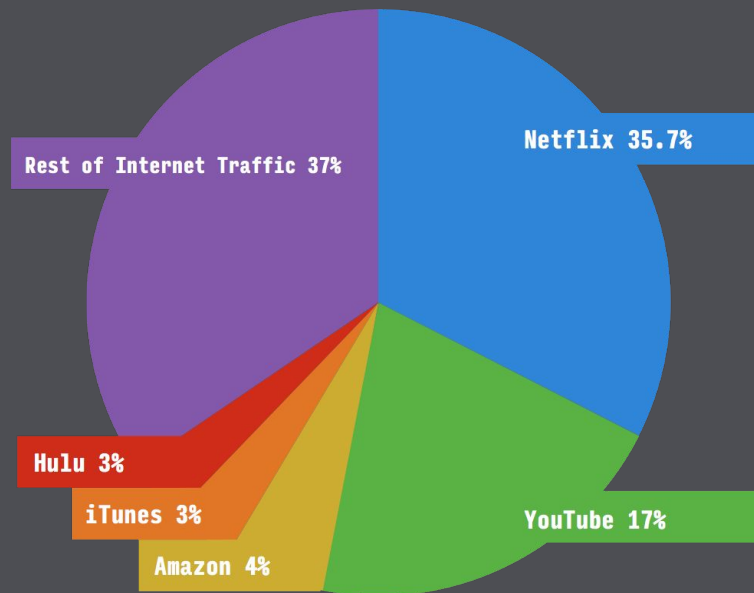


AV1 Image File Format (AVIF)

Nathan Egge <negge@mozilla.com>
81st JPEG Meeting - Vancouver, BC
October 14, 2018

Slides: <https://xiph.org/~negge/AVIF2018.pdf>

North America Internet Traffic



82% of Internet traffic by 2021 Cisco Study

Alliance for Open Media (AOM)

Goals of the Alliance:

- Produce a video codec for a broad set of industry use cases
 - Video on Demand / Streaming
 - Video Conferencing
 - Screen sharing
 - Video game streaming
 - Broadcast
- Open Source and Royalty Free
- Widely supported and adopted
- At least 30% better than current generation video codecs

AOM Members



AOM Members / Browsers



AV1 Coding Tools Overview

- New high-level syntax
 - Easily parsed sequence header, frame header, tile header, etc
- New adaptive multi-symbol entropy coding
 - Up to 16 possible values per symbol
- New coefficient coder
 - LV-MAP exploits multi-symbol arithmetic coder
- More block sizes
 - Prediction blocks from 128x128 down to 4x4
 - Rectangular blocks
 - 1:2 and 2:1 ratios (4x8, 8x4, etc)
 - 1:4 and 4:1 ratios (4x16, 16x4, etc)
 - Transform sizes from 64x64 down to 4x4
 - Includes rectangular transforms 1:2, 2:1 and 1:4, 4:1 ratios
- More transform types
 - 16 possible transform types
 - Row and column chosen from: IDTX, DCT, DST, ADST
- More references
 - Up to 7 per frame (out of a store of 8)
- Spatial and temporal scalability
- Lossless mode
- Chroma subsampling
 - 4:4:4, 4:2:2, 4:2:0, monochrome
- More prediction modes
 - Intra
 - 8 main directions plus delta for up to 56 directions
 - Smooth HV modes interpolate across block
 - Palette mode with index map up to 8 colors
 - Chroma from Luma intra predictor
 - Intra Block Copy
 - Inter
 - Expanded reference list (up to 7 per frame)
 - Allow ZEROMV predictor, which isn't always (0,0)
 - Compound mode
 - Inter-Intra prediction
 - Depends on difference between pixel prediction
 - Smooth blending limited to certain intra modes
 - Wedge codebook (Inter-Inter, or Inter-Intra)
 - Warped motion local affine model with neighbors
 - Global motion affine model across entire frame
- Loop filtering
 - Deblocking filter
 - Constrained Directional Enhancement Filter
 - Loop restoration
- Film grain synthesis

Profiles

Main

- 8-bit and 10-bit
- 4:0:0 and 4:2:0 chroma subsampling

High

- 8-bit and 10-bit
- 4:0:0, 4:2:0 and 4:4:4 chroma subsampling

Professional

- 8-bit, 10-bit and 12-bit
- 4:0:0, 4:2:0, 4:2:2 and 4:4:4 chroma subsampling

Levels

For a given sequence, place limits on:

- frame size (width and height)
- maximum picture size (area in samples)
- maximum display rate (samples per second)
- maximum decode rate (samples per second)
- average rate (Mbits per second)
- high rate (Mbits per second)
- maximum number of tiles
- maximum number of tile columns

High Level Syntax

Sequence Header

Frame Header

Tile Group

Tile

Tile

Tile Group

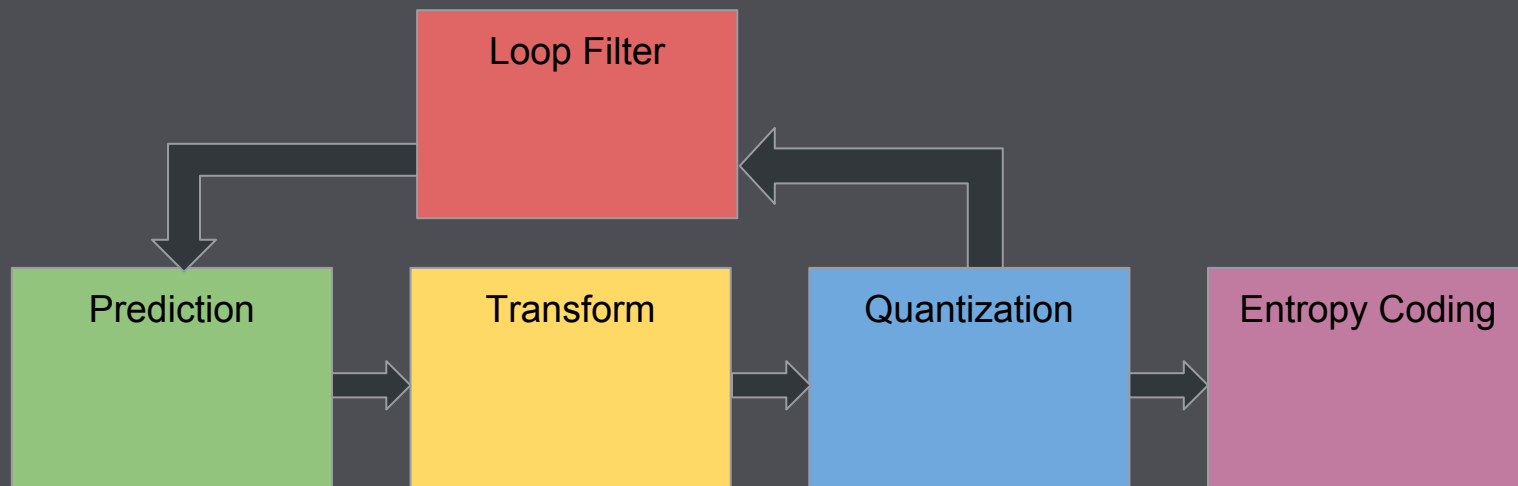
Tile

Tile

Colors and HDR

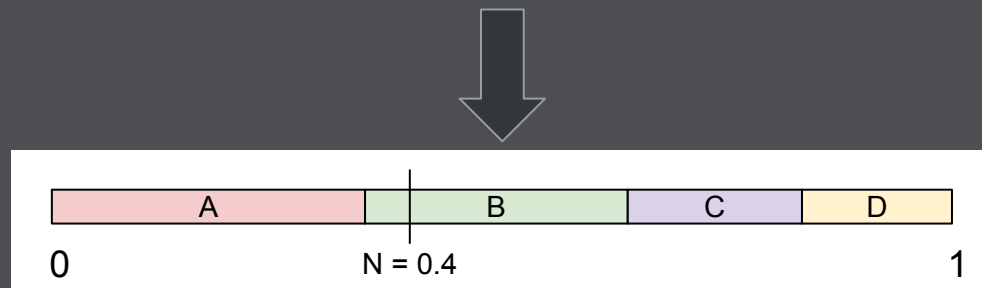
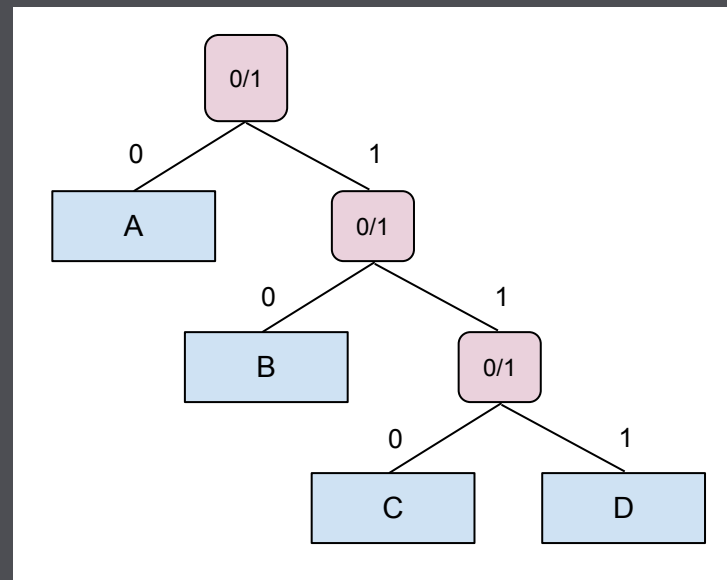
- Colorspace, color matrix, transfer functions, etc. can be encoded directly in the bitstream
 - Chroma siting and levels too
- HDR metadata can be added through the Metadata OBU syntax

Codecs 101



Multi-Symbol Entropy Coder

- Arithmetic Range Coder
- Code both binary symbols and multi-symbols
 - Alphabet sizes up to 16
- Improve EC throughput with high rate streams
 - Instead of 1 bit per cycle, decode up to 4
- Use 8x9 -> 17 bit multiples when coding
 - 15-bit CDFs shifted down before multiply
 - Adaptation still occurs with 15-bit precision
- Fast adaptation mode for first few symbols



Transform Types

VP9 has two types: DCT and ADST

- Chosen independently for horizontal / vertical directions
- Signaled once per prediction block

AV1 has four types:

- DCT
- ADST
- FlipADST (mirror image of ADST)
- Identity (no transform)

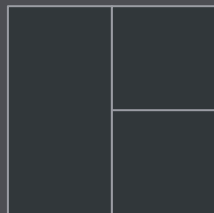
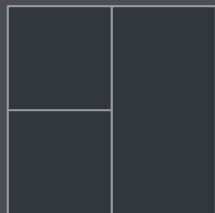
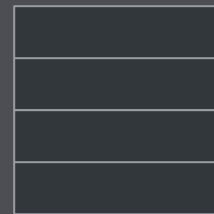
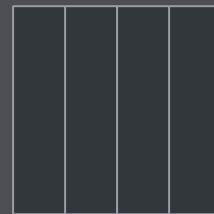
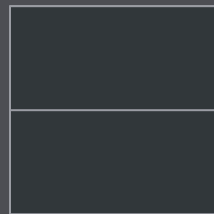
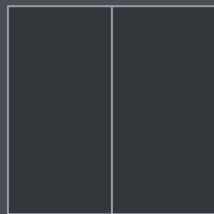
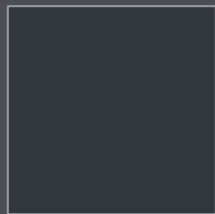
Still chosen independently for horizontal / vertical directions

- Total of 16 possible combinations
- Not all combinations allowed in all contexts (e.g., no FlipADST for intra)

Signaled once per transform block

Prediction Block Structure

10 different splitting modes

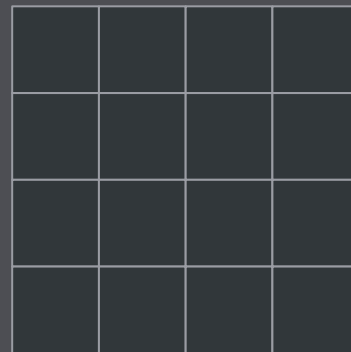


- Last (4-way) split is recursive

Transform Block Sizes: Intra

Signaling mostly unchanged from VP9

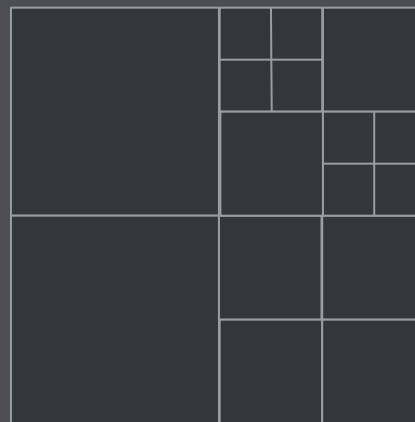
- One transform size per prediction block
- For rectangular prediction blocks, largest rectangular transform that fits allowed, e.g., 1:2, 2:1, 1:4 and 4:1 ratio transform blocks
- Transform sizes go up to 64x64
 - Only upper left 32x32 region allowed to be non-zero



Transform Block Sizes: Inter

Signaling completely different from VP9

- Four way quad tree splitting
- For rectangular prediction blocks, largest rectangular transform that fits also allowed
- Available sizes same as intra

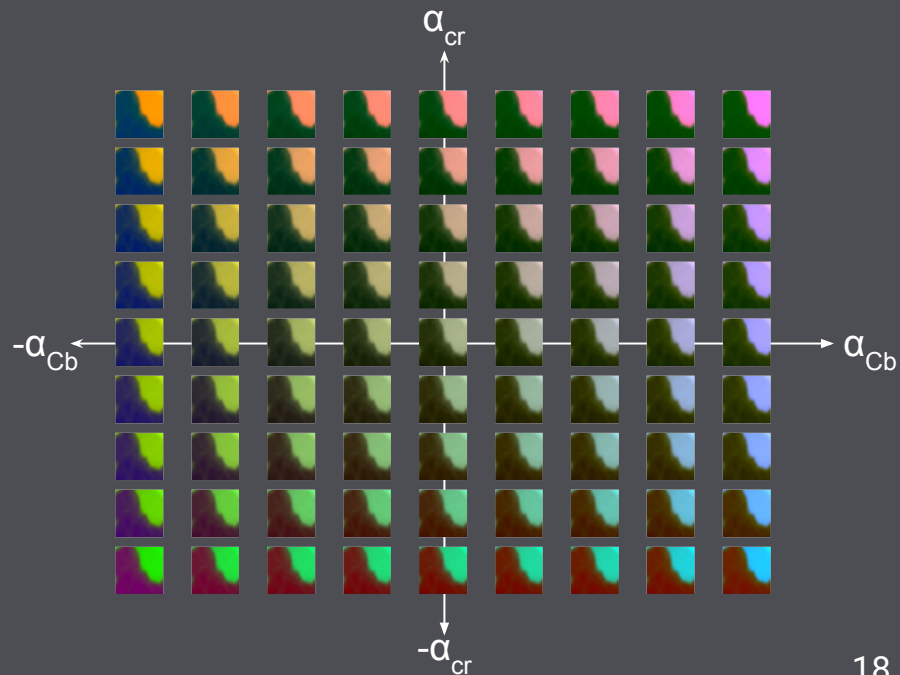
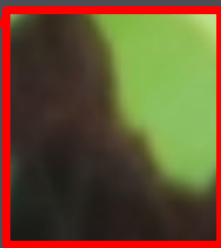


Intra Prediction Modes

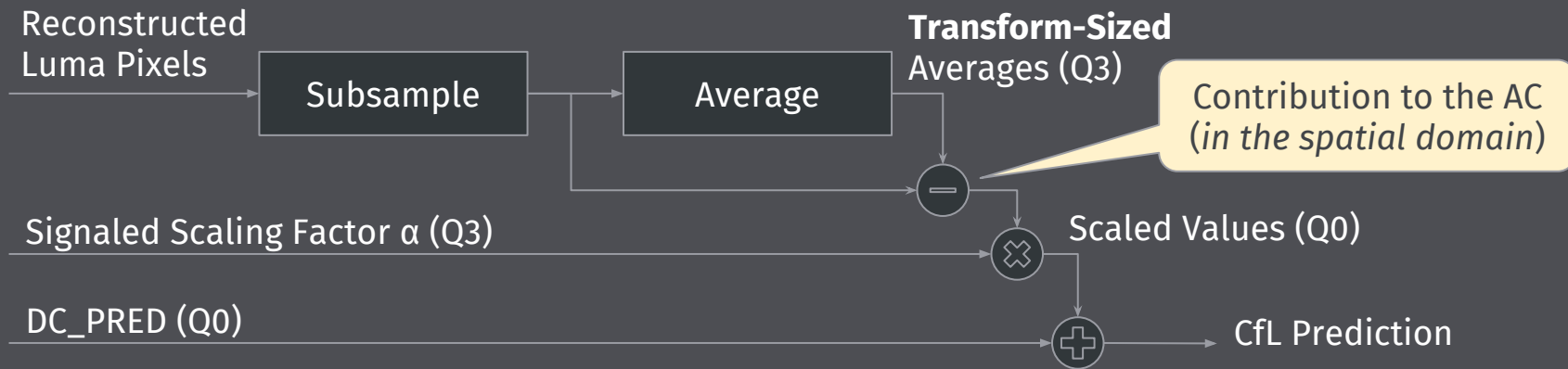
- More directional modes
 - 8 main directions plus delta for up to 56 directions
 - Not all modes available at smaller sizes
- Smooth H + V modes
 - Smoothly interpolate between values in left column (resp. above row) and last value in above row (resp. left column)
- Intra filter modes
- Paeth predictor mode
- Palette mode
 - Color index map with up to 8 colors
 - Separate palettes for Y, U and V planes
 - Palette index coded using context model for each pixel in the block
 - Pixels predicted in 'wavefront' order to allow parallel computation
- Intra Block Copy
- Chroma from Luma

Chroma from Luma Intra Prediction

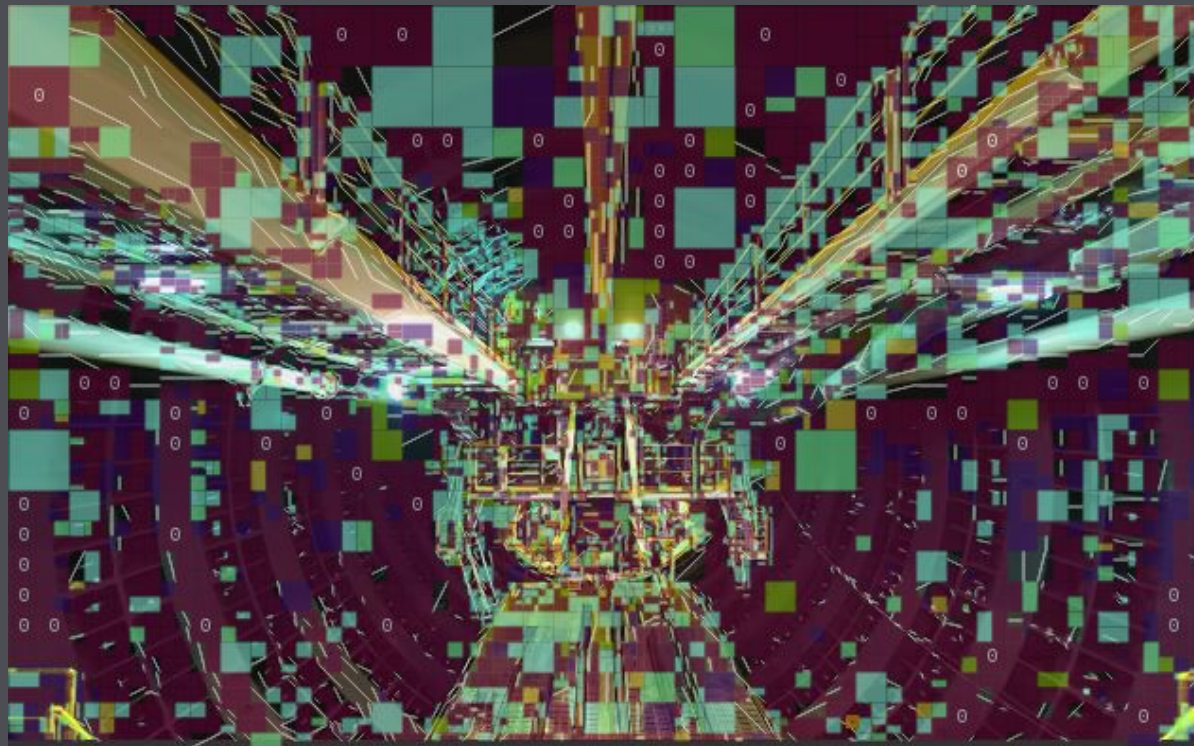
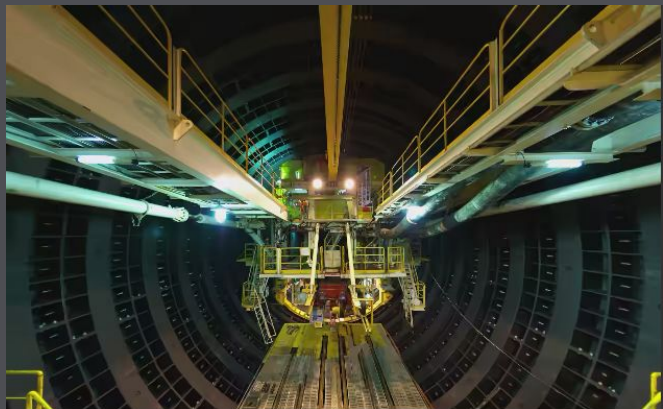
- Predict chroma channel based on decoded luma
 - Encoder signals best correlation constants: α_{cb} and α_{cr}
- Good for screen content or scenes with fast motion



Chroma from Luma Algorithm



UV Mode Selection Example (<https://goo.gl/6tKaB8>)



Ohashi0806shield.y4m
QP = 55

Awesome for Gaming (Twitch dataset)

	BD-Rate (%)						
	PSNR	PSNR-HVS	SSIM	CIEDE2000 ¹	PSNR Cb	PSNR Cr	MS SSIM
Average	-1.01	-0.93	-0.90	-5.74	-15.55	-9.88	-0.81

<https://arewecompressedyet.com/?job=no-cfl-twitch-cpu2-60frames%402017-09-18T15%3A39%3A17.543Z&job=cfl-inter-twitch-cpu2-60frames%402017-09-18T15%3A40%3A24.181Z>

Notable Mentions

	BD-Rate (%)						
	PSNR	PSNR-HVS	SSIM	CIEDE2000 ¹	PSNR Cb	PSNR Cr	MS SSIM
Minecraft	-3.76	-3.13	-3.68	-20.69	-31.44	-25.54	-3.28
GTA V	-1.11	-1.11	-1.01	-5.88	-15.39	-5.57	-1.04
Starcraft	-1.41	-1.43	-1.38	-4.15	-6.18	-6.21	-1.43



Minecraft

MINECRAFT_10_120f.y4m



GTA V

GTAV_0_120f.y4m



Starcraft

STARCRAFT_10_120f.y4m

Motion Vector Coding

- Each frame has a list of 7 previous frames to reference (out of a pool of 8)
 - Can reference non-displayed frames, so many possible structures
- Construct list of top 4 MVs for a given reference / reference pair from neighboring area
- Complicated entropy coding scheme

Compound Prediction

($\frac{1}{2}$, $\frac{1}{2}$) weights like VP9

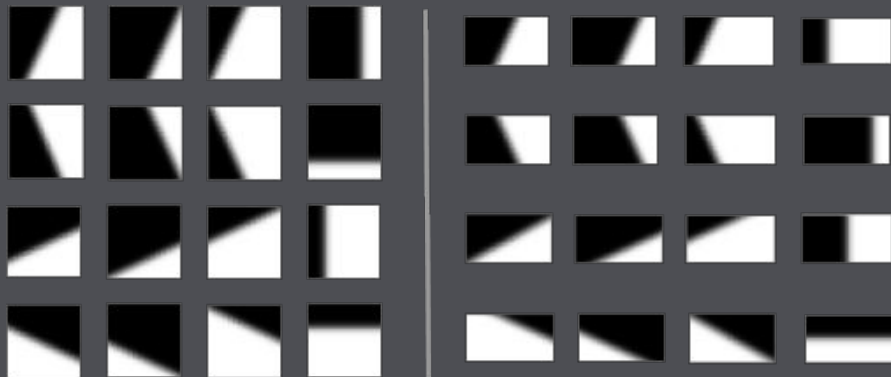
Inter-inter compound segment

- Pixel weights depend on difference between prediction pixels

Inter-intra gradual weighting

- Smoothly blends from inter to intra prediction
- Only a limited set of intra modes allowed (DC, H, V, Smooth)

Wedge codebook (inter-inter or inter-intra)



Square Codebook

Rectangular Codebook

Global Motion

- Defines up to a 6-parameter affine model for the whole frame (translation, rotation and scaling)
- Blocks can signal to either use the global motion vector or code a motion vector like normal
 - If global motion isn't used, default is 0,0

Warped Motion

- Use neighboring blocks to define same motion model within a block
 - Decomposed into two shears with limited range
 - Similar complexity to subpel interpolation

Segmentation IDs

- Up to 8 possible segment labels (3 bits)
 - Value set per label, e.g., filter strength, quantizer, reference frame, skip
 - Signaled per prediction block, down to 8x8
- Can either predict segment ID temporally or spatially (chosen per frame)
 - Spatial prediction
 - Useful to change quantizer/loop filter strength
 - Useful for adaptive quantization, e.g., for activity masking
 - Temporal prediction
 - Useful for predicting temporal properties, e.g., skip
 - Useful for temporal RDO, e.g., MB-tree

Activity Masking

- Exploit spatial prediction of segment IDs
- Use higher quantizer in regions with more activity
 - 8 possible segment IDs, each can have a different quantizer
- Can do in one-pass or two-pass over the frame (like rate control)

Temporal RDO (e.g., MB-tree)

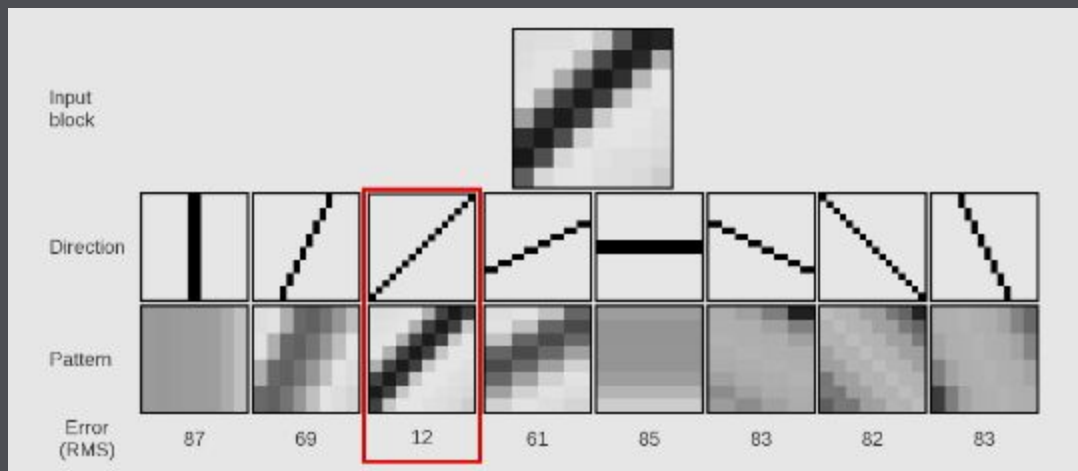
- Exploit temporal prediction of segment IDs
- Run look-ahead and encode down-sampled version of input
 - Trace motion vectors to find parts of the video that are highly coherent
 - Cap rate for a GOP structure and allocate bits to improve overall quality
- Can use choices to inform search at full frame size
 - limit block size, transform type, motion search, compound modes, etc.

Deblocking Filter

- Similar to what is in VP9
- Changed the order edges are filtered to make hardware easier
- More flexible strength signaling
 - Separate H + V strength for luma
 - Separate C_b and C_r strengths for chroma
 - Can be adjusted on a per-super block basis
- NB: deblocking filter crosses tile boundaries

Constrained Directional Enhancement Filter (CDEF)

- Merge of Daala's directional deringing filter (DERING) and Thor's constrained lowpass filter (CLPF)
 - Both encoder and decoder search for the direction that best matches
 - Primary filter run along direction, and secondary conditional replacement filter run orthogonally
 - Strength is signaled in the bitstream
- Results exceed both DERING and CLPF alone, as well as applying DERING + CLPF sequentially



Loop Restoration

- Enhanced and simplified loop filters from VP10
- Two filter choices per superblock
 - Separable Wiener filter with explicitly coded coefficients
 - Self-guided filter
- Runs in a separate pass after CDEF
 - Showed best metrics of any approach tested
 - Uses deblocking filter output outside of superblock boundaries to minimize line buffers

Spatial and Temporal Scalability

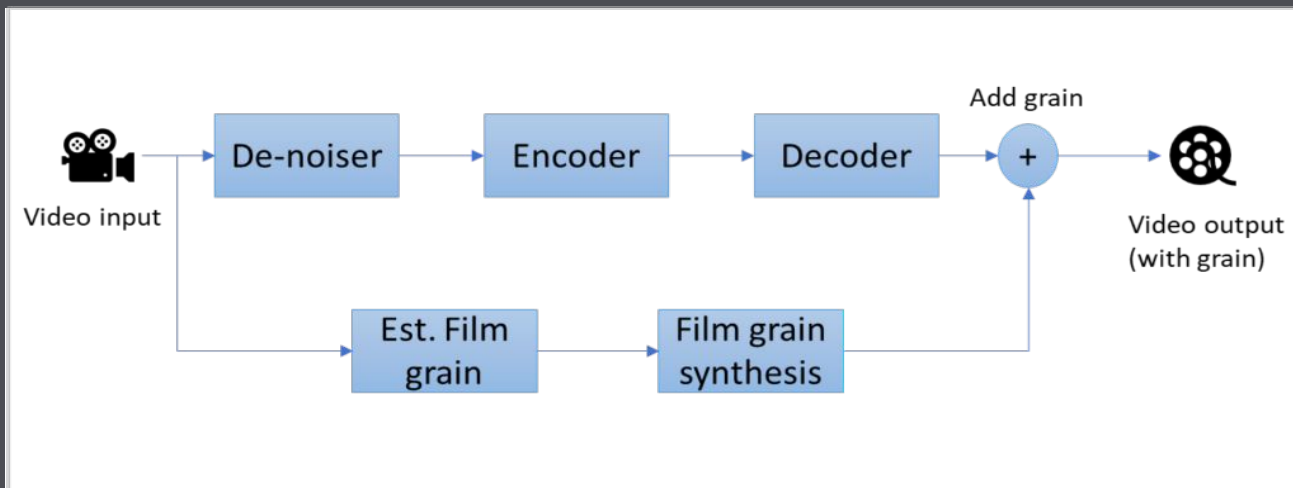
- Each frame can have a `spatial_id` and a `temporal_id`
 - When `spatial_id = 0` and `temporal_id = 0` it is called a base layer
 - When `spatial_id > 0` and `temporal_id > 0` it is called an enhancement layer
- Idea is that decoder will simply display the frames from the highest layer
 - Higher layer frames can reference lower layer frames
- Designed to be used by a special “Selective Forwarding Unit” server that hands out the appropriate scalable layer to a client

Frame Super-Resolution

- Not actually super-resolution
- Instead
 - Code at reduced resolution
 - Run deblocking filter and CDEF, but not Loop Restoration filter
 - Upsample with simple upscaler
 - Run Loop Restoration filter at full resolution
- Only horizontal resolution reduction allowed
 - Simplifies hardware (no new line buffers)
- Allows for gradual bitrate scaling

Film Grain Synthesis

- Grain parameters signaled per frame
- Synthesized film grain applied after decoding (not in loop)
- Could be applied using GLSL + PRNG based texture



AOM Members / Hardware



Designed for Hardware Implementations

Hardware members involved from the very beginning

Feedback incorporated into a number of tools

- Per symbol probability adaptation
- Smaller multipliers in entropy coder
- Single pass bitstream writing
- Fewer line buffers in CDEF and LR
- Only allow horizontal scaling for super-resolution

AOM Members / Real-Time Conferencing



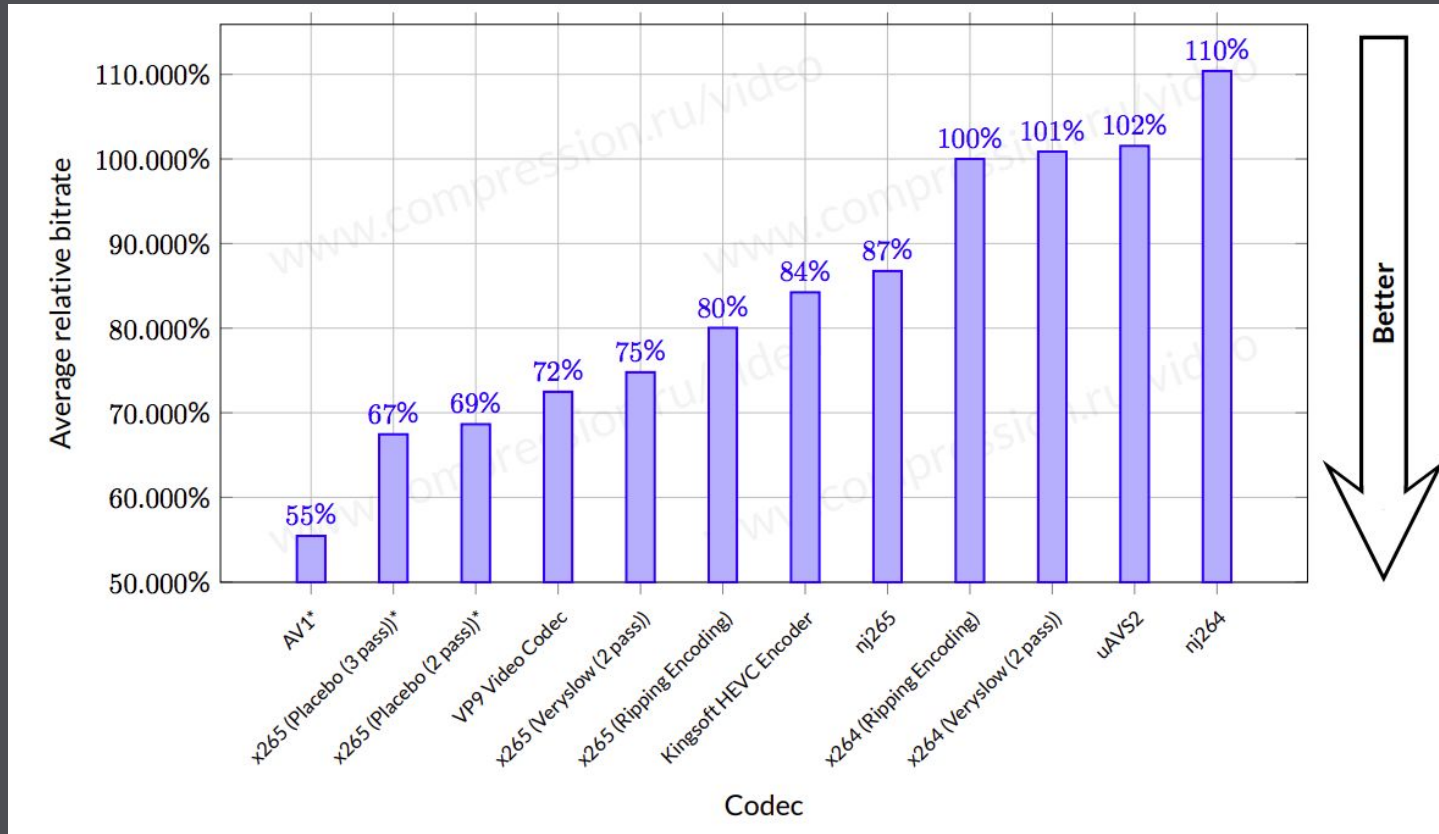
Designed for Low-Latency

Per symbol adaptation replaces symbol counts in VP9

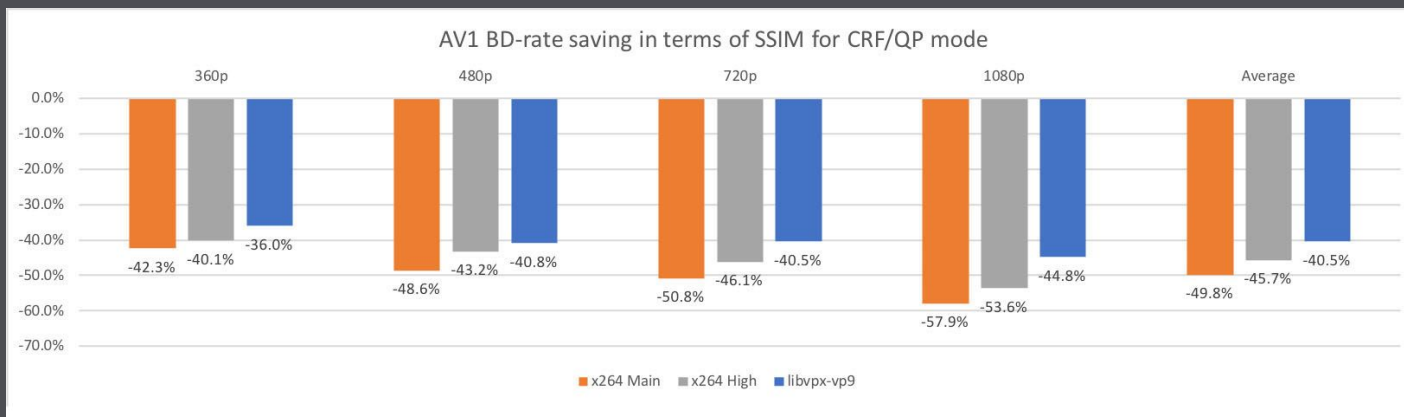
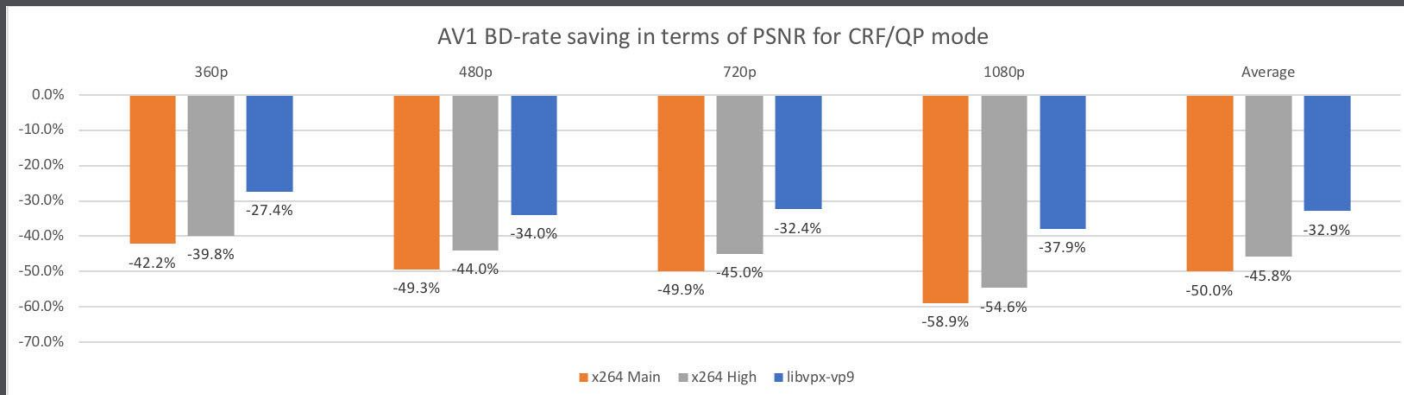
Can write bitstream with subframe latency

Removed signaling from frame header that forced whole frame buffering

Moscow State University (SSIM - June 2017)



Facebook Study (April 2018)



AVIF High Level Syntax

- AVIF [1] describes how to put AV1 [2] into HEIF [3] using ISOBMFF [4]
- Normatively specifies:
 - Profile and Level for AVIF Baseline and AVIF Advanced
 - How to add alpha transparency
- Payload is exactly AV1 with a single bit set

[1] <https://aomediacodec.github.io/av1-avif/>

[2] <https://aomediacodec.github.io/av1-spec/av1-spec.pdf>

[3] <https://www.iso.org/standard/66067.html>

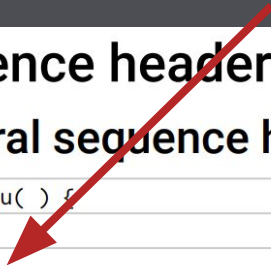
[4] <https://aomediacodec.github.io/av1-isobmff/>

AVIF High Level Syntax

- AVIF [1] describes how to put AV1 [2] into HEIF [3] using ISOBMFF [4]
- Normatively specifies:
 - Profile and Level for AVIF Baseline and AVIF Advanced
 - How to add alpha transparency
- Payload is exactly AV1 with a single bit set

5.5. Sequence header OBU syntax

5.5.1. General sequence header OBU syntax



sequence_header_obu() {	Type
seq_profile	f(3)
still_picture	f(1)
reduced_still_picture_header	f(1)
if (reduced_still_picture_header) {	
timing_info_present_flag = 0	
decoder_model_info_present_flag = 0	
initial_display_delay_present_flag = 0	

[1] <https://aomediacodec.github.io/av1-avif/>

[2] <https://aomediacodec.github.io/av1-spec/av1-spec.pdf>

[3] <https://www.iso.org/standard/66067.html>

[4] <https://aomediacodec.github.io/av1-isobmff/>

AVIF High Level Syntax

- AVIF [1] describes how to put AV1 [2] into HEIF [3] using ISOBMFF [4]
- Normatively specifies:
 - Profile and Level for AVIF Baseline and AVIF Advanced
 - How to add alpha transparency
- Payload is exactly AV1 with a single bit set (optionally two if you want smaller files)

5.5. Sequence header OBU syntax

5.5.1. General sequence header OBU syntax

sequence_header_obu() {	Type
seq_profile	f(3)
still_picture	f(1)
reduced_still_picture_header	f(1)
if (reduced_still_picture_header) {	
timing_info_present_flag = 0	
decoder_model_info_present_flag = 0	
initial_display_delay_present_flag = 0	



[1] <https://aomediacodec.github.io/av1-avif/>

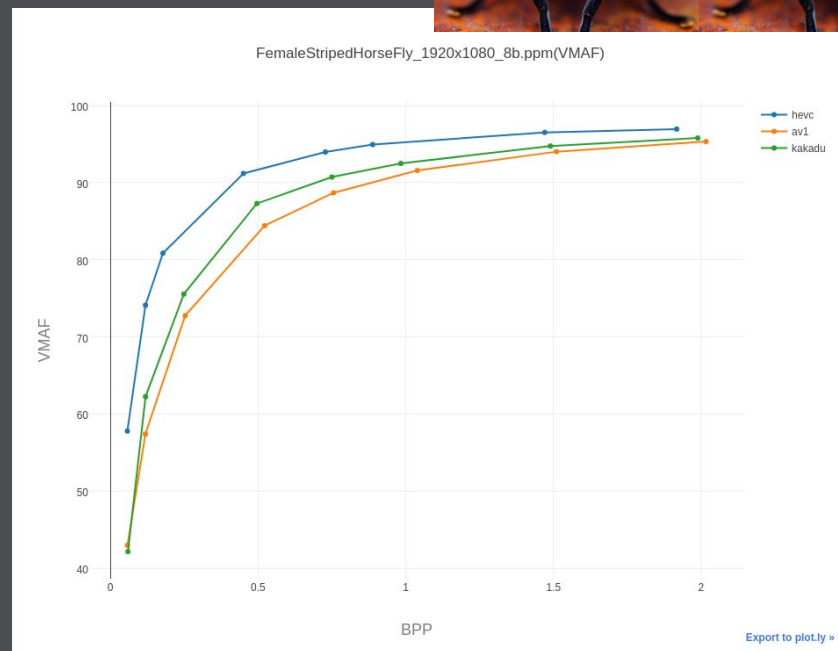
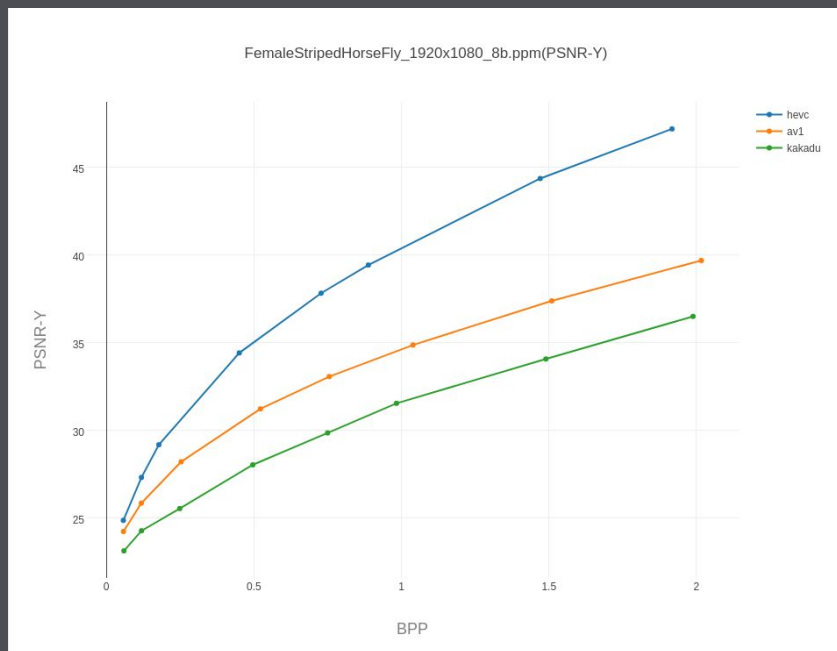
[2] <https://aomediacodec.github.io/av1-spec/av1-spec.pdf>

[3] <https://www.iso.org/standard/66067.html>

[4] <https://aomediacodec.github.io/av1-isobmff/>

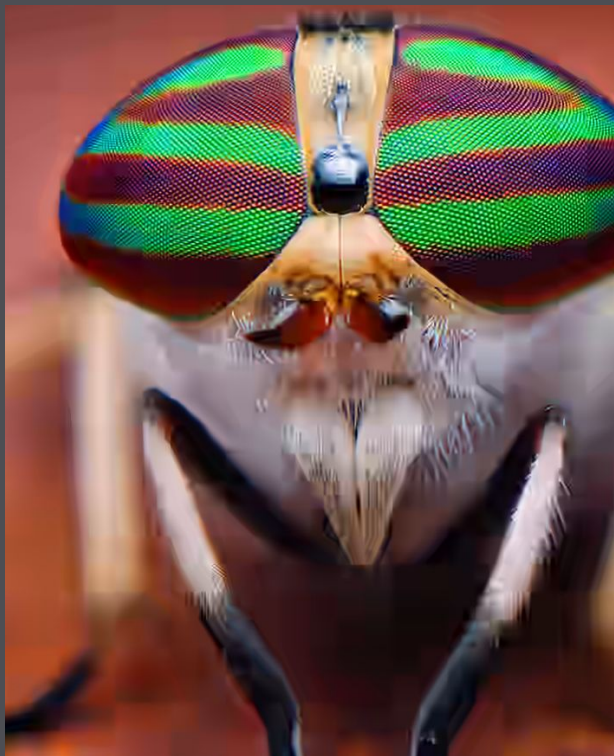
Results: FemaleStripedHorseFly_1920x1080_8b.ppm

- Image where AV1 (libaom) performed poorly



Results: FemaleStripedHorseFly_1920x1080_8b.ppm

- Image where AV1 (libaom) performed poorly



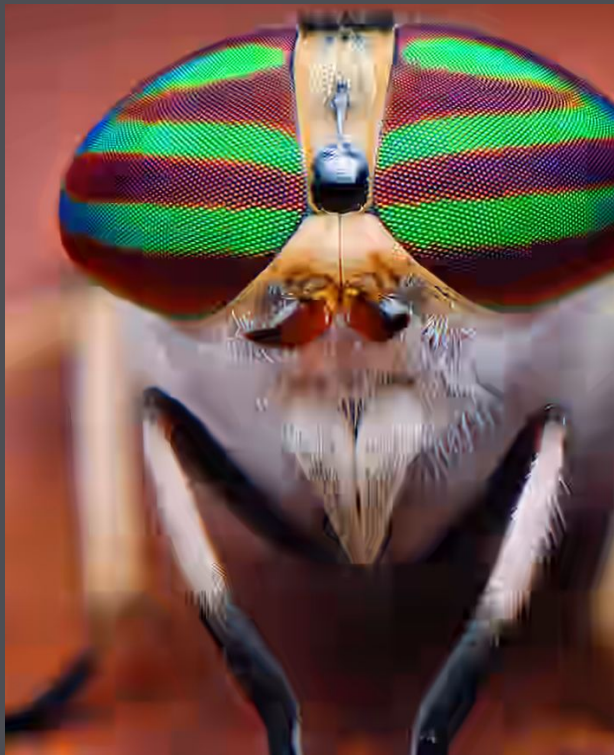
Results: FemaleStripedHorseFly_1920x1080_8b.ppm

- Image where AV1 (libaom) performed poorly

AV1

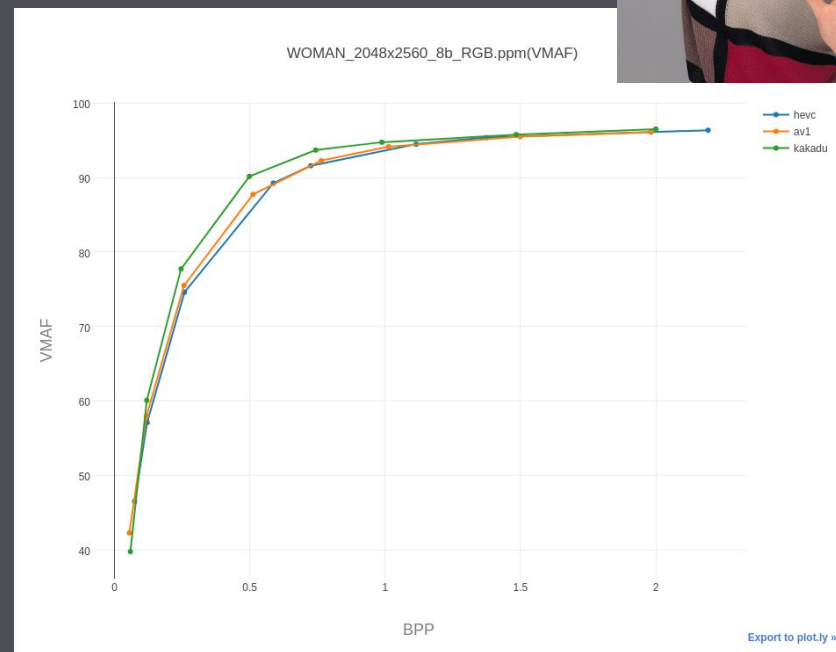
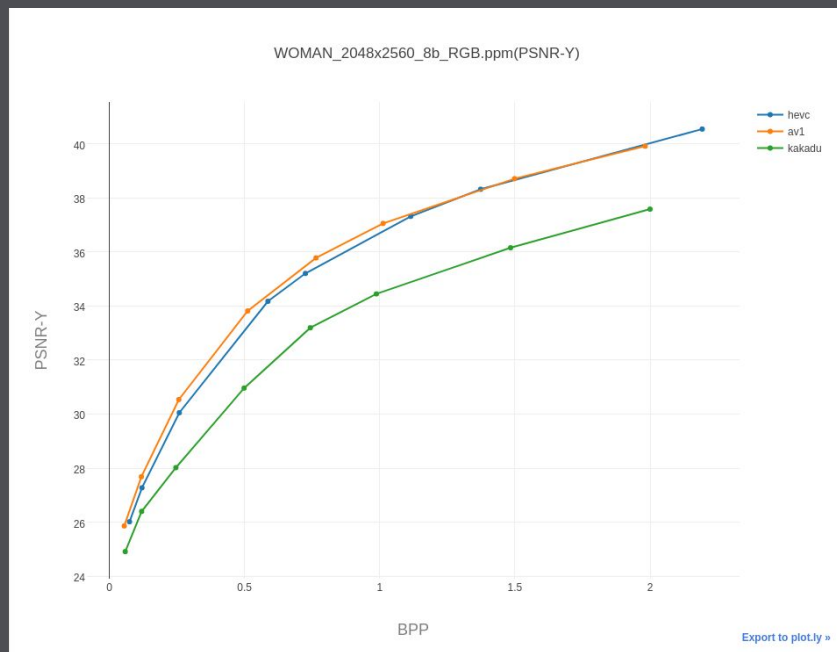
HEVC

Kakadu



Results: WOMAN_2048x2560_8b_RGB.ppm

- Image where AV1 (libaom) performed well



Results: WOMAN_2048x2560_8b_RGB.ppm

- Image where AV1 (libaom) performed well



Results: WOMAN_2048x2560_8b_RGB.ppm

- Image where AV1 (libaom) performed well

Kakadu

AV1

HEVC



Encoder Complexity:

FemaleStripedHorseFly_1920x1080_8b.ppm

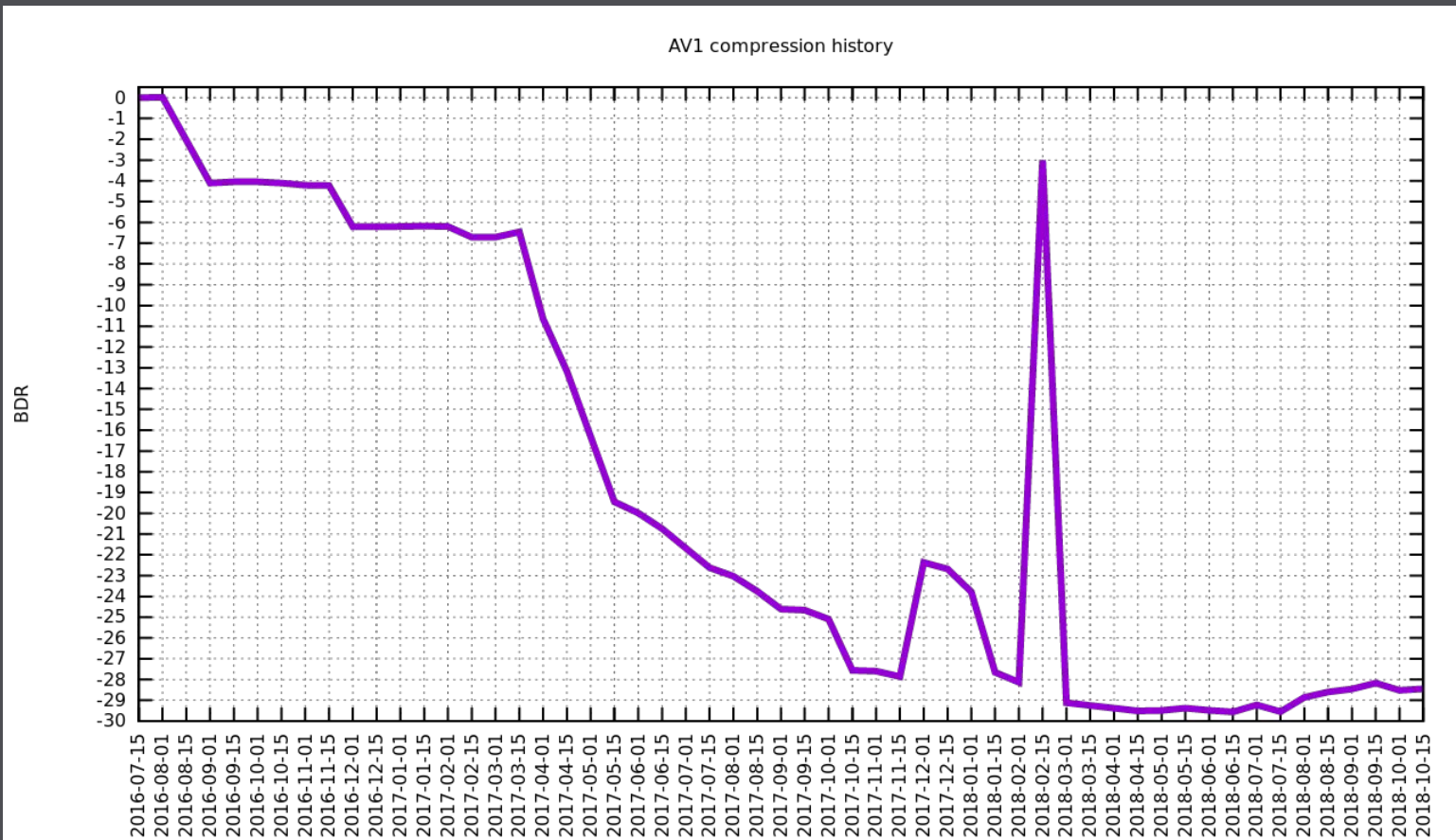
Target	AV1 (libaom-f7e0b7f3-20181012)			HEVC (HM-16.18+SCM-8.7)		
	QP	Time (s)	BPP	QP	Time (s)	BPP
0.06	61	42.062	0.0579089	45	38.507	0.0574730
0.12	56	63.782	0.1184683	41	46.593	0.1188580
0.25	48	102.878	0.2532291	37	60.138	0.2545640
0.5	39	158.155	0.5221064	33	68.876	0.4504398
0.75	34	212.129	0.7554861	29	75.824	0.7277777
1.0	29	231.838	1.0392554	26	77.757	0.9730671
1.5	22	271.657	1.5096682	21	87.923	1.4704976
2.0	15	300.162	2.0165663	16	102.384	2.0446952

Encoder Complexity:

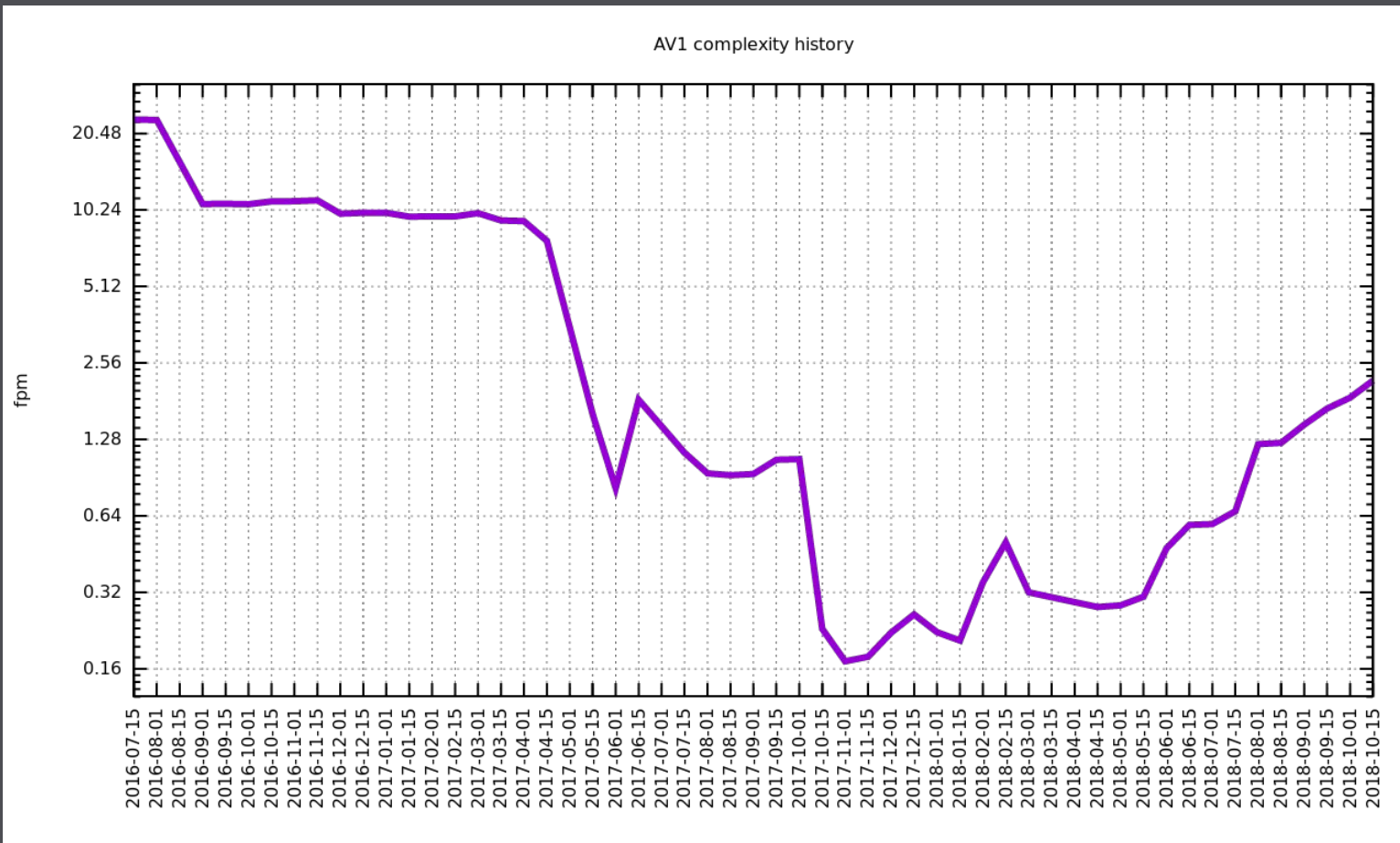
WOMAN_2048x2560_8b_RGB.ppm

Target	AV1 (libaom-f7e0b7f3-20181012)			HEVC (HM-16.18+SCM-8.7)		
	QP	Time (s)	BPP	QP	Time (s)	BPP
0.06	59	128.336	0.0551818	47	238.042	0.0747238
0.12	54	185.598	0.1186187	45	256.295	0.1209228
0.25	45	314.710	0.2571548	41	290.965	0.2588790
0.5	35	549.508	0.5118804	36	366.766	0.5244338
0.75	28	723.997	0.7644210	33	406.594	0.7252929
1.0	23	910.718	1.0125549	30	445.665	0.9879150
1.5	14	1243.480	1.5892593	26	524.937	1.5374099
2.0	11	1372.122	1.9819168	24	572.788	1.9489959

AV1 (libaom) Compression History



AV1 (libaom) Complexity History



Design Goals of Rust AV1 Encoder (rav1e)

- Clean room implementation
 - Avoid pitfalls of just porting libaom algorithms
- Sensible default behavior
 - Expose simple knobs for “quality” and “complexity”
 - Encoder controls for finer tuning, but keep to a minimum
- Use high level language (no C++)
 - Rust chosen because of focus on high performance systems programming
 - Memory safety, “fearless” concurrency, zero-cost abstractions, etc.
 - Optimized llvm backend, safe-ish SIMD, C calling convention to extern
 - Keep program control readable, focus on algorithms
- Cover all use cases
 - Start with VOD, but also work towards live streaming and interactive VC

rav1e Live Encoding

Shown at IBC in Sept 2018

- 640x480 @ 30 fps
- Single tile / thread
- Simplified feature set

The image shows a computer monitor displaying a live encoding demonstration. The screen is divided into several windows:

- Top Left:** A small window showing a live video feed of a trade show booth.
- Top Right:** A browser window displaying a YouTube video player with a woman's face. A 'YouTube AV1 STREAMING' text box is overlaid on the video.
- Bottom Left:** A terminal window titled 'rav1e REAL-TIME AV1 ENCODING' showing a system monitor window with a table of process statistics.
- Bottom Center:** A text box with the text 'YouTube AV1 STREAMING'.
- Bottom Right:** A YouTube video player interface showing a 'SUBSCRIBE' button and video recommendations.

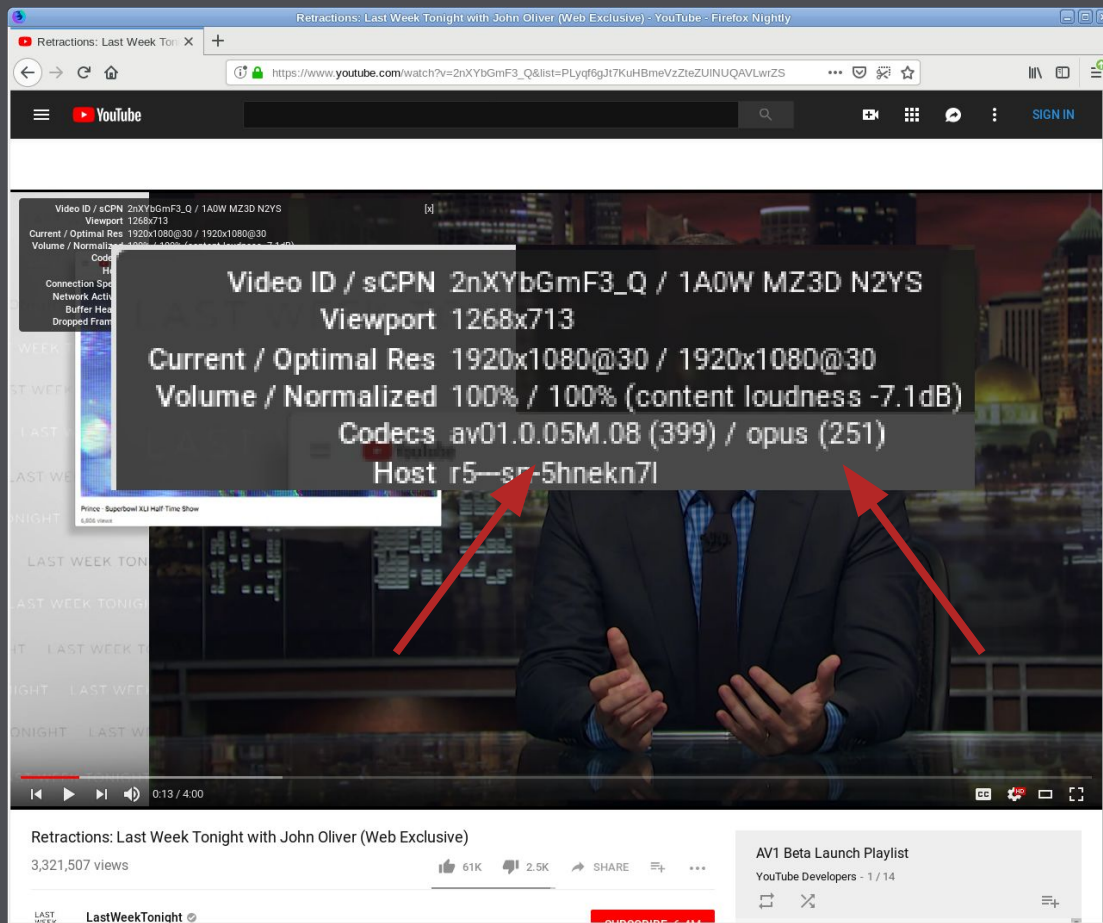
The system monitor window in the bottom left shows the following process statistics:

Process Name	% CPU	Private Memory	Threads	Open File Descriptors	FDs	User
nvml	0.0	12.4M	1	0	0	nvidia
nvml	48.7	7.0G	2	0	0	nvidia
nvml	8.7	1.0G	55	174	1947	nvidia
Google Chrome Helper	58.2	44.7G	18	56	1164	nvidia
Firefox Nightly	4.3	11.0G	27	177	1037	nvidia
WebContent	4.4	13.0G	71	131	884	nvidia
MOZApplication	4.3	7.5G	9	38	200	nvidia
nshtml	2.6	1.0G	2	81	200	nvidia
Google Chrome Helper	5.0	98.2G	209	601	0	nvidia
nvml	2.7	72.0G	27	447	1642	nvidia
Google Chrome Helper	1.7	2.4G	11	28	5197	nvidia
AudioMonitor	1.6	1.4G	21	11	84	nvidia
Compositor	1.6	7.3G	5	2	819	nvidia
Google Chrome	1.6	26.1G	8	129	148	nvidia
Google Chrome Helper	1.3	9.0G	37	24	472	nvidia
System	3.7%	0M	Threads:	4253		
nvml	20.9%		Processes:	533		
nvml	70.3%					

dav1d AV1 decoder

- Goals of dav1d project
 - Fastest software decoder possible on all platforms
 - Speed through concurrency, e.g., frame-threading, tile-threading, etc.
 - Speed through implementation, e.g., hand-written SIMD, minimum copies
- Encoder contains within it a full decoder for reconstruction
 - Re-use SIMD for inverse transforms, loop filters, intra prediction

AV1 Content on Youtube



Firefox Nightly (64 beta)

- Supports AV1 in MP4 / ISOBMFF Matroska and WebM formats
- Behind about:config flag:
`media.av1.enabled = true`
- General availability: 2018-Dec-11

Chrome (70)

- Enabled by default on desktop platforms and Android
- General availability: 2018-Oct-17

AV1 software decoders on billions of devices by the end of the week!

Compressed Bitstream Requirements

Core requirements

- ✓ Significant compression efficiency improvement over coding standards in common use at equivalent subjective quality.
- ✓ Hardware/software implementation-friendly encoding and decoding (parallelization, memory, complexity, power usage).
- ✓ Support for alpha channel / transparency coding.
- ✓ Support for animation image sequences.
- ✓ Support for 8-bit and 10-bit bit depth.
- ✓ Support for high dynamic range coding.
- ✓ Support for wide color gamut encoding.
- ✓ Support for efficient text and graphics compression.

Desirable requirements

- ✓ Support for higher bit depth (e.g. 12 to 16-bit integer or floating-point HDR) images.
- ✓ Support for different color representations, including Rec. BT.709, Rec. BT.2020, Rec. BT.2100, LogC.
- ✗ Support for embedded preview images.
- ✓ Support for very low file size image coding (e.g. <200 bytes for 64×64 pixel images).
- ✓ Support for lossless alpha channel coding.
- ✓ Support for a low-complexity encoding option.
- ✓ Support for region-of-interest coding.

Questions?