# Reference Material

Background on Digital Media
http://xiph.org/video/vid1.shtml http://xiph.org/video/vid2.shtml

Excellent Overview of Media Compression at
http://people.xiph.org/~tterribe/pubs/lca2012/auckland/intro_to_video1.pdf ;
https://www.xiph.org/daala/

HEVC Information http://hevc.hhi.fraunhofer.de/
http://www.atlanta-smpte.org/HEVC-Tutorial.pdf

H.264 Information http://www.itu.int/rec/T-REC-H.264

Tools : www.ffmpeg.org http://www.videolan.org/

VP9 Presentation at Google IO 2013 : http://www.youtube.com/watch?v=K6JshvblIcM

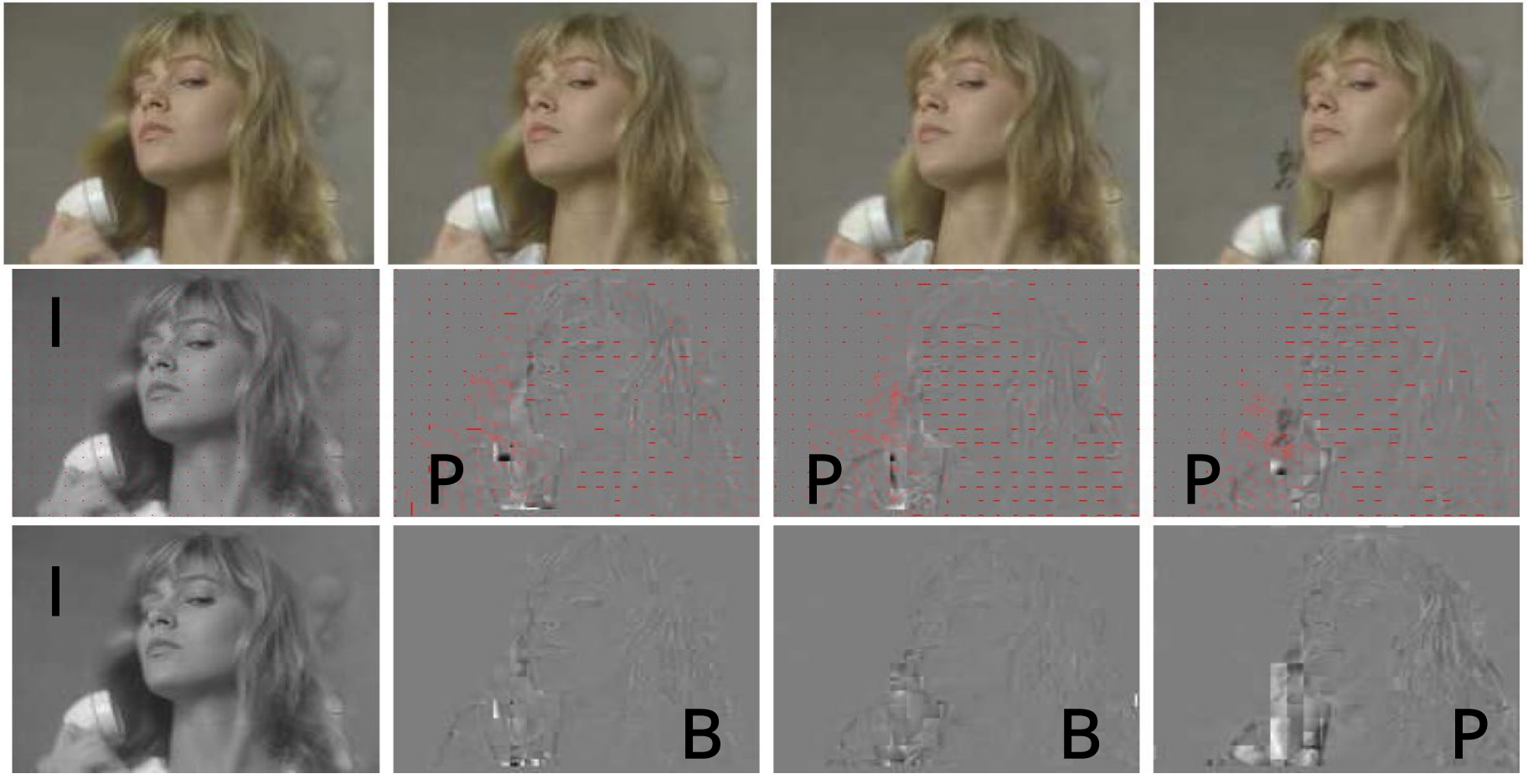
Wikipedia actually quite good on this stuff

Motion Picture Engineering

Prof. A. Kokaram

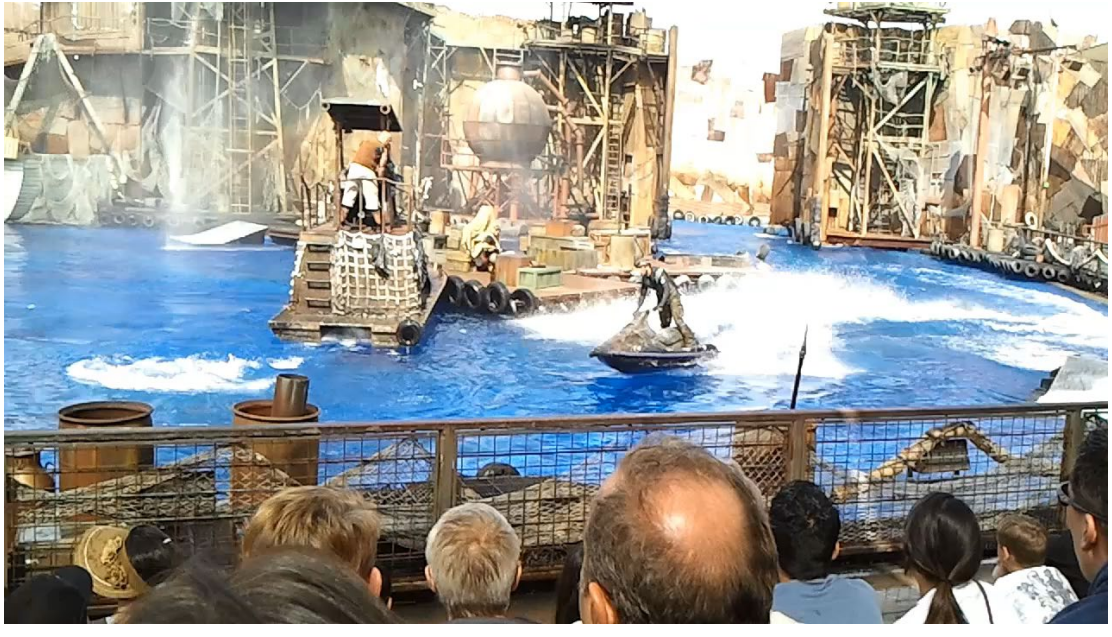
# MODERN VIDEO STANDARDS

# MC Difference Sequence : Smarter Prediction



# Examine .MP4 container with H.264 essense

Video recorded and packaged by a Samsung Galaxy SII
(Using www.codecvisa.com for easy to use diagnostic, could also use ffmpeg)



Starting from 1st frame : I, P, P, B, P, B, B, P, P, P, P, P, P, B, B, P, B, B, P, B, B ....

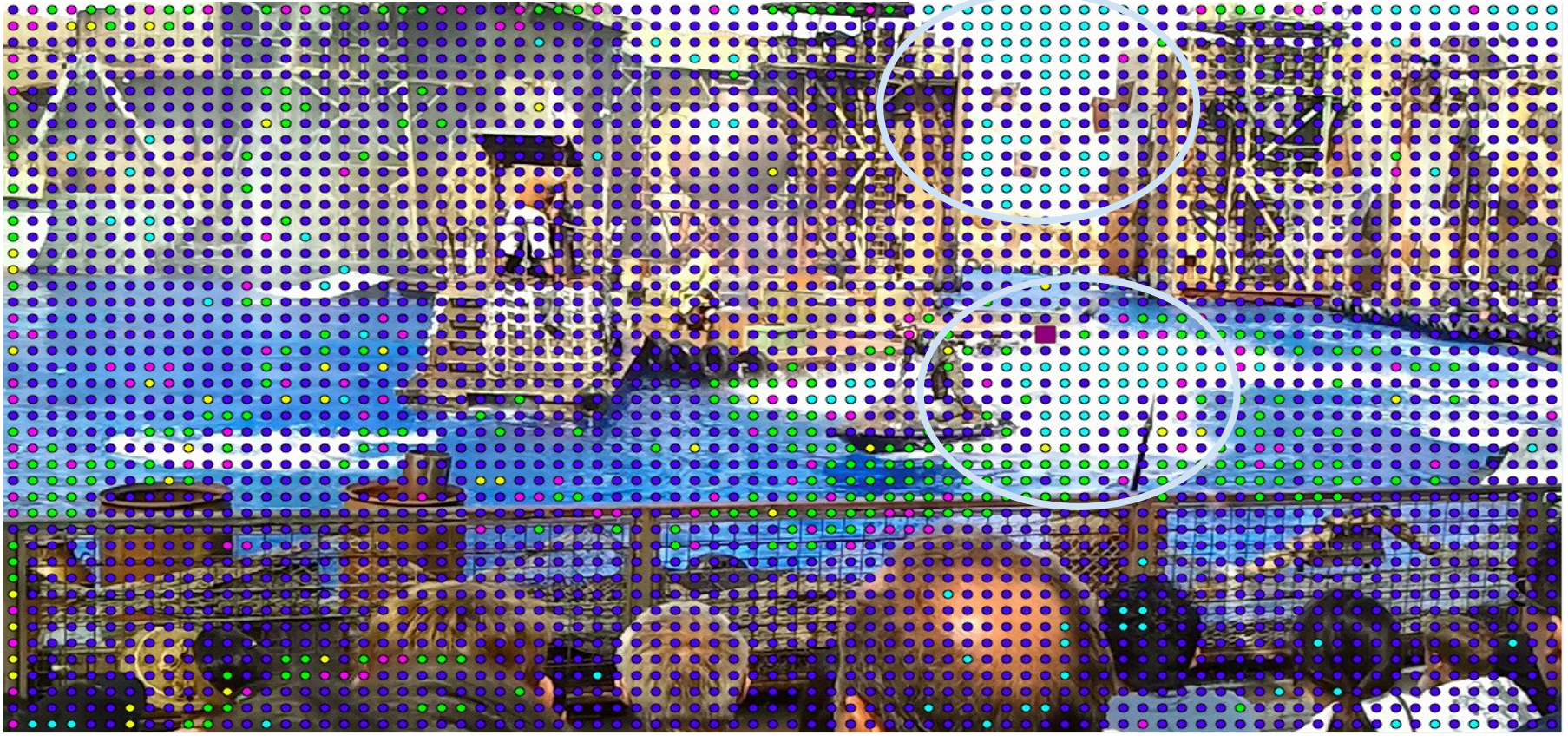
I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Frame 0, Intra-Frame (PTS -0, DTS - 0) : 1.3 Mbits

Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4

I, P, P, B, P, B, B, P, P, P, P, P, P, B

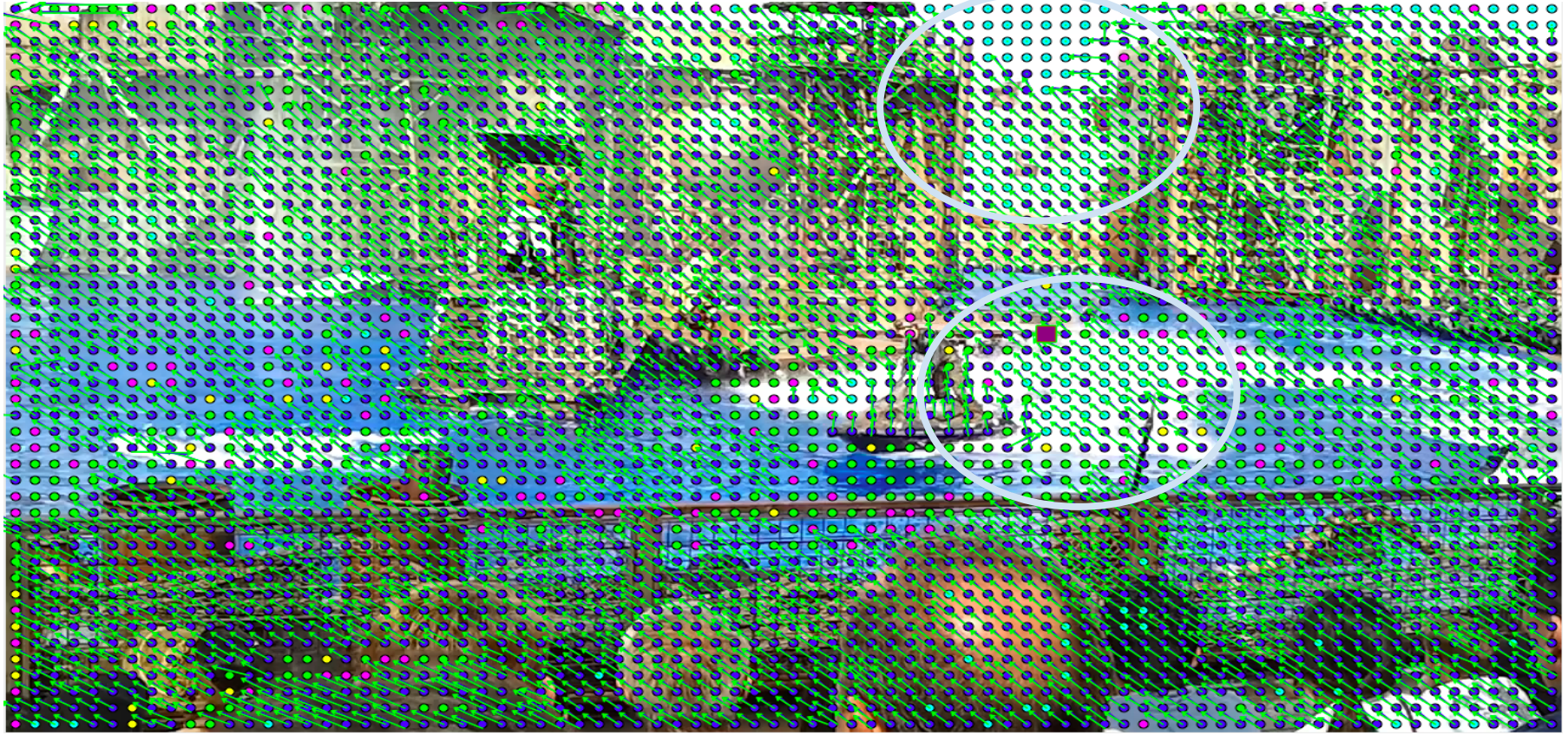
# Frame 1, P-Frame (PTS -1, DTS - 1) : 0.6 MBits



Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 P_MB_16x16 (8) P_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

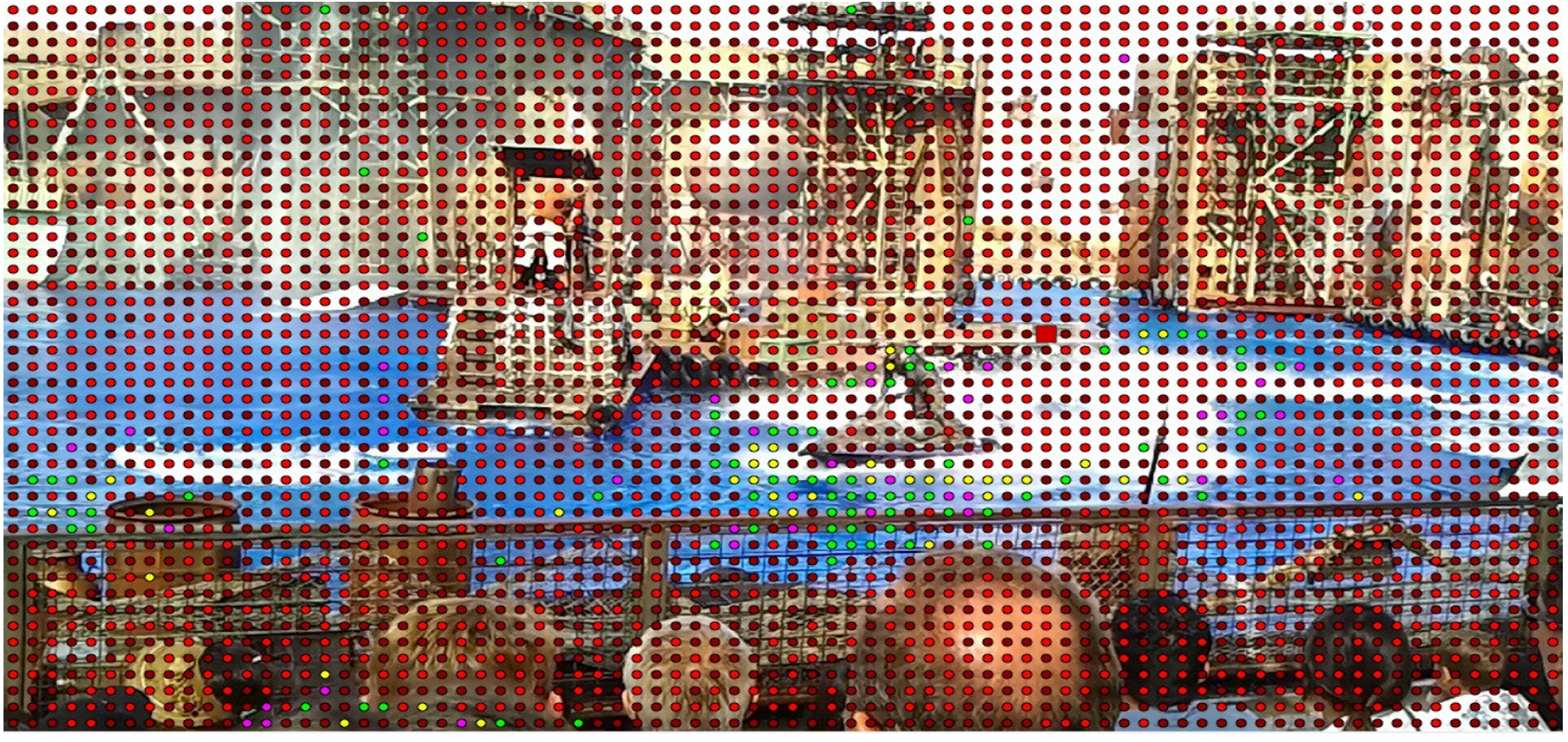
# Frame 1 Motion, P-Frame (PTS -1, DTS - 1)



Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 P_MB_16x16 (8) P_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

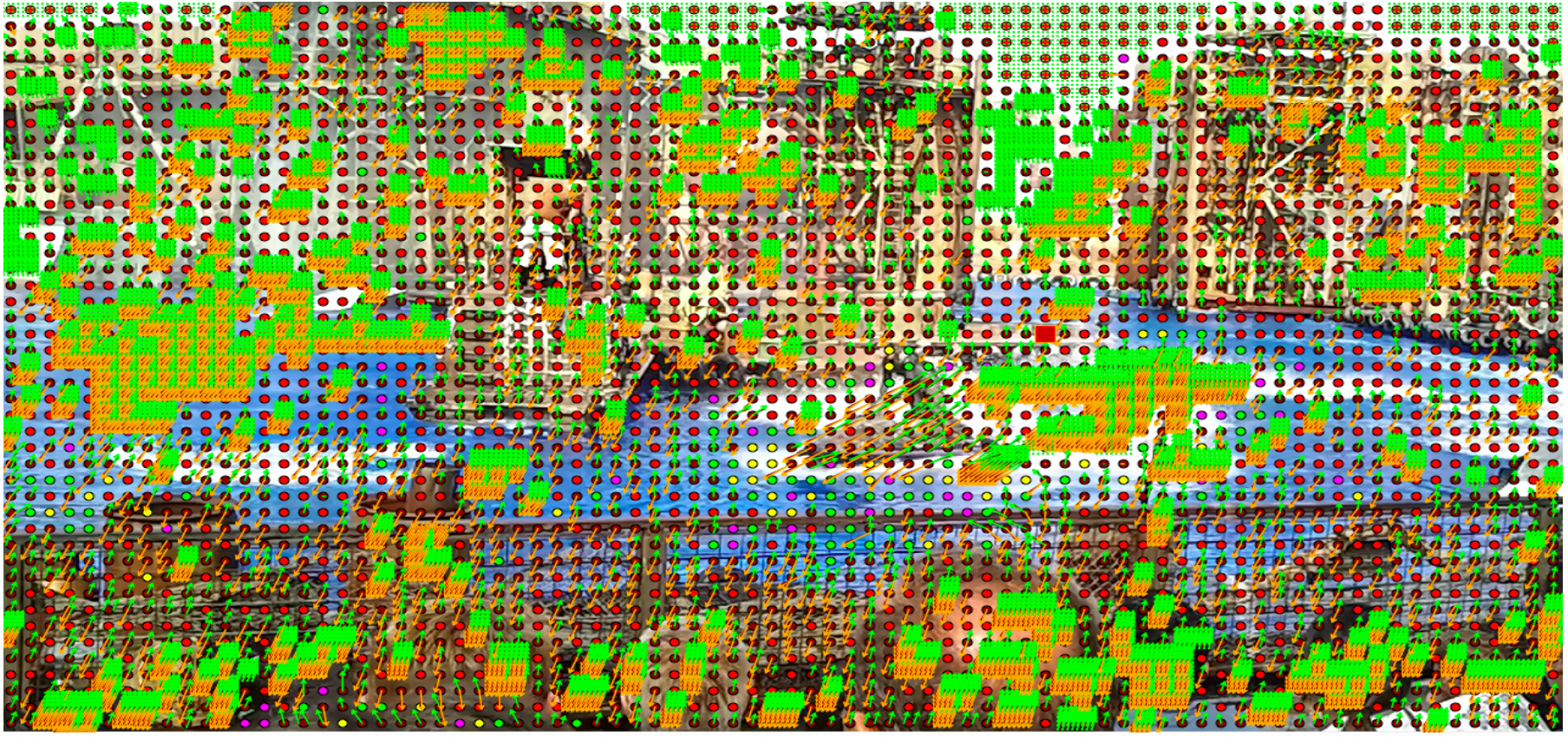
# Frame 3, B-Frame (PTS -3, DTS - 4) : 0.2 Mbits



Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Frame 3, Motion Vectors



Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

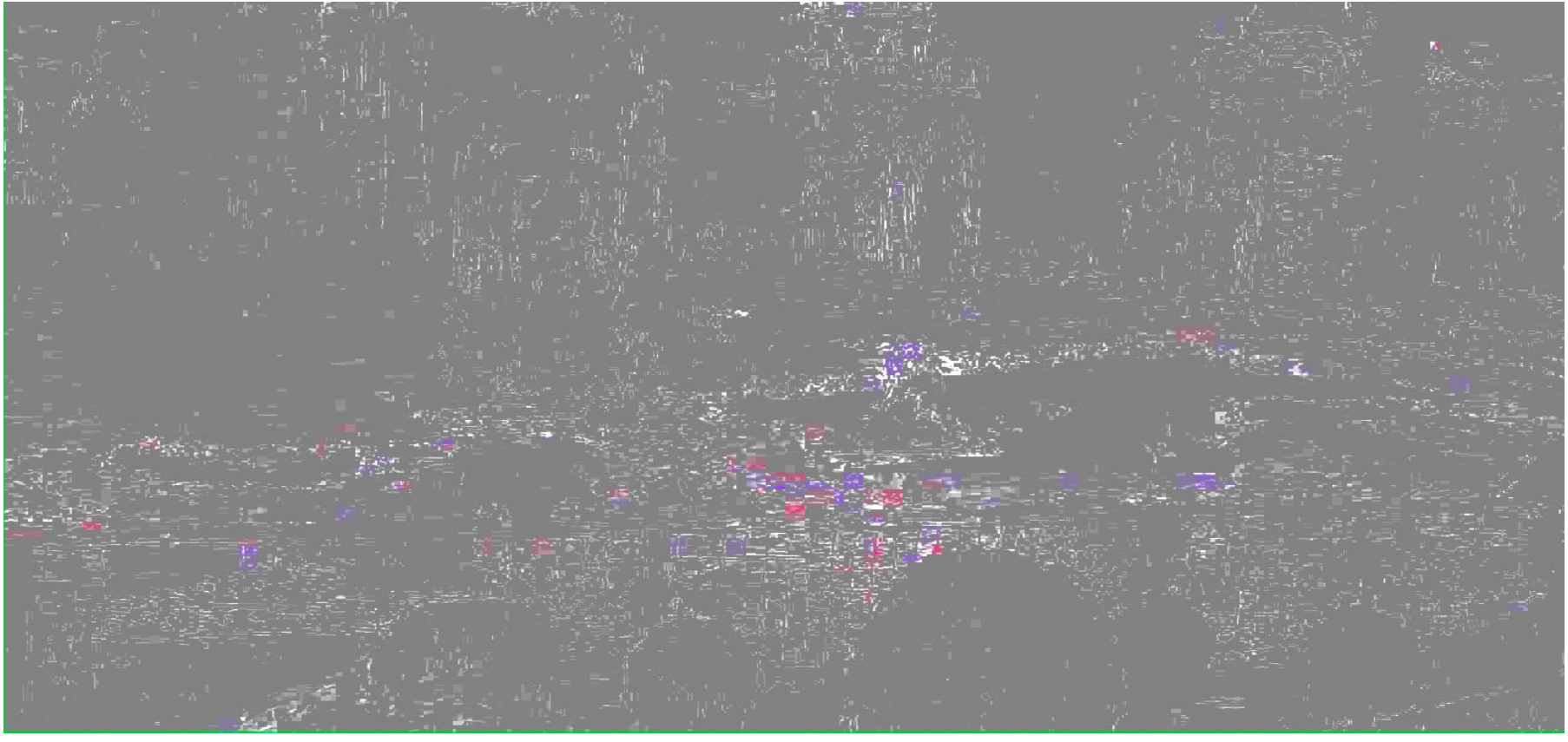
I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Predicted Frame 3, B-Frame, uses Frames 1 (I) and 4 (P)

Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Residual at Frame 3, B-Frame, [i.e. Inverse DCT of MBs]



Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Final Decoded Frame 3, B-Frame

Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Predicted Frame 3, B-Frame, uses Frames 1 (I) and 4 (P)

Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Predicted Frame 7, P-Frame, uses Frame 4 (P)

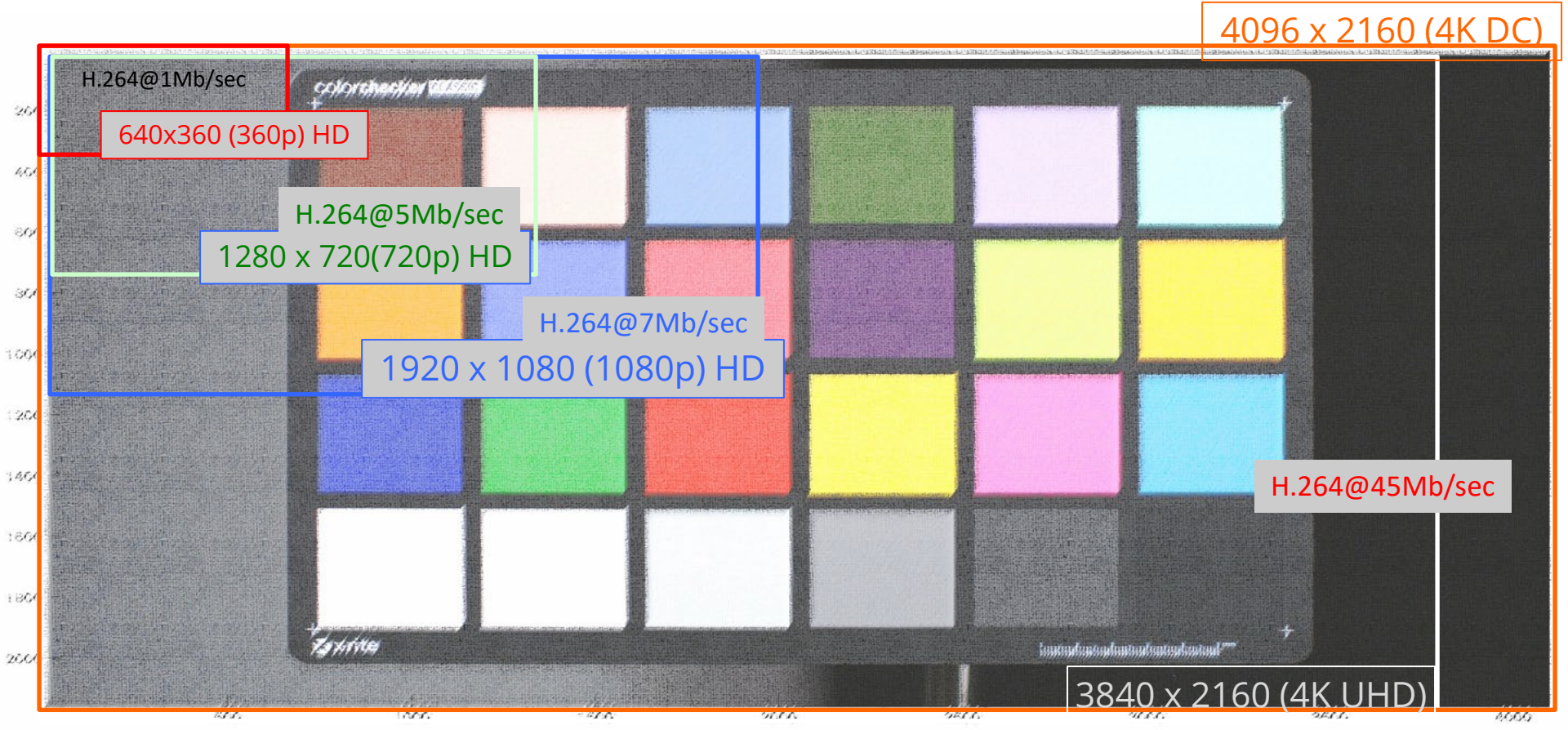
Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

I, P, P, B, P, B, B, P, P, P, P, P, P, B

# Final Decoded Frame 7, P-Frame, uses Frame 4 (P)

Intra_MB_16x16 Intra_MB_8x8 Intra_MB_4x4 B_MB_16x16 (8) B_MB_Skip

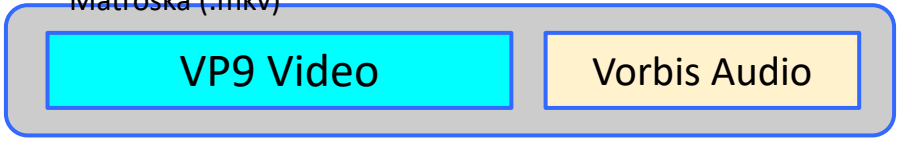
# Picture Formats/Sizes and “rules of thumb bitrates” to remember



The two sides met at Picture Coding Symposium 2013

# VP9 and HEVC

- VP9 from Google Open Web Platform Project WEBM
- Developed by Chrome Media team
- Started in 2011
- Open source project developed entirely in public domain
- Unorganised progress, compression gains measured against test set every day
- Bitstream finalised June 2013
- Launched at Google IO 2013
- *YT streams 4K to an LG 4K TV Set in CES 2014*
- No Royalty payment needed
- Bitstream documented by the code!




- HEVC is latest evolution of H.26x (H.265)
- MPEG Consortium
- Very organised, documents available
- Bitstream documented and available online at ITU
- Highly academic driven. See Special Issue of IEEE CSVT 2012 on HEVC. Also won best paper prize (Wiegand).
- Goal 50% lower bitrate at same quality as H.264
- Draft since 2012, finalised in 2013.
- *Akami/Qualcomm/Elemental Demo HEVC streaming at CES 2014.*

# Why the fuss?

- Licensing fees for H.264 and variants are significant and affect the cost of hardware codecs and hence phones
- Streaming media companies also pay licensing for encoding.
- MPEG/ITU : Important “trusted” *international standardisation bodies*. Enabled the underbelly of the digital media marketplace in the late 90’s and since. If MPEG say that a bitstream is like so, then it must be so?
- Google/YT : *A private company* committed to Open Source code. As CPUs get faster and consumer lower power codecs become viable in software. Open source software used by a large community _can_ be more robust and _faster_ than software from a closed community. Google/YT are even releasing *open source hardware IP*!

# The News from 2015

http://youtube-eng.blogspot.ca/2015/04/vp9-faster-better-buffer-free-youtube.html

MPEG-LA announced HEVC licensing terms in October 2015.
[http://www.mpegla.com/main/programs/HEVC/Pages/Agreement.aspx]

20c per unit over 100K units, max of $25M/year backdated from 2013!

Conservative 35 Million Iphones sold in 1 Quarter. So Apple already at $25M Cap.

Scared everybody into looking at VP9 again. Netflix using VP9 for all 480p and below.

**MPEG LA's HEVC Licensing Terms Are Flawed, Will Prevent Adoption**

There's no reason for Adobe to add HEVC support to Flash, and that keeps HEVC from becoming relevant for general purpose streaming.

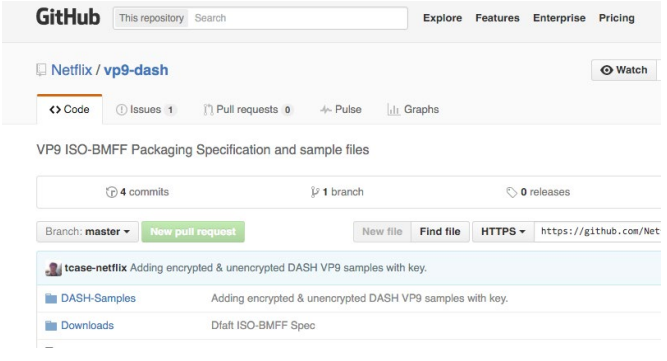
**By** *Jan Ozer*

**For the rest of the June 2014 issue of Streaming Media magazine please** *click here*

**Page 1**

The Producer's View

As you might know, MPEG LA announced the licensing terms for HEVC back in January. At the time, I thought, as most observers did, that those terms would provide long-needed certainty. Upon further reflection, I think they're flawed and will delay, if not prevent, the adoption of HEVC in the streaming space. For HEVC to flourish there, MPEG LA needs to dramatically revamp the structure for free players.



GitHub This repository Search Explore Features Enterprise Pricing

Netflix / vp9-dash Watch

Code Issues 1 Pull requests 0 Pulse Graphs

VP9 ISO-BMFF Packaging Specification and sample files

4 commits 1 branch 0 releases

Branch: master New pull request New file Find file HTTPS https://github.com/Net

tcase-netflix Adding encrypted & unencrypted DASH VP9 samples with key.

DASH-Samples Adding encrypted & unencrypted DASH VP9 samples with key.

Downloads Dfaft ISO-BMFF Spec

# The Later News

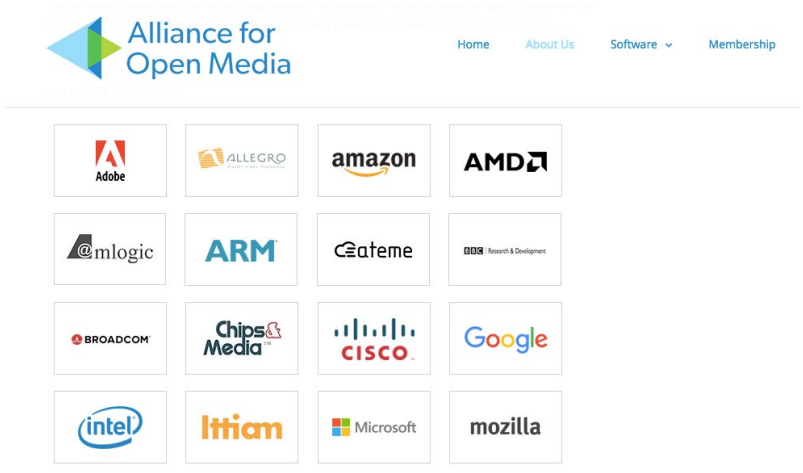
Alliance for Open Media

Announced Sept 2015

http://aomedia.org/

# The 2016 News

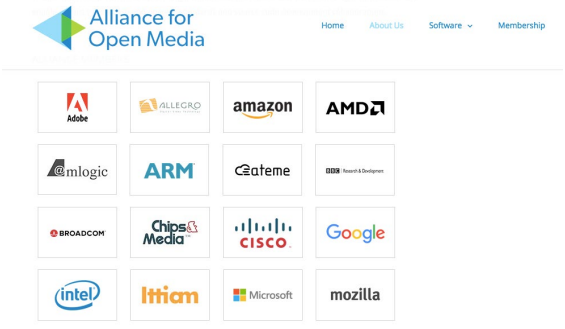
- 16 companies involved in AOM Codec
- Netflix commits to VP9 (Oct 2016)
- MPEG “Future Video Coding” Launches (H.266)



# @Netflix in Los Gatos Nov 2016


# The Latest News

- (2020) AV1 Rolls out in Netflix, YouTube
- (2019) AV2 development begins, MPEG5 Essential Video Coding begins
- (2018) AV1 Bitstream frozen this month?
- (2018) Film grain synthesis is part of the codec .. Required by Netflix.
- (2018) Qualcomm and other hardware manufacturers are not that keen on this extra bit of complexity
- (2017) AOM Codec now called AV1
- (2017) Apple announced that it will commit to HEVC (uh oh)
- (2017) Facebook joined AOM
- See here for more tension




Wonderful exposition of history of MPEG is here

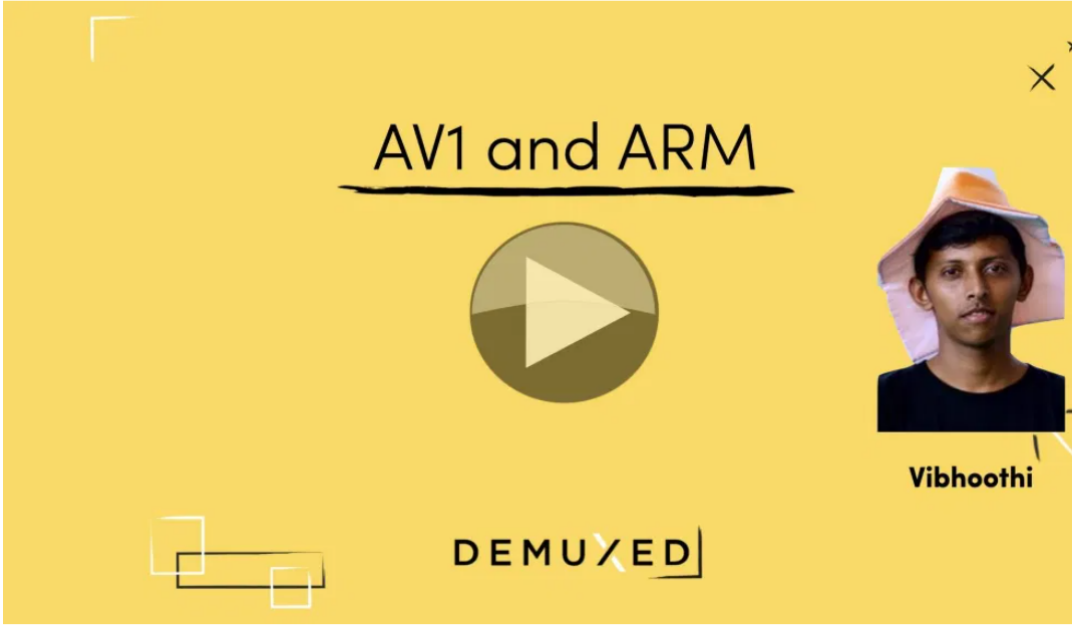
## Really Really Latest News

Dav1D AV1 encoder in software available since 2020

See here for a detailed exposition by Viboothi (Major contributor to the Open Source AV1 initiative)

# Video: AV1 and ARM

Posted on 16th March 2021 by Russell Trafford-Jones




AV1's no longer the slow codec it was when it was released. Real-time encodes and decodes now practical with open-source software implementations called rav1e for encoding and da for decoding. We've also seen in previous talks the SVT-AV1 provides real-time encoding and WebRTC now has a real-time version with the AV1 codec.

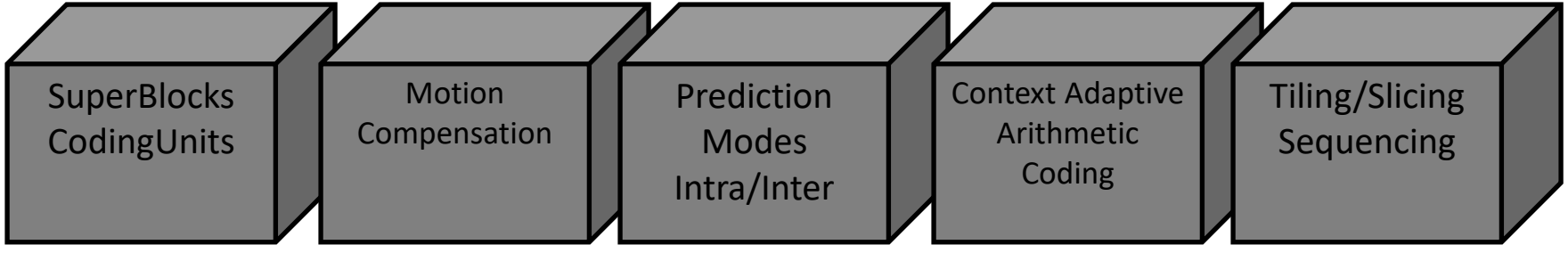
# Profiles for VP9/HEVC

| VP9 | HEVC |
| --- | --- |
| 4:2:0 Launched 2013<br>4:2:2 Launched 2015<br>4:4:4 Launched June 2014 | 4:2:0 4:2:2 4:4:4<br><br>Main (see next slide) |
| 10 bit 12 bits Launched June 2014 | Main 10<br>10 bit |

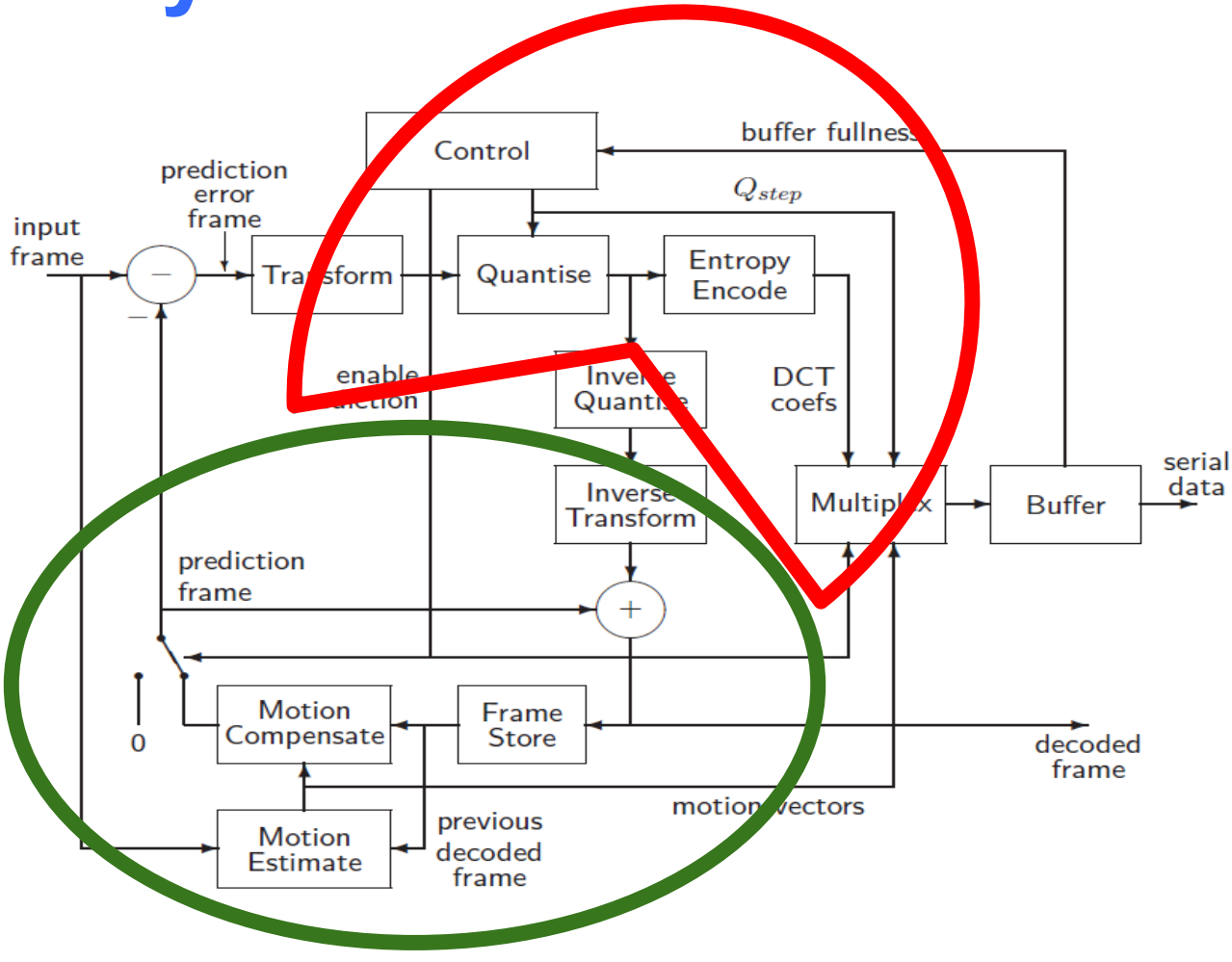
# Levels, Tiers for HEVC

| Level | Max Luma Picture Size (samples) | Max Luma Sample Rate (samples/s) | Main Tier Max Bit Rate (1000 bits/s) | High Tier Max Bit Rate (1000 bits/s) | Min Comp. Ratio |
|---|---|---|---|---|---|
| 1 | 36 864 | 552 960 | 128 | – | 2 |
| 2 | 122 880 | 3 686 400 | 1500 | – | 2 |
| 2.1 | 245 760 | 7 372 800 | 3000 | – | 2 |
| 3 | 552 960 | 16 588 800 | 6000 | – | 2 |
| 3.1 | 983 040 | 33 177 600 | 10 000 | – | 2 |
| 4 | 2 228 224 | 66 846 720 | 12 000 | 30 000 | 4 |
| 4.1 | 2 228 224 | 133 693 440 | 20 000 | 50 000 | 4 |
| 5 | 8 912 896 | 267 386 880 | 25 000 | 100 000 | 6 |
| 5.1 | 8 912 896 | 534 773 760 | 40 000 | 160 000 | 8 |
| 5.2 | 8 912 896 | 1 069 547 520 | 60 000 | 240 000 | 8 |
| 6 | 35 651 584 | 1 069 547 520 | 60 000 | 240 000 | 8 |
| 6.1 | 35 651 584 | 2 139 095 040 | 120 000 | 480 000 | 8 |
| 6.2 | 35 651 584 | 4 278 190 080 | 240 000 | 800 000 | 6 |

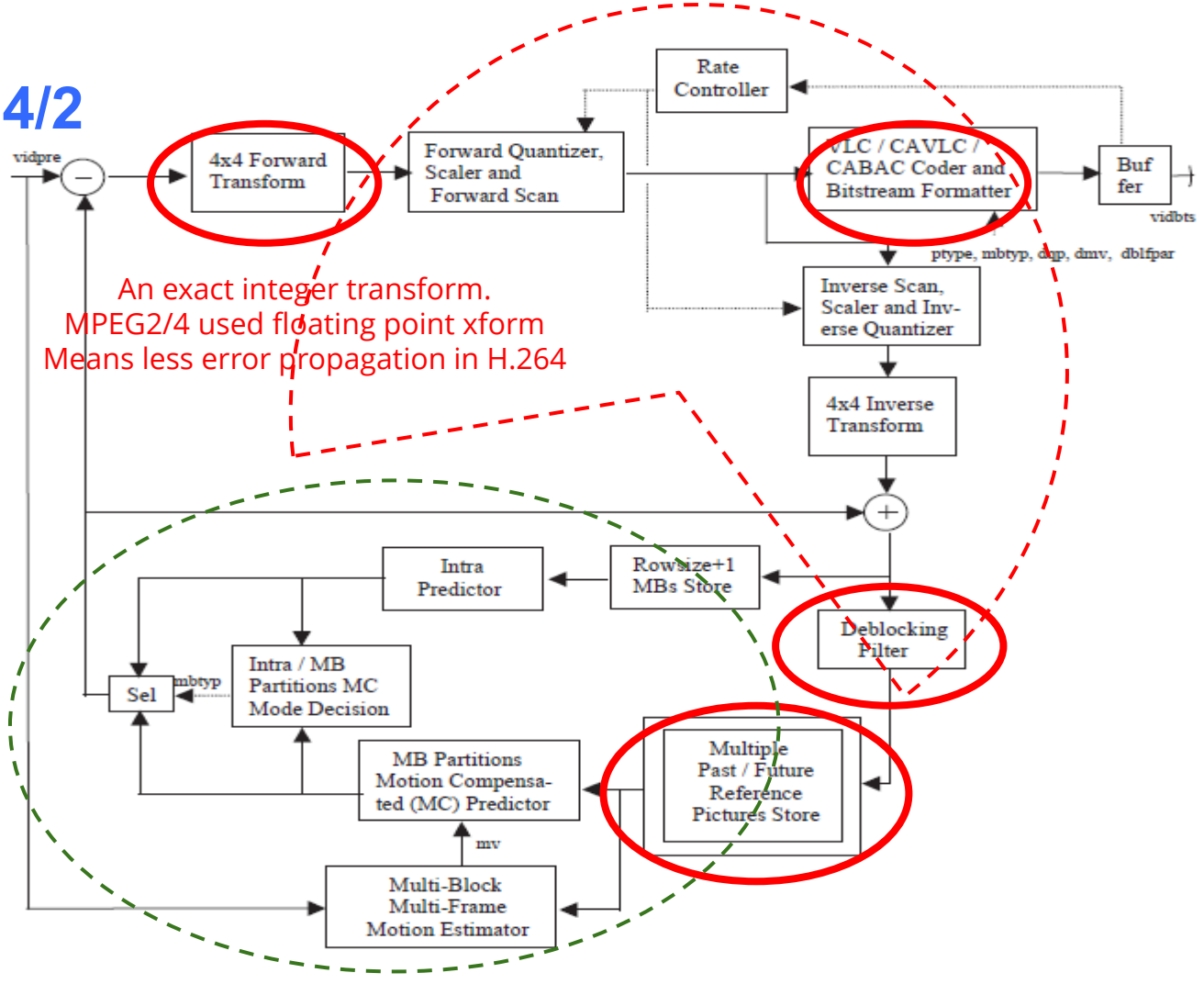
# Subset of The New Tools



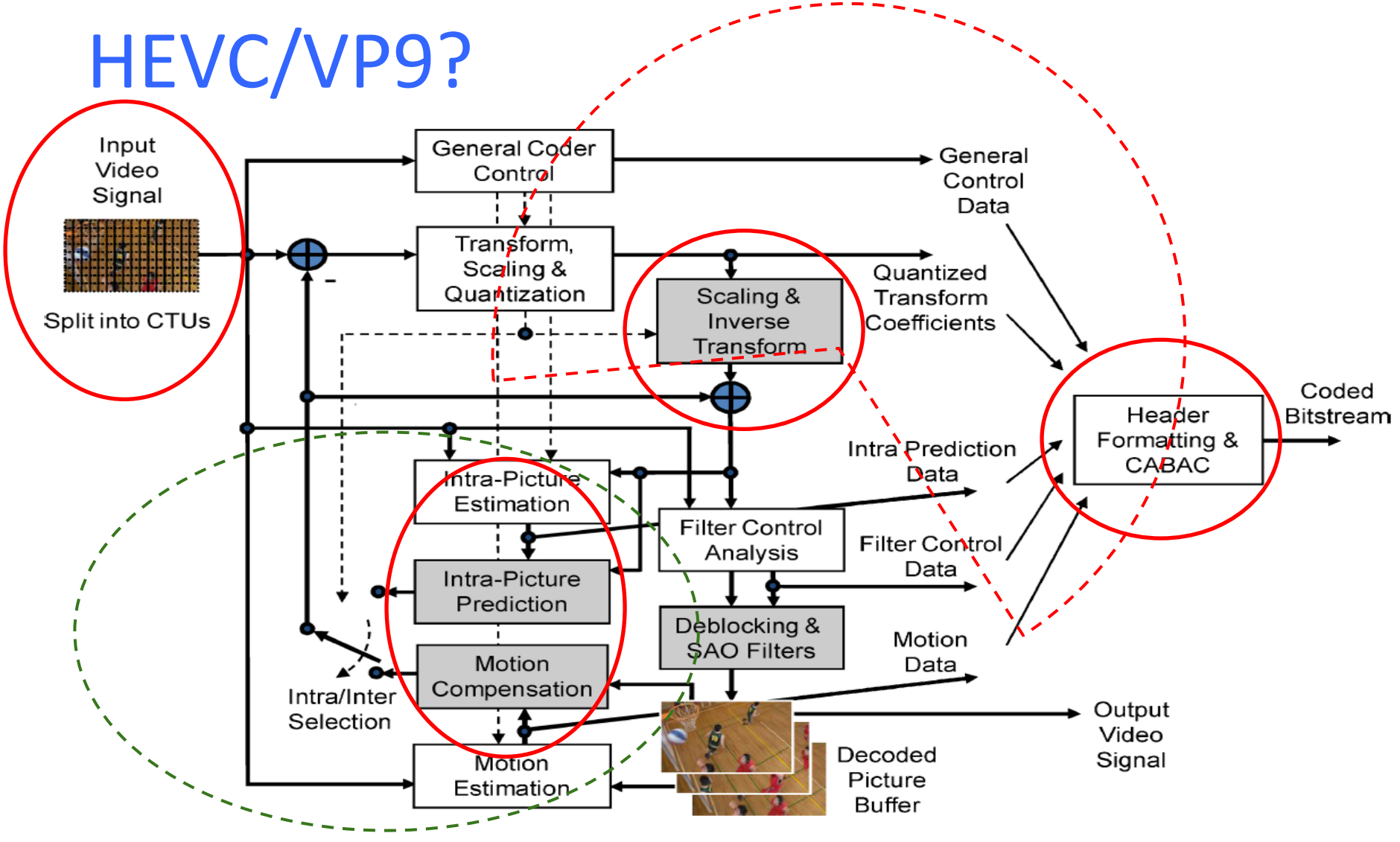
# Vanilla Hybrid Encoder Block Diagram



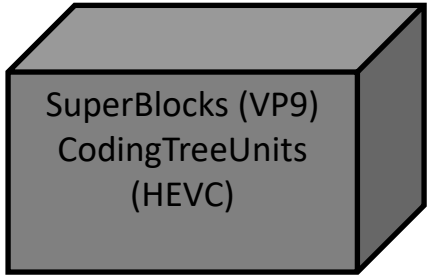
# H.264 Vs MP4/2

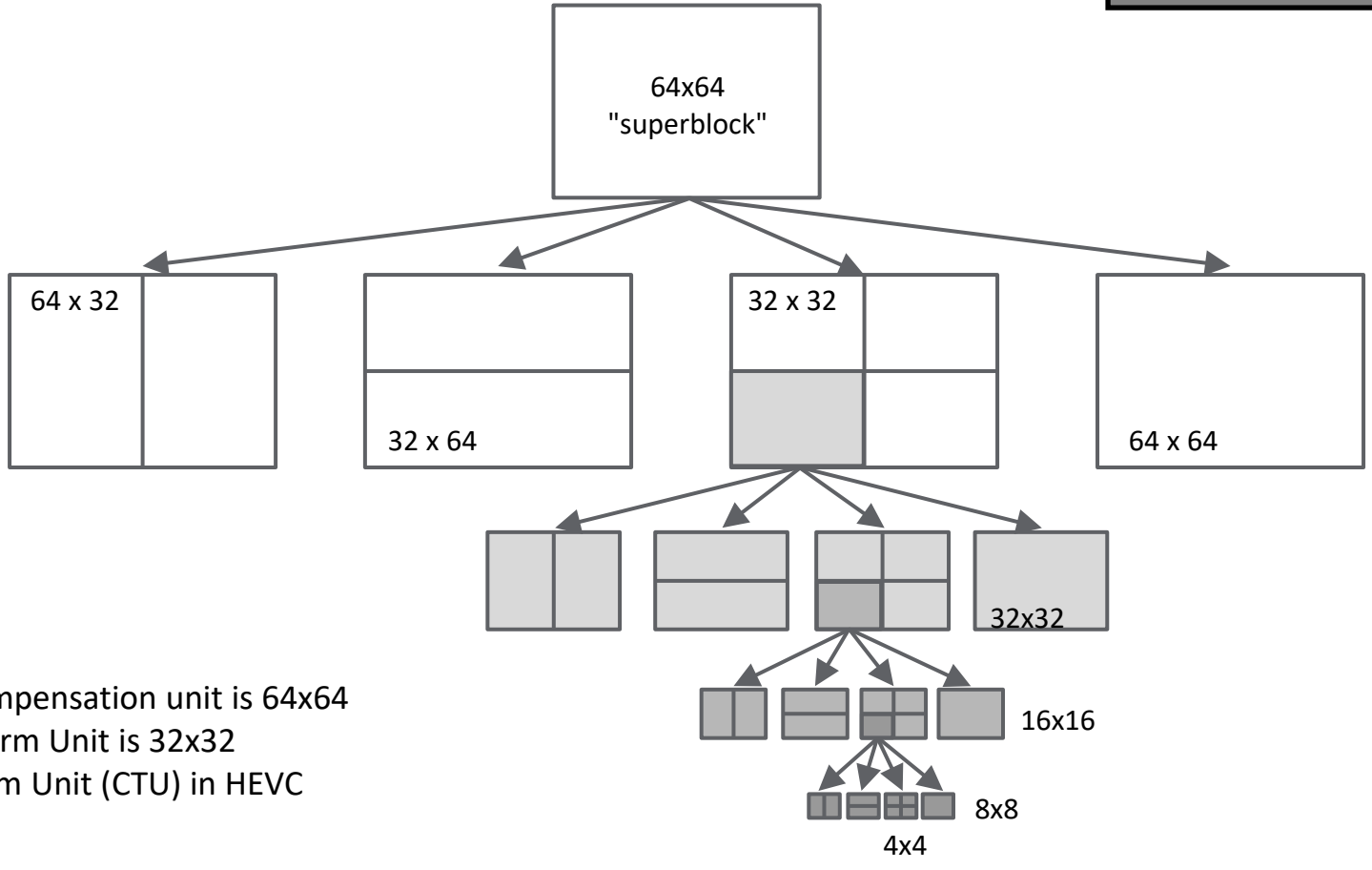

Rate Controller
vidpre
4x4 Forward Transform
Forward Quantizer, Scaler and Forward Scan
VLC / CAVLC / CABAC Coder and Bitstream Formatter
Buf fer
vidbts
ptype, mbtyp, cbp, dmv, dblfpar
Inverse Scan, Scaler and Inv- erse Quantizer
4x4 Inverse Transform
An exact integer transform.
MPEG2/4 used floating point xform
Means less error propagation in H.264
Intra Predictor
Rowsize+1 MBs Store
Deblocking Filter
Sel
mbtyp
Intra / MB Partitions MC Mode Decision
MB Partitions Motion Compensa- ted (MC) Predictor
Multiple Past / Future Reference Pictures Store
mv
Multi-Block Multi-Frame Motion Estimator

# HEVC/VP9?


Input Video Signal
Split into CTUs
General Coder Control
General Control Data
Transform, Scaling & Quantization
Quantized Transform Coefficients
Scaling & Inverse Transform
Header Formatting & CABAC
Coded Bitstream
Intra Prediction Data
Intra-Picture Estimation
Filter Control Analysis
Filter Control Data
Intra-Picture Prediction
Deblocking & SAO Filters
Motion Data
Motion Compensation
Intra/Inter Selection
Output Video Signal
Motion Estimation
Decoded Picture Buffer

# Superblocks (VP9)
# Coding Units (HEVC)



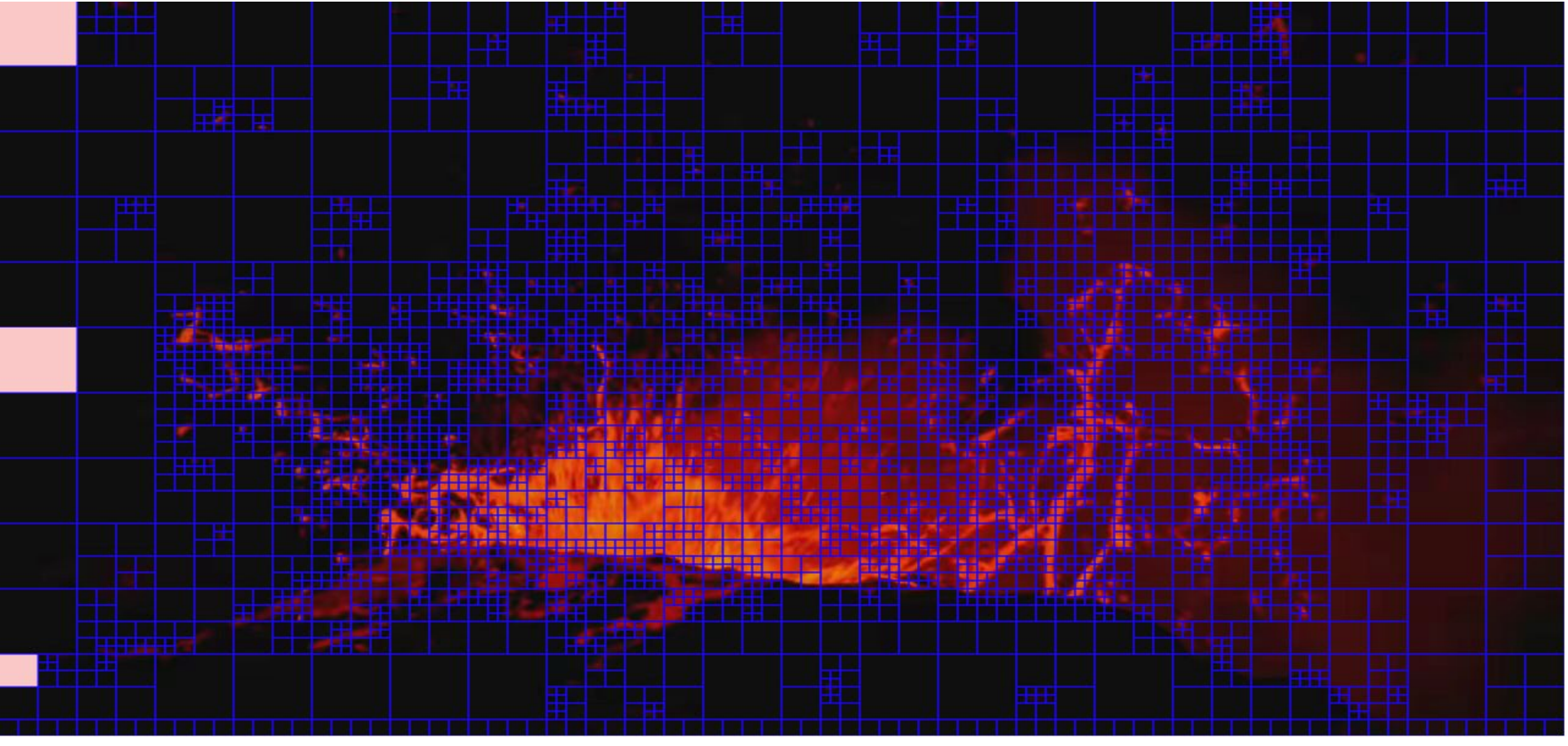




Max motion compensation unit is 64x64
Maxium Transform Unit is 32x32
Coding Transform Unit (CTU) in HEVC

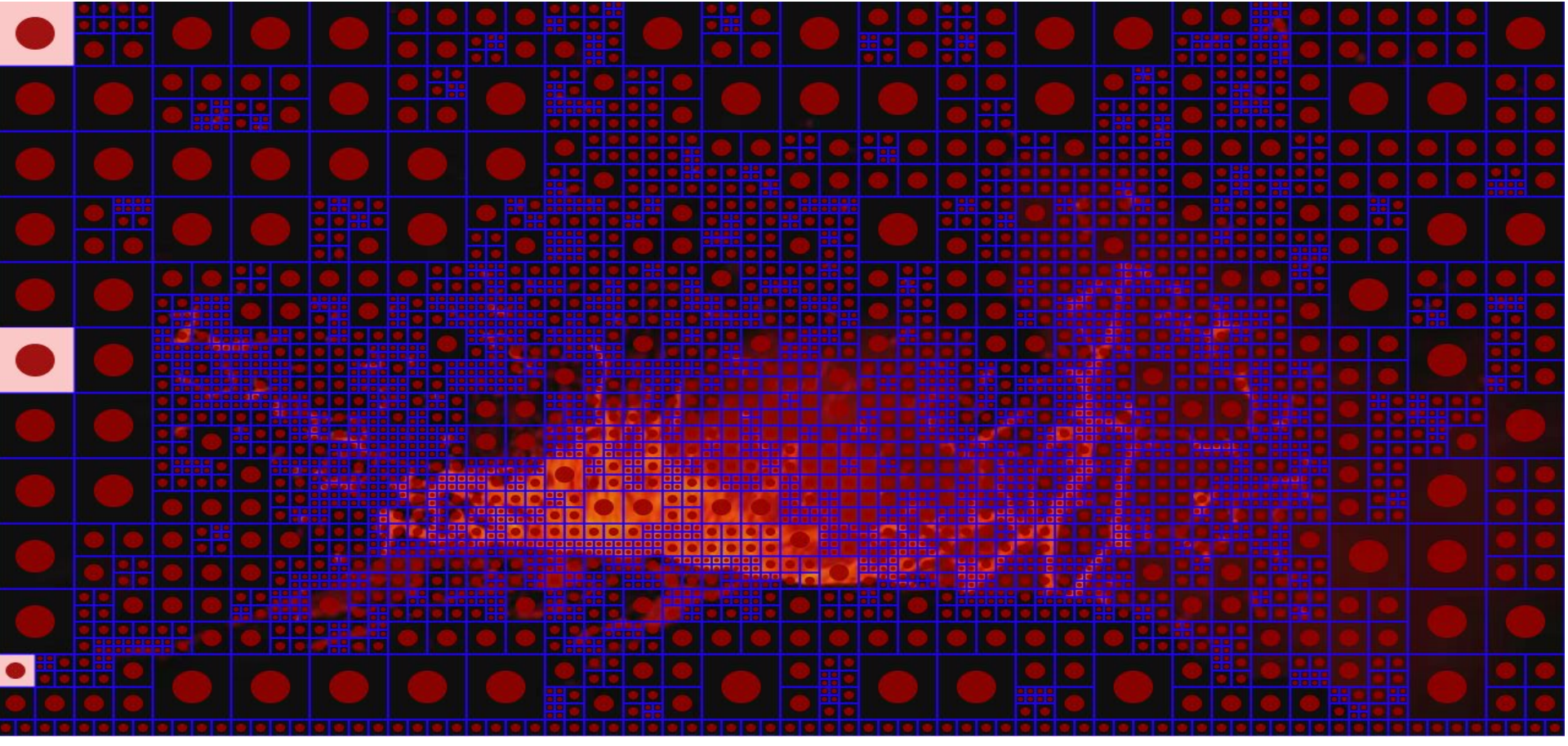
720p HEVC (H.265) – Intra Frame 0  103kbits, Intra Coding Units

IDR (Instantaneous Decoding Refresh) Picture

# 720p HEVC (H.265) – Intra Frame 0  Showing CU's



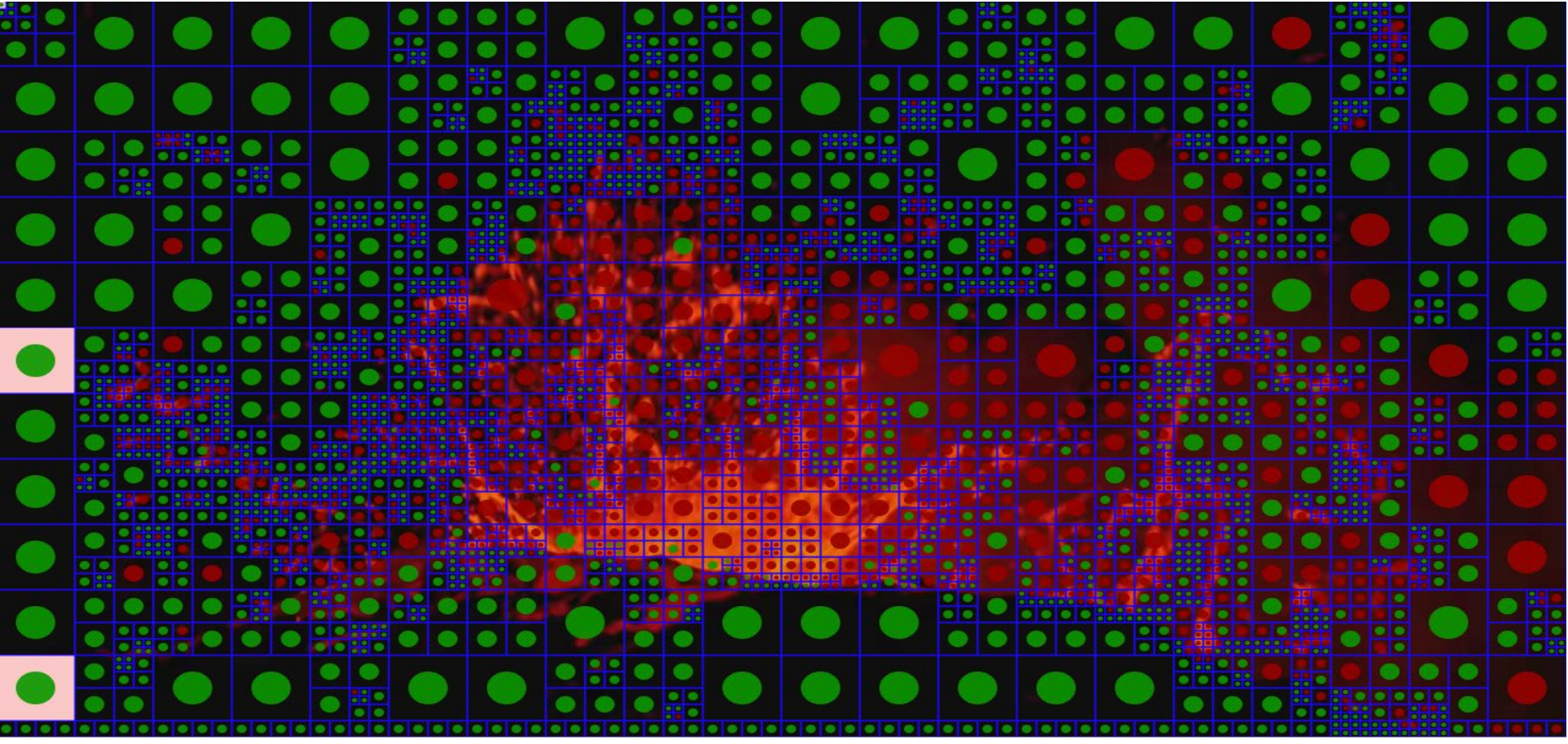
# 720p HEVC (H.265) – Intra Frame 0  Showing CU's Intra



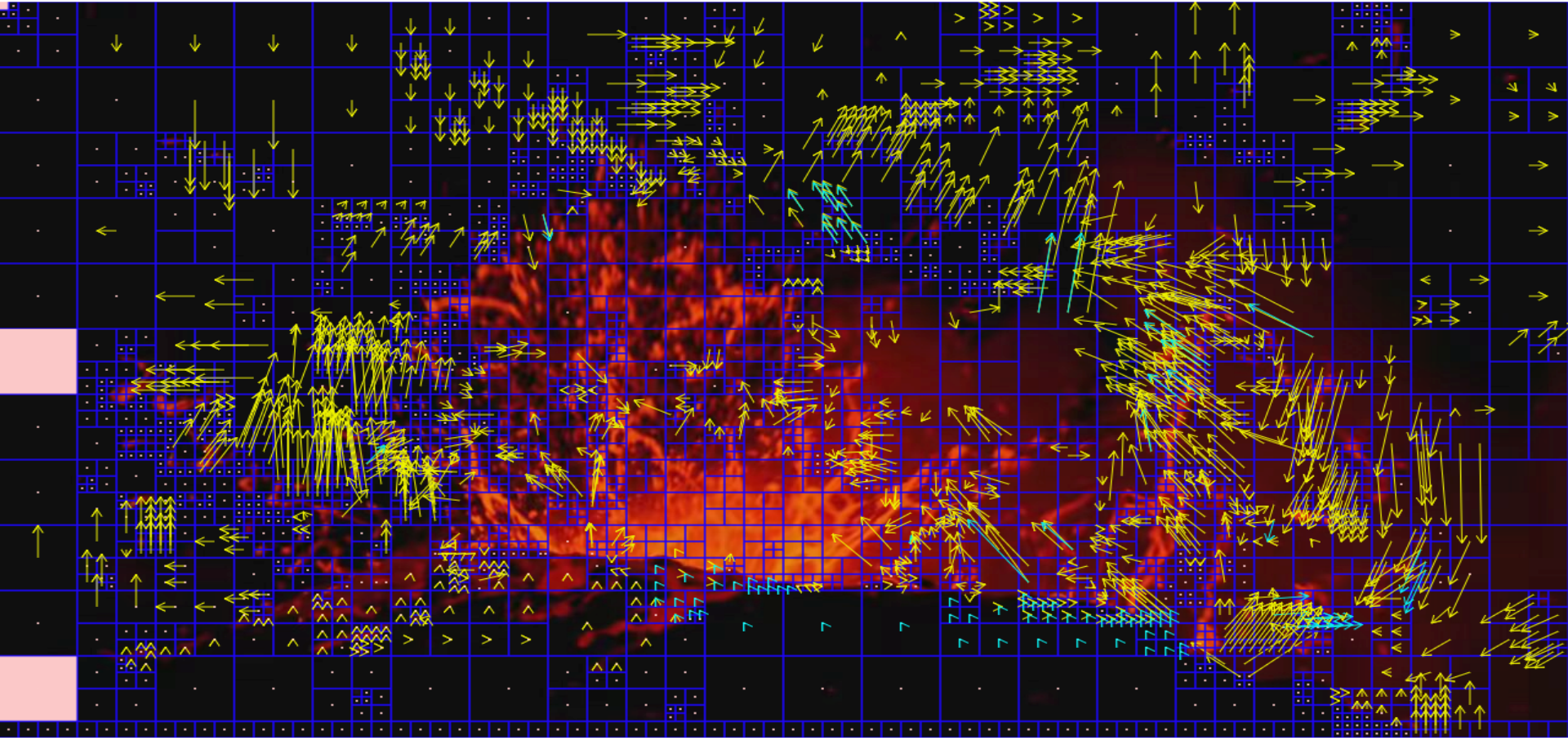
# 720p HEVC (H.265) – Frame 0 I-Frame

# 720p HEVC (H.265) – Frame 8 B-Frame 94.5 kbits

# 720p HEVC (H.265) – Frame 8 B-Frame, CUs and Type I/P



# 720p HEVC (H.265) – Frame 8 B-Frame Motion

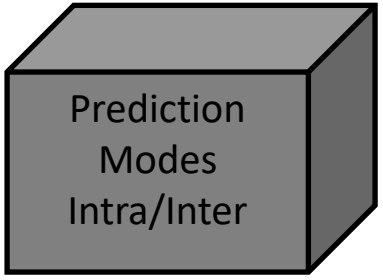


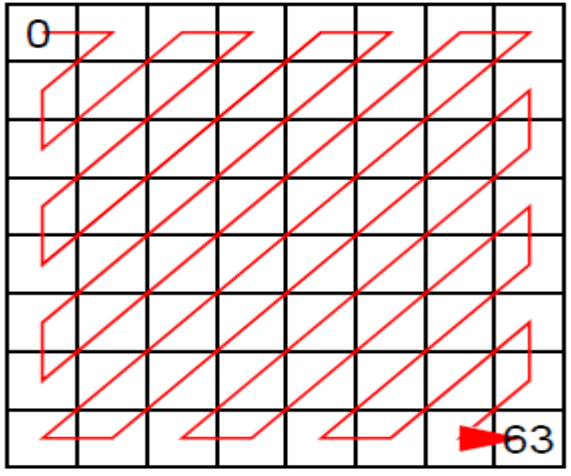
# 720p HEVC (H.265) – Frame 0 I-Frame

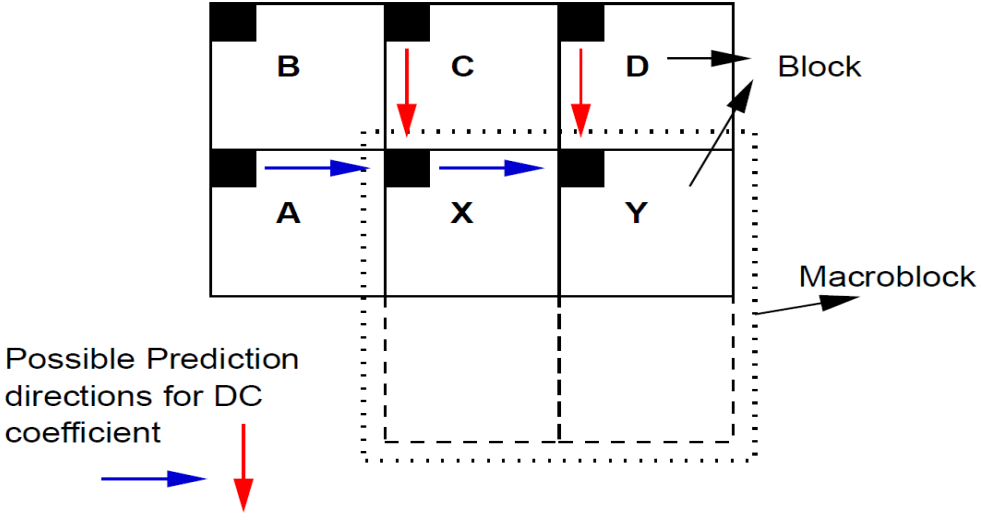
# 720p HEVC (H.265) – Frame 8 Predicted B-Frame

# 720p HEVC (H.265) – Frame 8 Final B-Frame

# Intra Prediction Modes



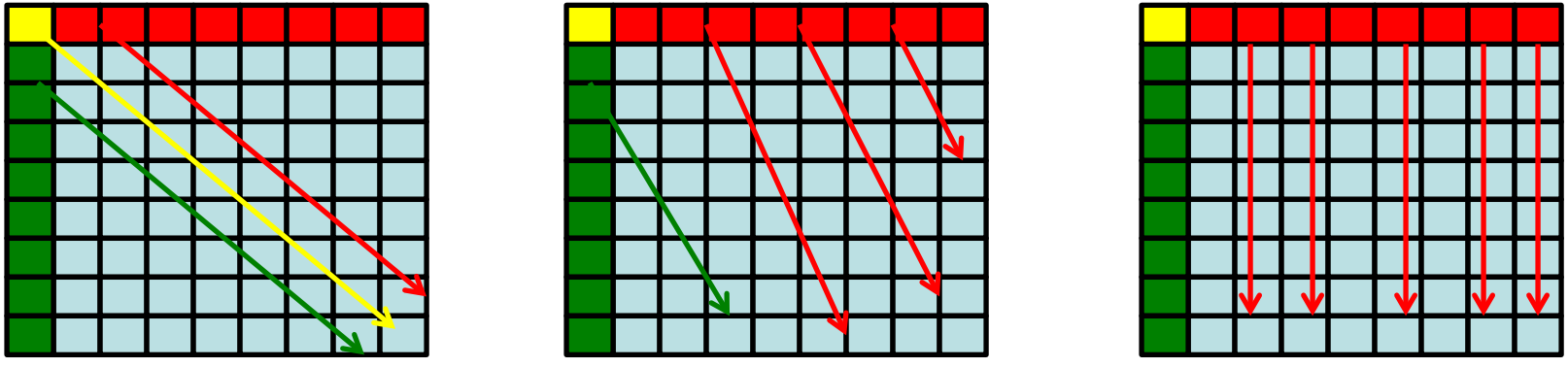




DC Coefficient encoded differentially like JPEG
MPEG2 only allows prediction from left block,
Rest of coeffs encoded using run,length huffman codes (JPEG, MPEG2,4)

MPEG2 allows prediction spatially as well as temporally
H.264 also CABAC or CAVLC for difference between prediction and the observed coefficients

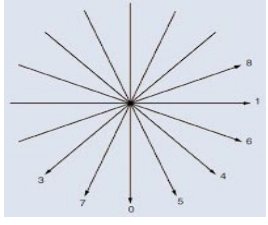
# H.264 generates intra-block predictions using pixelwise copies



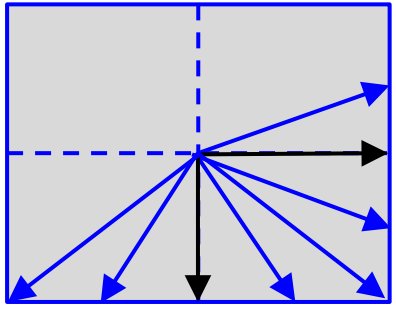
VP9 Uses Hybrid DCT/ADST for Intra Modes not DCT

Uses DCT for INTER Modes

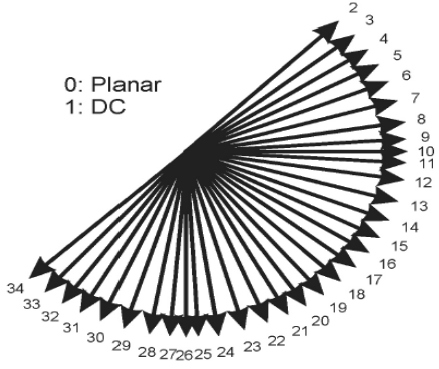
# VP9/HEVC pushes this further (accounting for 10% over H.264)



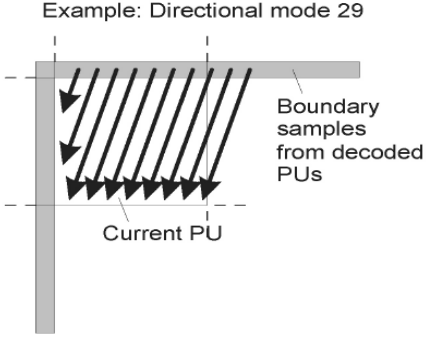
DC + 8 Directions
H.264



VP9 8 directions, along with DC_PRED and TM_PRED modes

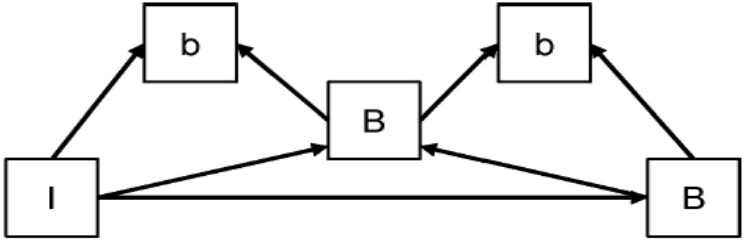



HEVC : DC + Planar + 33 Directions



# Inter Prediction

- VP9 allows coding a frame with a max of 3 reference frames, that are chosen per frame from a pool of 8
  - Every coded frame replaces one of the frame buffers as specified in frame header.
  - Certain frames can be designated *invisible* (Altref)



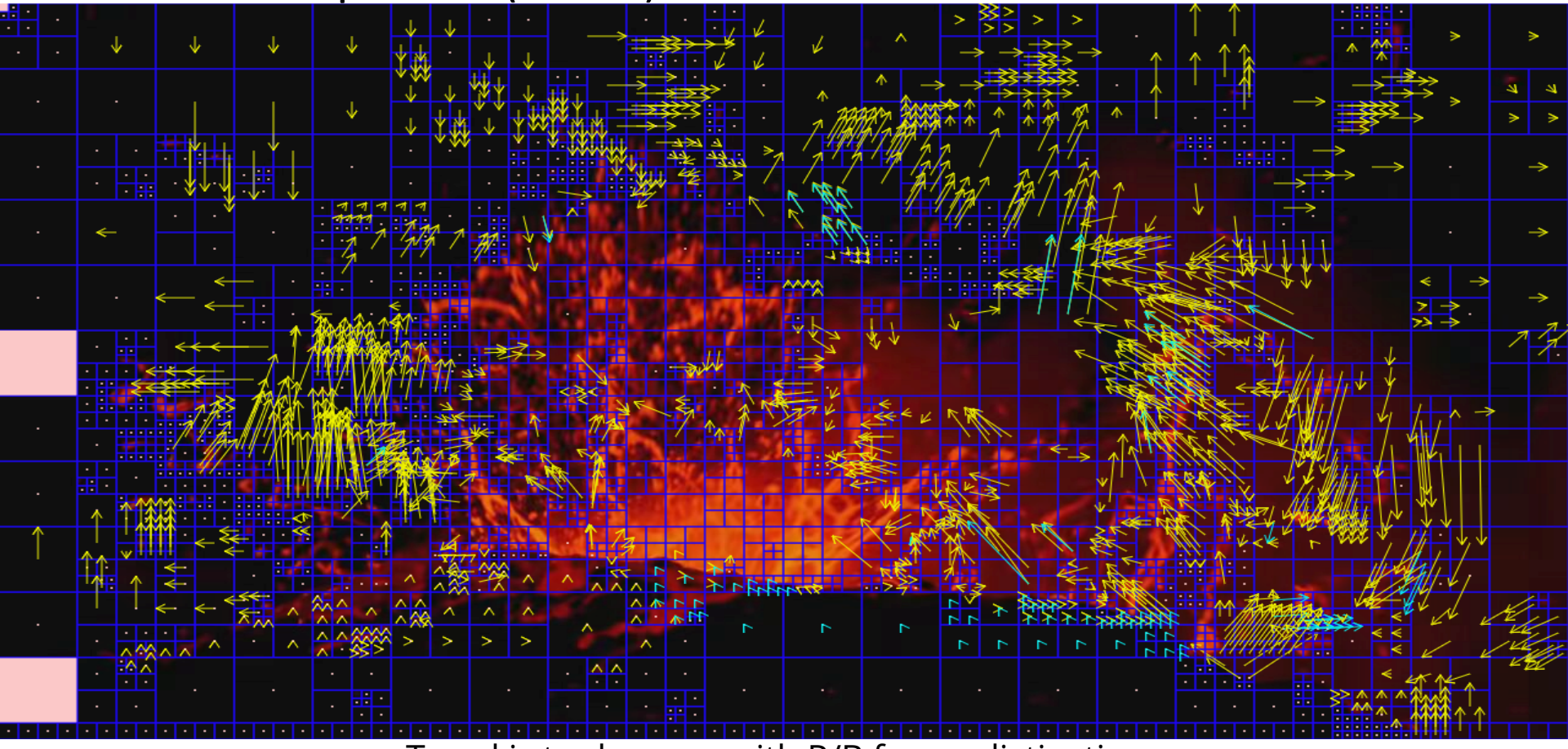

HEVC

| | | | | | |
|---|---|---|---|---|---|
| POC | 0 | 1 | 2 | 3 | 4 |
| Decoding order | 1 | 4 | 3 | 5 | 2 |
| RPS | – | [0,2] | [0,4] | [2,4] | [0] |

Picture Order Count

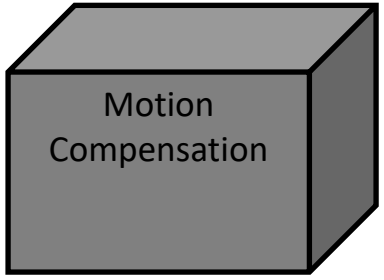
Reference Picture Set

# 720p HEVC (H.265) – Frame 8 B-Frame Motion



Trend is to do away with P/B frame distinction
allow prediction from one or more reference pictures on a CU basis

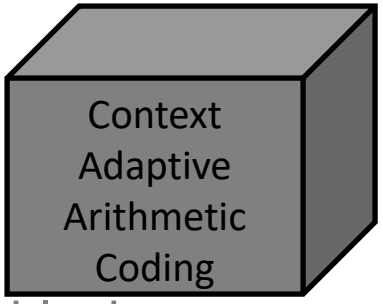
# VP9 and HEVC Motion Compensated Interpolation




- Fractional motion compensation critical for video coding performance
- VP9 supports ⅛ pel motion (1/16 in U and V)
- Frame level flag indicates whether ⅛ is to be used
- Companded motion - use ⅛ pel only for small motion as indicated by reference MV magnitude
- Use 3 different 8-tap filters + bilinear (regular/sharp/smooth)

- HEVC ¼ pel accuracy
- Use 8tap and 7tap filters to interpolate picture directly
- H.264 used two-step interpolation

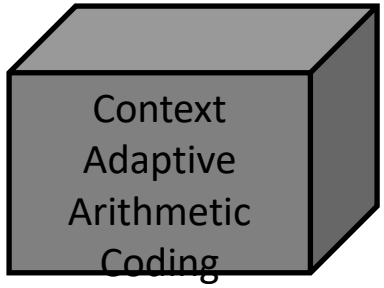
# Context Adaptive Binary Arithmetic Coding




- VP9 Updates per frame only
- HEVC updates per block
- Uses a temporal smoothing of the symbol p.d.f's combined with simple p.d.f. models
- No temporal smoothing

Gain of 2-5% over CAVLC

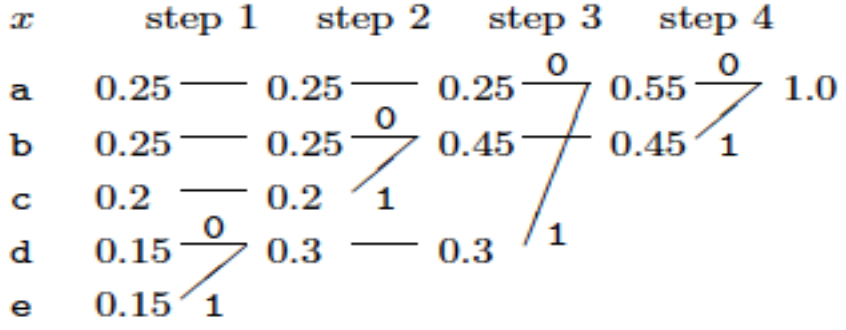
# Context Adaptive Binary Arithmetic Coding




- Given the data to be coded a module is needed to assign bits to each datum for transmission.
- Up till 1998 the common thing to was Huffman Coding.
- Since 1999 the world switched to Arithmetic Coding thanks to a series of papers by Radford Neal et al (since 1987)
- Arithmetic coding achieves the Shannon coding limit, is very efficient and separates the process of coding from the process of p.d.f. measurement.
- All modern media compression uses some form of Arithmetic Coding. H.264/HEVC/VP9 use “CABAC”

Gain of 2-5% over CAVLC

# Huffman Coding Example




| $a_i$ | $p_i$ | $h(p_i)$ | $l_i$ | $c(a_i)$ |
|---|---|---|---|---|
| a | 0.25 | 2.0 | 2 | 00 |
| b | 0.25 | 2.0 | 2 | 10 |
| c | 0.2 | 2.3 | 2 | 11 |
| d | 0.15 | 2.7 | 3 | 010 |
| e | 0.15 | 2.7 | 3 | 011 |

$$\{\,\mathtt{a},\quad \mathtt{b},\quad \mathtt{c},\quad \mathtt{d},\quad \mathtt{e}\,\}$$
$$\{\,0.25, 0.25, 0.2, 0.15, 0.15\,\}$$

- Incurs an extra 0 or 1 bit per symbol
- Does not adapt to changing symbol p.d.f
- Typical patch up is to group symbols into blocks and code those

# DC Coeff coding in JPEG, MPEG2/4

| DC Coef Difference | Size | Typical Huffman codes for Size | Additional Bits (in binary) |
|---|---|---|---|
| $0$ | $0$ | $00$ | $-$ |
| $-1, 1$ | $1$ | $010$ | $0, 1$ |
| $-3, -2, 2, 3$ | $2$ | $011$ | $00, 01, 10, 11$ |
| $-7, \cdots, -4, 4, \cdots, 7$ | $3$ | $100$ | $000, \cdots, 011, 100, \cdots, 111$ |
| $-15, \cdots, -8, 8, \cdots, 15$ | $4$ | $101$ | $0000, \cdots, 0111, 1000, \cdots, 1111$ |
| $\vdots$ | $\vdots$ | $\vdots$ | |
| $-1023, \cdots, -512, 512, \cdots, 1023$ | $10$ | $1111\,1110$ | $00\,0000\,0000, \cdots, 11\,1111\,1111$ |
| $-2047, \cdots, -1024, 1024, \cdots, 2047$ | $11$ | $1\,1111\,1110$ | $000\,0000\,0000, \cdots, 111\,1111\,1111$ |

# Run/length huffman coding for AC Coeffs

As an example, let us code the following $8 \times 8$ block:

$$\begin{matrix} -13 & -3 & 2 & 0 & 0 & 0 & 1 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Code Run of zeros together with the SIZE of the non-zero coefficient at the end.
Max length of run is 15.
ZRL for any run more than 15 zeros, so ZRL = 15 zeros followed by a 0

Converting this to (DC Size) or (Run,Size) and values for the Additional Bits gives:

| (4) | −13 | (0, 2) | −3 | (0, 3) | 6 | (2, 2) | 2 | (3, 1) | −1 | (ZRL) | (1, 1) | 1 | (EOB) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 101 | 0010 | 01 | 00 | 100 | 110 | 11111001 | 10 | 111010 | 0 | 11111111001 | 1100 | 1 | 1010 |

## **CABAC in H.264 :** Responsible for 9-14% gain over run/length with fixed tables (baseline H.264)

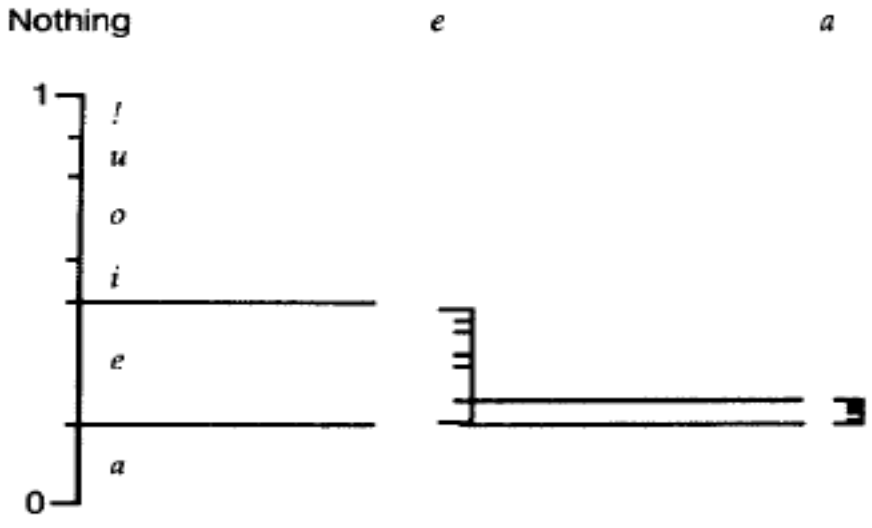
Context Adaptive Binary Arithmetic Coding / Variable Length Coding

Basic idea is to use different VLC tables that depend on a categorization of the nature of a block. This is done through some analysis of a significance map of a block and using the state of the blocks to the top and left of the current one.

*Context Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard*, Detlev Marpe, Heiko Schwarz, and Thomas Wiegand, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 13, NO. 7, JULY 2003

# Arithmetic Coding (A Fast Introduction)

| Symbol | Probability | Range |
|---|---|---|
| *a* | .2 | [0, 0.2) |
| *e* | .3 | [0.2, 0.5) |
| *i* | .1 | [0.5, 0.6) |
| *o* | .2 | [0.6, 0.8) |
| *u* | .1 | [0.8, 0.9) |
| *!* EndOfBlock | .1 | [0.9, 1.0) |

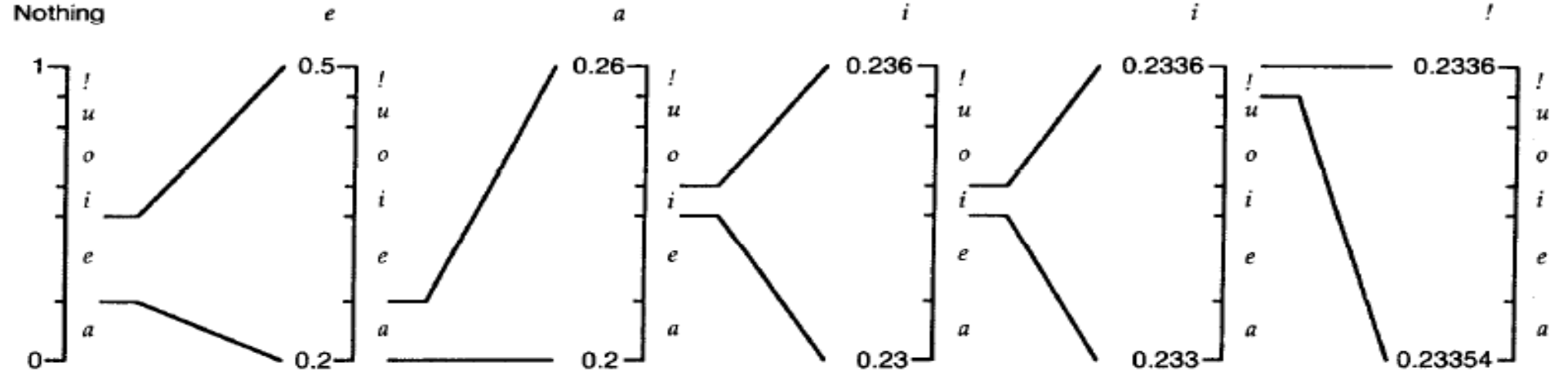



Encode e, a, i, i, !

# Arithmetic Coding (A Fast Introduction)

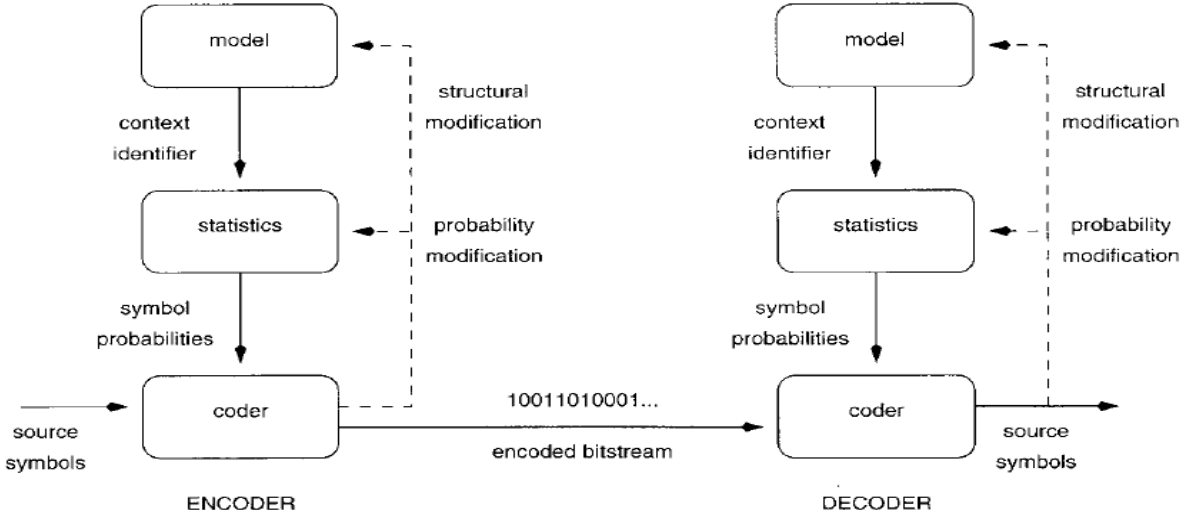
Arithmetic Coding for Data Compression, Witten, Radford, Neal, Cleary, *Communications of the ACM*, vol. 30, no. 6, pp. 520-540, June 1987.

Encode e, a, i, i, !




Decoder is a matter of recursive interval “spotting”
Gets arbitrarily close to the Shannon Limit for long streams of bits
Can change p.d.f. as you collect more “context”
More complicated than Huffman but recent work uses integer arithmetic instead

# Arithmetic Coding: Separating Coding from Modelling

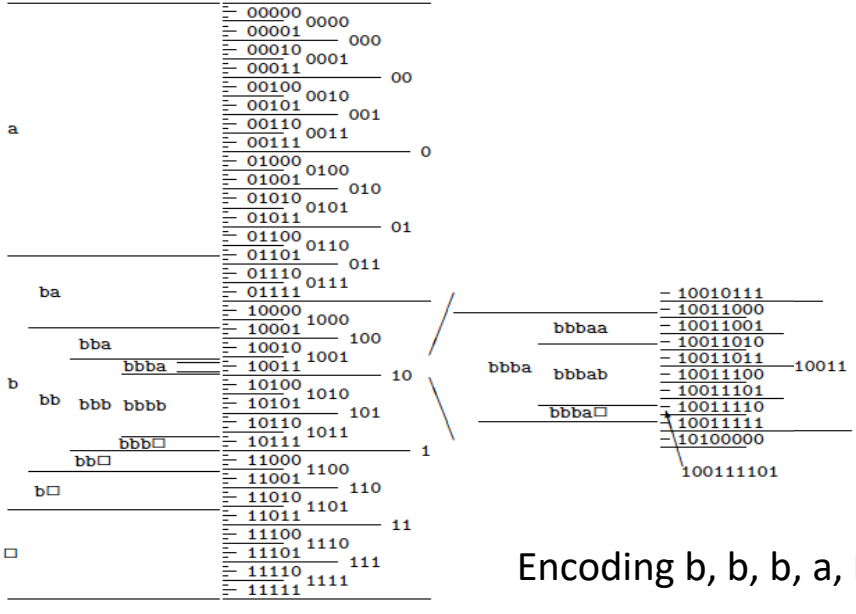



Arithmetic Coding Revisited, Moffat, Neal and Witten, *ACM Transactions on Information Systems*, Vol. 16, No. 3, July 1998, Pages 256–294.

This paper proposed practical schemes and led to CABAC used today

# Arithmetic Coding: Using integer representation

| Context (sequence thus far) | Probability of next symbol | | |
|---|---|---|---|
| | $P(\mathtt{a}) = 0.425$ | $P(\mathtt{b}) = 0.425$ | $P(\square) = 0.15$ |
| b | $P(\mathtt{a} \mid \mathtt{b}) = 0.28$ | $P(\mathtt{b} \mid \mathtt{b}) = 0.57$ | $P(\square \mid \mathtt{b}) = 0.15$ |
| bb | $P(\mathtt{a} \mid \mathtt{bb}) = 0.21$ | $P(\mathtt{b} \mid \mathtt{bb}) = 0.64$ | $P(\square \mid \mathtt{bb}) = 0.15$ |
| bbb | $P(\mathtt{a} \mid \mathtt{bbb}) = 0.17$ | $P(\mathtt{b} \mid \mathtt{bbb}) = 0.68$ | $P(\square \mid \mathtt{bbb}) = 0.15$ |
| bbba | $P(\mathtt{a} \mid \mathtt{bbba}) = 0.28$ | $P(\mathtt{b} \mid \mathtt{bbba}) = 0.57$ | $P(\square \mid \mathtt{bbba}) = 0.15$ |

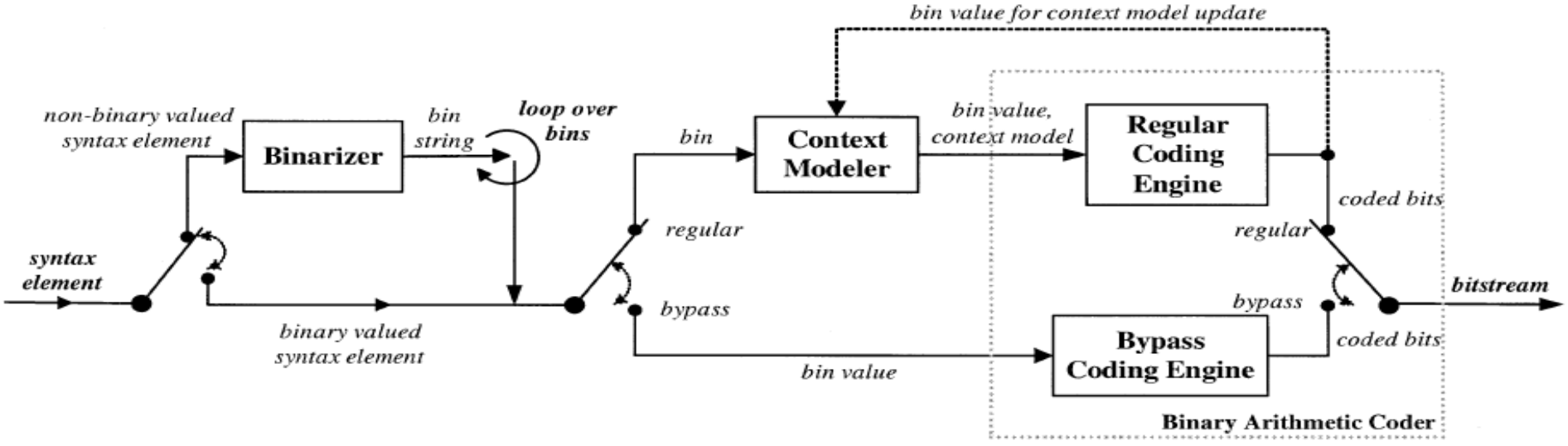



Use binary intervals
Ship bits when stable

See pages 112-114 of
*Information Theory* by MacKay

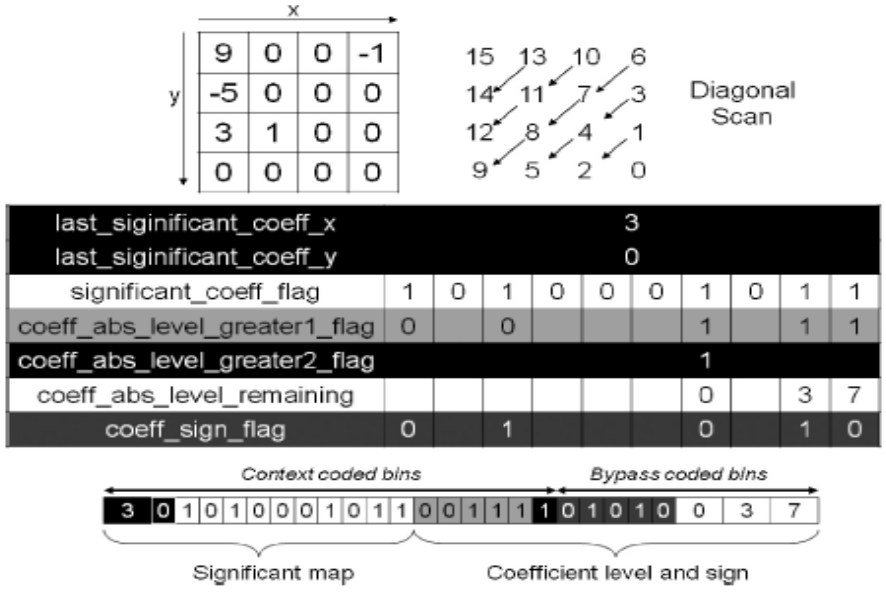
Encoding b, b, b, a, EoB

# CABAC in H.264 and HEVC




Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard, Detlev Marpe, Heiko Schwarz, and Thomas Wiegand, *IEEE Trans Vid Ccts and Sys*, Vol 13, No 7, July 2003, pp 620-636

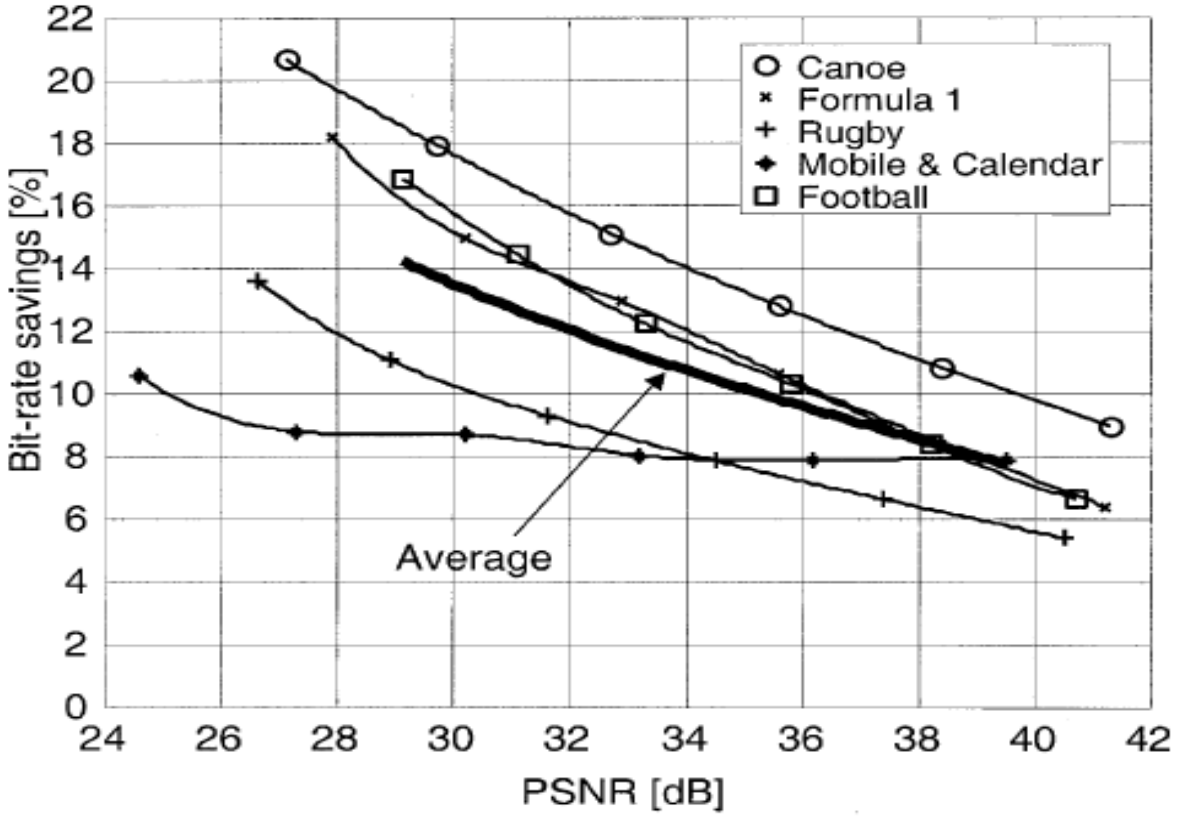
# CABAC for DCT in H.264 and HEVC



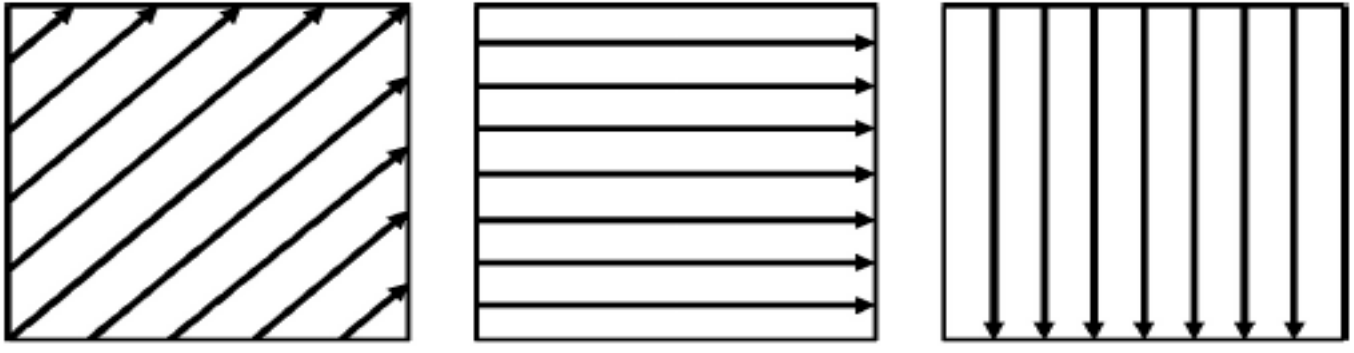

Key Challenges: High throughput, Low complexity and Low Memory (esp for hardware)

High Throughput CABAC Entropy Coding in HEVC, Vivienne Sze and Madhukar Budagavi, *IEEE Trans Vid Ccts and Sys*, Vol 22, No 12, Dec 2012, pp 1778-1791

# CABAC in H.264 Vs CAVLC H.264 Baseline



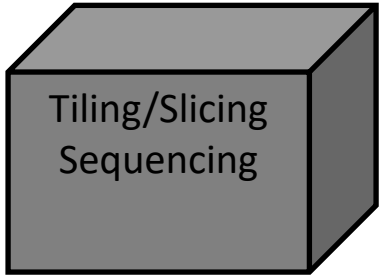
# 3 Scan Patterns in HEVC
# HEVC and VP9 _not_ defined for Interlaced video

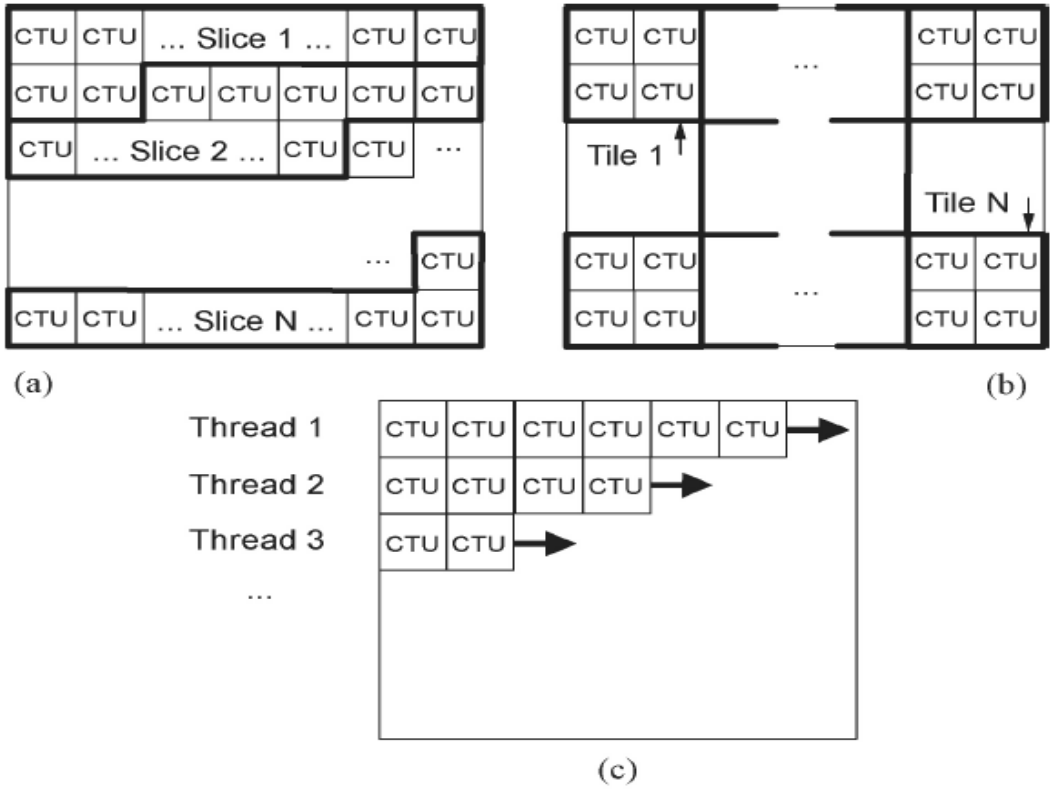


## VP9 CABAC

- Very similar to CABAC in HEVC in spirit
- Different models (some parametric)
- Different binary symbols (different coding tree)
- Update models only every frame instead of every symbol
- 2-3% Improvement over CAVLC (Huffman++)
- Compromises made for fast operation in hardware

# Sequencing (HEVC)



# A note on synchronisation

The codes are all Variable Length Codes
This is why synchronisation is a problem.

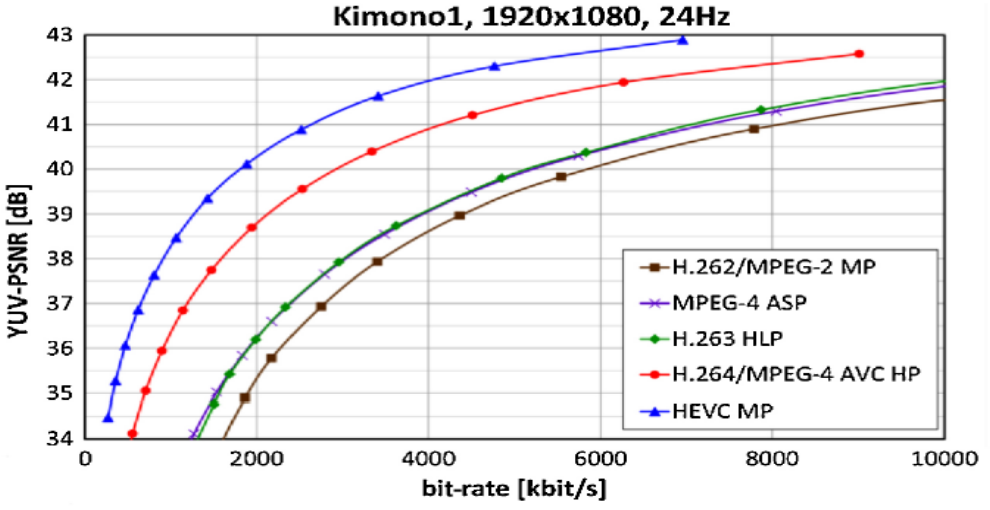
So if you forget where you are in the bitstream somehow, you will start to confuse one code with another.

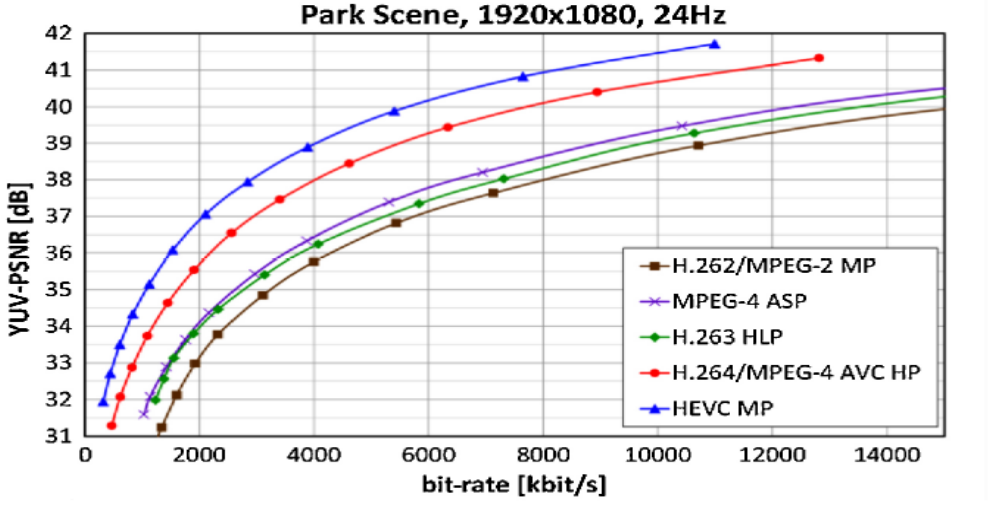
If you detect a problem, you can skip to the next unique code you can recognise. This can be a special code or just a clever decoder spotting the next macroblock header,

MPEG4 uses the Resynchronisation marker : 0000 0000 0000 0000 1

MPEG2 uses the slice header. H.264 uses video packet header.
VP9 and HEVC use SLICES for resync

# HEVC Performance



# More Reading

Overview of the High Efficiency Video Coding (HEVC) Standard, Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han and Thomas Wiegand, *IEEE Trans Vid CCts and Sys*, Vol 22, No 12, Dec 2012, pp 1649-1668

Information Theory, Inference and Learning Algorithms, David MacKay, CuP 2003, Online vesion at http://www.inference.phy.cam.ac.uk/mackay/itila/

Excellent video lectures by Prof. MacKay http://www.inference.phy.cam.ac.uk/itprnn_lectures/ [Lecture 5 of most relevance to CABAC]

Google's WEBM project http://www.webmproject.org/

VP9 : http://www.webmproject.org/vp9/

# VP9 Vs H.264

https://x20web.corp.google.com/~jimbankoski/vp9/same_bitrate.html

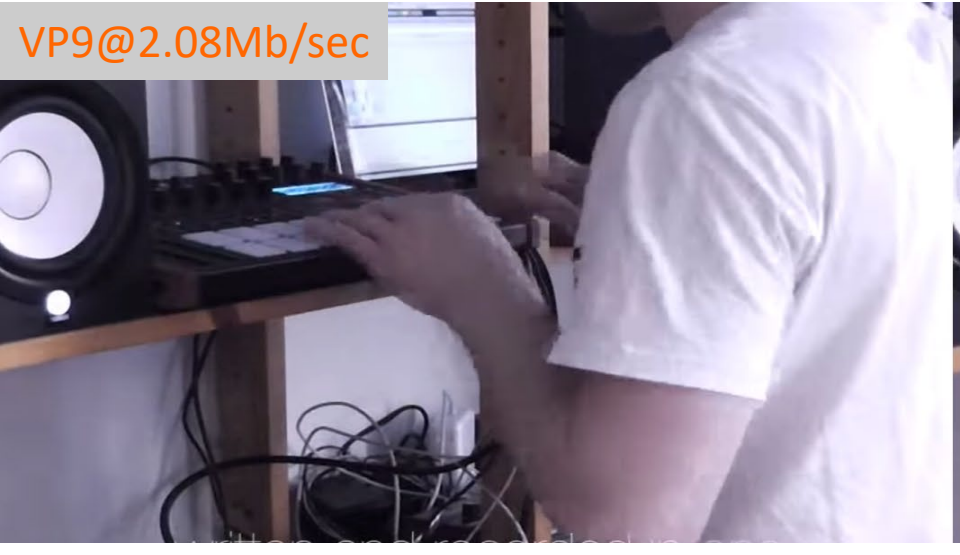
- Have to use a browser to play this since Chrome is the only VP9 player easily usable in a presentation like this

720p

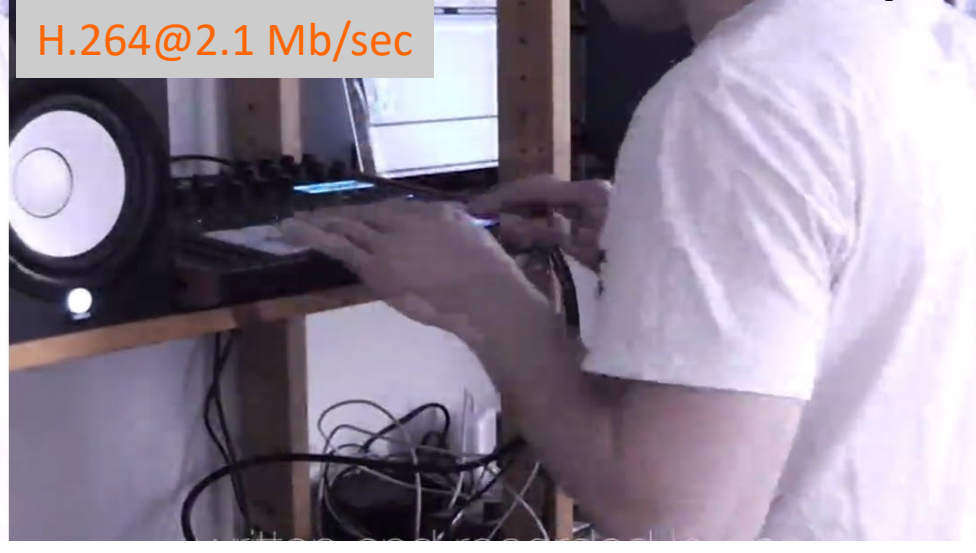
# VP9 Vs H.264

720p

VP9@2.08Mb/sec



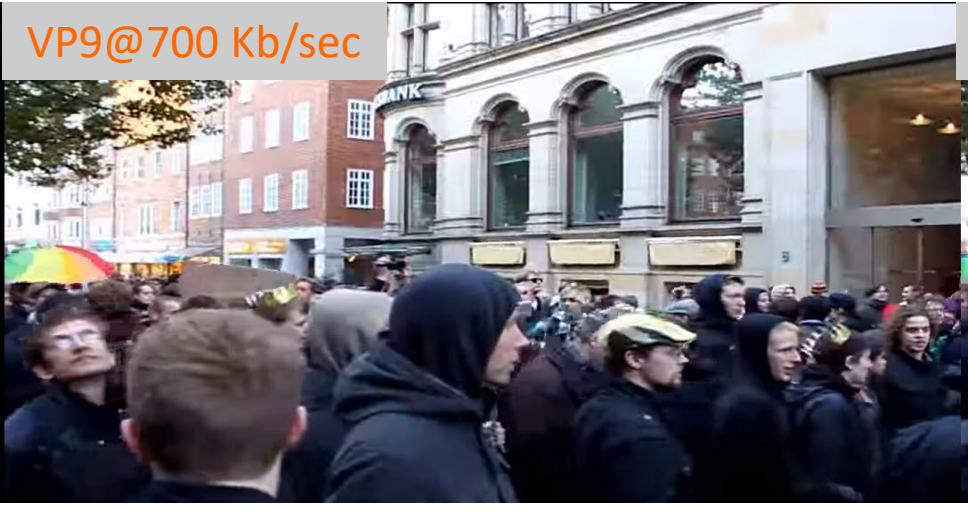
H.264@2.1 Mb/sec



# VP9 Vs H.264

720p

VP9@700 Kb/sec



H.264@726 Kb/sec

# Final Comments

- Video is breaking the internet
- Need to keep improving compression, keep increasing bandwidth, keep reducing cost
- HEVC and VP9 are the new wave of compression algorithms
- Based on new tools : larger blocks, quadtree, arithmetic coding
  VP9 already supported in Chrome. HEVC at first had better hardware traction but VP9 has caught up.
- NOTE: Standards do NOT specify coding methods only the BITSTREAM SYNTAX. Standards docs are pretty tricksy to read (try it), you are trying to describe program code in words. That is why Google's Chrome Media doesn't write alot of them.
- Next : Trading off bandwidth/compression/cost