HEVC: A coding standard that we (will) use

when watching TV/video-streams

David Svoboda

CBIA

Centre for Biomedical Image analysis
Faculty of Informatics, Masaryk University, Brno, Czech Republic

May 7, 2019

Content

CBIA

Motivation

Mobile data and TV broadcasting are competitors

- Capacity of mobile data . . . increasing
- Quality of TV channels . . . increasing

The bandwidth of air (as a transmission medium) is limited!

A brief history of digital data transmission in the Czech republic

- 2005 . . . start of digital transmission (DVB-T)
- 2011 . . . end of analogue transmission
- 2014 . . . mobile providers start LTE
- 2017 . . . start of digital transmission of G2 (DVB-T2)
- 2021 . . . end of digital transmission of G1 (DVB-T)

CBIA

Basic facts

DVB-T = Digital Video Broadcasting (Terrestrial)

- DVB-T . . . based on MPEG-2 or H.264/MPEG-4 AVC
- DVB-T2 . . . based in H.265/HEVC

HEVC = High Efficiency Video Coding

HEVC addressed two key issues:

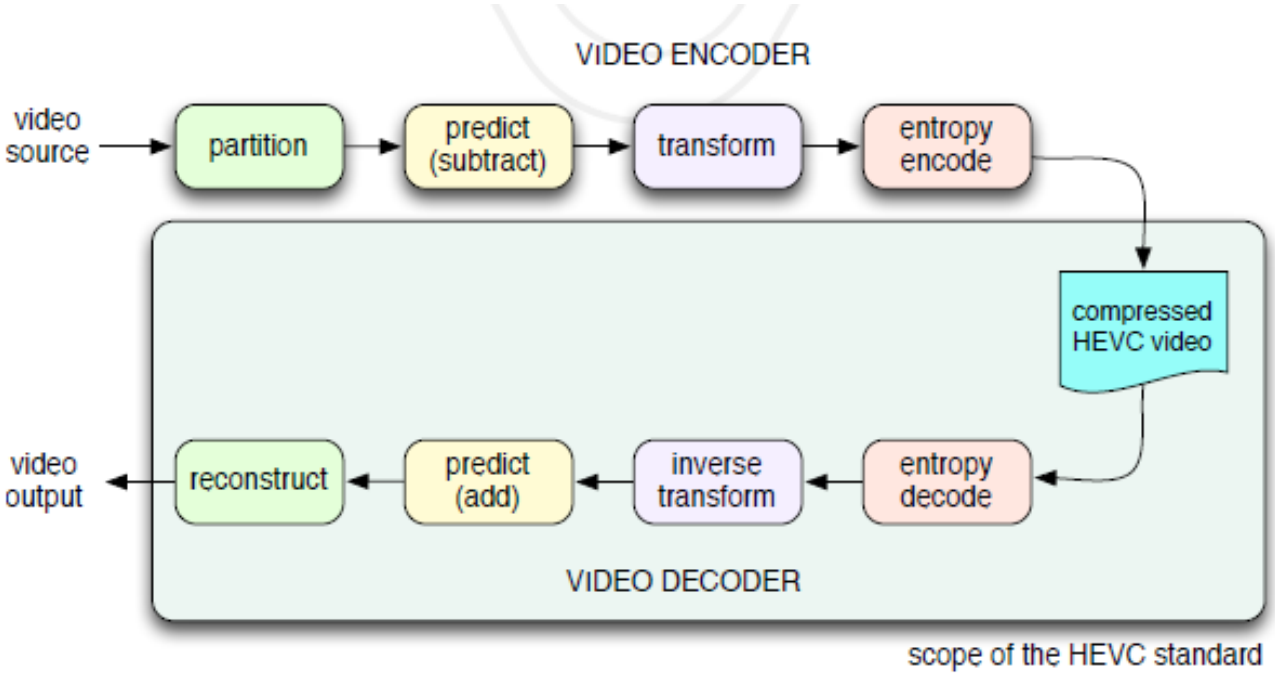
- increase video resolution while keeping the bitrate *or* improve the data compression (up to 50%) while keeping the same level of video quality (*compared to H.264*)
- increase use of parallel processing architectures

CBIA

Specification

Basic building blocks of HEVC standard

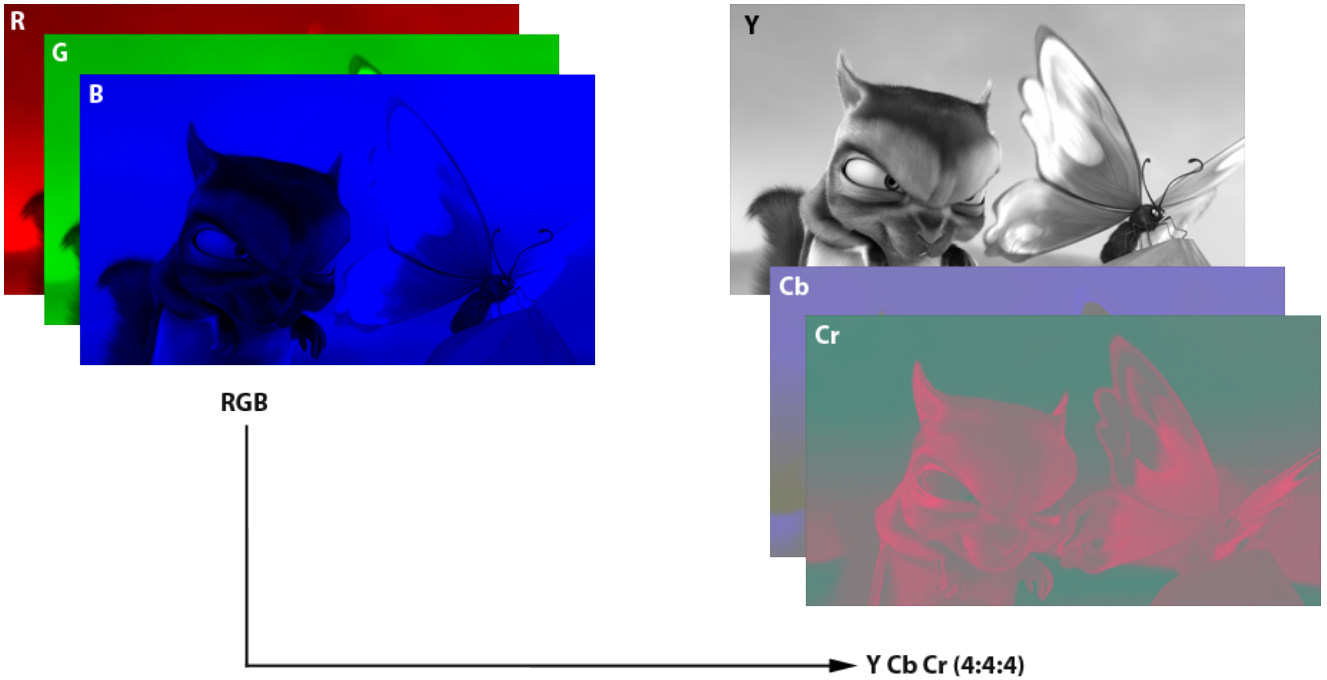
- Color representation
- Picture partitioning
- Intra prediction
- Inter prediction (motion vectors)
- Transform coding (of residuals)
- Entropy coding (of transform coefficients)
- Deblocking filters (part of reconstruction)



Specification

Color representation

- Use of YCbCr model rather than RGB or CMYK

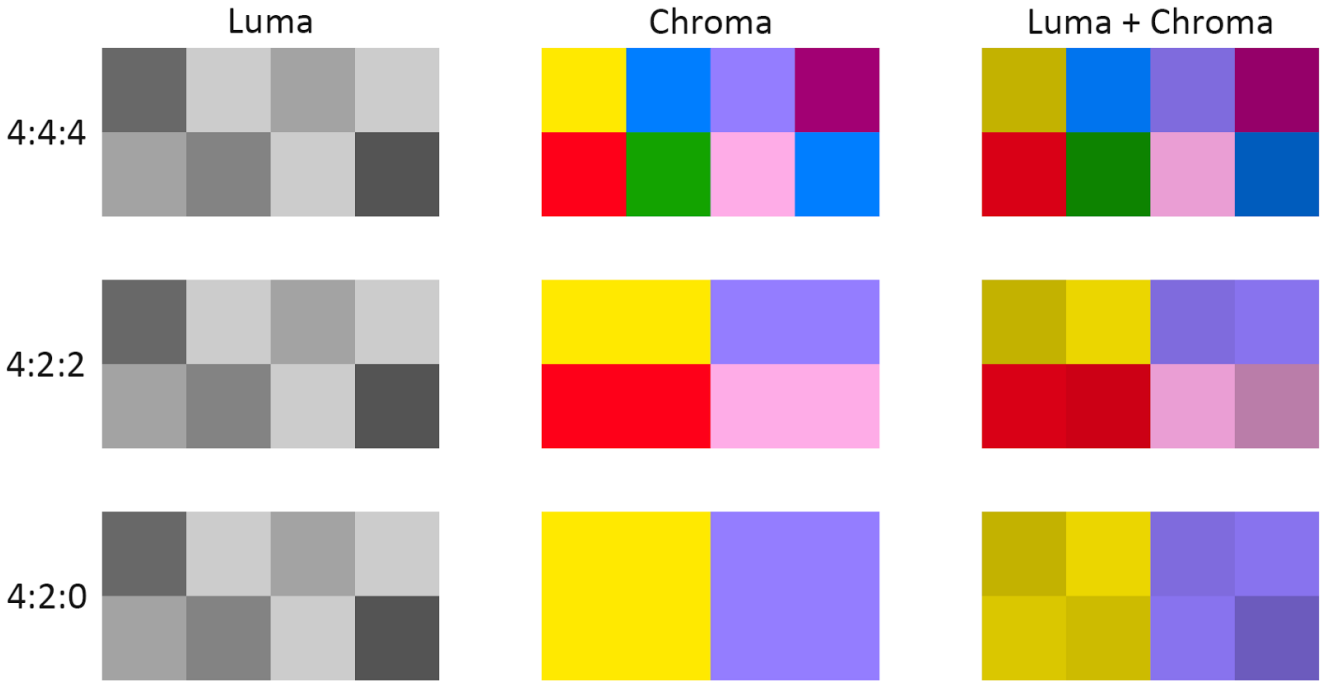


- Human visual system less sensitive to color than to structure and texture $\Rightarrow$ full resolution luma, lower resolution chroma

Specification

Color representation (cont'd)

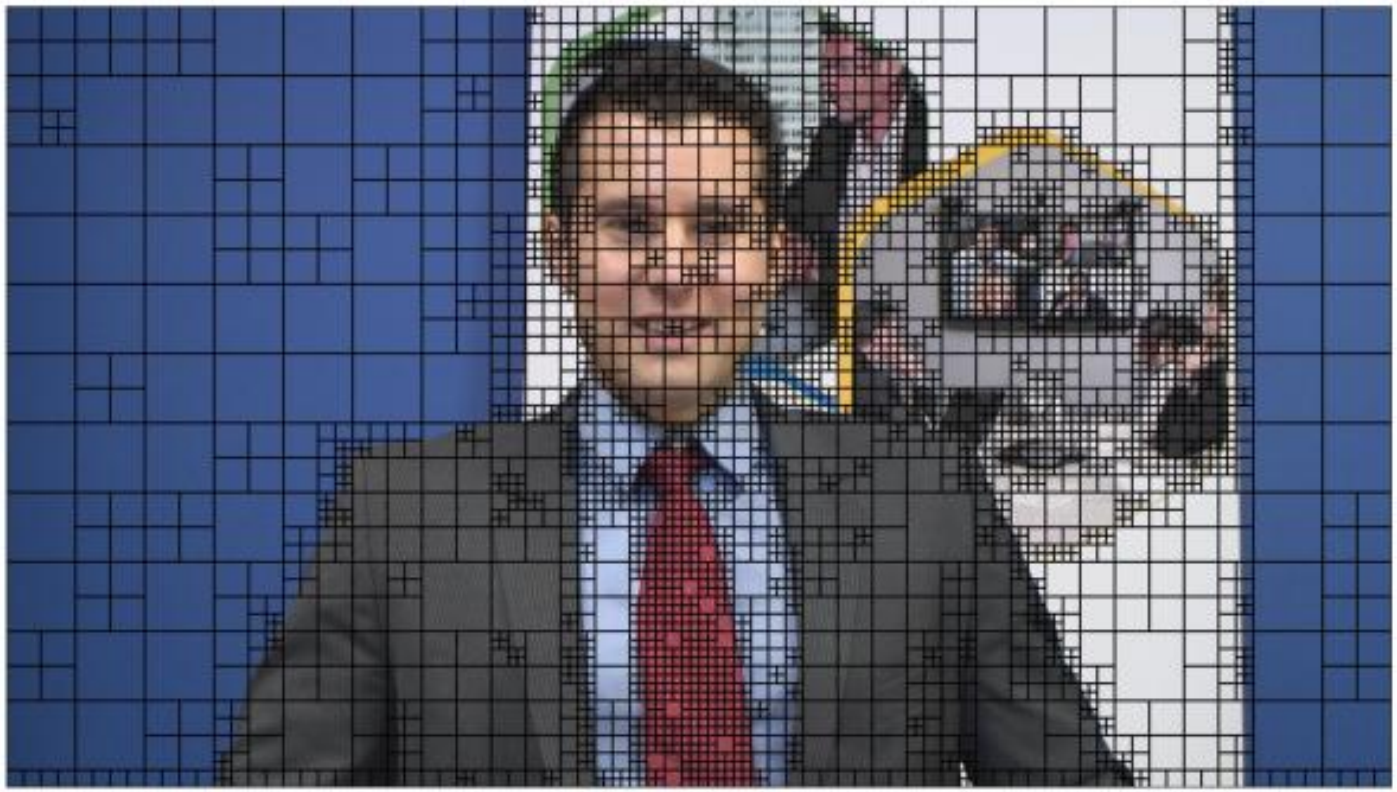
- Chroma sub-sampling 4:2:0 has been accepted as the standard format for consumer video



Specification

Picture partitioning

- The image is split into coding tree units (CTU) of adaptive size ($16 \times 16, 32 \times 32, 64 \times 64$).
- Each CTU can be further split into smaller coding units (CU) based on quad-tree algorithm.

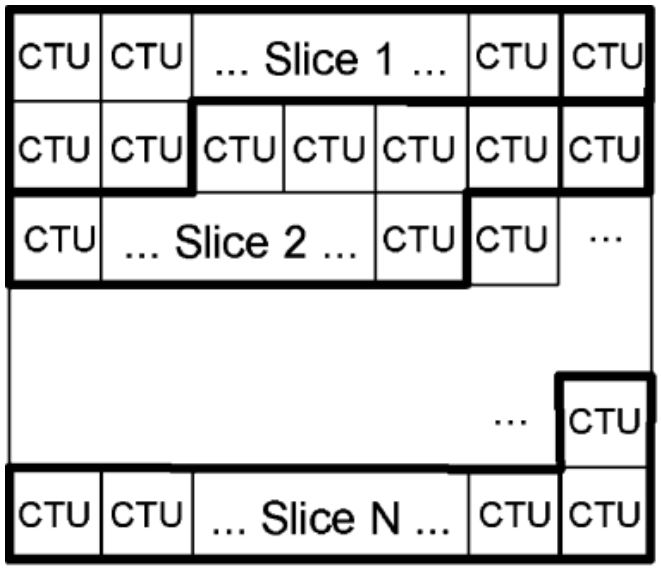


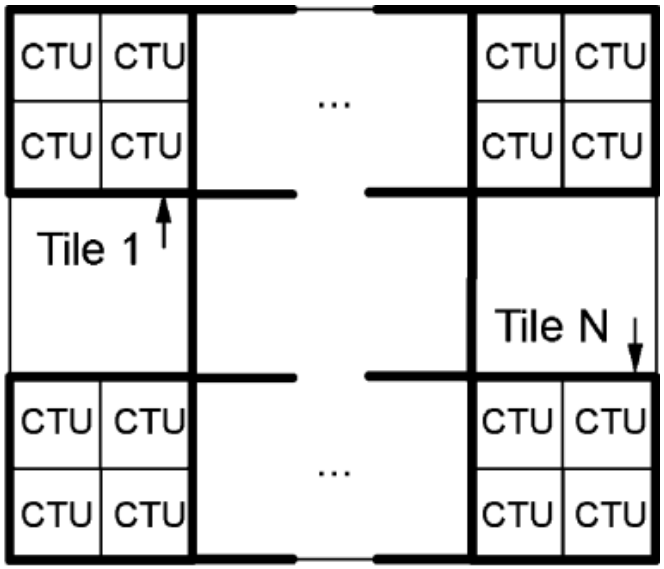
CBIA

Specification

Picture partitioning

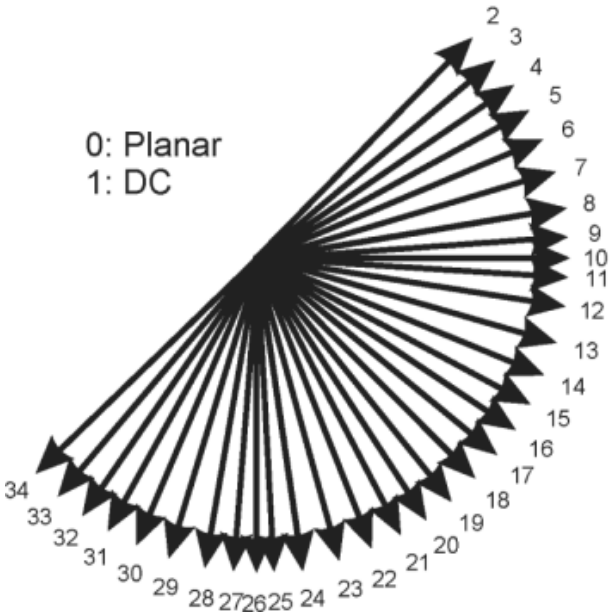
The image is initially split into slices/tiles

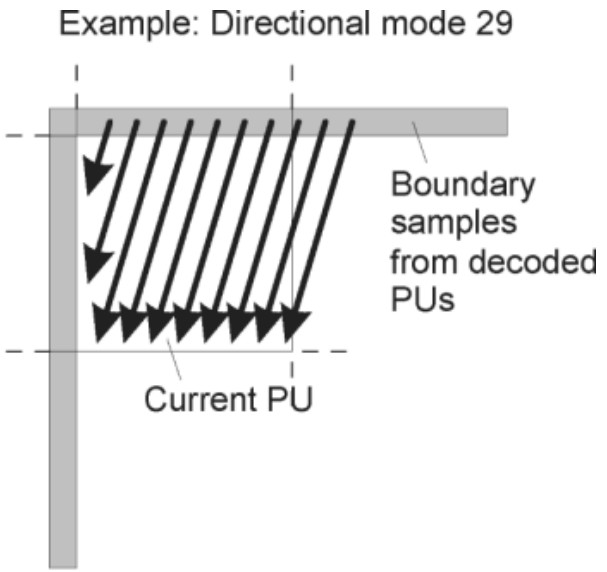




- Each slice is computed independently, i.e. no prediction (relationship) across the slice boundaries.
- Purpose of tiles is the resynchronization after data losses.
- Tiles are regular rectangular regions.
- The main purpose is the support for parallel processing.

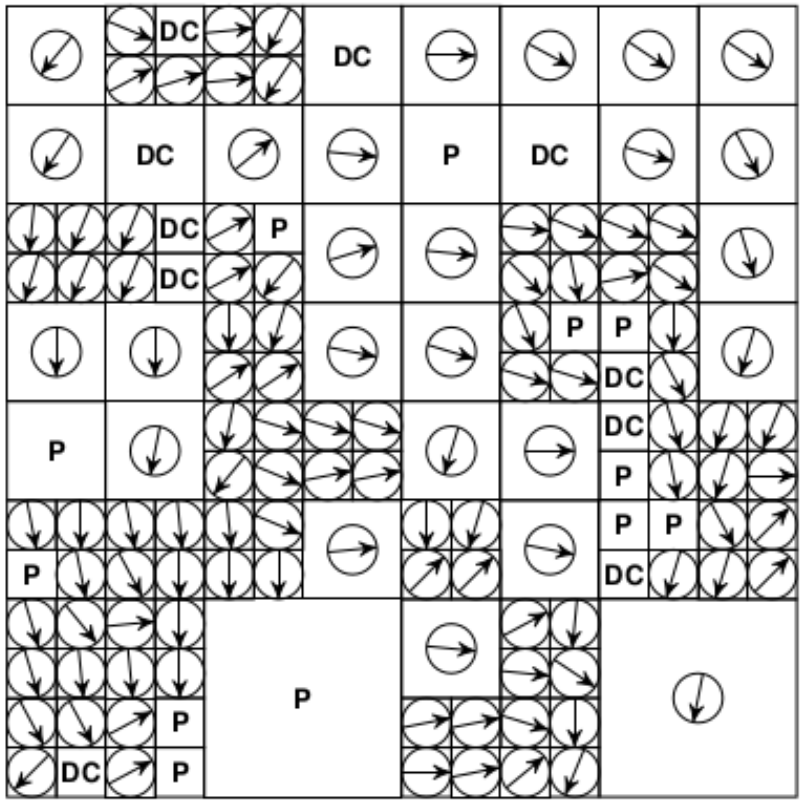
Each CU can be predicted from neighbouring image data in the same video frame.





Specification

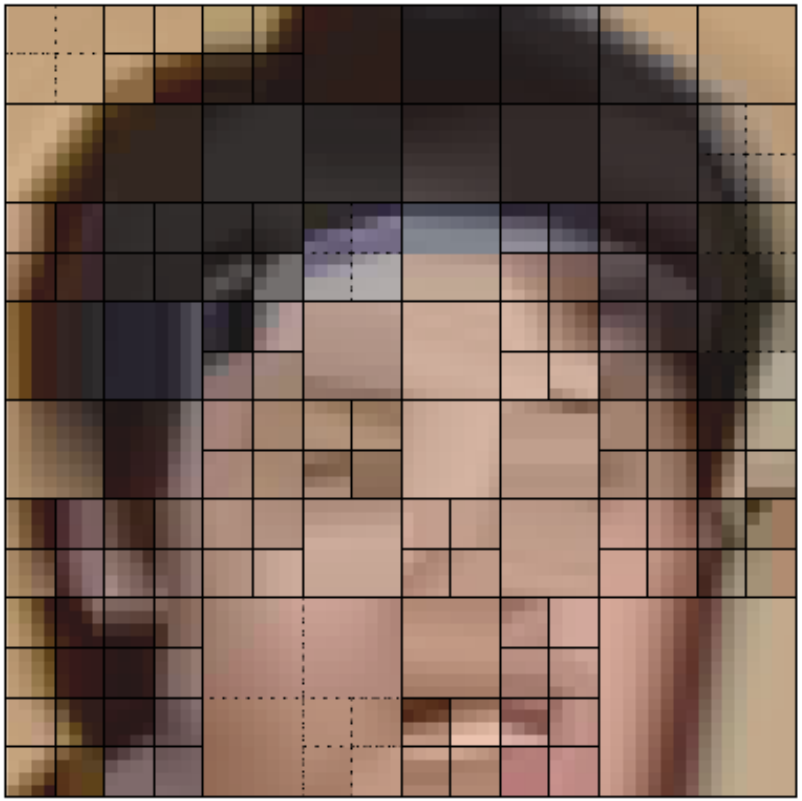
Intra prediction

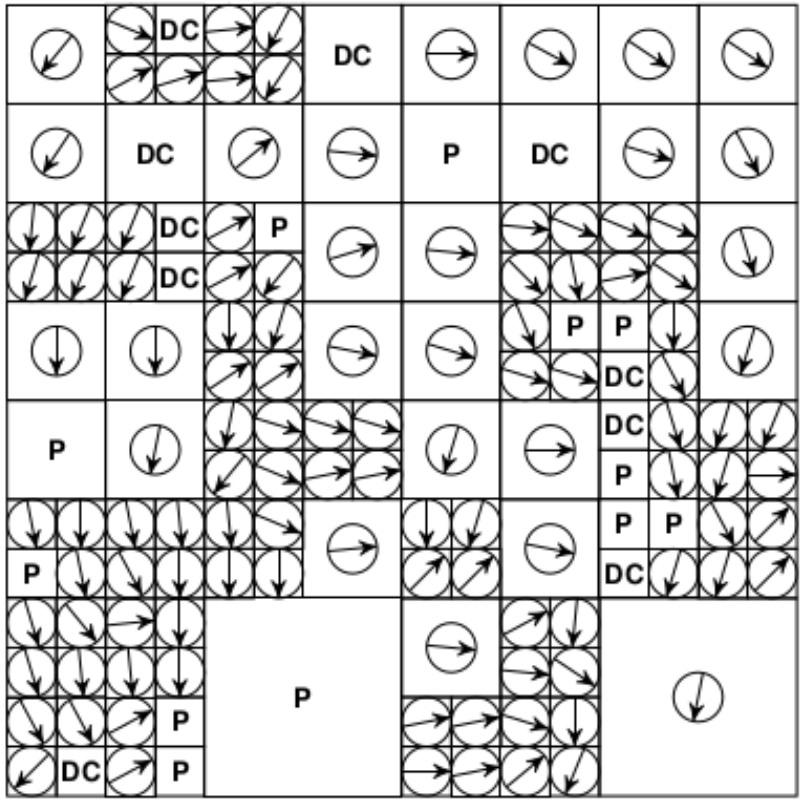


original

Specification

Intra prediction

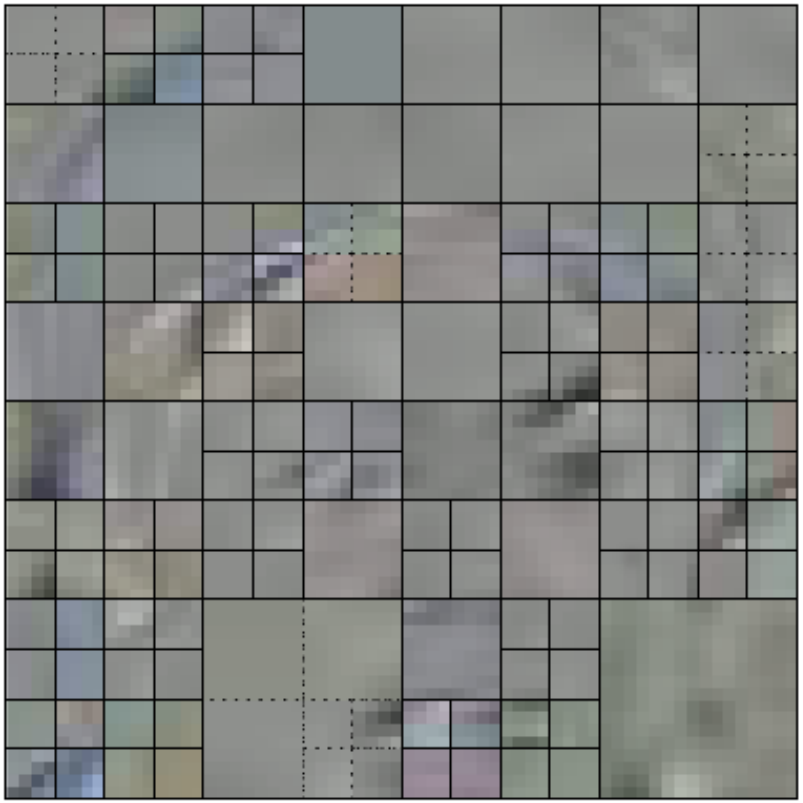


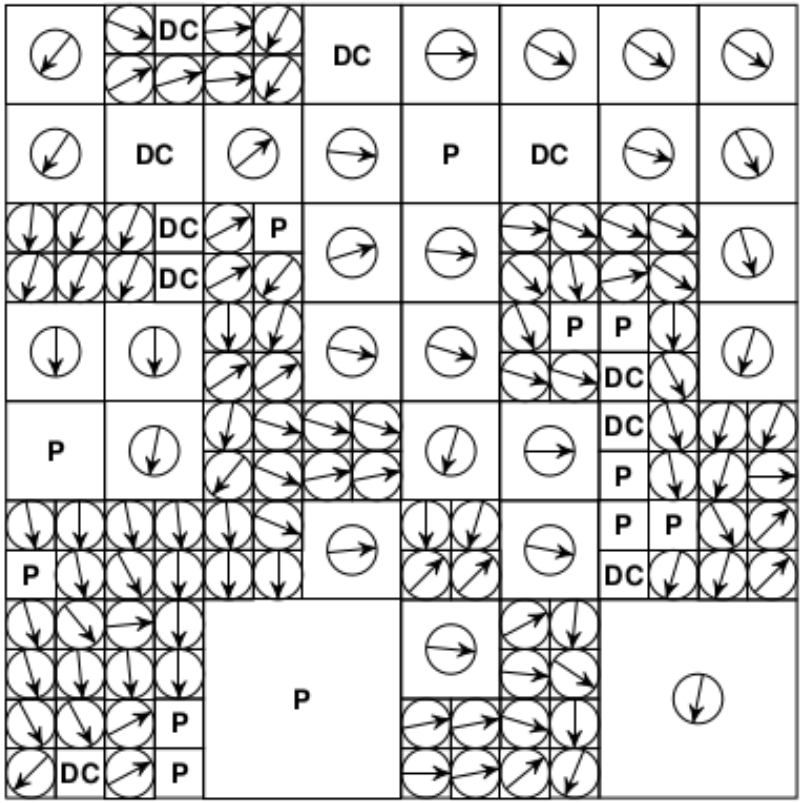


prediction

Specification

Intra prediction





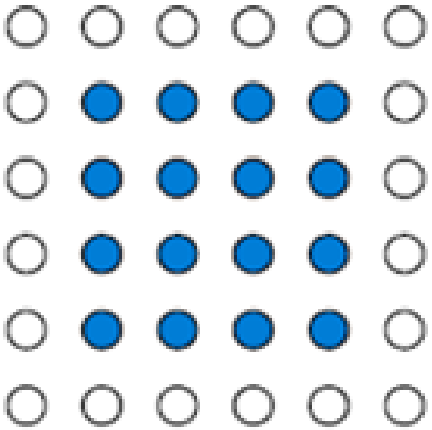
residual

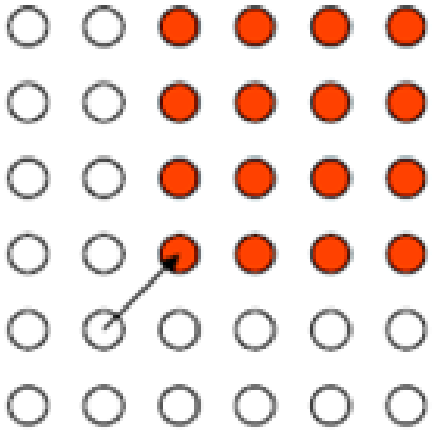
Specification

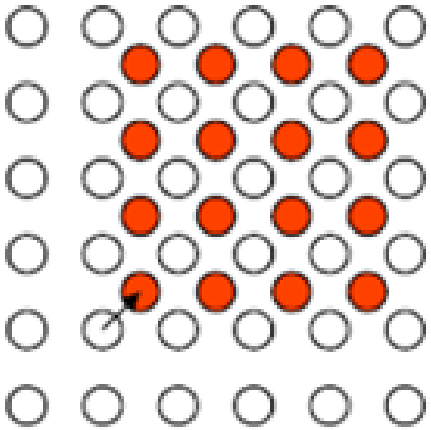
Inter prediction – Motion vectors

- Intra prediction . . . spatial prediction
- Inter prediction . . . spatial & temporal prediction

Inter prediction . . . predicting from image data in one or two reference pictures (before or after the current picture in display order), using motion compensated prediction.



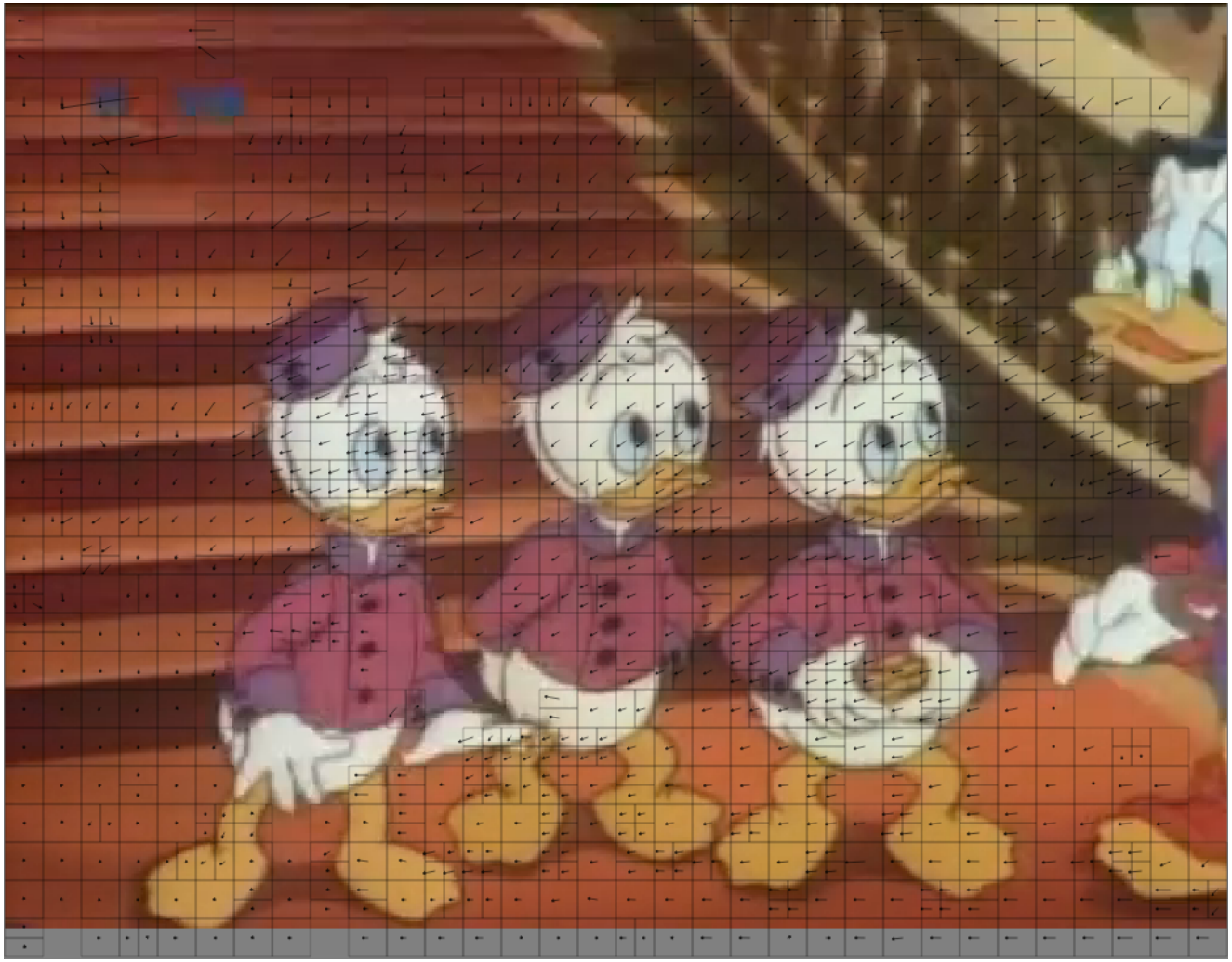




CBIA

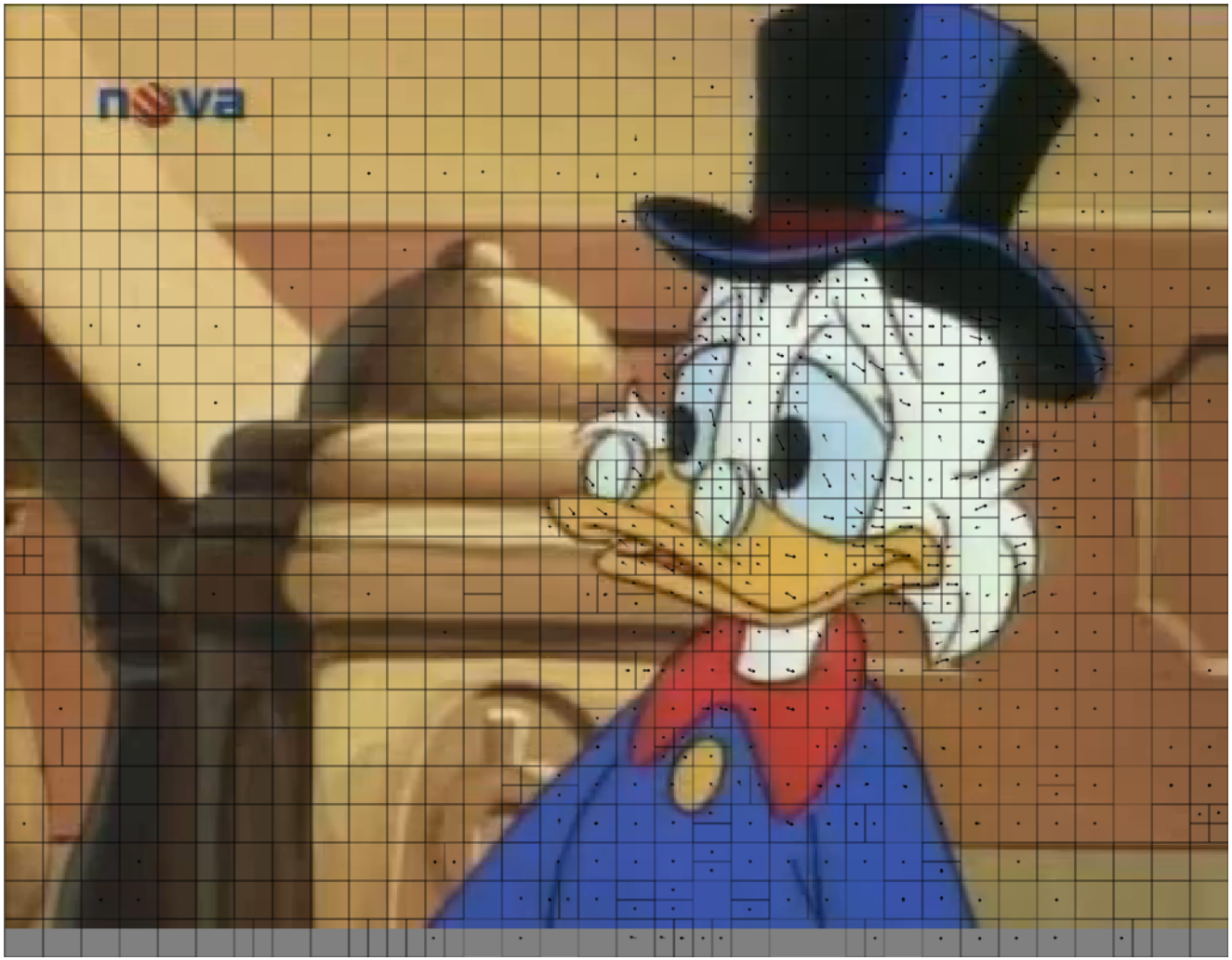
Specification

Inter prediction – Motion vectors



Specification

Inter prediction – Motion vectors



Specification

Transform coding

The discrete cosine transform applied to residuals is followed by quantization.

Specification

Entropy coding

CABAC – context-adaptive binary arithmetic coding

- The quantized DCT coefficients are submitted to arithmetic coding
- Context allows the adaptability when encoding the final bitstream

A simple example of entropy coding

Let's suppose we have a block with 8-levels coded with 3-bit binary code

i	$p(i)$	3-bit code	$len(i)$	new code	$new\ len(i)$
$l_0 = 0$	0.19	000	3	11	2
$l_1 = 1$	0.25	001	3	01	2
$l_2 = 2$	0.21	010	3	10	2
$l_3 = 3$	0.16	011	3	001	3
$l_4 = 4$	0.08	100	3	0001	4
$l_5 = 5$	0.06	101	3	00001	5
$l_6 = 6$	0.03	110	3	000001	6
$l_7 = 7$	0.02	111	3	000000	6

Using new code brings better (lower) average number of bits per pixel:

$$L_{avg} = 2(0.19) + 2(0.25) + 2(0.21) + 3(0.16) + 4(0.08) + 5(0.06) + 6(0.03) + 6(0.02) = 2.7 \text{ bits}$$

Specification

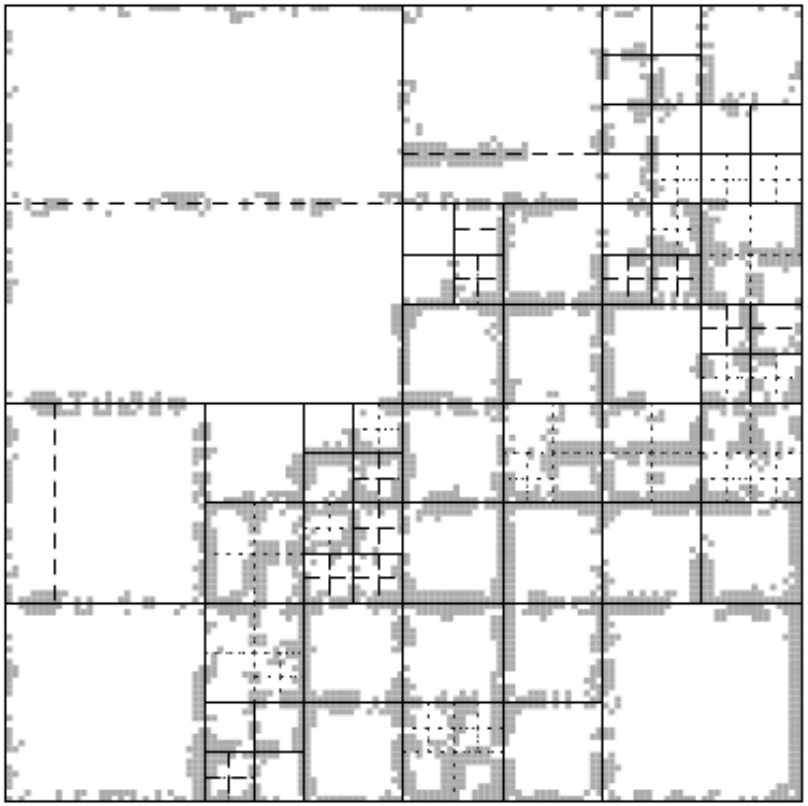
Deblocking filters

(a) Original

(b) Reconstruction with deblocking

Specification

Deblocking filters

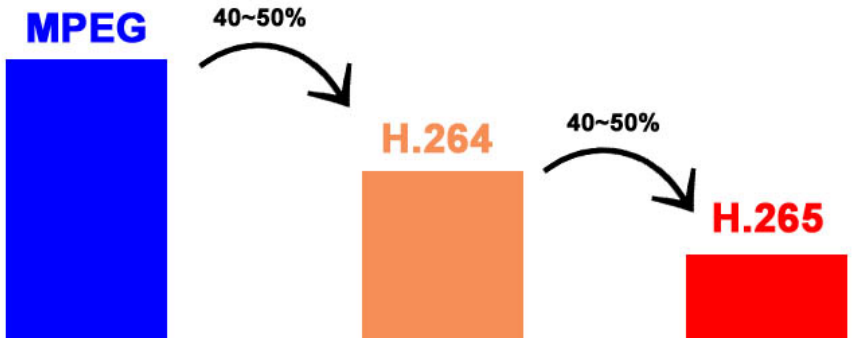


(c) Structure, deblocked samples

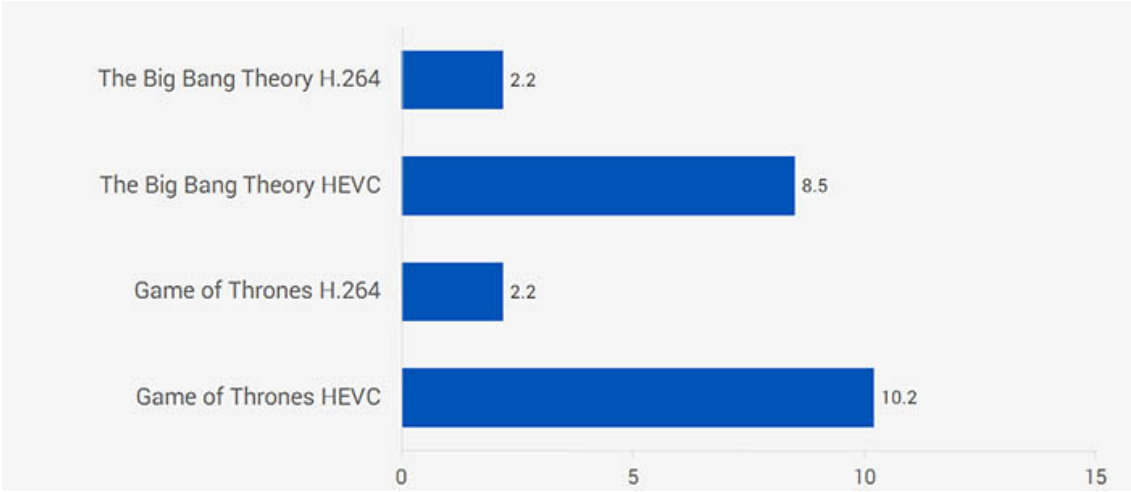
(d) Reconstruction without deblocking

Pros and cons

+ efficiency (50% compared to H.264)



+ speed/parallelism
− computationally intensive



CPU usage

Recommended sources

- short (but illustrative) paper:
  Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. 2012. Overview of the High Efficiency Video Coding (HEVC) Standard. IEEE Trans. Cir. and Sys. for Video Technol. 22, 12 (December 2012), 1649-1668
- book with complete specification:
  Sze, Vivienne, ed., Madhukar Budagavi ed., and Gary Joseph Sullivan ed. High Efficiency Video Coding (HEVC): Algorithms and Architectures. Berlin: Springer, 2014.
- codec/decoder source codes:
  `http://x265.org/`, `https://www.libde265.org/`
- software for video analysis (Win/Linux/Mac):
  `https://ient.github.io/YUView/`
- software for video transcoding (Win/Linux/Mac):
  `https://handbrake.fr/`
  currently installed on Alfa – you can try :-)

CBIA

Thank you for your attention . . .