18 departments in the following areas:
  - Information Technologies
  - Wood and Biogenic Technologies
  - Business and Tourism
  - Media and Design
  - Health and Social Work
- Department of Information Technolgy and Systems Management (ITS)

About ITS

- Bachelor and Master degree programs
- Specializations (Master degree program)
  - Embedded Signal Processing
  - Adaptive Software Systems
  - Convergent Network and Mobility
  - E-Health
- Bachelor and Master exchange programs with ESIGETEL (→ separate presentation)

Overview

- Introduction to image and video coding
- The H.264 standard
- Real-time aspects of H.264 video coding
- H.264 error resilience tools
- Practical work

Introduction to image and video coding

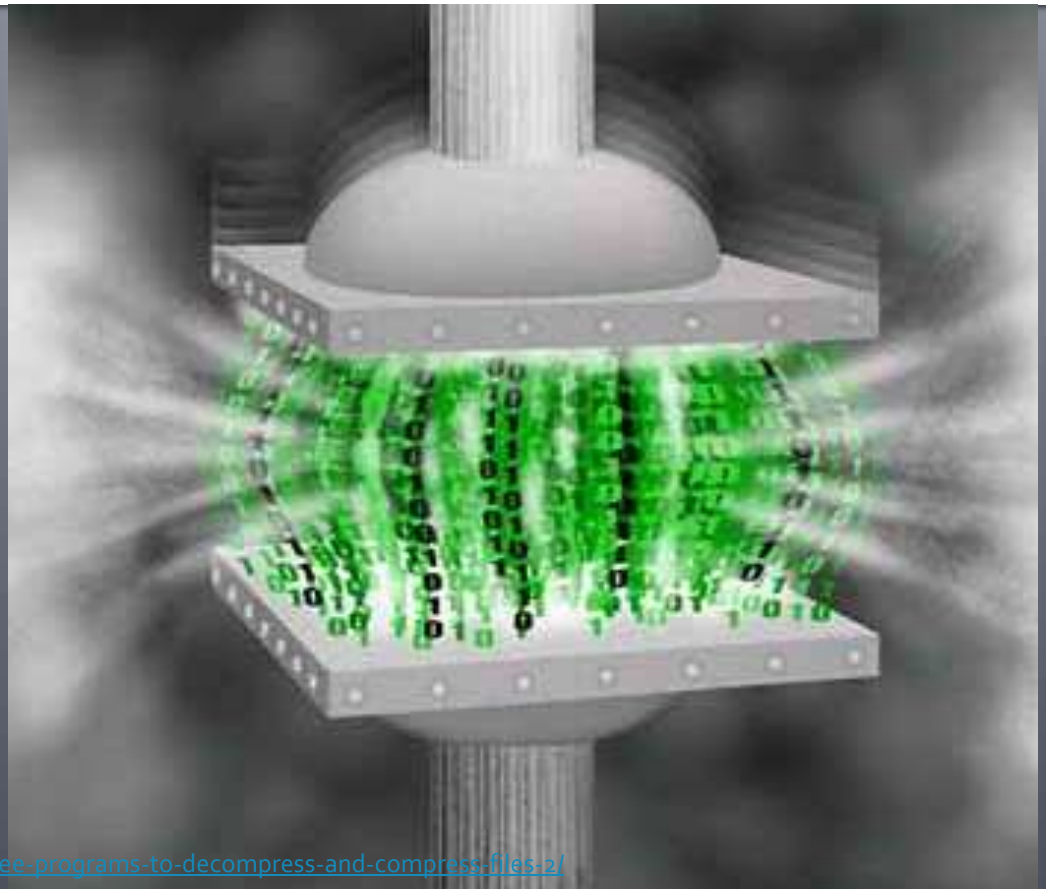
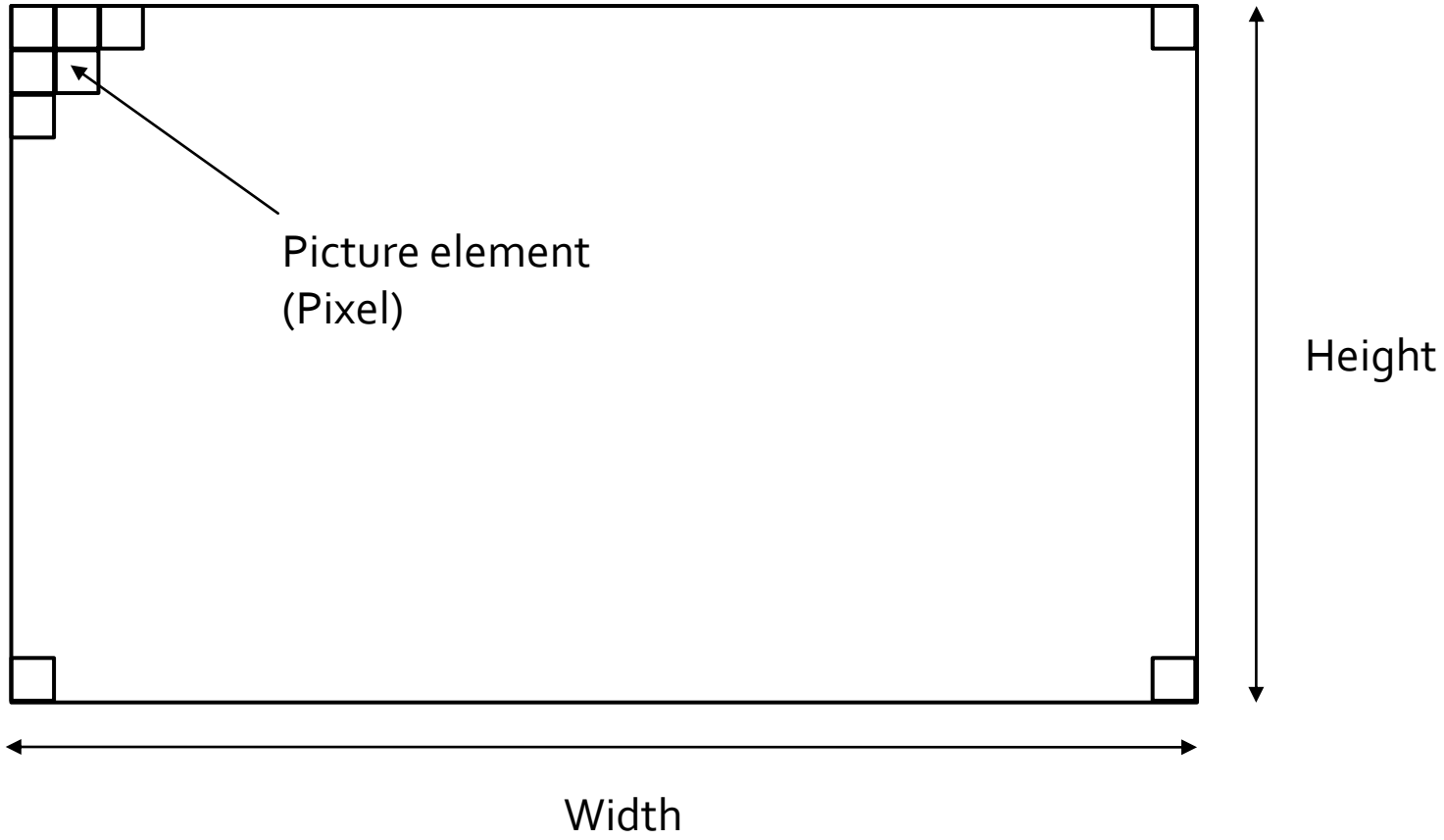


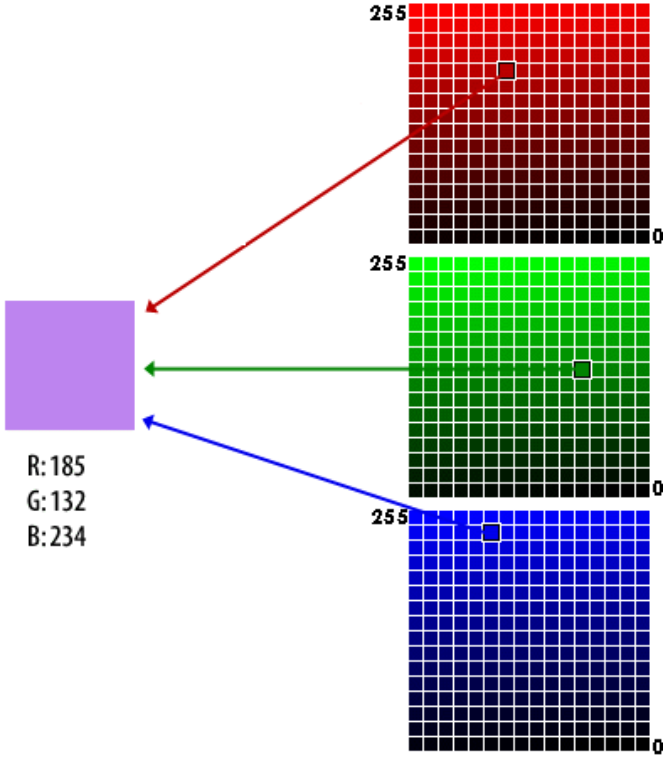
Image source: http://blog.toggle.com/free-programs-to-decompress-and-compress-files-2/

Digital images



Digital color images

- R, G and B information for each pixel
- Sum of components forms perceived color



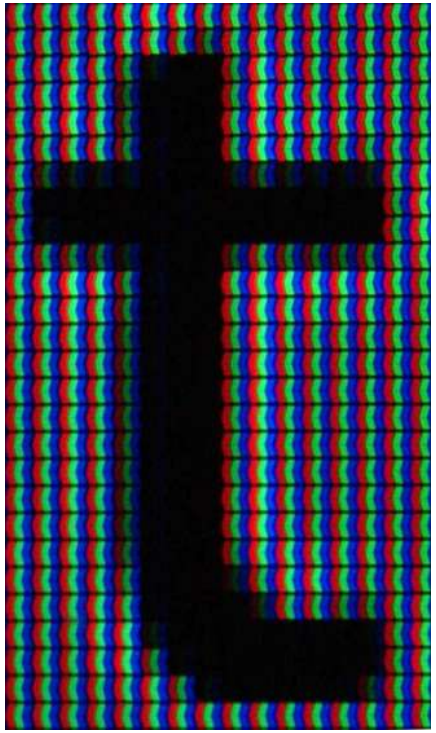


Image sources: Unknown. Please contact me if you know the sources

YUV color space

- Alternative representation of color information
- Luminance (Y) component and 2 color difference signals (U/C_B and V/C_R, "chroma")
- Conversion from and to RGB possible
- Why?
  - Compatibility to black/white television signals (luminance only, no chroma)
  - Possibility to easily reduce chroma resolution as the human visual system is more sensitive to luma

YUV images

Y

U

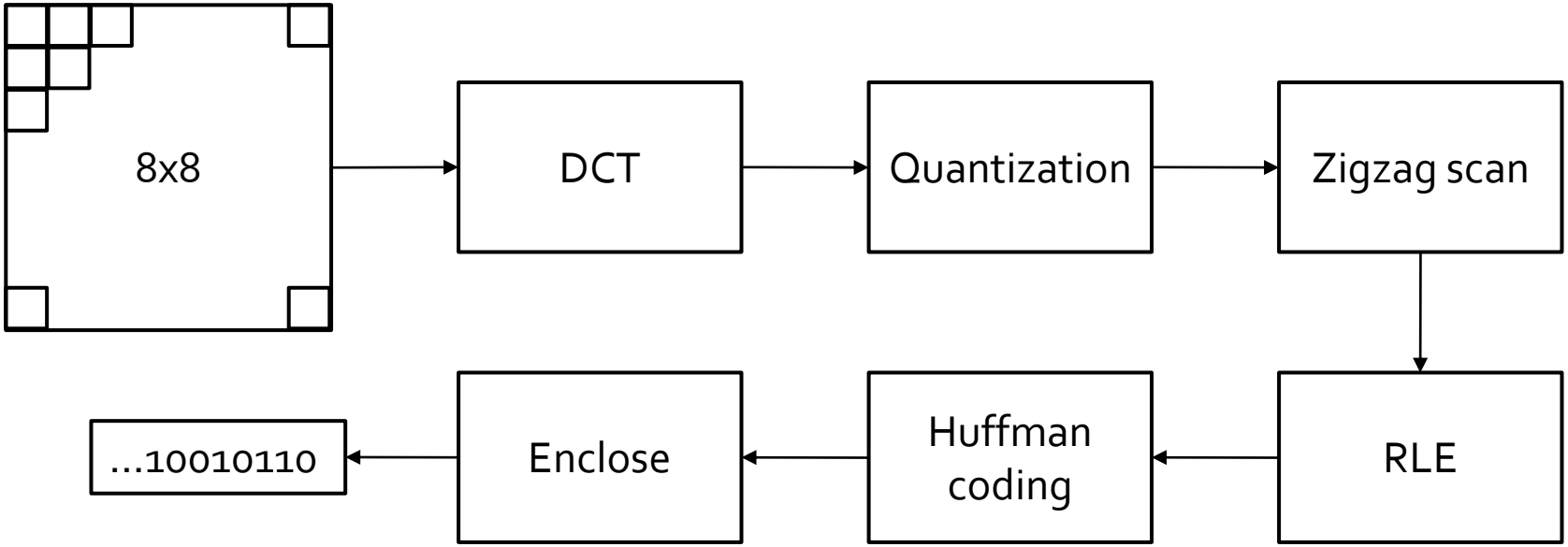
V

+ +

Image source: http://en.wikipedia.org/wiki/File:Barn-yuv.png

JPEG image compression

- Image split into blocks of 8x8 luma samples (macroblocks)
- Each macroblock is processed separately



1D DCT

- Discrete cosine transform (one-dimensional)
- Transformation of signal to frequency domain
- Cosines of different frequencies as basis functions → weighted sum of bases forms signal

- DCT:

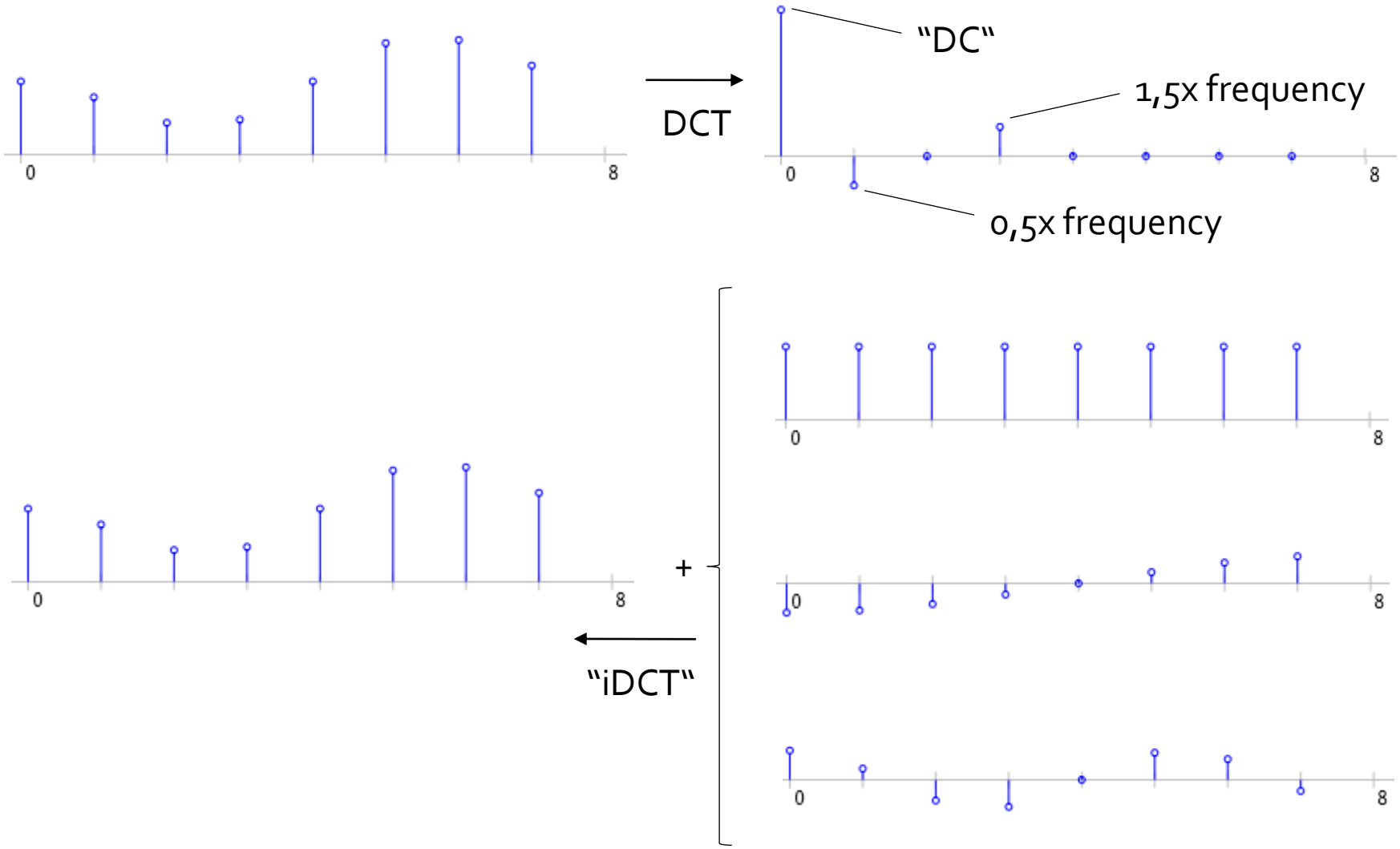
$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right] \qquad k = 0, \ldots, N$$

- iDCT

$$X_k = \frac{1}{2}x_0 + \sum_{n=1}^{N-1} x_n \cos\left[\frac{\pi}{N}n\left(k + \frac{1}{2}\right)\right]$$

Image source: http://en.wikipedia.org/wiki/Discrete_cosine_transform

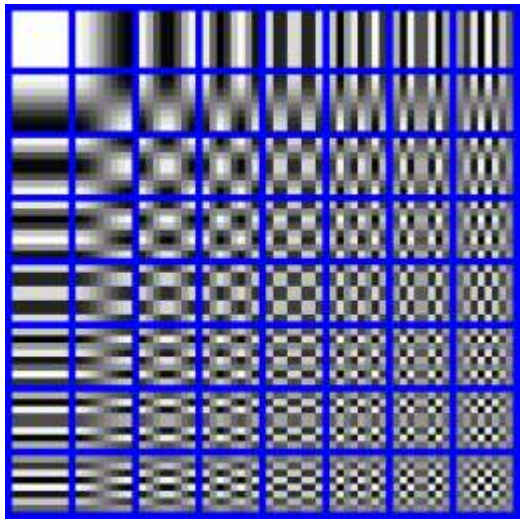
1D DCT example



2D DCT

- Two-dimensional extension of 1D DCT
- Spatial frequencies in gray images: change of gray (signal) value per pixel
- Basis images:

Higher vertical frequencies



Higher horizontal frequencies

Image source: http://en.wikipedia.org/wiki/File:Dctjpeg.png

2D DCT example

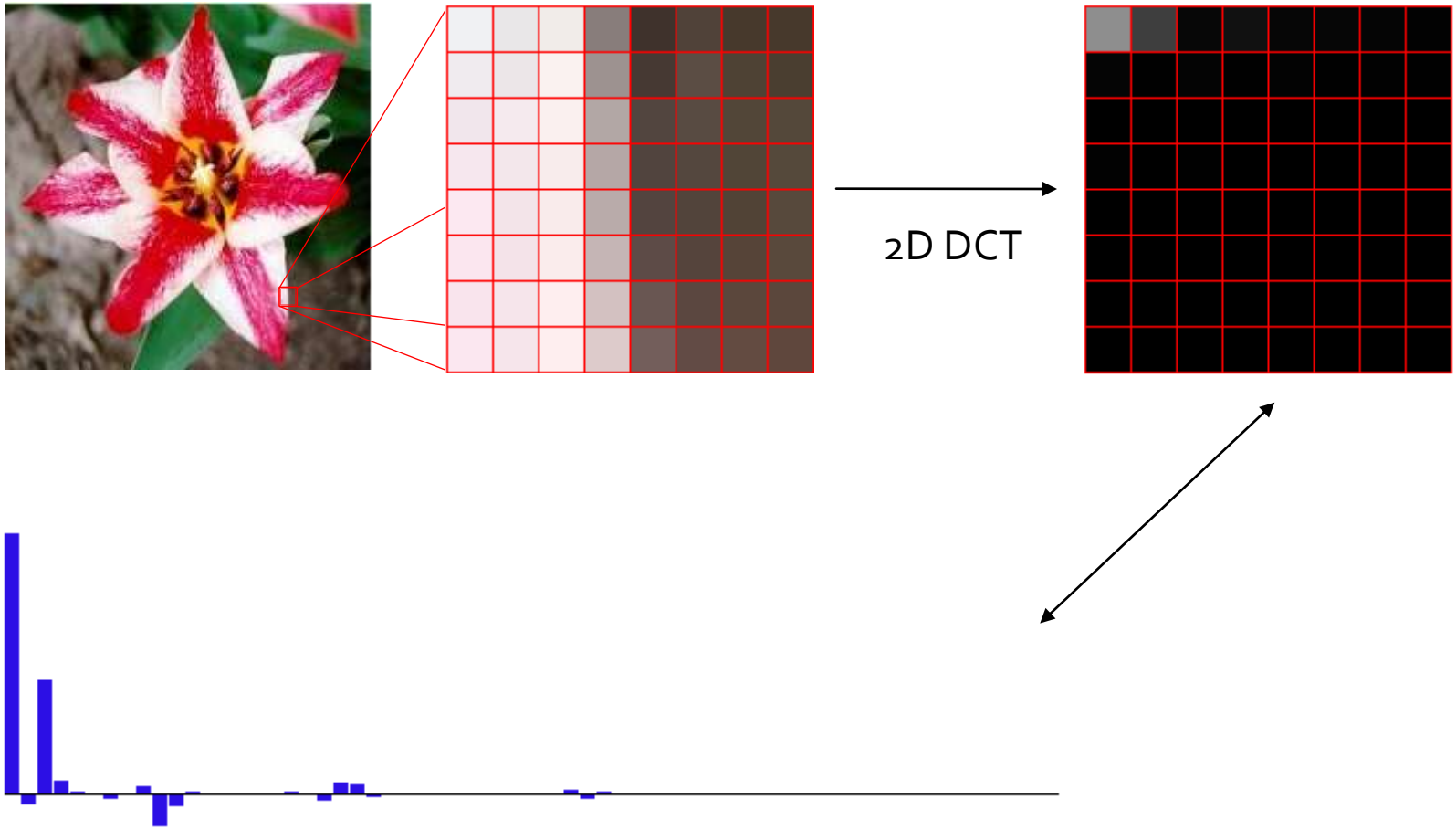


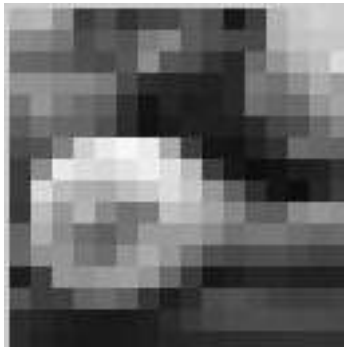
Image source: http://www.comp.nus.edu.sg/~cs5248/l01/DCTdemo.html

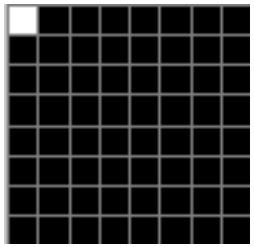
DCT properties

- Theoretically lossless (beware of floating point rounding errors!)
- Decorrelates input signal → simplifies compression
- Human visual system merely perceives loss of high frequencies → simplifies compression
- JPEG's 2D DCT works on shifted input data (8 bit values are shifted to a [-128;127] range)

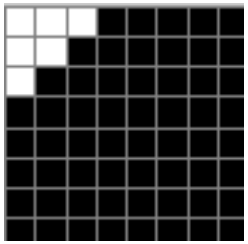
Effect of high frequency loss

Original

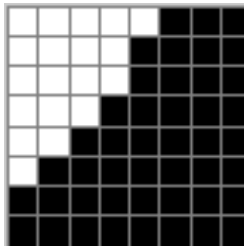




1 DCT coefficient
(all others zero)



6 DCT coefficients
(all others zero)



19 DCT coefficients
(all others zero)

Image source: Matlab DCT demo

Quantization

- Reduce number of possible values instead of eliminating coefficients in transform domain
- Division by factor, followed by rounding → less bits required for representation
- Quantization matrix Q specifies division factors for every coefficient → higher quantization factors for higher frequencies

$$B_{j,k} = \text{round}\left(\frac{G_{j,k}}{Q_{j,k}}\right) \text{ for } j = 0, 1, 2, \cdots, 7; k = 0, 1, 2, \cdots, 7$$

Image source: http://en.wikipedia.org/wiki/Jpeg

Quantization example

$$G = \begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.13 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ -48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.88 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix}$$

Quantization with Q

$$\text{round}\left(\frac{-415.38}{16}\right) = \text{round}\,(-25.96) = -26$$

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

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

Image source: http://en.wikipedia.org/wiki/Jpeg

JPEG quantization

- Quality in per cent (%) from 0 to 100 with according predefined quantization matrices
- Custom quantization matrices possible
- Separate quantization matrices for luma and chroma (higher quantizers for chroma)
- Be aware: 100% quality is not lossless (quantization matrix for 100% quality contains many 1 values, but not only!)

Quantization effects

Quality 100%

Quality 50%

Quality 25%

Quality 0%

Image source: http://www.fh-salzburg.ac.at

Zigzag scan

- Subsequent zeros can be compressed more efficiently → scan matrix in an order which makes subsequent zeros more likely
- Higher frequencies are likely to be zeroed by quantization
- Zigzag scan reorders coefficients from low to high frequencies

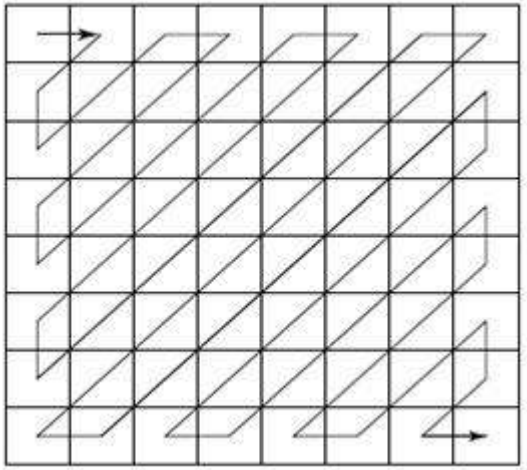
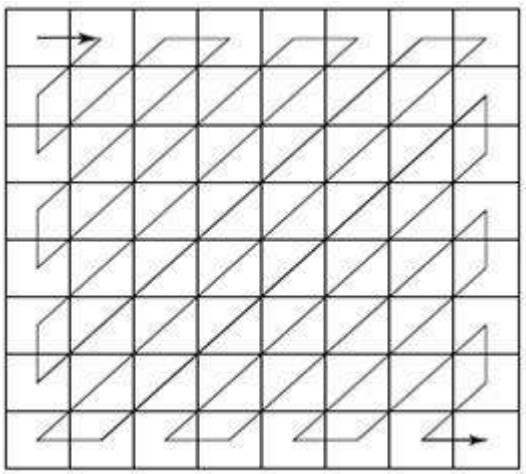


Image source: http://3.bp.blogspot.com/_6kpJik-Ua70/RkIT7NZCL_I/AAAAAAAAADE/acWC3QN2L24/s320/fig1.22.JPG

Zigzag scan example

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



−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
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							

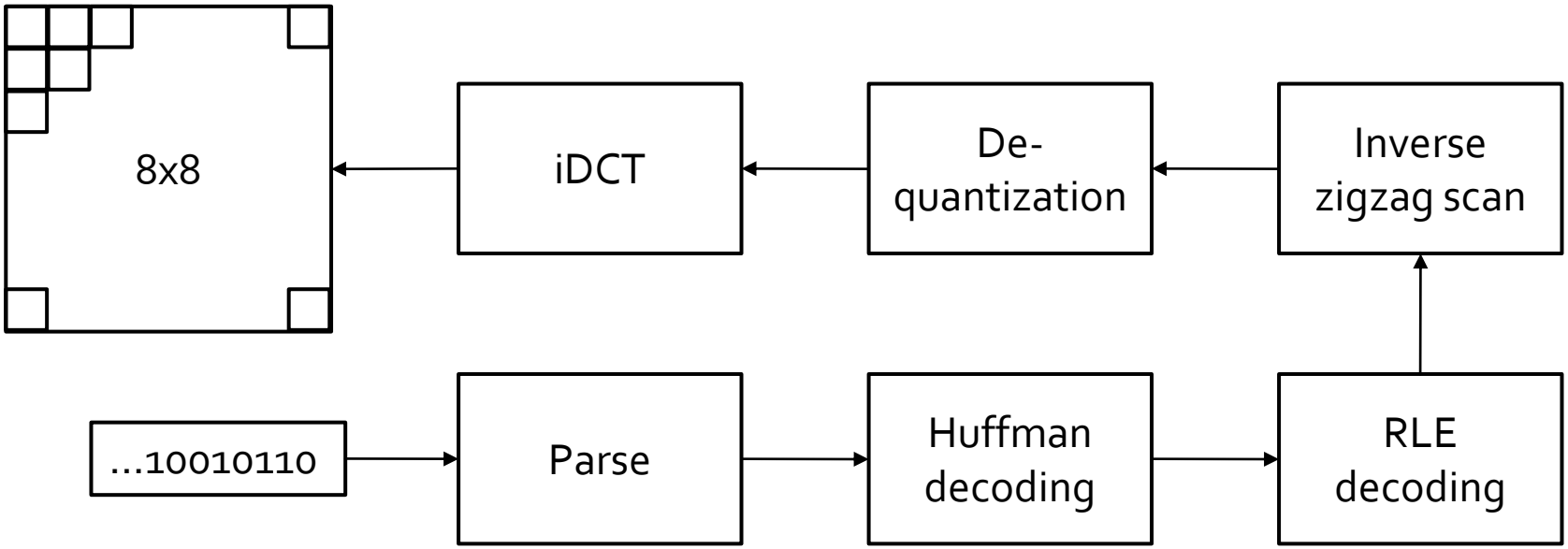
Image sources: http://en.wikipedia.org/wiki/Jpeg; http://3.bp.blogspot.com/_6kpJik-Ua70/RkIT7NZCL_I/AAAAAAAADE/acWC3QN2L24/s320/fig1.22.JPG

Entropy coding

- Lossless coding of coefficients scanned in zigzag order
- Run-length encoding (RLE) for sequences of zeros with special „EOB" marker at the end to indicate that the rest of the coefficients is zero
- Huffman coding for actual coefficient differences based on their probability
- Custom Huffman tables possible

JPEG decoding

- Inverse encoding process
- Dequantization is lossless; loss of data occured in quantization process during encoding
- Zero coefficients remain zero → artifacts



Compression artifacts I

- Blocking (macroblocks are encoded separately → edges don't fit together smoothly)
- Blurring (loss of too many high frequencies)

Blocking

Blurring

Image sources: http://en.wikipedia.org/wiki/File:Lenna.png; http://www.fh-salzburg.ac.at

Compression artifacts II

- Ringing (edges cannot be accurately represented by „continuous" base functions)
- Over- and undershooting („minus" red is cyan)

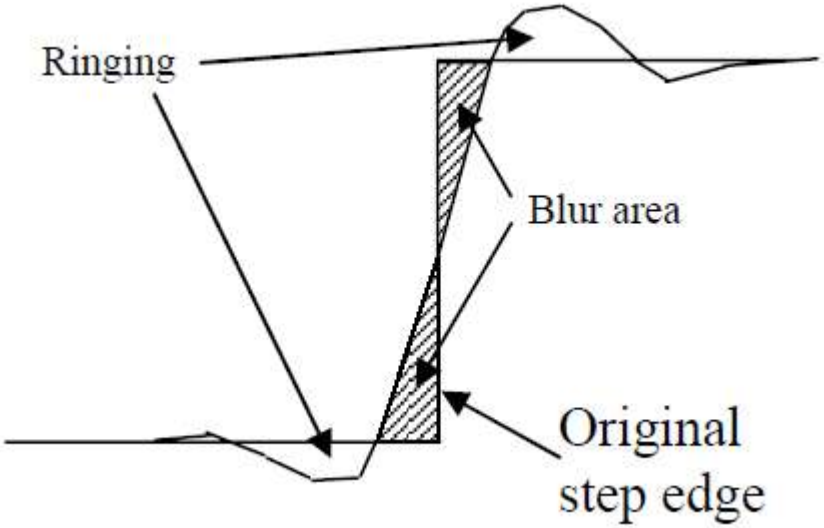


Image sources: http://en.wikipedia.org/wiki/File:Asterisk_with_jpg-artefacts.png; Punchihewa, G. A. D., Bailey, D. G., and Hogson, R. M.: Objective Evaluation of Edge Blur and Ringing Artefacts: Application to JPEG and JPEG 2000 Image Codecs. In Proceedings of Image and Vision Computing New Zealand 2005, pages 61-66, Dunedin, New Zealand, 2005

Compression artifacts III

- Basis function artifacts (very high quantization makes single basis functions visible)
- Stair case artifacts (diagonal edges cannot be represented accurately by horizontal and vertical base functions)

Quality measurement

- Different methods for objective and subjective quality measurement
- Subjective measurement using humans who rate the video quality on a defined scale (costly and time consuming)
- Objective measurement using mathematical formulas can only approximate subjective measurements but is less time consuming

Objective quality metrics

- Measure difference between original image/video and its encoded/decoded version
- Sum of absolute differences (SAD)
- Sum of absolute total differences (SATD)
- Sum of squared differences (SSD)
- Mean squared error (MSE)
- (Y-)PSNR derived from MSE (logarithmic)
- More metrics

MSE and PSNR

- MSE for m times n samples for original samples I and coded samples K

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

- Y-PSNR (luma only), $MAX_I = 2^8 - 1$ for 8 bit samples

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

- PSNR calculation for color pictures

Image source: http://en.wikipedia.org/wiki/PSNR

PSNR value examples

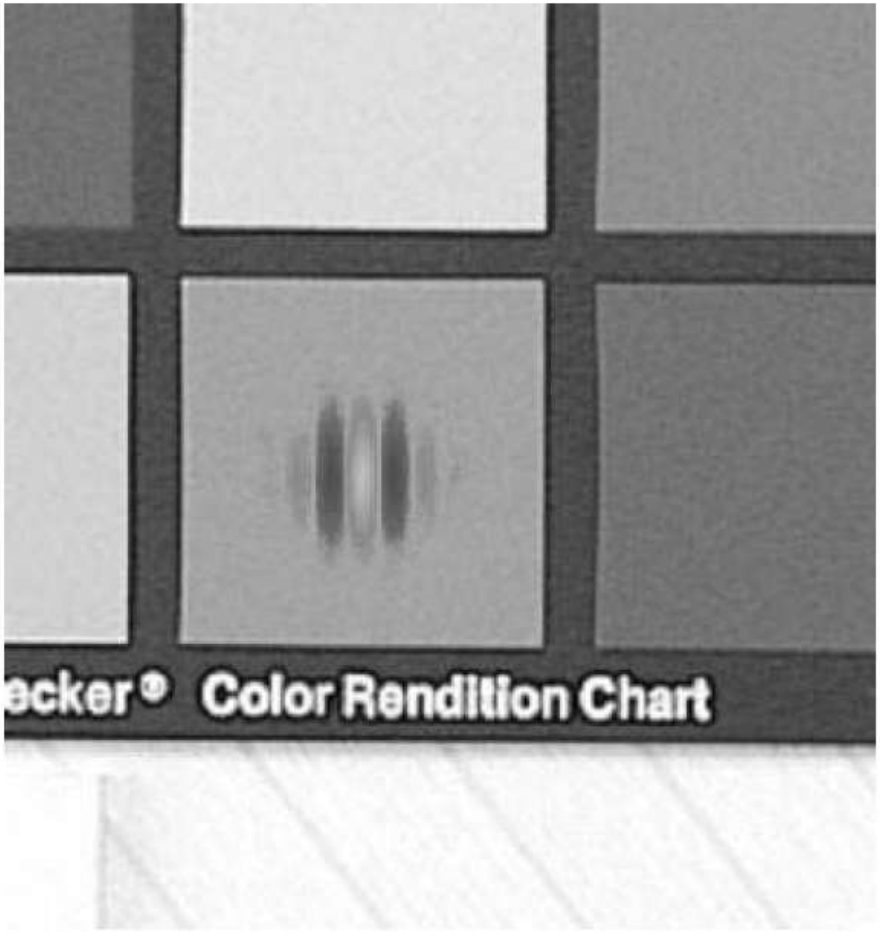
- Theoretically values between 0 and inf. dB
- Typical range: 25dB to 40dB
- 20dB or less usually indicate severe distortion

QP: 50 | PSNR: 20,45dB

QP: 30 | PSNR: 31,88dB

QP: 10 | PSNR: 45,63dB

One problem with objective metrics



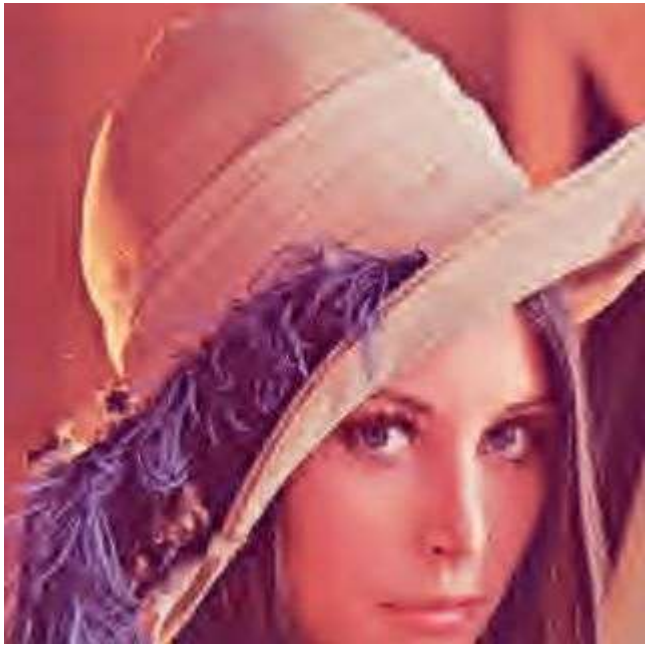
Both coded pictures have about the same MSE. Can you see why?

Image source: Nadenau, M.: Integration of Human Color Vision Models into High Quality Image Compression. PhD thesis, Ecole Polytechnique Federale de Lausanne, 2000

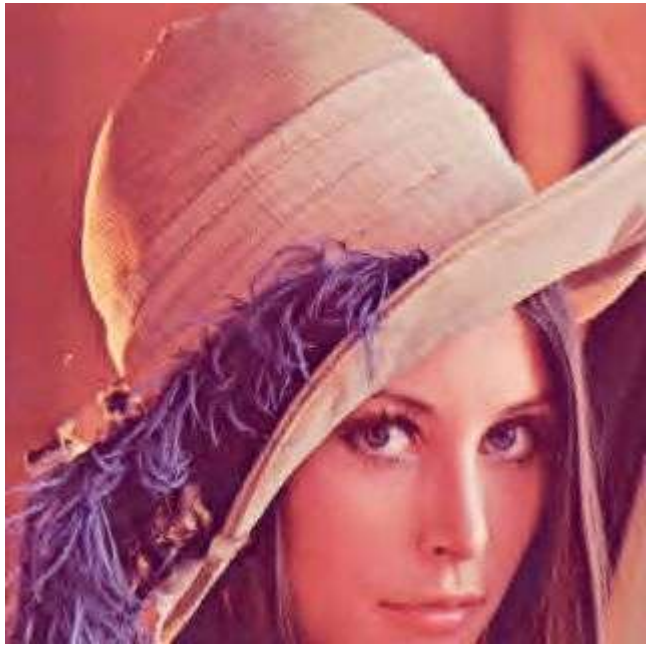
Structural similarity (SSIM)

- Approximate subjective quality measurement by measuring structual similarity of image blocks (typically 8x8 samples)
- Take contrast, blurriness and other paramters into consideration (similar to human eye)
- More complex to calculate than Y-PSNR, but also higher correlation with subjective quality measurements (typical trade-off)

Y-PSNR/SSIM example I



Y-PSNR: 32,08dB
SSIM: 0,865

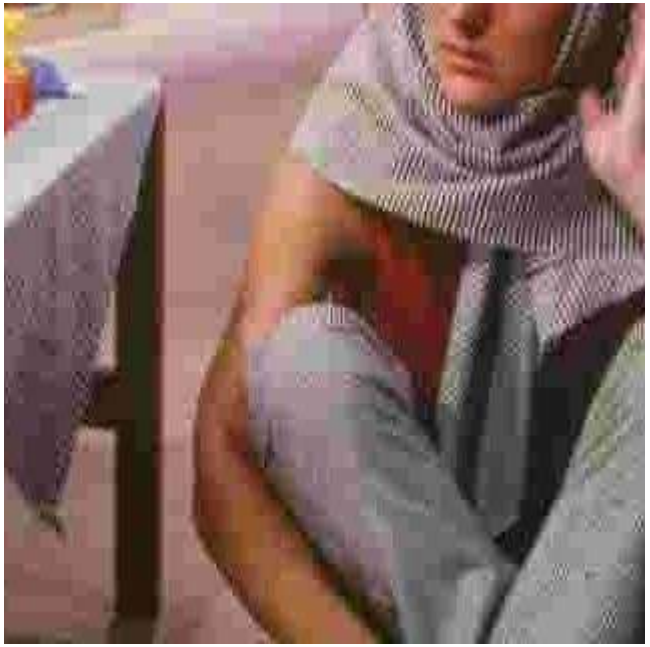


Y-PSNR: 35,50dB
SSIM: 0,921

Y-PSNR range: 0 to inf. dB (best)
SSIM: -1 to 1 (best)

Y-PSNR/SSIM example II

Y-PSNR: 30,81dB
SSIM: 0,866



Y-PSNR: 29,12dB
SSIM: 0,822

Y-PSNR range: 0 to inf. dB (best)
SSIM: -1 to 1 (best)

Video coding

- Videos („motion pictures“) are sequences of images with high temporal correlation
- Video coding uses image coding techniques and makes use of this temporal correlation
- Motion is perceived at ca. 20 pictures per second, fluid perception requires a higher frame rate (depends on content, individual, and lightning conditions)

Temporal correlation

- Pictures are similar to one another
- Exploit this property by coding differences between pictures instead of the pictures

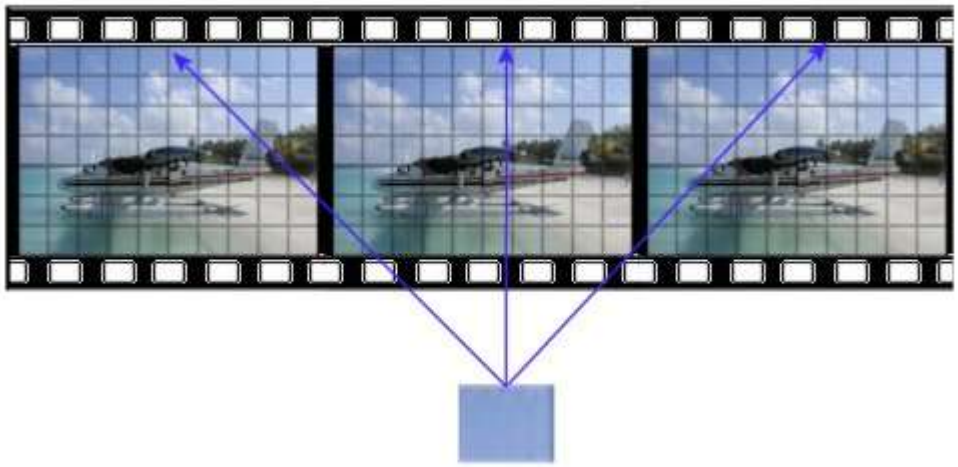


Image source: http://www.labdv.com/images/library/dv_basics/mpeg_explained/tempcompress.jpg

Pictures and picture types

- Pictures (frames) can be of different types
  - I (Intra) pictures/frames: Fully coded similar to JPEG algorithms described earlier ("key" frames, relevant for scene changes and fast forwarding)
  - P (Predicted) pictures/frames: Only differences to previous pictures/frames are coded (use of temporal correlation between pictures)
  - More picture types to be described later

Motion estimation

- Used in P pictures to find similar macroblocks in previously coded pictures
- Motion estimation: find best match

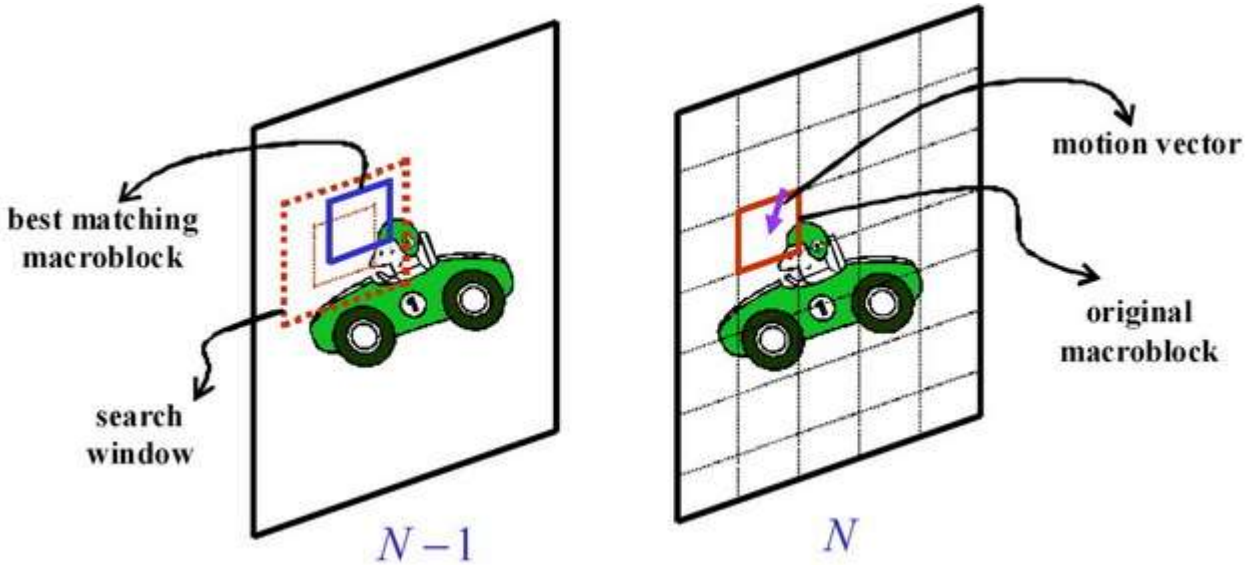
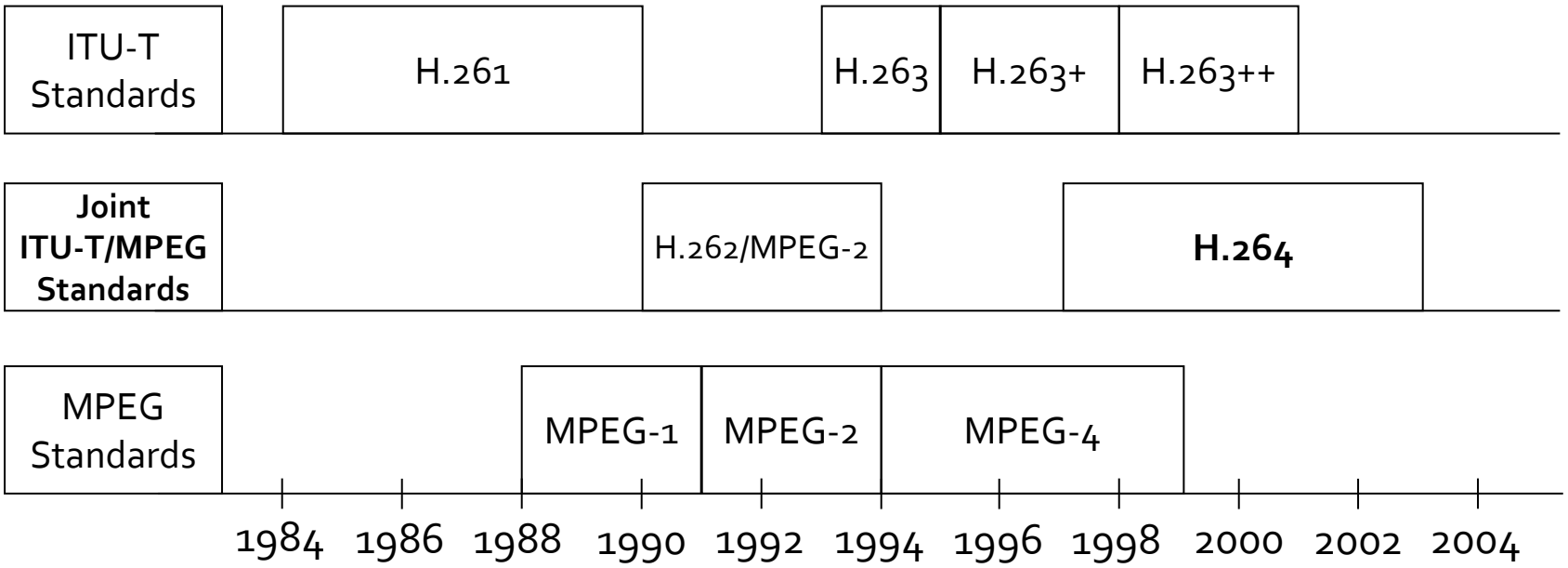


Image source: http://www-sipl.technion.ac.il/UploadedFiles/BlockBasedMotionEstimation.jpg

Motion coding and decoding

- Calculate difference between macroblock match and current macroblock
- Difference is treated like a regular macroblock (transform, quantization etc.)
- To be saved additionally: motion vector and match picture "number" (or distance)
- Reconstruction by applying difference to reference macroblock in decoder

Video coding standards time line



The H.264 standard

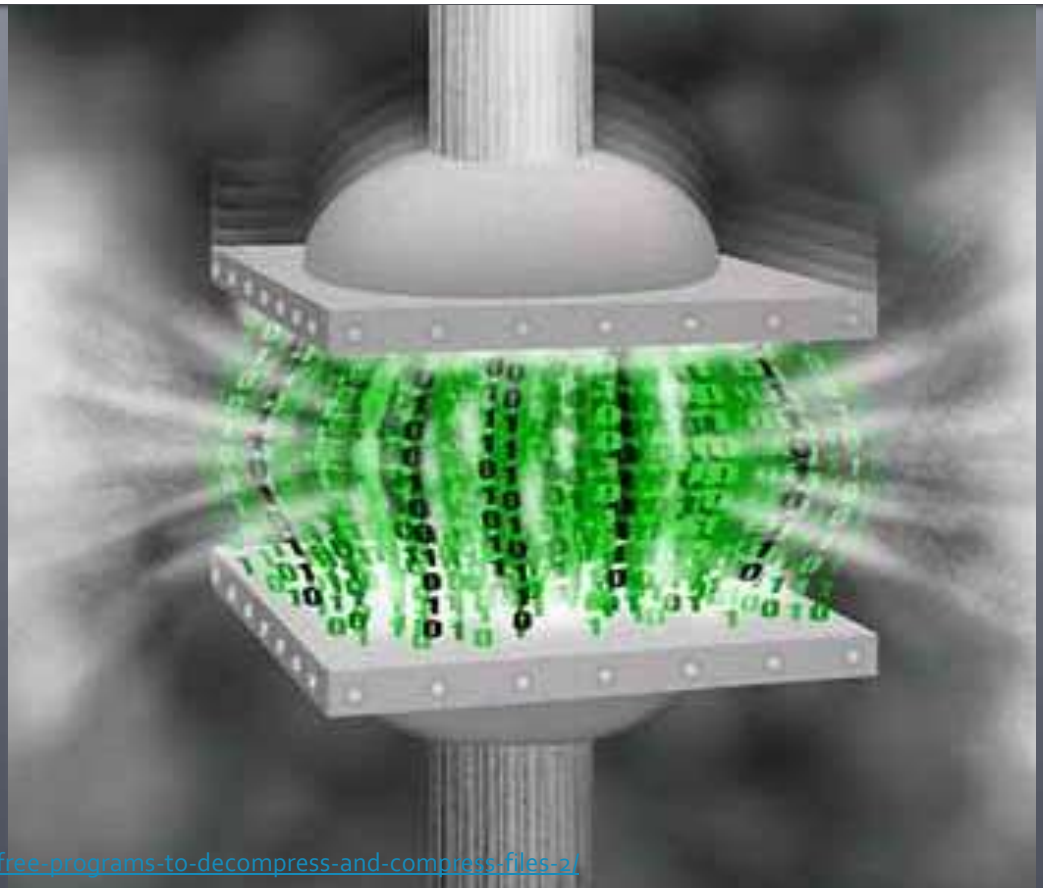


Image source: http://blog.toggle.com/free-programs-to-decompress-and-compress-files-2/

H.264

- State-of-the-art video (de)coding standard
- Only specifies how decoding is to be done
  - Encoder design can be arbitrary as long as the syntax of the generated bit stream is valid
  - No guarantee for a specific quality as encoder decisions are not specified

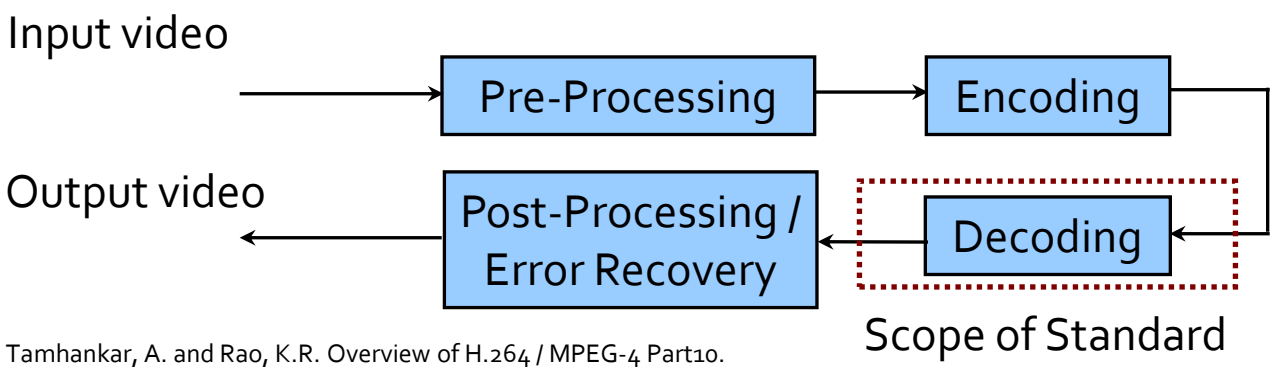


Image source: Kwon, S., Tamhankar, A. and Rao, K.R. Overview of H.264 / MPEG-4 Part10. http://www.powershow.com/view/19bc9-NDQyM/Overview_of_H_264_MPEG4_Part10_powerpoint_ppt_presentation (14.10.2010), 2004.

H.264 coding structure

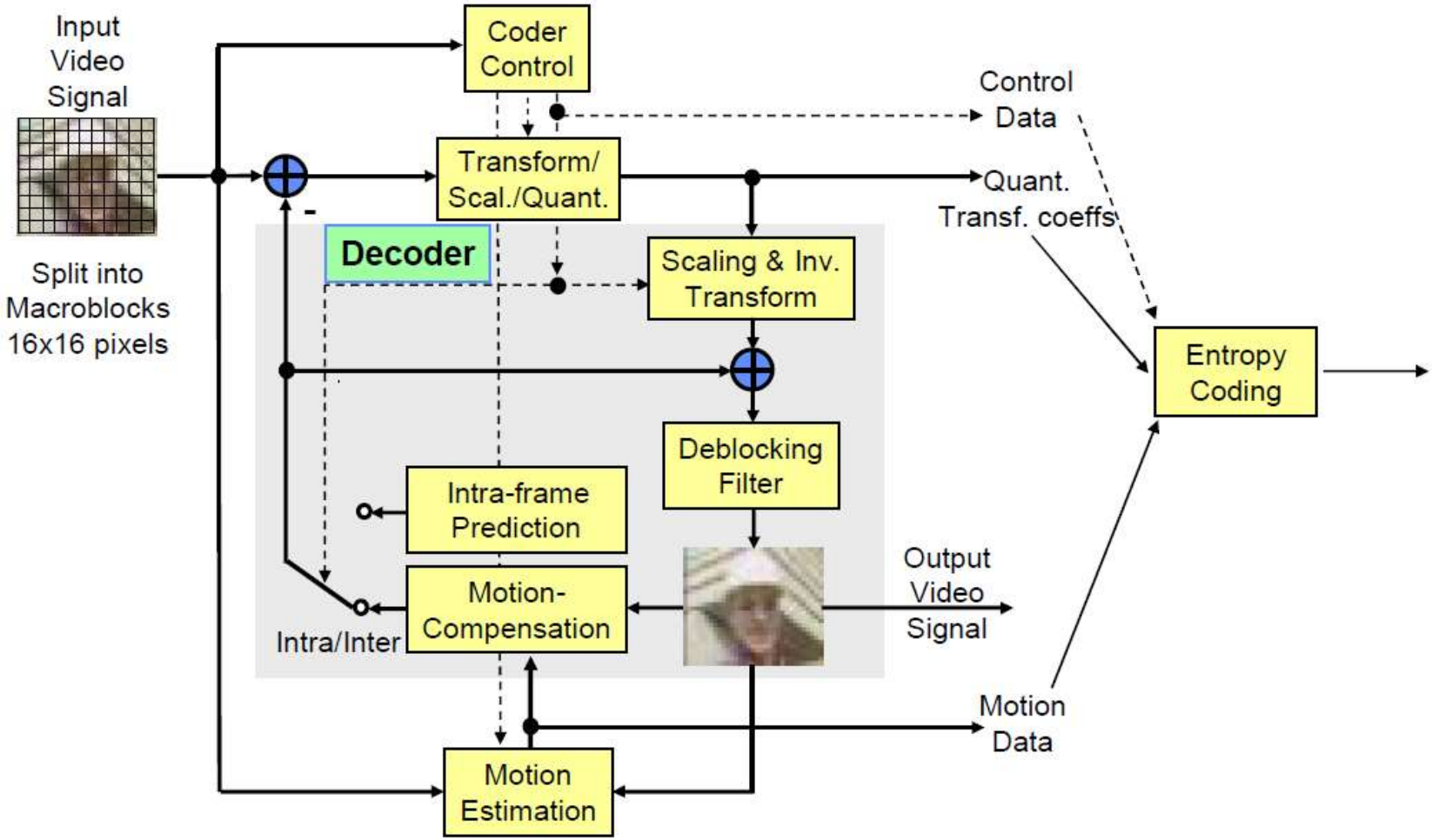
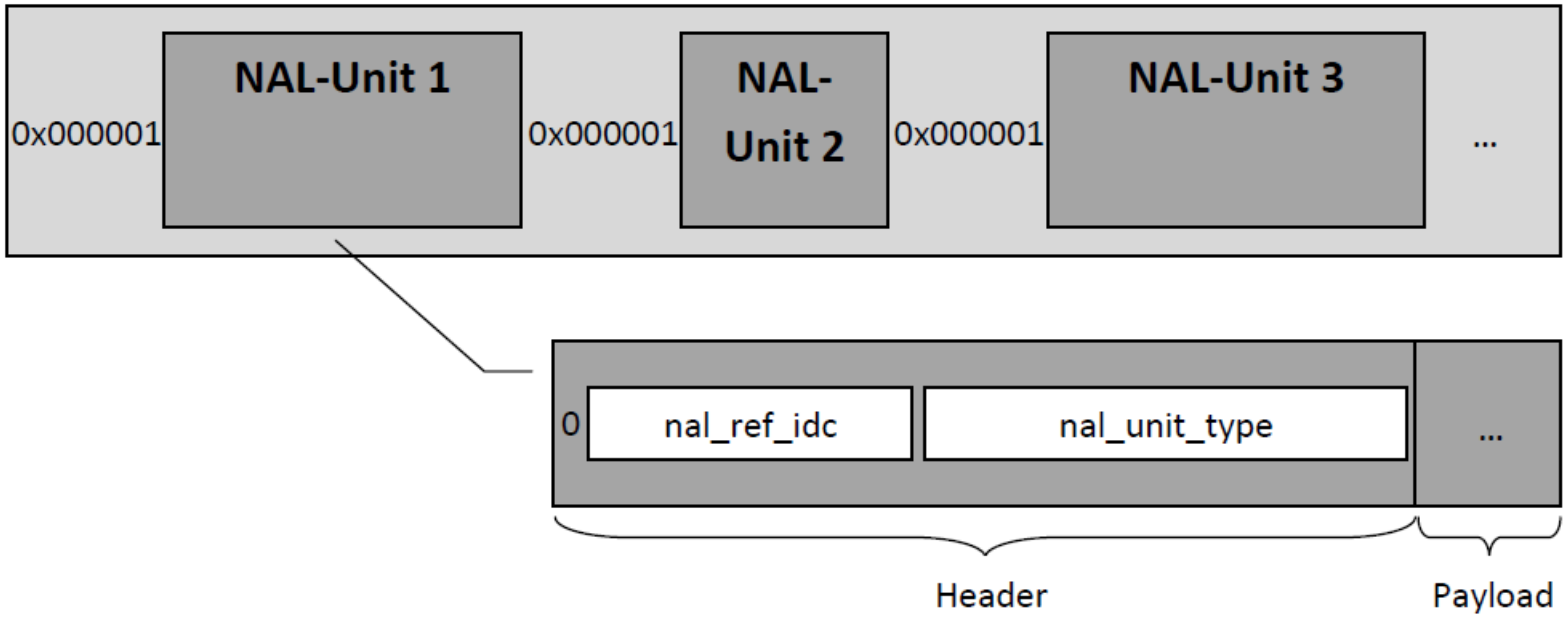


Image source: Wiegand, T. and Sullivan, G. J.: The H.264 | MPEG-4 AVC Video Coding Standard. http://ip.hhi.de/imagecom_G1/assets/pdfs/H264_03.pdf (14.10.2010), 2004

NAL units (NALUs)

- Store pictures (or parts of them)
- Allow for easy transmission over networks (one NALU per package)



Differences to JPEG compression

- 16x16 macroblocks instead of 8x8 blocks
- 16 4x4 blocks per macroblock which are transformed separately
- Integer transform (similar to DCT) to avoid floating point rounding errors and allow for better performance
- Different scanning order
- Additional filtering
- More efficient entropy coding

Intra prediction

- Intra coded macroblocks are not fully coded, but predicted from their neighbours (use of spatial correlation within picture)
- Either 4x4 or 16x16 block prediction possible
- Blocks at picture borders are also predicted
- Intra coded macroblocks can also appear in P pictures if no appropriate match is found during motion estimation

16x16 intra prediction modes

- Encoder can choose between single 16x16 intra prediction or 16 separate 4x4 predictions
- Less signalling bits for 16x16 modes required
- 4 possible 16x16 modes, 9 4x4 modes

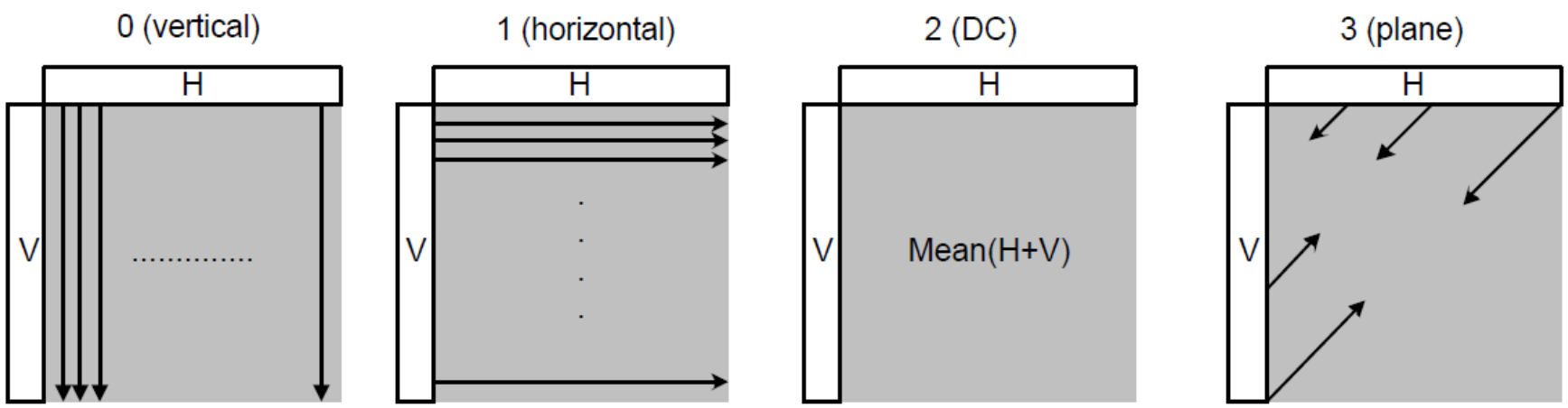


Image source: Richardson, I. E. G.: H.264 / MPEG-4 Part 10: Intra Prediction. http://www.vcodex.com/h264.html (16.1.2008), 2003.

16x16 intra prediction example

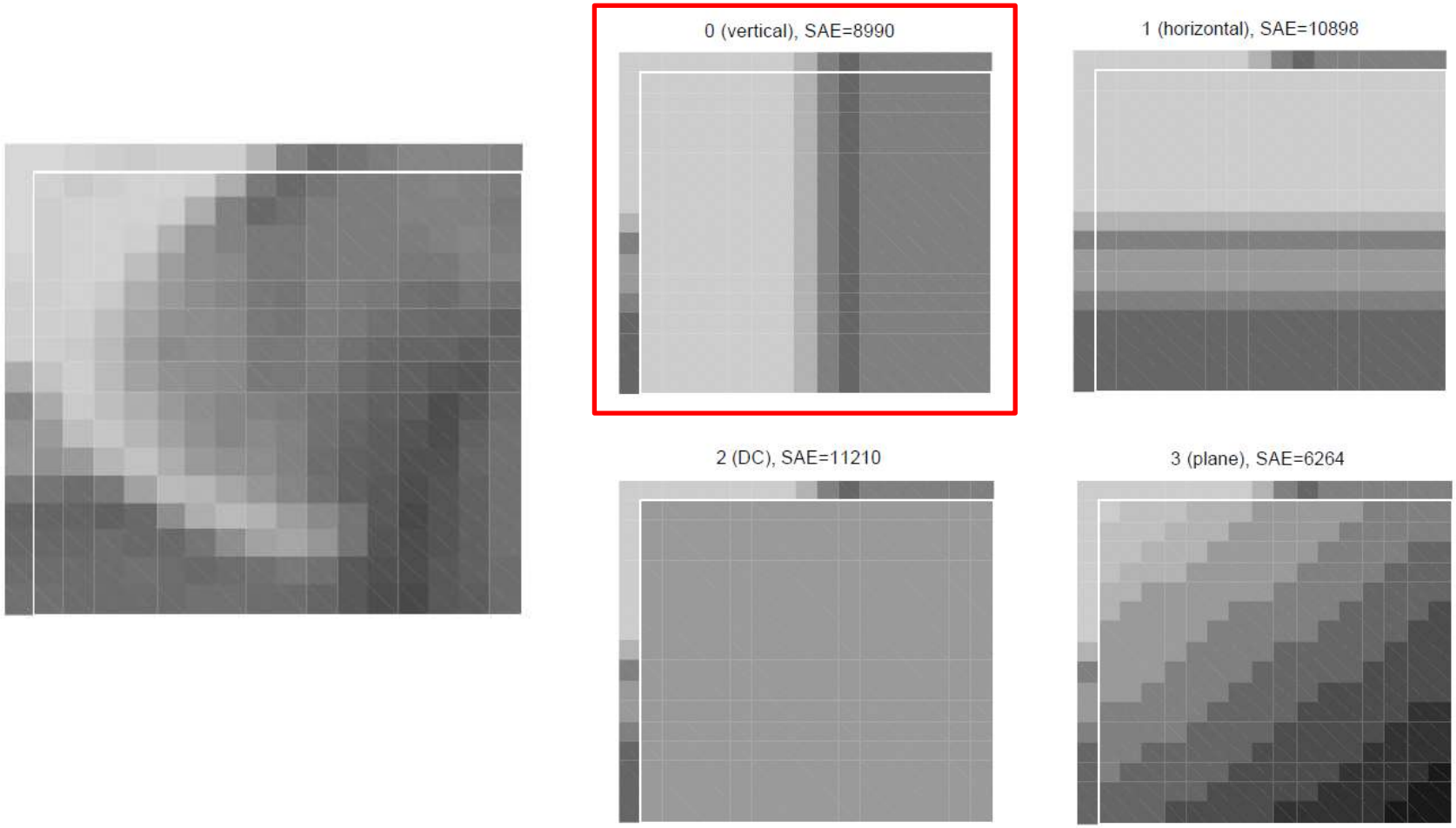
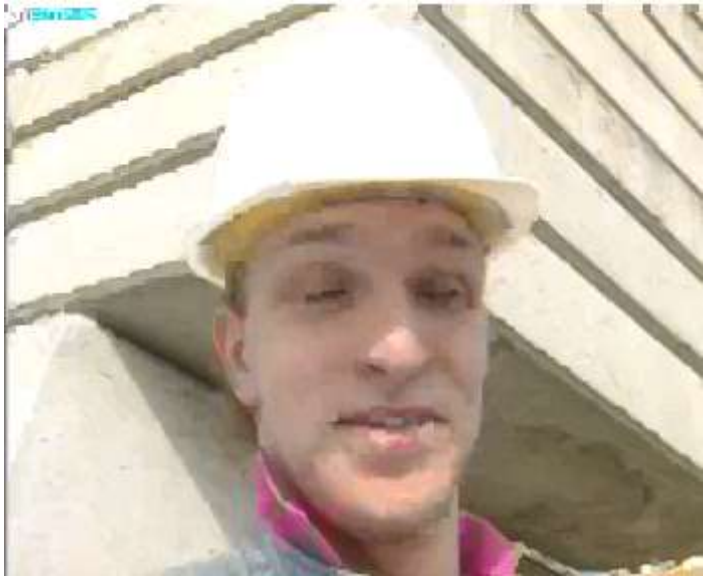


Image source: Richardson, I. E. G.: H.264 / MPEG-4 Part 10: Intra Prediction. http://www.vcodex.com/h264.html (16.1.2008), 2003.

Intra prediction efficiency

- Prediction vs. final picture



- 16x16 modes in helmet area, rest: 4x4 modes

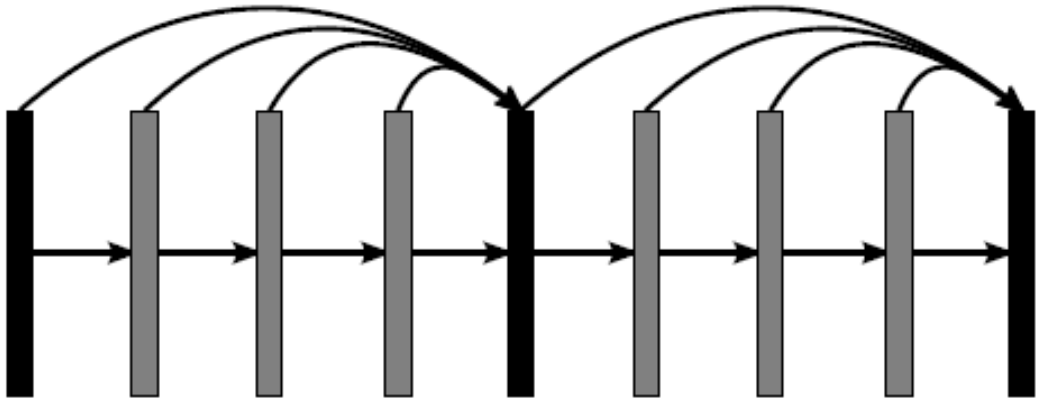
Image source: Richardson, I. E. G.: H.264 / MPEG-4 Part 10: Intra Prediction. http://www.vcodex.com/h264.html (16.1.2008), 2003.

B pictures

- Bidirectional prediction also allows prediction based on "future" frames
- Allow references to past or to "future" pictures, depending on what costs less bits
- Coding order has to be adapted as real "future" prediction is not possible
- Coding order and display order differ if B pictures are used

Coding order vs. display order I

- H.264 allows coding order to differ arbitrarily from display order
- References have to be coded first



- Coding order: **0 2 3 4 1 6 7 8 5**
- Display order: 0 1 2 3 4 5 6 7 8

Inter prediction

- Recall: motion estimation and compensation
- Sophisticated partitioning of picture
- Use of multiple reference pictures
- Sophisticated management of reference pictures
- Recall: picture types I, P and B

IDR pictures

- "Instantaneous Decoding Refresh"
- Coded like I pictures, but in special NAL unit
- Difference to I pictures: prediction border (no references allowed to frames before IDR)
- Used as resynchronisation points
  - At the beginning of a video stream
  - Fast forward
  - After package loss

H.264 implementations

- Large number of commercial encoders and decoders, few free implementations
- Both encoder and decoder are very complex
- Decoders are equal in quality (at least in theory) as the standard defines how decoding has to be done → decoder speed matters
- Encoder quality and feature implementations vary as the encoder design can be arbitrary

Popular implemenations

- Decoders
  - ffmpeg (open source)
  - CoreCodec CoreAVC (commercial)
- Encoders
  - x264 (open source)
- Codecs (encoders and decoders combined)
  - MainConcept (commercial)
  - DivX H.264 (free and commerical version available)
  - Nero Digital (commercial)

Real-time aspects of H.264 video coding

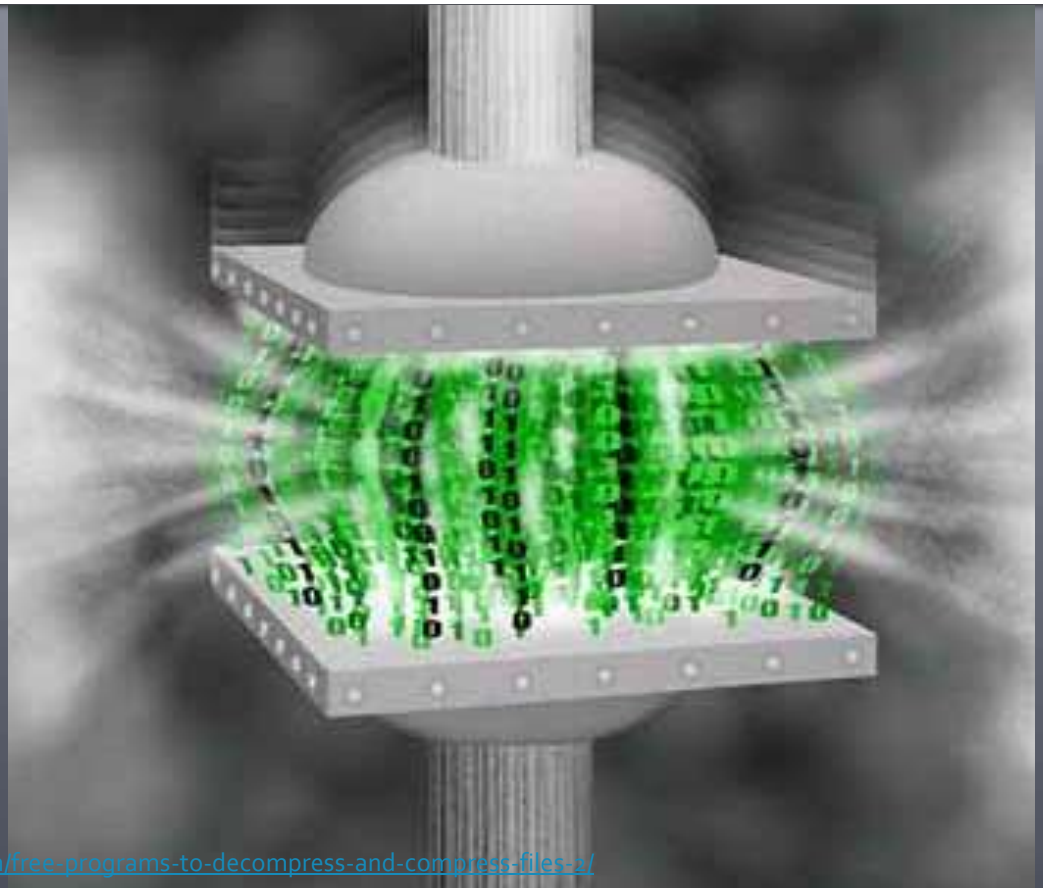


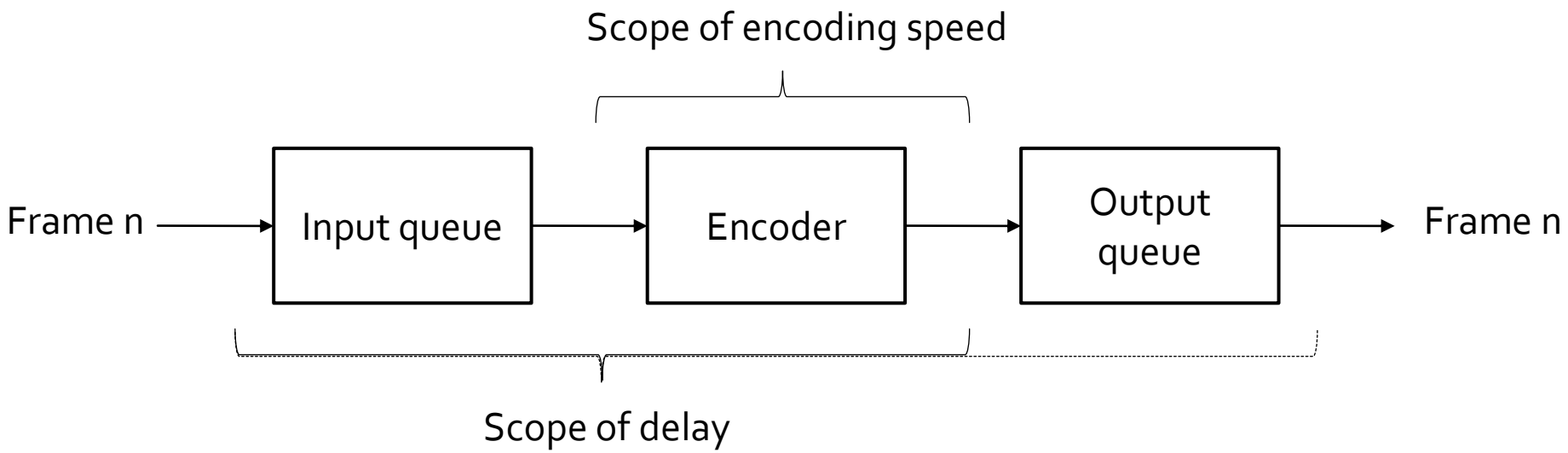
Image source: http://blog.toggle.com/free-programs-to-decompress-and-compress-files-2/

Definition of real-time

- Video coding concept of real-time is different than in classical real-time systems with soft or hard deadlines
- Frame (processing) rate of encoder has to be greater than or equal to frame rate of video in time average (in theory!)
- Delay issues have to be coped with separately

Encoding speed and delay

- Encoding speed is frame processing rate of encoder only (also depends on input)
- Delay includes queues and more (ambiguous)



Delay

- Multiple definitions
  - Encoding delay (duration of pure frame encoding)
  - Encoder delay (duration from frame input to output by encoder; greater than or equal to encoding delay – beware of reordering!)
  - End-to-end delay (duration between recording and display for live streams; should be minimal)
- Definition of delay depends on application
- Telemedicine applications require low delay

Live end-to-end delay sources

Camera processing delay

Camera-to-encoder transmission delay

Encoder delay

Packaging and transmission delay

Decoder delay

Display delay

Image sources: Unknown. Please contact me if you know the source; http://www.popjolly.com/wp-content/uploads/2010/04/surgeons-300x194.jpg

Delay example I

- Example: 25 fps recording (40ms per frame)
- Average recording delay for one frame: 20ms
- Average encoder delay also ca. 20ms (jitter)
- Transmission depends on channel and buffers: >40ms (longer for TCP/IP and similar channels)
- Decoder has to decode whole picture: average delay of 20ms (jitter due to different frames)
- 50 Hz display adds another min. 10ms delay

Delay example II

- Total delay depends on various sources
- At 25fps, the minimum delay is at least 100ms (nearly 3 whole frames)
- Frame rate has significant impact (increasing the frame rate lowers the delay)
- Some sources of delay cannot be minimized
- Some can, but require additional efforts (extra hardware/software/money etc.)

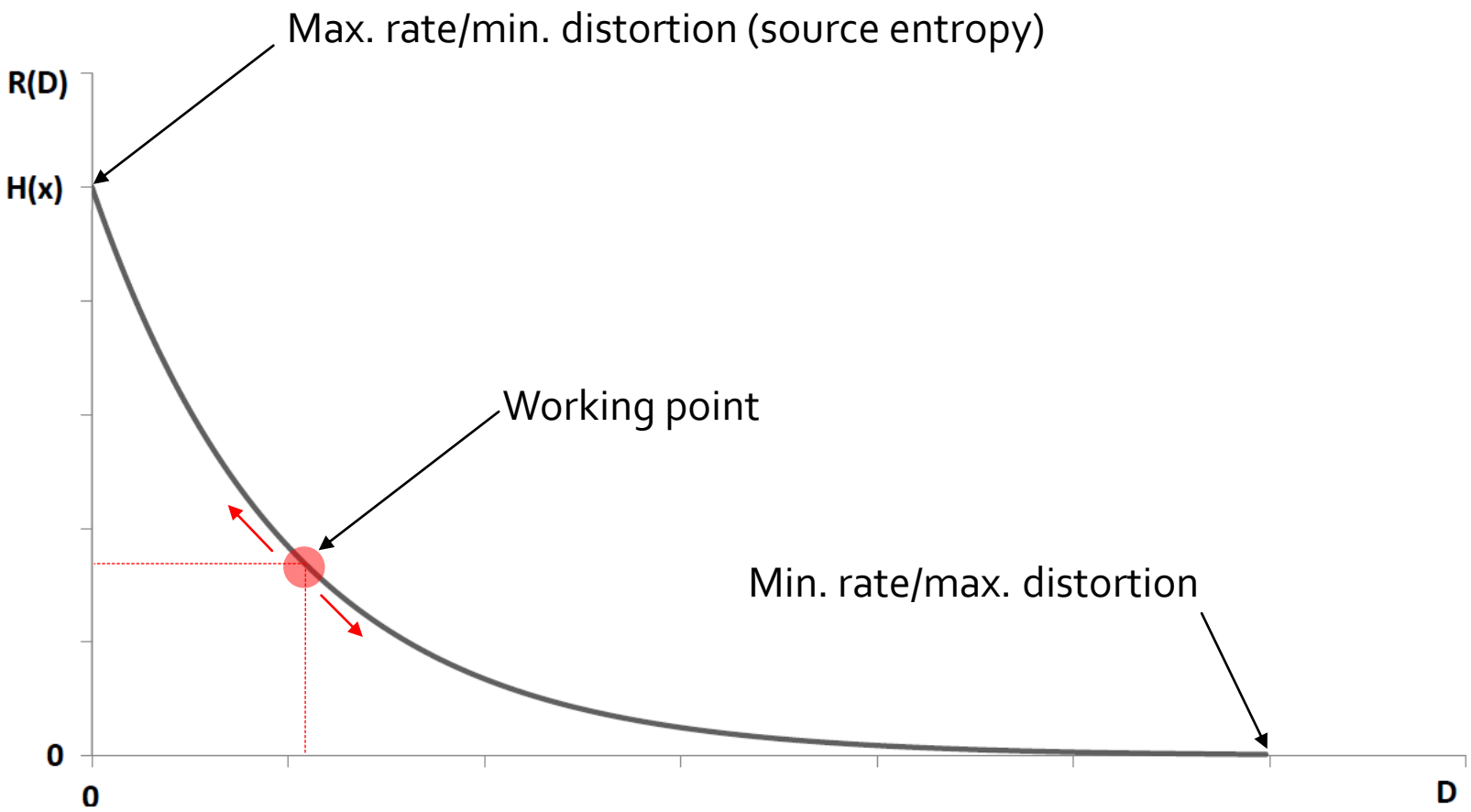
How to lower end-to-end delay

- Lower delay of each delay source
- Focus on main sources of delay
  - Transmission (faster connections, prioritized package handling etc.)
  - Encoding (faster encoder, different parameters)
- Decoding complexity is influenced by encoder and its parameters ("ultra low delay" (ULD) encoders/decoders help)
- Transmission delay is a network layer issue

Rate-distortion optimization (RDO)

- Video coding is always a trade-off between quality and bit rate
- Quality is "inverse" distortion
- Quality goal: best quality at given bit rate
- Bit rate goal: lowest bit rate at given quality
- Rate distortion optimization: achieve goals
- Base encoder decisions on RDO

Rate distortion curve



Effect of coding tools/parameters

- RDO requires to "try" each mode (affects time)
- Higher resolution of input video entails more encoding operations (affect time)
- B pictures require picture reordering (affects delay); combined with RDO there are yet more modes to test (affect time)
- Search range and number of pictures for motion estimation prolong search (affect time)

Practical parameters example

- x264 (open source H.264 encoder) uses presets to specify coding speed vs. compression efficiency trade-off
- --preset parameter allows "ultrafast" to "veryslow" (and even "placebo")
- Default settings: 3 B pictures, 3 reference pictures for motion estimation with a range of 16x16 each (and some more restrictions)

Practical parameters example I

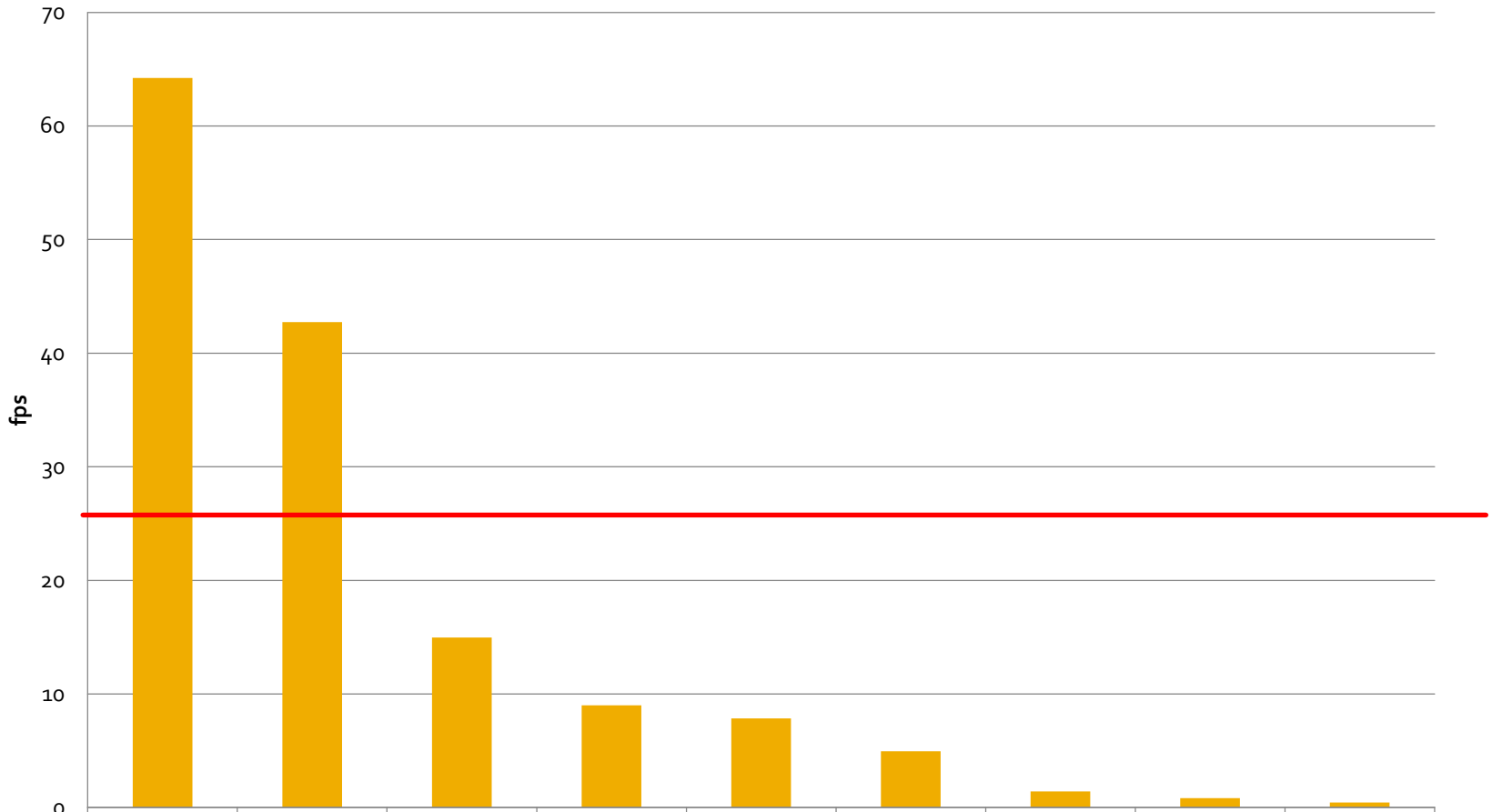
- Important "ultrafast" parameter changes:
  - No B pictures
  - 1 reference picture for motion estimation
  - Full pixel diamond motion estimation (very fast)

Practical parameters example II

- Important "veryslow" parameter changes:
  - 8 B pictures with adaptive B picture decision
  - 16 reference pictures (maximum allowed)
  - ME range 24x24
  - RDO for all modes and picture types
  - Psychovisual optimizations (e.g. Trellis quantization)
  - Psychovisual RDO

x264 preset speed comparison

Intel Core 2 Quad Q6600 720p encoding speed



	ultrafast	veryfast	faster	fast	medium	slow	slower	veryslow	placebo
fps	64,23	42,74	14,98	9,01	7,85	4,94	1,42	0,84	0,45

Encoder speed comparison (MSU)

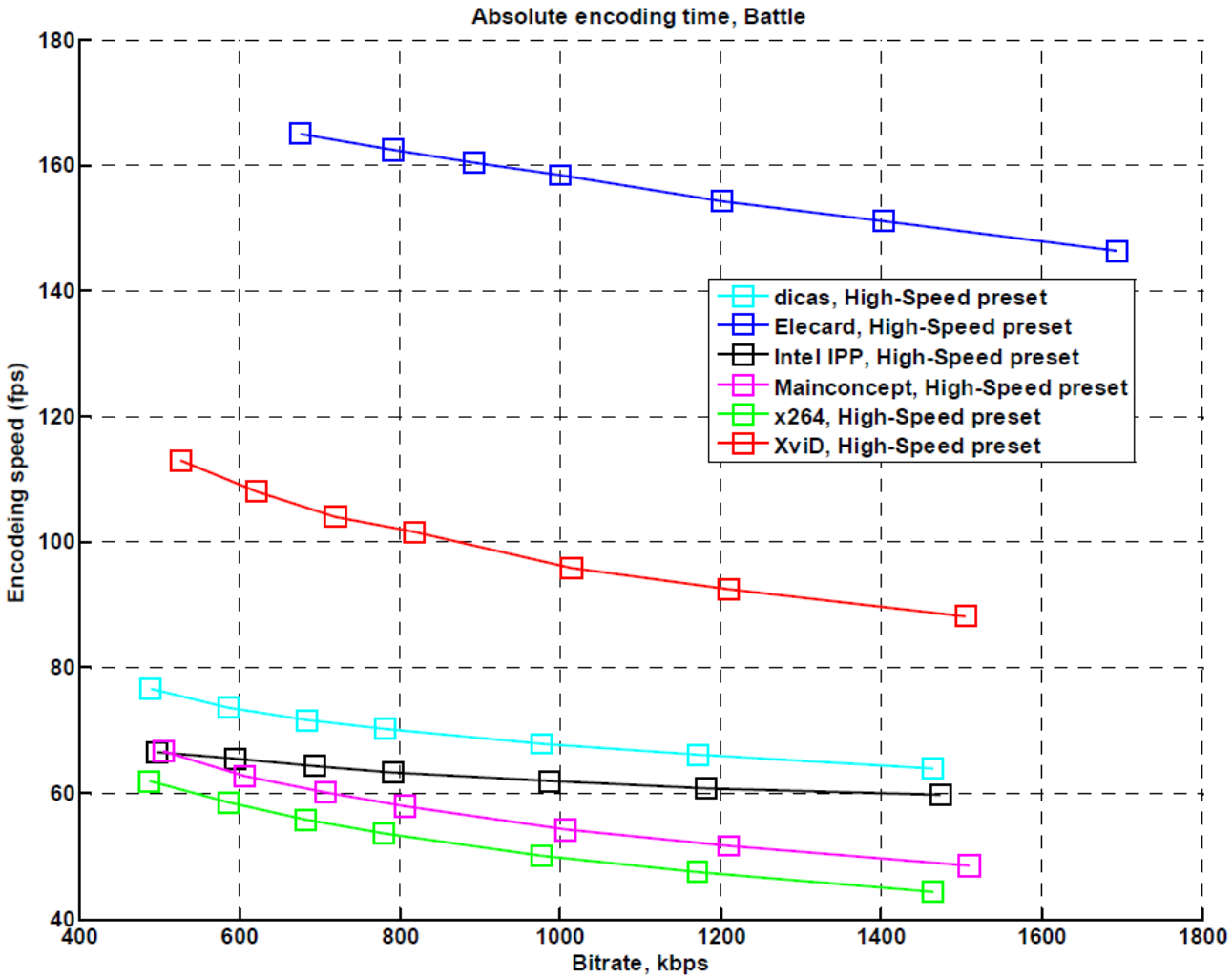


Image source: CS MSU Graphics & Media Lab Video Group: MPEG-4 AVC/H.264 Video Codecs Comparison. http://compression.ru/video/codec_comparison/pdf/msu_mpeg_4_avc_h264_codec_comparison_2009.pdf (14.11.2010), 2009.

H.264 error resilience tools

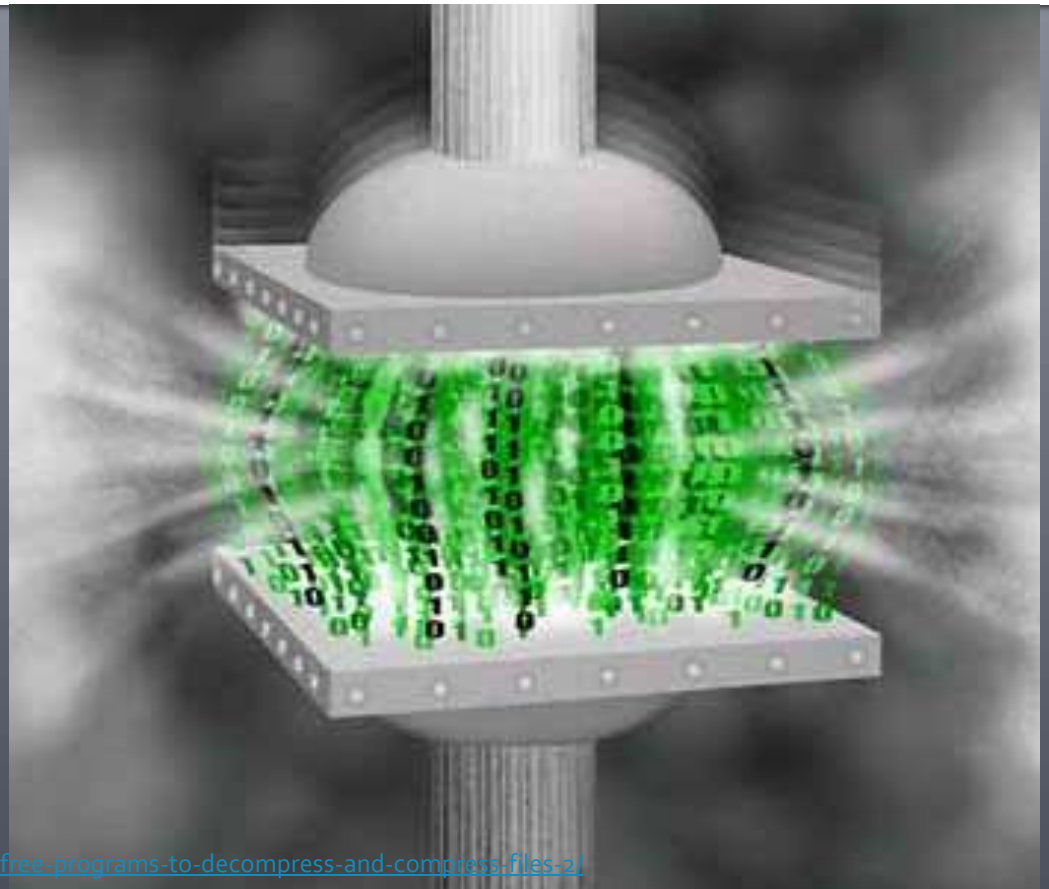


Image source: http://blog.toggle.com/free-programs-to-decompress-and-compress-files-2/

Bit errors in video coding

- Bit errors can cause corrupt macroblocks or macroblock groups during decoding
- Errors can propagate throughout the picture through intra prediction
- Errors can propagate to the next/previous pictures through inter prediction
- Prediction mechanisms favor propagation
- Error resilience tools minimize errors, but cost extra bits (less efficient compression)

Small burst error effects example

Image source: Boulos, F., Wei Chen; Parrein, B.; Le Callet, P.: A new H.264/AVC error resilience model based on Regions of Interest. 17th International Packet Video Workshop, 2009, pp.1-9, 2009

Large burst error effects example

Image source: Tsai, Y. C., Lin, C.-W. and Tsai, C.M.: H.264 error resilience coding based on multi-hypothesis motion-compensated prediction, Signal Processing: Image Communication, Vol.22, Issue 9, pp. 734-751, 2007

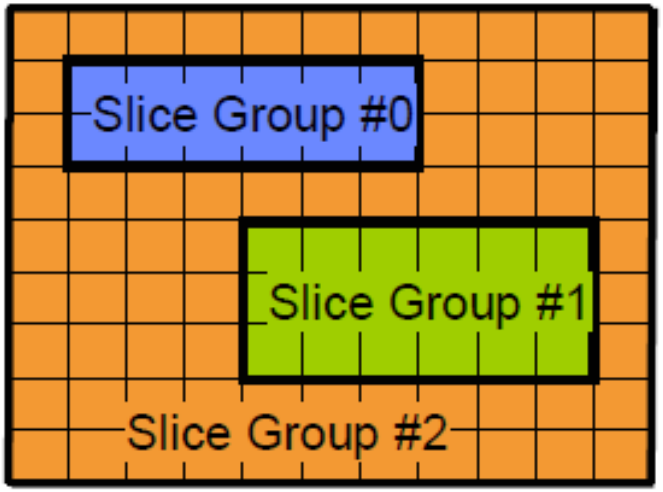
Tool: Data partitioning

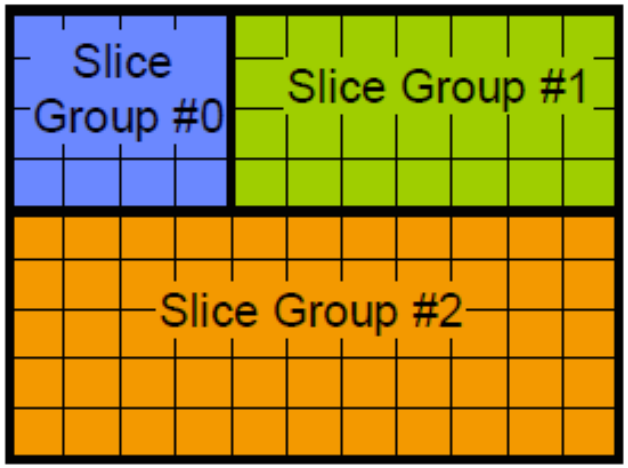
- Slice data (encoded macroblocks) is split into three data partitions: A, B and C
- Partition A: slice header and macroblock header information (coding mode etc.)
- Partition B: macroblock data for intra blocks
- Partition C: macroblock data for inter blocks
- Each partition is encapsulated in a separate NAL unit → priorization of A over B/C possible

Slices

- Separation of picture into separate independent parts (slices)
- No prediction between slices allowed → independency
- Slice types within a picture can differ (I, P, B)
- Slices can be grouped into slice groups

Slices and slice groups





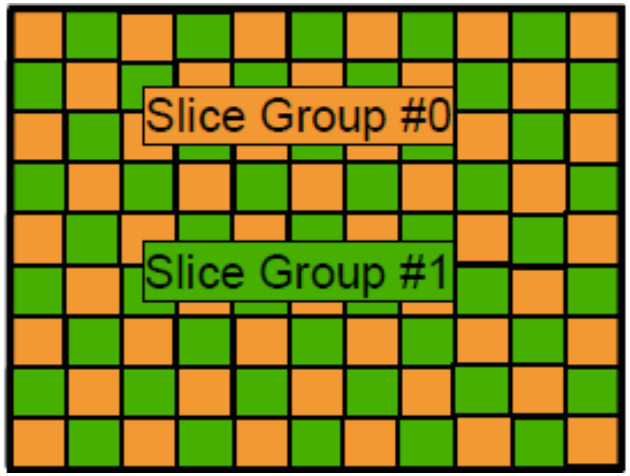


Image source: Wiegand, T. and Sullivan, G. J.: The H.264 | MPEG-4 AVC Video Coding Standard. http://ip.hhi.de/imagecom_G1/assets/pdfs/H264_03.pdf (14.10.2010), 2004

Tool: Redundant slices

- Possibility to code macroblocks more than one time and send them as additional, but redundant slice (lower quality possible)
- Decoder favors primary slice and discards the redundant slice if both are available
- If the primary slice is not available, the decoder uses the according redundant slice (if available) → reconstruction possible

Tool: Flexible macroblock ordering

- Defines fine-grain macroblock association with slice groups
- Different scanning patterns possible
- Dispersing macroblocks reduces sensitivity to burst noise on transmission channel
- Both implicit and explicit assignment to slice groups possible through multiple map types

FMO map types I

- 7 FMO map types for different use cases
- Type 0: run length based (macroblocks)
- Type 1: dispersed (p.e. checkerboard)
- Type 2: rectangular regions of interest

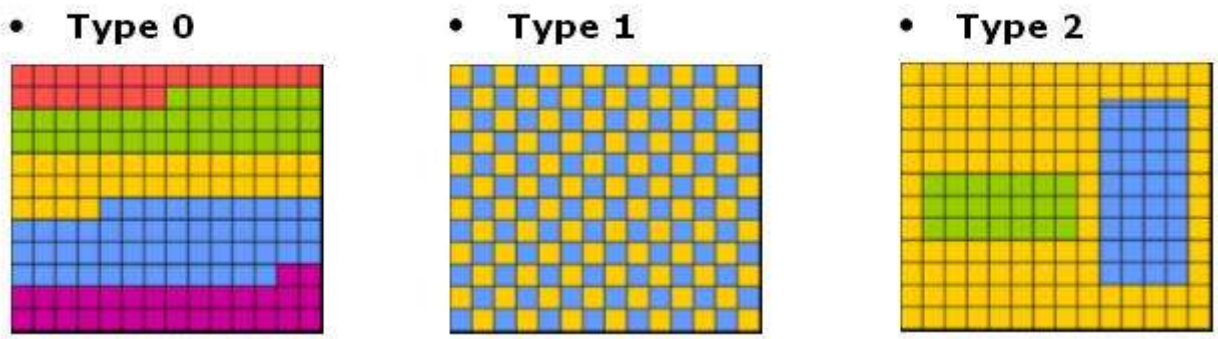


Image source: http://en.wikipedia.org/wiki/File:Tipos_FMO.jpg

FMO map types II

- Types 3-5: slice groups grow and/or shrink over time (growth/shrink rate specified)
- Type 6 allows arbitrary (explicit) grouping
- Arbitrary slice ordering (ASO)

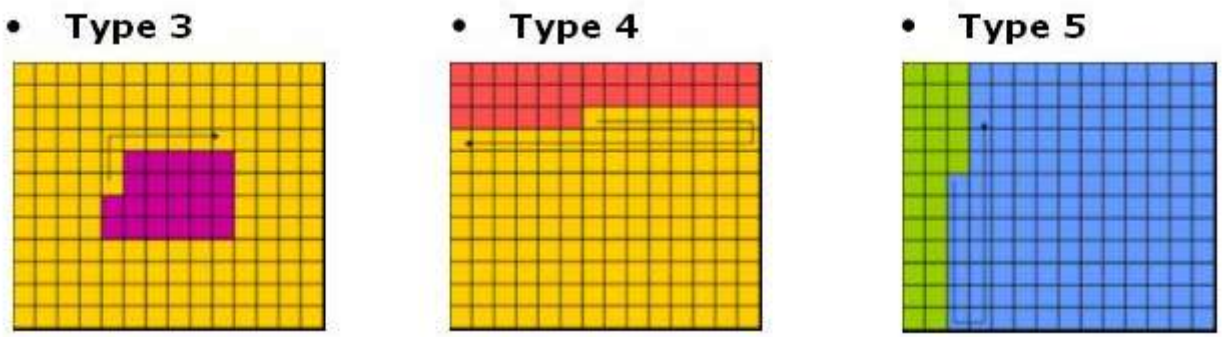


Image source: http://en.wikipedia.org/wiki/File:Tipos_FMO.jpg

Tool: Arbitrary slice ordering

- Slices don‘t have to be transmitted in scanning order (arbitrary ordering possible)
- Reordering may reduce effect of burst noise as it does not effect subsequent picture regions
- Priorization of slices or slice groups possible as every slice is contained in a separate NAL unit
- Fine-grain macroblock association possible with flexible macroblock ordering (FMO)

Arbitrary slice ordering example

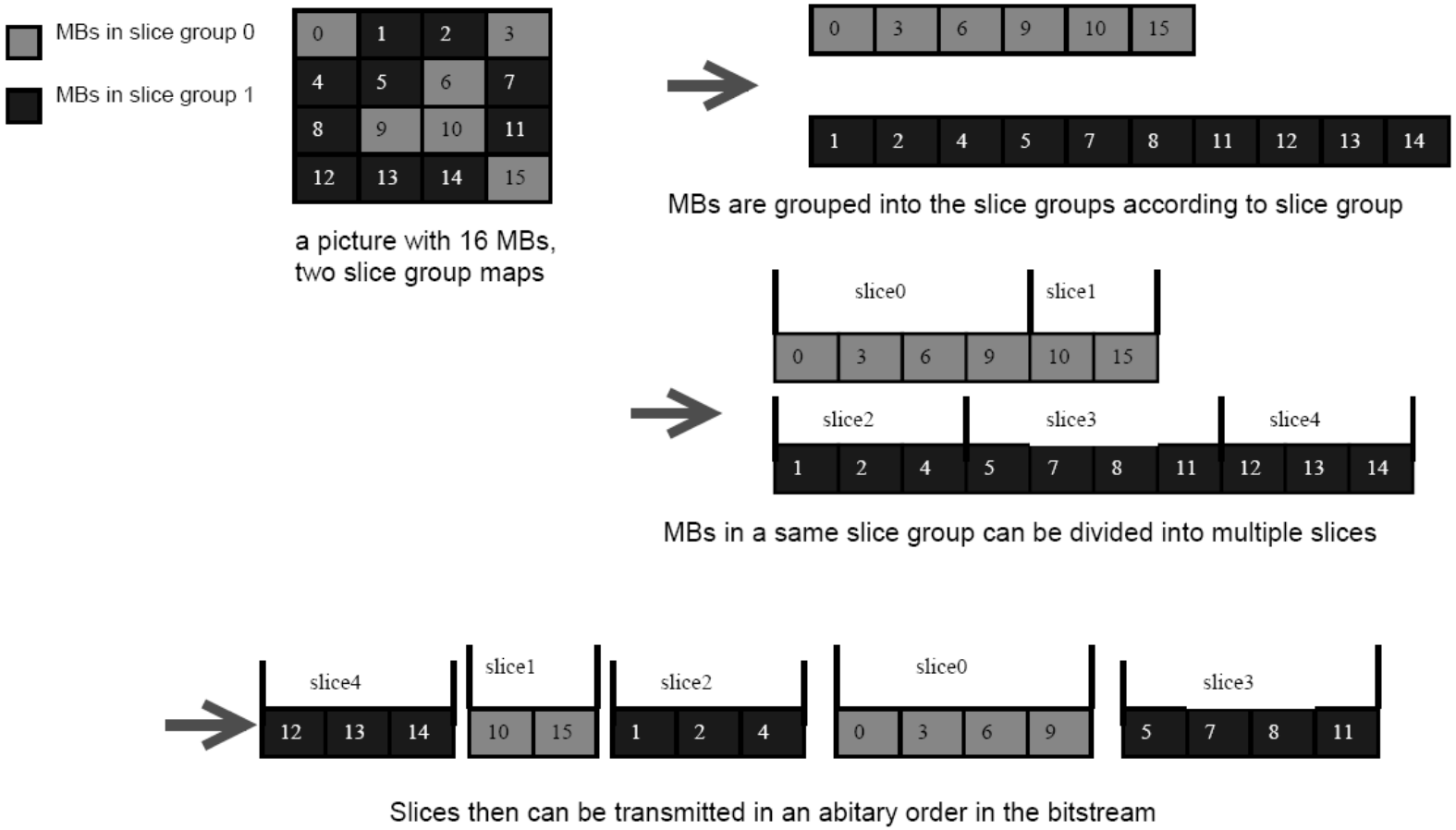


Image source: Jain, A. Introduction to H.264 Video Standard. http://www.powershow.com/view/ce503-OTc3M/Introduction_to_H264_Video_Standard_powerpoint_ppt_presentation (14.10.2011), 2011.

Effect of error resilience tools

- Redundant slices require additional coding (affect time)
- ASO is theoretically time-neutral as the number of coded blocks/slices stays the same
- FMO may slow down the encoder due to cache misses due to dispersed macroblocks in slice groups (affects time)
- Decoding complexity increases (affects time)

Error concealment

- H.264 specifies that corrupt NAL units are to be discarded (syntax errors or semantical errors can be detected, some others cannot)
- Error concealment may be applied to all areas where data is missing (optional)
- Standard proposes concealment methods
- Different methods proposed by various research groups

Error detection

- Syntax error detection
  - Illegal values of syntax elements
  - Illegal synchronisation/header (0x000001)
  - Coded macroblock data contains more elements than specified (p.e. more than 16 in a 4x4 block)
  - Illegal entropy code (words)
- Semantics error detection
  - Out of range luma or chroma values
  - Invalid states during decoding

Data partition loss concealment

- Recall: data partitions A, B and C to prioritize information (A is the most important partition)
- Recall: B/C contain intra/inter coded data

Available Partition(s)	Concealment Method
A and B	Conceal using MVs from Partition A, and texture from Partition B; intra concealment is optional.
A and C	Conceal using MVs from Partition A and inter info from Partition C; inter texture concealment is optional.
A	Conceal using MVs from Partition A
B and/or C	Drop Partitions B and C. Use motion vectors of the spatially above MB row for each lost MB

Image source: Kumar, S., Xu, L., Mandal, M. K. and Panchanathan, S.: Error Resiliency Schemes in H.264/AVC Standard. Elsevier Journal of Visual Communication & Image Representation (Special issue on Emerging H.264/AVC Video Coding Standard), Vol. 17(2), 2006

Data partitioning effects I

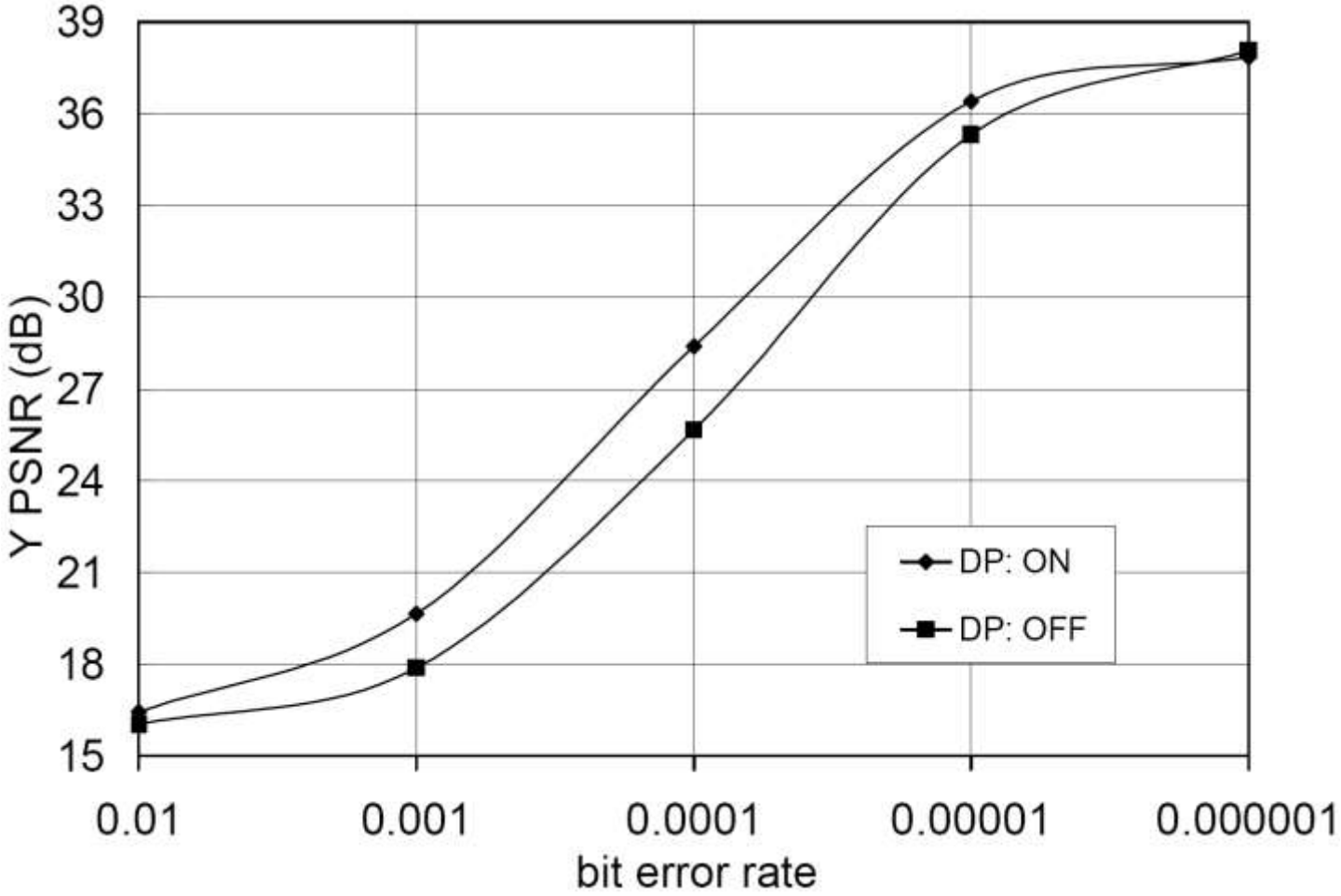


Image source: Kumar, S., Xu, L., Mandal, M. K. and Panchanathan, S.: Error Resiliency Schemes in H.264/AVC Standard. Elsevier Journal of Visual Communication & Image Representation (Special issue on Emerging H.264/AVC Video Coding Standard), Vol. 17(2), 2006

Data partitioning effects II

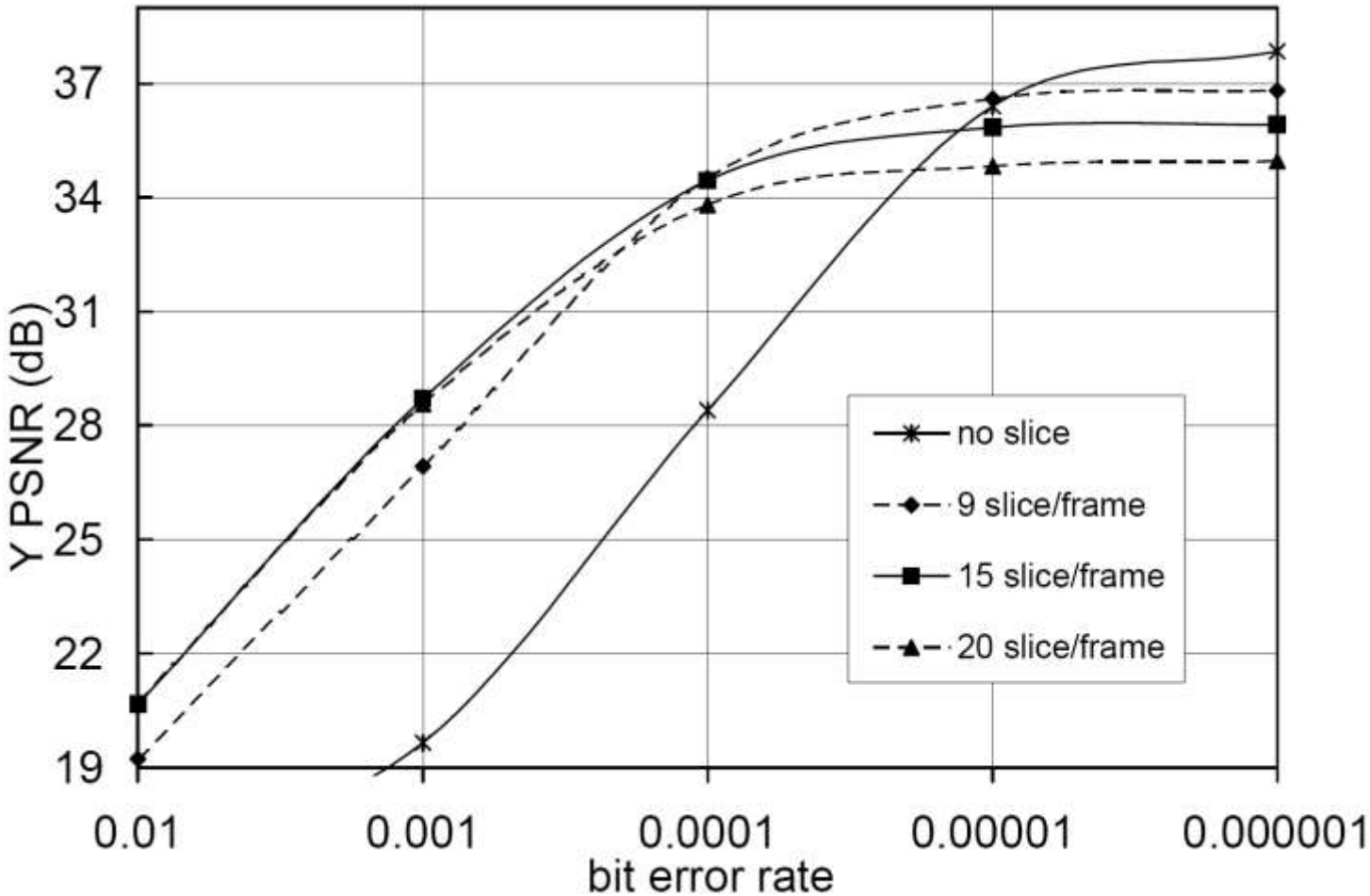


Image source: Kumar, S., Xu, L., Mandal, M. K. and Panchanathan, S.: Error Resiliency Schemes in H.264/AVC Standard. Elsevier Journal of Visual Communication & Image Representation (Special issue on Emerging H.264/AVC Video Coding Standard), Vol. 17(2), 2006

FMO effects

- Checkerboard pattern allows interpolation if one slice group gets lost or damaged
- Concealment: use previous pictures‘ blocks

Original

Decoded (err.)

“Copy“ concealment

Image source: Kolkeri, V. S., Koul, M. S., Lee, J. H. and Rao, K. R.: Error Concealment Techniques in H.264/AVC For Wireless Video Transmission In Mobile Networks. http://students.uta.edu/ms/msk3794/docs/ERROR_CONCEALMENT_TECHNIQUES-JAES2009.pdf (14.10.2011), 2008

Simple intra concealment

- Weighted pixel averaging from neighbouring macroblock samples
- Neighbours have to be available
- If neighbours are also interpolated, prediction becomes worse

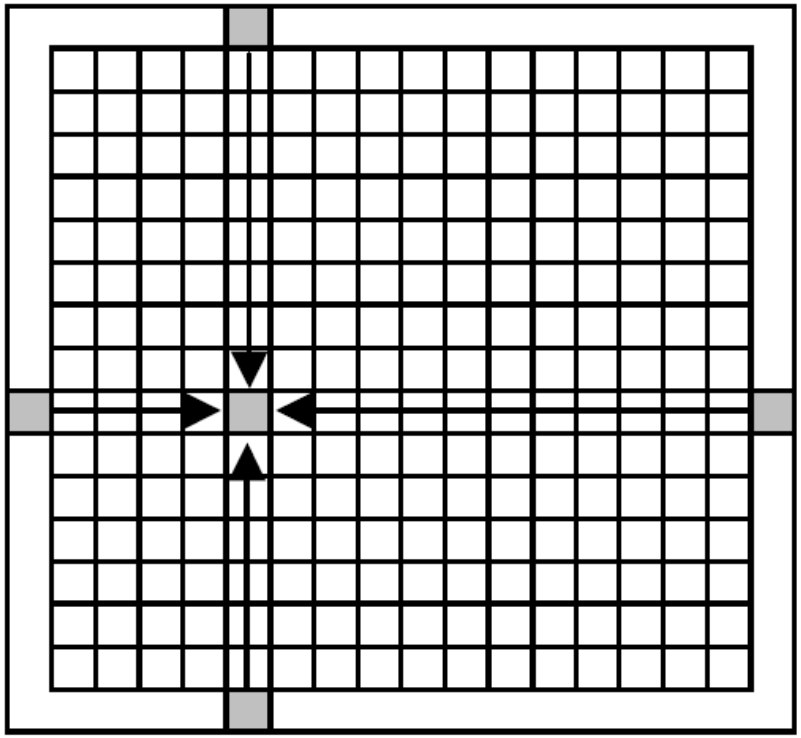


Image source: Kumar, S., Xu, L., Mandal, M. K. and Panchanathan, S.: Error Resiliency Schemes in H.264/AVC Standard. Elsevier Journal of Visual Communication & Image Representation (Special issue on Emerging H.264/AVC Video Coding Standard), Vol. 17(2), 2006

Simple inter concealment

- Motion vectors tend to correlate in small regions and over a small number of pictures
- If motion vectors are lost, they may be predicted from the neighbouring blocks and/or the reference frame's co-located macroblocks
- Motion vector prediction (based on median, average or similar techniques) often fails
- In some cases: copy co-located reference block

Advanced concealment I

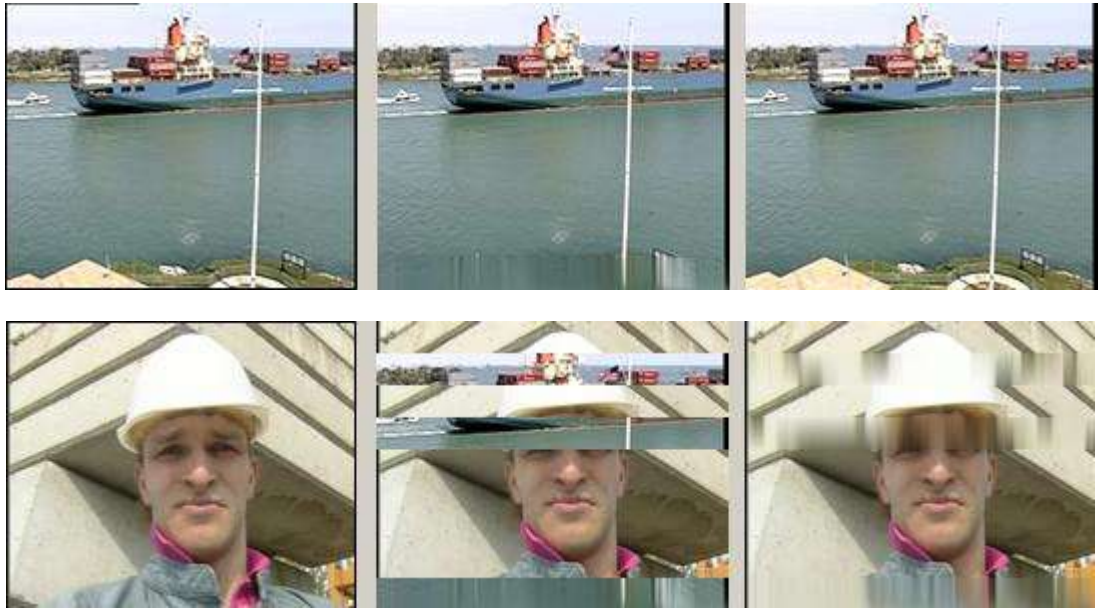
- Based on scene cut detection to force zero motion vectors for lost blocks

Picture n-1

20% loss

Picture n

20% loss

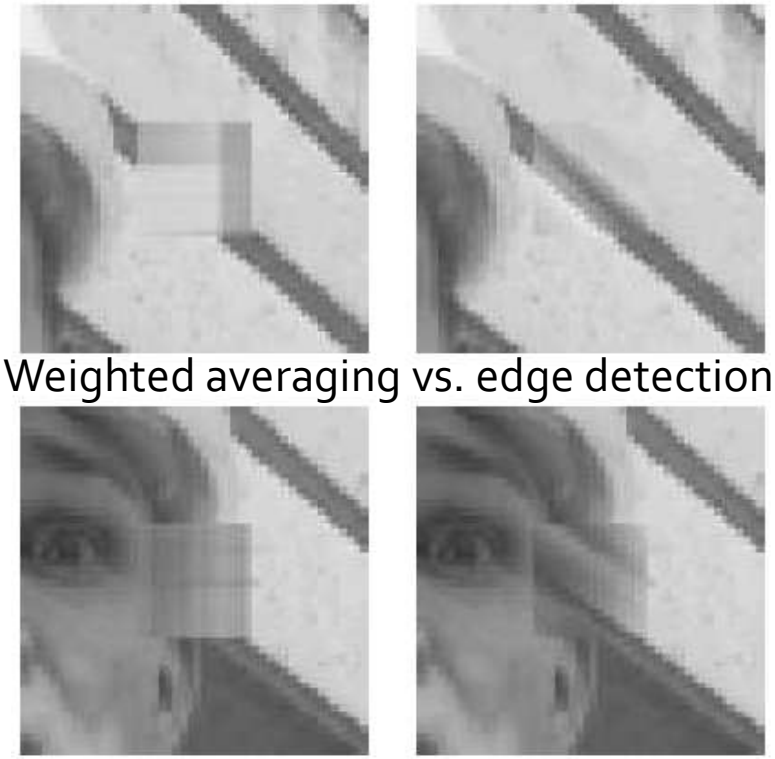


Original Decoded (err.) Concealed

Image source: Su, L., Zhang, Y., Gao, W., Huang, Q. and Lu, Y.: Improved error concealment algorithms based on H.264/AVC non-normative decoder. 2004 IEEE International Conference on Multimedia and Expo, vol.3, pp. 1671-1674, 2004

Advanced concealment II

- Dominant edge detection from neighbouring macroblocks for directional interpolation



Weighted averaging vs. edge detection

Image source: Nemethova, O., Al-Moghrabi, A. and Rupp, M.: Flexible Error Concealment for H.264 Based on Directional Interpolation. 2005 International Conference on Wireless Networks, Communications and Mobile Computing, vol.2, pp.1255-1260, 2005

Advanced concealment III

- Theoretical methods not for practical use
- Example: determine regions of interest (green box) in the picture through subjective tests and force intra prediction for affected blocks

Image source: Boulos, F., Wei Chen; Parrein, B.; Le Callet, P.: A new H.264/AVC error resilience model based on Regions of Interest. 17th International Packet Video Workshop, 2009, pp.1-9, 2009

Loss measurement

- Complete loss of macroblocks or whole slices is hard to measure (as effect on the picture(s))
- Complete loss is nearly always noticeable, but concealment can hide it up to a certain extent
- Y-PSNR and SSIM make sense in error concealed pictures, but not in lost ones
- Measuring quality of pictures with completely lost macroblocks or slices is difficult

Practical tasks I

- Handling uncompressed (YUV) video files
  - Opening, viewing and comparing
  - File size calculations and considerations
- Getting to know x264, an H.264 encoder
  - Basic H.264 encoding (default settings)
  - Original/output file comparison
  - Effect of options on coding time and quality
  - Output file formats and compatibility

Practical tasks II

- Getting to know the testbed for error resilience
  - "Network" simulator with virtual, error prone transmission channel (on a NALU basis)
  - Extended version of x264 with error resilience tools
  - Extended version of ffmpeg with error concealment
- Testing effects of error resilience tools (increase in bit rate, increase in encoding time)
- Testing effects of error concealment methods (loss prevention, method comparison)

Thank you for your attention!

Questions?