

VO Videoverarbeitung

The Insides of Video Coding

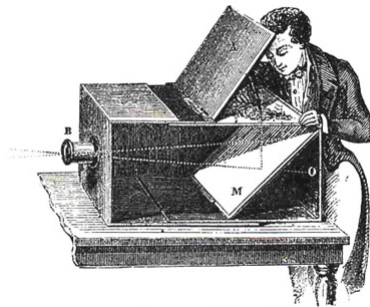
SS 2008

Gastvortrag: Florian Seitner
(seitner@ims.tuwien.ac.at)

Overview

- Evolution of video coding
- Video coding techniques
- Real-time issues
- Parallelization

When did it start...?



When did it start...?

“Interaction” between the short-comings of the human eye and technical advances

- Ancestor of the modern slide projector
- China, 2nd century
- Projected paintings



“Laterna Magica”
(Magic lantern)

When did it start...?

“Interaction” between the short-comings of the human eye and technical advances

- Rotating cylinder with slits
- Images “blur” together
- Objects appear thinner than they are!



Zoetrope¹, 1834

Chemical film (1860s)

- Celluloid film
- Still pictures
- Black and white

8 hours exposure time!
Sun-light?



(Nicéphore Niépce, 1826)

Motion Capture Camera (1880s)

- "Silent era"
- Pianist, orchestra



Roundhay Garden Scene
(1888, Louis Le Prince)

VO Videoverarbeitung (The insides of video coding)

7

"Talking pictures" (1920s)

- Early days of hollywood
- Attach synchronized(!) soundtrack to film
- F.W. Murnau
(*Nosferatu*, *The last laugh*)
- Charles Chaplin, Buster Keaton
- ...
- **Color television (1960s)**



Nosferatu, 1922
(Max Schreck as Count Orlock)

VO Videoverarbeitung (The insides of video coding)

8

Video Evolution

- Projection of paintings
- Impression of movement (paintings)
- Still images of the "real" world
- Moving images
- Sound
- Color

The Digital Age (1980s)

- Projection of paintings
- Impression of movement (paintings)
- Still images of the "real" world
- Moving images
- Sound
- Color
- Video storage & transmission
- Information reduction
- Complexity vs. quality
- New applications

Information Reduction

- Removal of irrelevant information
 - Not noticeable by human observer
 - Exploit shortcomings of the human visual system (HVS)
- Removal of redundant information
 - No information loss
 - Information theory
 - Spatial & temporal similarities

The Human Visual System (HVS) (1)

- Reconstruct the spatial composition of a scene from overlapping, light, shade, motion and experience.
2-D images → 3-D scene
- Limited temporal resolution
 - Sampling with 25 frames/s
 - Displaying with 50 frames/s
- Limited spatial resolution
 - Determines resolution & distance to display

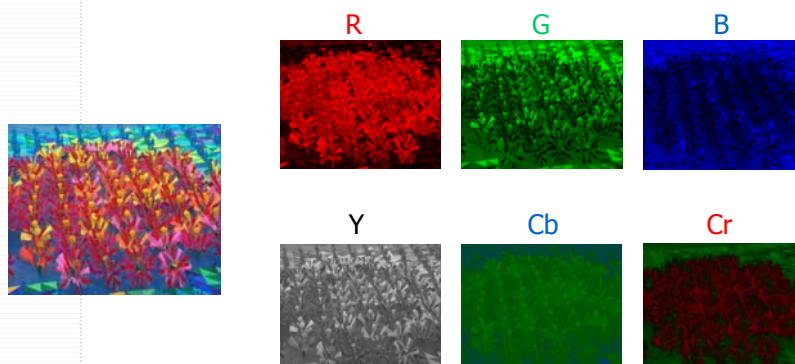
The Human Visual System (HVS) (2)

- Reduced spatial sensitivity to color information (Chroma)
- Sensor delivers RGB data
- Transformation RGB → YUV / YCbCr color space

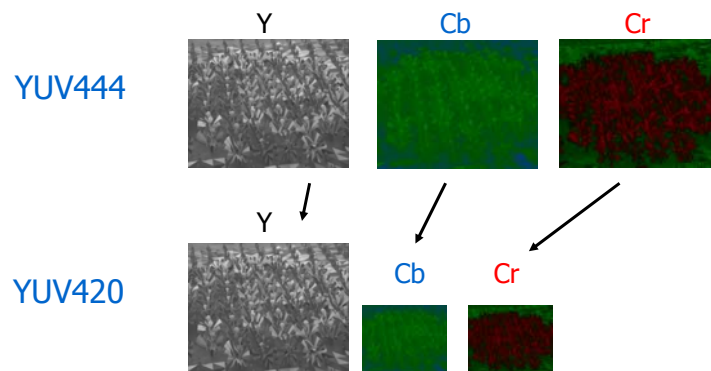
$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168 & -0.331 & 0.5 \\ 0.5 & -0.418 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}$$

- Signal changes on the luminance channel (Y) result in zero activity of the chromatic channels of the HVS
- Sub-sampling of color information

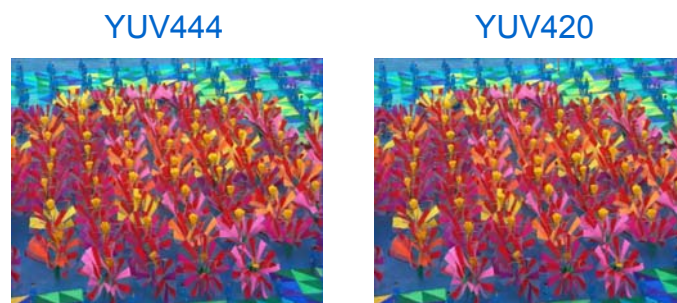
Color Conversion RGB → YCbCr



Color Sub-Sampling (1)

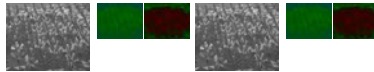


Color Sub-Sampling (2)

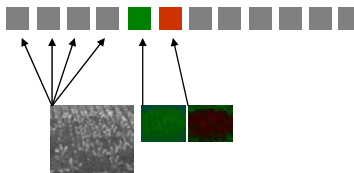


Raw YUV420 Files

Planar



Interleaved



VO Videoverarbeitung (The insides of video coding)

17

The Digital Age

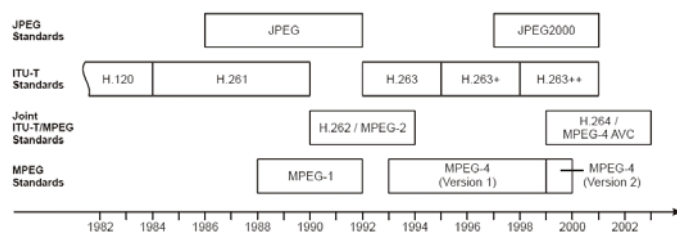
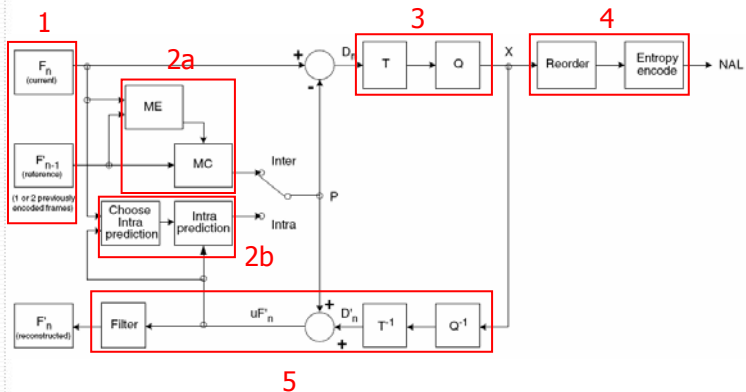


Figure 2.1: Historical development of video standards

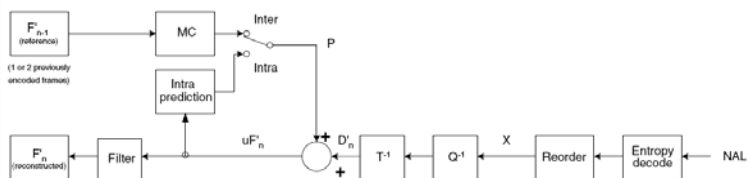
VO Videoverarbeitung (The insides of video coding)

18

Block-based video coding (H.264 Encoder)



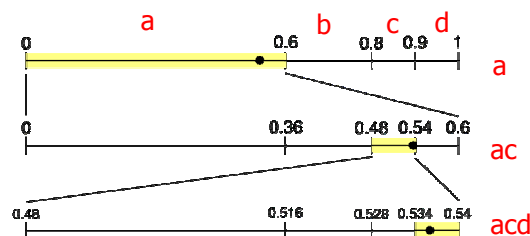
Block-based video coding (H.264 Decoder)



Entropy coding

- Remove statistical redundancy
- Variable length coding (VLC)
 - Probability-based code word generation
- Arithmetic Coding (AC)
- Context-Adaptive coding

Arithmetic Coding



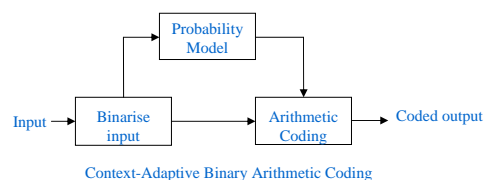
- Any number in the range [0.534, 0.54] represents the symbol sequence **acd**.
- Use the number which can be coded most efficiently in binary representation.

Binary-Arithmetic Coding

- Binary arithmetic coding (2 probability states)
- Binarization of syntax elements (SEs)
 - Syntax elements: MV, residuals, coding mode, etc.
 - SE → bins
 - Different SE binarizations

Context-Adaptive Coding

- Context-adaptive VLC / BAC
- Assumption:
 Spatially close syntax elements have similar values!
- Exploit similarity between similar values!
- Adaptive probability models for each bin

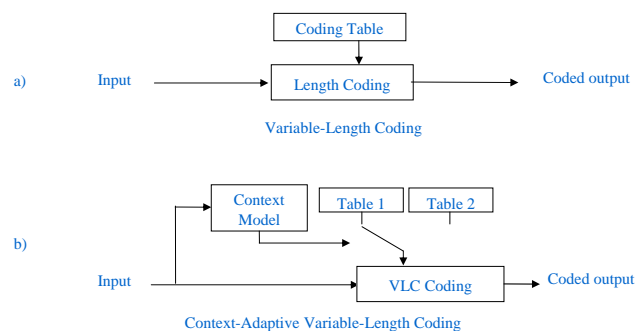


Context-Adaptiv Coding

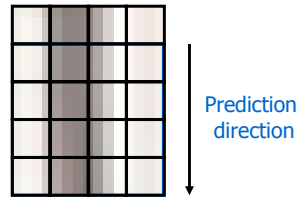
- Context-adaptive VLC / BAC
- Assumption:
 Spatially close syntax elements have similar values!
- Exploit similarity between similar values!
- Adaptive probability models for each bin

MV1 → Bins: 00110110
 MV2 → Bins: 00110001
 ↓
 Bin model 1-5 correct prediction

Context-Adaptive Variable-Length Coding (CAVLC)



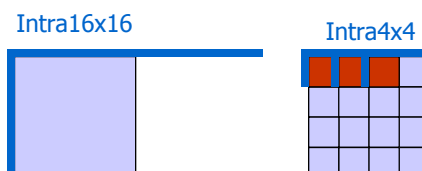
Intra prediction (1)



Predicted block Similar border pixels

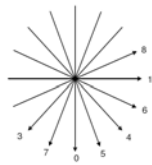
Intra prediction (2)

- Structural similarity between neighbouring pixels
- Different block sizes
 - Luma: 16x16, 4x4
 - Chroma: 8x8
- Fine structures → Smaller block sizes



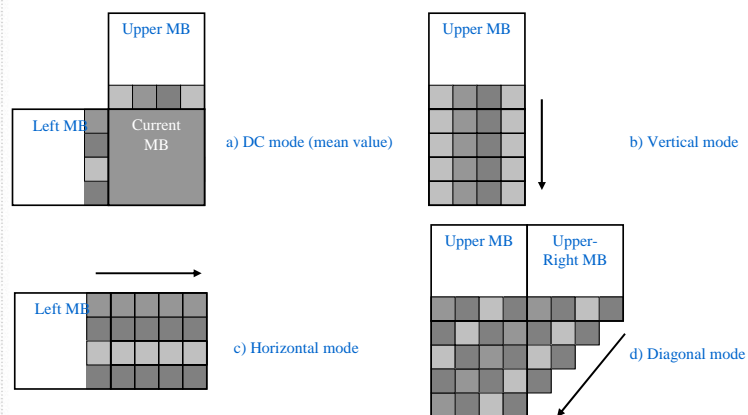
H.264 Intra modes

- H.264
 - 4 Intra16x16 modes for luma
 - 9 Intra4x4 modes for luma
 - 4 Intra8x8 modes for chroma
 - Smaller modes computationally more demanding
→ higher inter-dependencies!



Directions for Intra4x4 luma prediction
(2 = DC mode)

H.264 Intra 16x16 modes



Intra modes / Bitrate



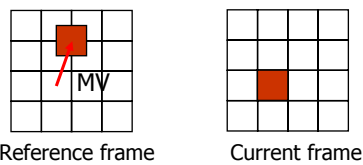
Few bits for coding macroblock

VO Videoverarbeitung (The insides of video coding)

31

Motion Estimation (1)

- Register each macroblock in previous frame
→ Motion vector (MV)



- SAD as correlation measurement
- Computationally demanding (up to 80% of processing time)
- Search strategy not covered by standards

VO Videoverarbeitung (The insides of video coding)

32

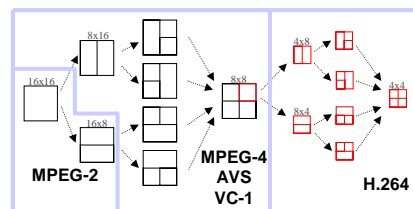
Motion Estimation (2)

- MV resolution:
 - Motion typically subpixel-wise
 - Subpixel precision by up-scaling images

Features	MPEG-2	MPEG-4 ASP	H.264
MV resolution	½ pixel	¼ pixel	¼ pixel

Motion Estimation (3)

- Variable block sizes:



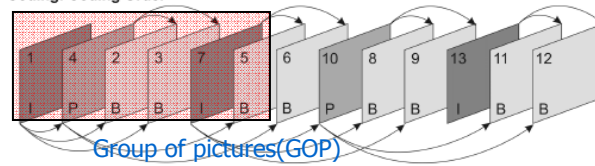
- Comparison:

Features	MPEG-2	MPEG-4 ASP	H.264
Vector block size	16x16, 16x8	16x16, 8x8	16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4

Motion Estimation (4)

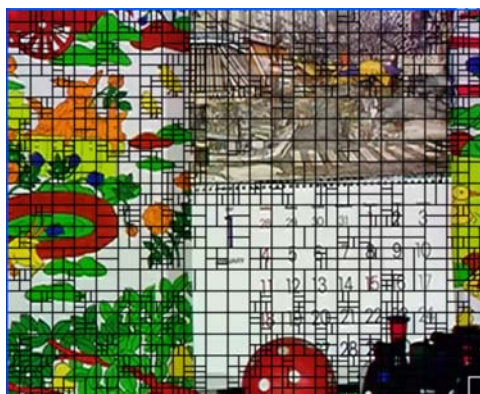
- Multiple reference frames

IBBP-Coding: Coding Order

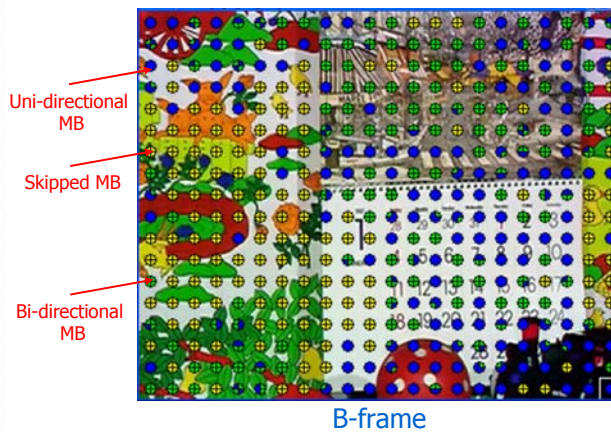


Features	MPEG-2	MPEG-4 ASP	H.264
P-frames	1	1-4	1-5
B-frames	1 / direction	Not in SP	multiple / direction

Temporal prediction (Partition size)



Temporal prediction (Coding mode)



VO Videoverarbeitung (The insides of video coding)

37

H.264 Intra vs. Inter prediction (1)

Intra

- Spatial correlation
- Pixel-based
- Simple pixel propagation

Inter

- Temporal correlation
- Block-based
- Structure and intensity propagation

VO Videoverarbeitung (The insides of video coding)

38

H.264 Intra vs. Inter prediction (2)

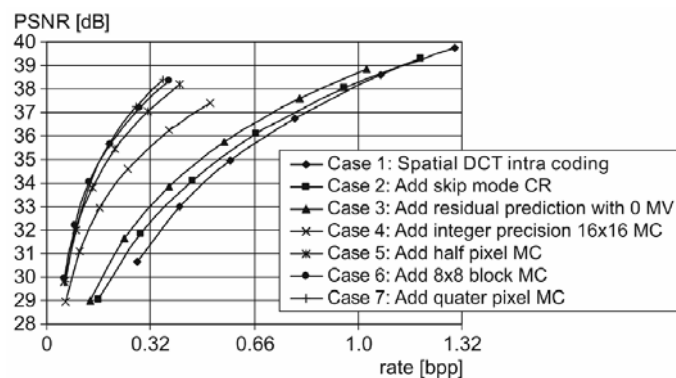
Intra

- Dependency on spatial neighbour MBs
- Small feed-back loop
- Unfiltered pixels as reference

Inter

- Dependencies on other frames
- Large feedback loop
- Filtered pixel as references (In-loop filtering)

Quality vs. Bitrate



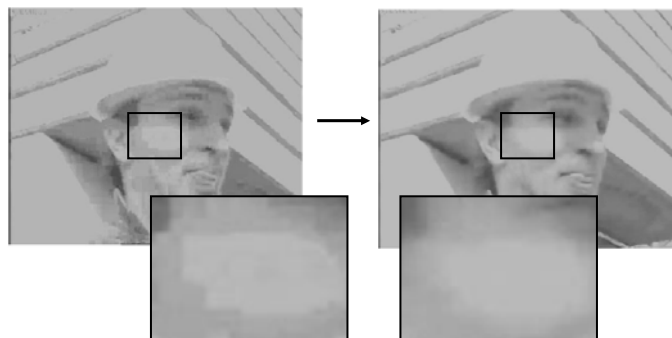
(Sullivan, Wiegand, 2003)

Discret Cosine Transform (DCT)

- Transform data to frequency domain
- Eye is less sensitive to high frequencies
- HERE the information loss takes place (quantization!)
- 4x4 Integer transform
 - Accuracy
 - Implementation independent
- Quantisation levels (0-51)...
- Rate control

Features	MPEG-2	MPEG-4 ASP	H.264
Transform	8x8 DCT	8x8 DCT	4x4, 8x8, Integer

Deblocking Filter (1)



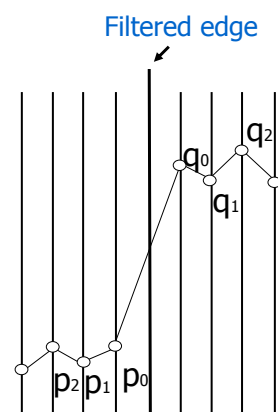
Deblocking Filter (2)

- Removal of blocking artifacts
- Artifacts caused by block-based coding
- No filtering over slice boundaries
 - Flag in H.264 allows exception!
- Post-processing vs. In-loop deblocking
- In-loop deblocking must be covered by standard

Features	MPEG-2	MPEG-4 ASP	AVS	VC-1 / WM-9	H.264
Deblocking	Post	Post	In-loop	In-loop	In-loop

H.264 Deblocking Filter (1)

- Removing of blocking artifacts
- Filtering on 4x4 blocks
- Heuristic for distinguish between „real“ edges & artifacts
 - Quantisation value QP
 - Filter strengths α , β



H.264 Deblocking Filter (2)

- Filtering p_0 and q_0

$$|p_0 - q_0| < \alpha(QP)$$

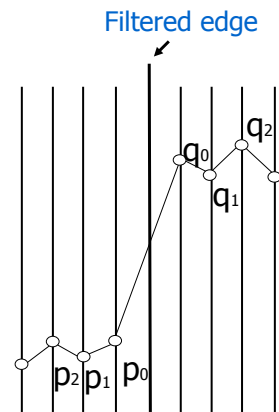
$$|p_1 - p_0| < \beta(QP)$$

$$|q_1 - q_0| < \beta(QP)$$

- Filtering of p_1 and q_1

$$|p_2 - p_0| < \beta(QP)$$

$$|q_2 - q_0| < \beta(QP)$$



Resource Scarcity

- Computation power (33 - 300 MHz)
- Local memories / caches (16 - 128 kB)
- External memory accesses
- Bus bandwidth
- Power consumption
- Chip area / cost

Main Applications

- Video conferencing and video telephony
- Video broadcasting
 - Digital video broadcast (DVB)
 - Digital media broadcast (DMB)
- Video storage and retrieval
- Streaming

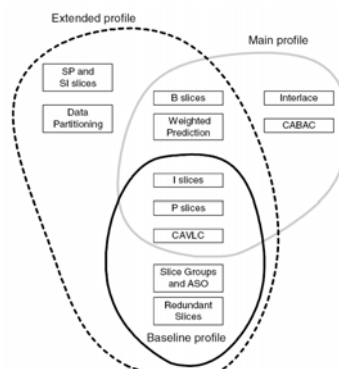
Application Requirements

- Coding efficiency
- Reliability
- Low-complexity
- Scalability
- Interlacing
- Low latency
- Lossless or near lossless
- Efficient transcoding

H.264 Levels & Profiles

- Levels
 - Restriction of resolution
 - Restriction of data throughput (Macroblocks/s)
- Profiles
 - Restriction of coding tools
 - Baseline, Main, High
 - Extended, FrExt → Error-resilience

H.264 Profiles



Numerical issues

- Integer transform
- In previous standards precision requirements
- 16-bit registers
- NO multiplications
 - shifts / adds for multiplications
 - Look-up tables for range divisions in binary arithmetic coding (figure!)

Dedicated HW extensions

- CAVLC / CABAC accelerator
- Pixel-based operations (SIMD, VLIW)
 - IDCT
 - Motion estimation / prediction
 - Filtering
- Intelligent DMA engines
 - 2D – transfers
 - Simple processing tasks (clipping, border expansion, etc.)

Standard Restrictions

- Application dependent
- Resolution dependent
 - 3GPP (Resolution, coding tools, etc.) → MMS
 - H.320 (ISDN-based video services)

IMPACT ON BITRATE!

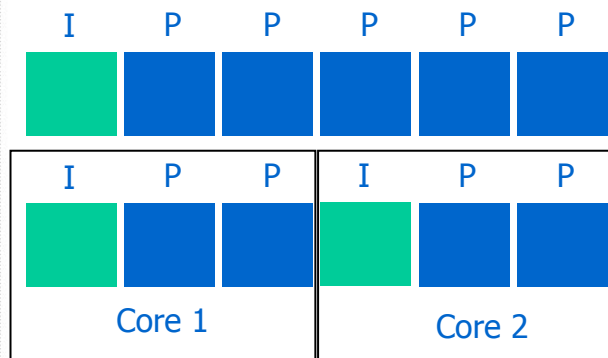
- Standard compliance required in H.264 decoder!

Reduction in Quality

- Simple ME algorithms
- Higher quantization
 - Reduces workload on entropy coder
- Only possible at encoder side!

IMPACT ON BITRATE!

Parallelization (Encoder)

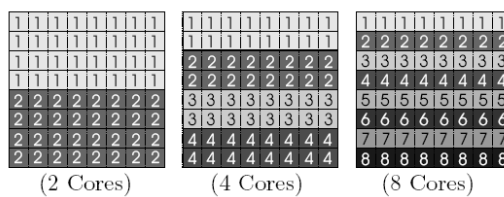


Parallel approaches (Decoder)

- Functional splitting
- Run decoding tasks on different CPUs
- Advantages
 - „Specialized“ blocks → typically smaller & less memory / cache
 - Strong algorithmic differences can be addressed by dedicated CPU extensions
 - Highly conditional parts (E.g. entropy coding, etc.)
 - Pixel-based signal processing parts (E.g. DCT, filtering)
- Problems
 - Unequal workload-balancing
 - Splitting interfaces must be defined & implemented
 - High data-transfers

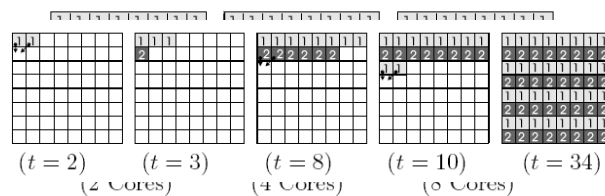
Parallel approaches (Decoder)

- Data-parallel approaches
 - Process luma / chroma data separately
 - Split frame into multiple slices and process them separately
→ Encoder has to support this!



Splitting approaches (Decoder) (1)

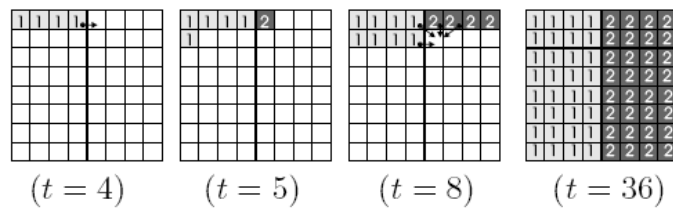
- Single-row splitting
- Each core works on one row



Encoder independent!

Splitting approaches (Decoder) (2)

- Multi-column approach



Encoder independent!

Attributes

1. Provided by Andrew Dunn
<http://www.andrewdunnphoto.com>

Thank you for your attention