VP9 Encoding for VOD and Live

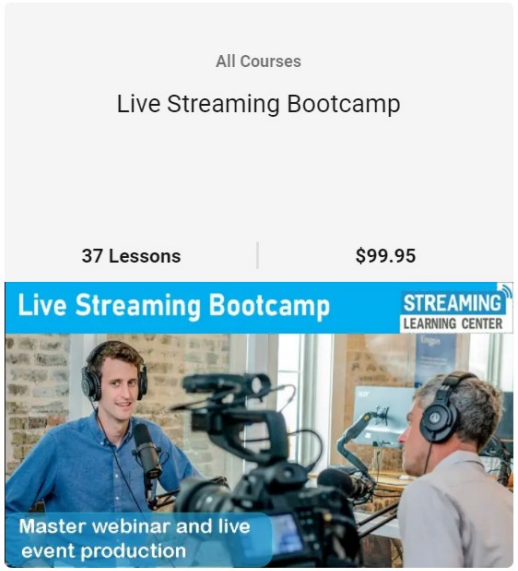
Jan Ozer
Streaming Learning Center
janozer@gmail.com

Agenda

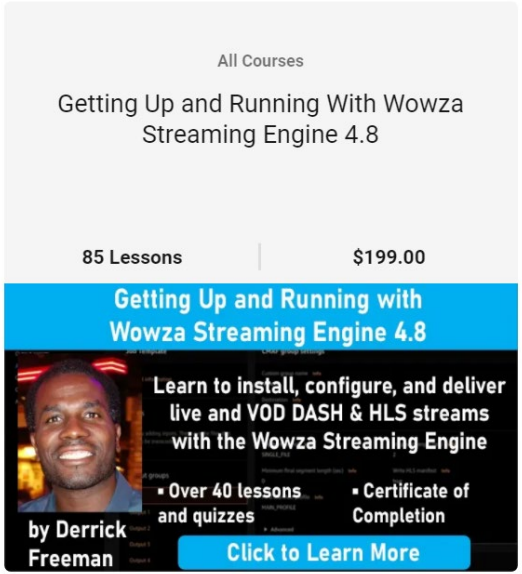
- Lesson 1: What VP9 is and where VP9-encoded videos play
- Lesson 2: VP9 encoding configuration options
- Lesson 3: How VP9 quality compares to H.264, HEVC, and AV1
- Lesson 4: What options are available for live:
  - VP9 origination
  - VP9 transcoding
- Resources
  - Didn’t get to produce section on DASH (if that’s your main interest, you probably shouldn’t stay - sorry)

Sponsored By: The Streaming Learning Center

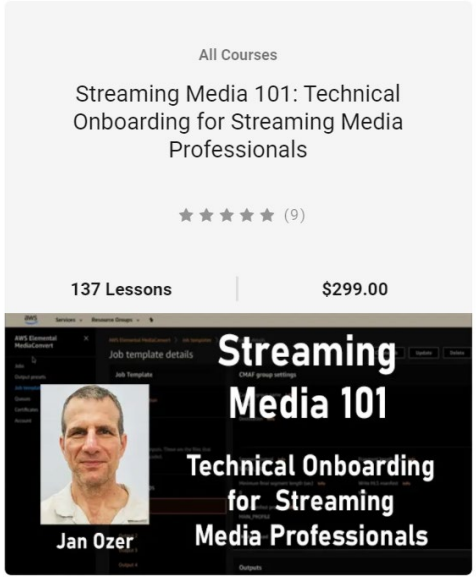
- Where Streaming Professionals Come to Learn



https://bit.ly/WSE_course



https://bit.ly/WSE_course



https://bit.ly/StreamingMedia101

Lesson: What VP9 is and where VP9-encoded videos play

- About VP9
- Where it plays
  - Browser
  - Mobile
  - Living Room

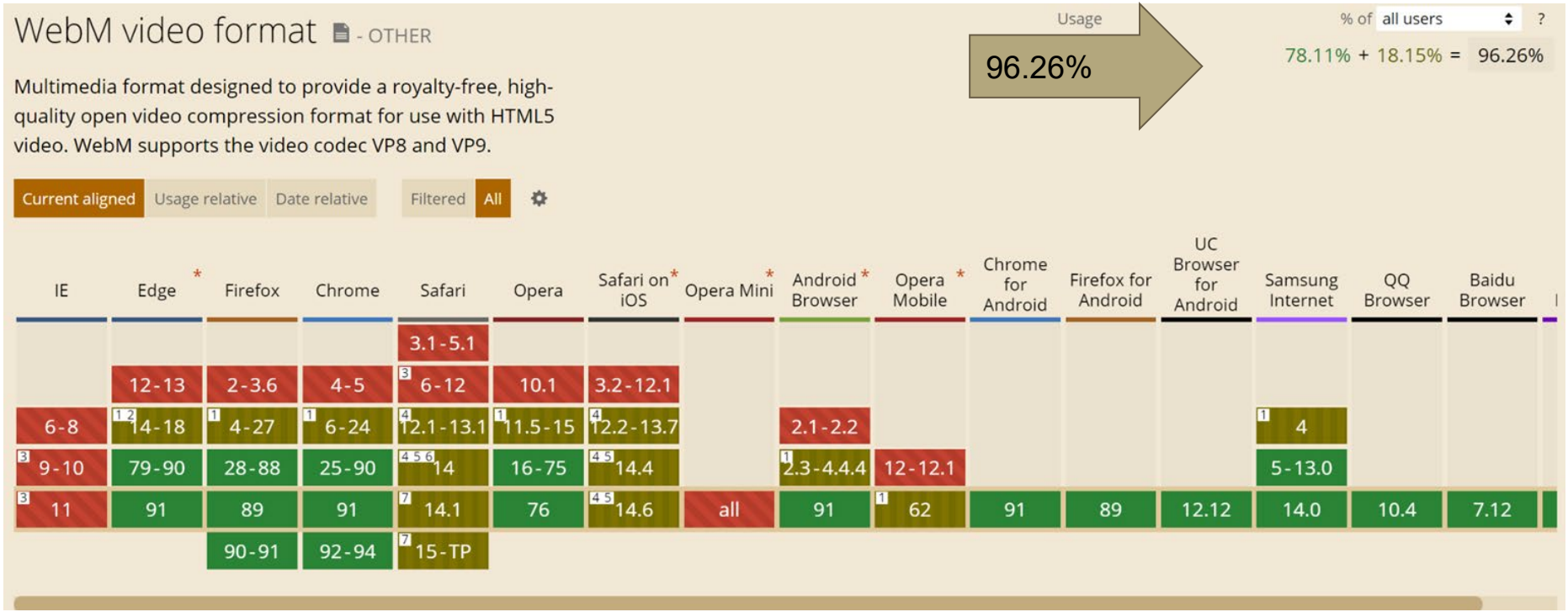
About VP9

- VP9 is an open-source codec produced by Google
  - From On2 acquisition that first produced VP8/WebM
  - Supposedly open-sourced and royalty free
  - Sisvel launched VP9/AV1 patent pool in 2019 (bit.ly/SV_VP9)
    - Applies to consumer hardware devices (not currently software)
    - Not currently seeking royalties on VP9-encoded content
    - Google disputes these claims so resolution is unclear

Lesson: Where VP9-encoded videos play

- Browser
- Mobile
- Living Room

VP9 Playback in a Browser



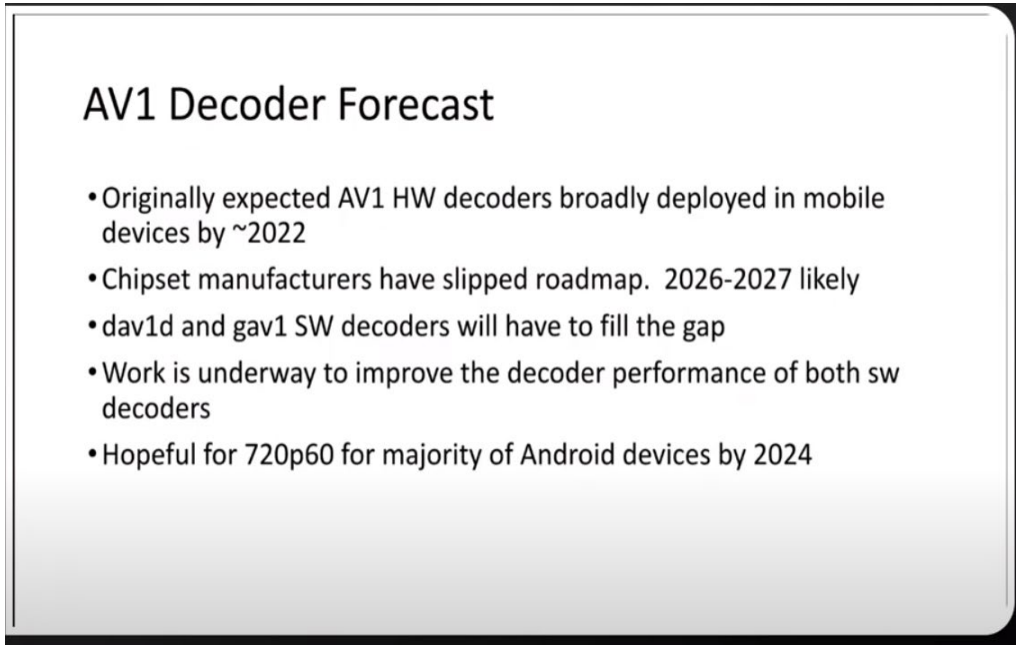
https://caniuse.com/?search=VP9

VP9 Playback in a Browser

	H.264	HEVC	VP9	AV1
CanIUse browser	98.23%	18.8%	96.26%	74.25%

- Still a meaningful lead over AV1 (but for how long?)
- HEVC is a total non-player in the browser

AV1 Hardware Slower than Forecast



JP's ChalkTalks - Scaling Video with AV1! With David Ronca (EP:11) (~31 minutes)

https://youtu.be/3qL5FdEBiGA

VP9 Playback - Mobile

	H.264	HEVC	VP9	AV1
Browser	Android/iOS	iOS	Android/iOS (?)	Android
App	Android/iOS	Android/iOS	Android/iOS	Android/iOS

- VP9 preferred over HEVC for Android browser
- Supported by iOS for 4K YouTube playback
- AV1 has battery life concerns

VP9 Playback - Living Room

	H.264	HEVC	VP9	AV1
Smart TV	All	All	Most (Samsung)	Nascent but growing
OTT	All	All	Many (Roku, Apple TV)	Very little
High Dynamic Range	Not preferred	Preferred	Not preferred	Not preferred

- Implemented mostly to enable 4K support from YouTube
- Not a preferred format for HDR which drives most premium living room viewers

Summary

- Sisvel pool likely not a deterrent for content producers
- Sweetspot appears to be:
  - Browser-based playback (JWPlayer OVP)
  - Download to mobile devices (Netflix)
  - Non-HDR content to the living room

Lesson 2: VP9 Encoding Options (libvpx-vp9)

- Starting point
- Working through the options
- Finalizing the string
- Comparing with
  - Google
  - AWS Elemental
  - Amazon Elastic Transcoder

Starting Point

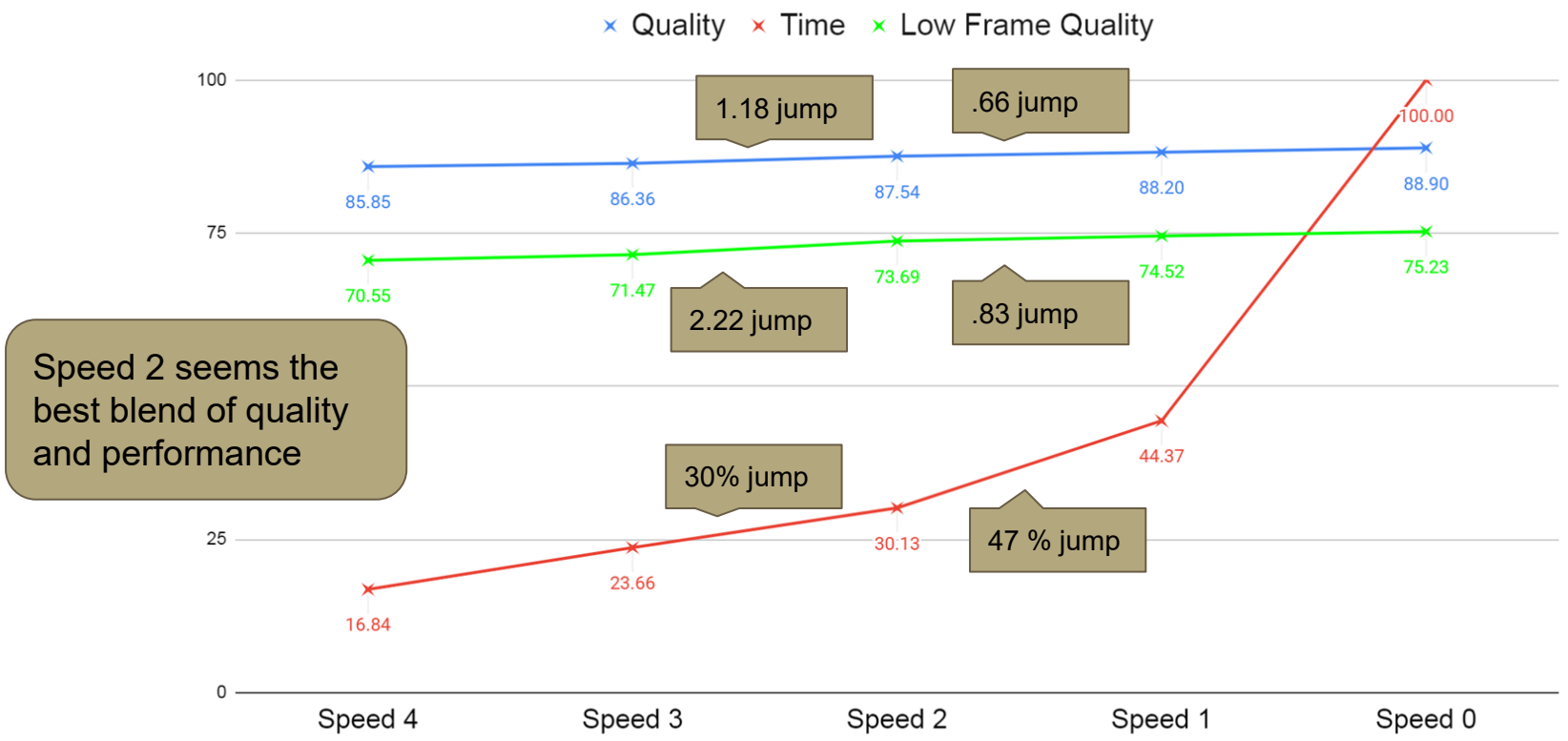
1. Baseline

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60  -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60  freedom_10_baseline.webm
```

- `-b:v 2500K` = Target bitrate
- `-g 60` = Keyframe interval
- `-keyint_min 60` = Minimum keyframe interval
- `-maxrate 5000K` = Maximum rate

Choose Preset Speed



Speed 2 seems the best blend of quality and performance

Speed - 4 or 2 First Pass?

Issue: Does using slower higher quality speed setting (2) for first pass impact quality or encoding time over faster speed setting (4)?

Default: 1

Findings: No impact on speed or quality

Decision: Go with 4 for first pass

Encoding Time	4 First Pass	2 First Pass	Delta
Freedom	0:02:00	0:02:01	0.83%
Football	0:02:29	0:02:31	1.34%
Average	0:02:15	0:02:16	**1.12%**
Bitrate	**4 First Pass**	**2 First Pass**	**Delta**
Freedom	2,398	2,398	0.00%
Football	2,484	2,484	0.00%
Average	2,441	2,441	**0.00%**
VMAF	**4 First Pass**	**2 First Pass**	**Delta**
Freedom	86.96	86.96	0.00%
Football	88.50	88.50	0.00%
Average	87.73	87.73	**0.00%**
Low Frame	**4 First Pass**	**2 First Pass**	**Delta**
Freedom	75.68	75.68	0.00%
Football	62.00	62.00	0.01%
Average	68.84	68.84	**0.00%**
Standard Deviation	**4 First Pass**	**2 First Pass**	**Delta**
Freedom	4.44	4.44	0.00%
Football	8.92	8.92	-0.03%
Average	6.68	6.68	**-0.05%**

Updated Command String

Insert different speeds

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 output.webm
```

Minrate:

Issue: Minimum bitrate specified in several Google recommended command strings. What is its impact on encoding?

Default: Not set

Findings:

- Specifying minimum bitrate boosts the bitrate over the target (bad for comparisons)
- Also boosts low frame quality appreciably (which is good)

Decision: Left out of command string for my tests because needed data rate accuracy; may want to consider for production (in Google command string).

Encoding Time	Baseline	Minrate	Delta
Freedom	0:02:00	0:02:01	0.83%
Football	0:02:29	0:02:46	11.41%
Average	0:02:15	0:02:24	**6.69%**
Bitrate	**Baseline**	**Minrate**	**Delta**
Freedom	2398	2613	8.97%
Football	2484	2824	13.69%
Average	2441	2719	**11.37%**
VMAF	**Baseline**	**Minrate**	**Delta**
Freedom	86.96	88.79	2.10%
Football	88.50	90.40	2.15%
Average	87.73	89.60	**2.13%**
Low Frame	**Baseline**	**Minrate**	**Delta**
Freedom	75.68	82.21	8.62%
Football	62.00	71.11	14.70%
Average	68.84	76.66	**11.36%**
Standard Deviation	**Baseline**	**Minrate**	**Delta**
Freedom	4.44	3.44	-22.68%
Football	8.92	7.66	-14.11%
Average	6.68	5.55	**-16.96%**

Threads

Issue: Threads control the amount of CPU allocated to encoding. What's the optimal setting?

Default: Not specified; appears to be over 1

Findings:

- There is no quality difference irrespective of thread count
- Encoding with one thread is much slower
- Unless you're running highly optimized encoder (e.g. one encoder for each thread) any value above 1 should be fine

Decision: Used 8

Encoding Time	Baseline	1 Threads	4 Threads	8 Threads	16 Threads	Delta
Freedom	0:02:00	0:03:44	0:01:58	0:01:57	0:01:57	91.45%
Football	0:02:29	0:05:00	0:02:25	0:02:23	0:02:26	107.69%
Average	0:02:15	0:04:22	0:02:12	0:02:10	0:02:12	100.38%
Bitrate	Baseline	1 Threads	4 Threads	8 Threads	16 Threads	Delta
Freedom	2,398	2,398	2,398	2,398	2,398	0.00%
Football	2,484	2,484	2,484	2,484	2,484	0.00%
Average	2,441	2,441	2,441	2,441	2,441	0.00%
VMAF	Baseline	1 Threads	4 Threads	8 Threads	16 Threads	Delta
Freedom	86.96	86.96	86.96	86.96	86.96	0.00%
Football	88.50	88.50	88.50	88.50	88.50	0.00%
Average	87.73	87.73	87.73	87.73	87.73	0.00%
Low Frame	Baseline	1 Threads	4 Threads	8 Threads	16 Threads	Delta
Freedom	75.68	75.68	75.68	75.68	75.68	0.00%
Football	62.00	62.00	62.00	62.00	62.00	0.00%
Average	68.84	68.84	68.84	68.84	68.84	0.00%
Standard Deviation	Baseline	1 Threads	4 Threads	8 Threads	16 Threads	Delta
Freedom	4.44	4.44	4.44	4.44	4.44	0.00%
Football	8.92	8.92	8.92	8.92	8.92	0.00%
Average	6.68	6.68	6.68	6.68	6.68	0.00%

Updated Command String

Insert threads

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 output.webm
```

Quality

Issue: Quality controls encoding speed/quality. What's the optimal setting?

Default: Good

Findings:

- Best boosts data rate and quality but
- Increases encoding time by 650%

Decision:

- Used good for my tests
- Added to encoding string even though default value (ease of comparison)
- Default config options in encoding string are default settings (not needed)

Encoding Time	Baseline	Good	Best	Delta
Freedom	0:01:57	0:01:57	0:13:44	604%
Football	0:02:23	0:02:27	0:19:02	699%
Average	0:02:10	0:02:12	0:16:23	656%
Bitrate	Baseline	Good	Best	Delta
Freedom	2,398	2,398	2,404	0%
Football	2,484	2,484	2,615	5%
Average	2,441	2,441	2,510	3%
VMAF	Baseline	Good	Best	Delta
Freedom	86.96	86.96	88.74	2%
Football	88.50	88.50	90.45	2%
Average	**87.73**	**87.73**	**89.60**	2%
Low Frame	Baseline	Good	Best	Delta
Freedom	75.68	75.68	80.86	7%
Football	62.00	62.00	69.68	12%
Average	**68.84**	**68.84**	**75.27**	9%
Standard Deviation	Baseline	Good	Best	Delta
Freedom	4.44	4.44	3.94	13%
Football	8.92	8.92	7.94	12%
Average	6.68	6.68	5.94	13%

Updated Command String

Insert Quality

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good output.webm
```

Tile/Columns

Issue: “Tile-columns “split the video-image into columns and rows. Higher values may improve multithreading capability and encoding speed, but the video quality decreases and the resulting file size is bigger.” (bit.ly/AOM_AV1)

Default: -1

Findings:

- 1 Tile-columns boost encoding time without quality benefit

Decision:

- Use 4 (most recommended setting)

Encoding Time	Baseline	-1 T/C	1 T/C	4 T/C	6 T/C	Delta
Freedom	0:01:57	0:01:58	0:02:29	0:01:57	0:01:59	27.35%
Football	0:02:23	0:02:34	0:03:14	0:02:25	0:02:24	35.66%
Average	**0:02:10**	**0:02:16**	**0:02:52**	**0:02:11**	**0:02:12**	31.92%
Bitrate	Baseline	-1 T/C	1 T/C	4 T/C	6 T/C	Delta
Freedom	2,398	2,398	2,399	2,398	2,398	0.04%
Football	2,484	2,484	2,484	2,484	2,484	0.00%
Average	**2,441**	**2,441**	**2,442**	**2,441**	**2,441**	0.02%
VMAF	Baseline	-1 T/C	1 T/C	4 T/C	6 T/C	Delta
Freedom	86.96	86.96	87.04	86.96	86.96	0.08%
Football	88.50	88.50	88.69	88.50	88.50	0.21%
Average	**87.73**	**87.73**	**87.86**	**87.73**	**87.73**	0.15%
Low Frame	Baseline	-1 T/C	1 T/C	4 T/C	6 T/C	Delta
Freedom	75.68	75.68	76.17	75.68	75.68	0.64%
Football	62.00	62.00	61.81	62.00	62.00	0.31%
Average	**68.84**	**68.84**	**68.99**	**68.84**	**68.84**	0.21%
Standard Deviation	Baseline	-1 T/C	1 T/C	4 T/C	6 T/C	Delta
Freedom	4.44	4.44	4.43	4.44	4.44	0.21%
Football	8.92	8.92	9.03	8.92	8.92	1.23%
Average	**6.68**	**6.68**	**6.73**	**6.68**	**6.68**	0.75%

Updated Command String

Tile-columns

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4
output.webm
```

Auto-Alt-Ref

Issue: Enable automatic alt reference frames or not

Default: -1, range is -1 to 6

Findings:

- Settings beyond 2 drop the data rate and impact VMAF
- Auto-alt-ref 1 delivers generally good performance and hits data rate target

Decision:

- Go with Auto-alt-ref 1

Encoding Time	Baseline	AAR - -1	AAR - 1	AAR - 2	AAR - 4	AAR - 6	Delta
Freedom	0:01:57	0:01:58	0:01:58	0:01:55	0:02:02	0:02:02	6.09%
Football	0:02:25	0:02:25	0:02:24	0:02:08	0:02:10	0:02:10	13.28%
Average	**0:02:11**	**0:02:12**	**0:02:11**	**0:02:02**	**0:02:06**	0:02:06	9.88%
Bitrate	Baseline	AAR - -1	AAR - 1	AAR - 2	AAR - 4	AAR - 6	Delta
Freedom	2,398	2,398	2,398	1,998	1,995	1,996	20.20%
Football	2,484	2,484	2,484	2,134	2,150	2,116	17.39%
Average	**2,441**	**2,441**	**2,441**	**2,066**	**2,073**	**2,056**	18.75%
VMAF	Baseline	AAR - -1	AAR - 1	AAR - 2	AAR - 4	AAR - 6	Delta
Freedom	86.96	86.96	86.96	84.88	85.07	85.16	2.46%
Football	88.50	88.50	88.50	86.45	86.90	86.54	2.38%
Average	**87.73**	**87.73**	**87.73**	**85.66**	**85.98**	**85.85**	2.42%
Low Frame	Baseline	AAR - -1	AAR - 1	AAR - 2	AAR - 4	AAR - 6	Delta
Freedom	75.68	75.68	75.68	75.31	73.53	73.53	2.93%
Football	62.00	62.00	62.00	60.03	62.00	60.41	3.29%
Average	**68.84**	**68.84**	**68.84**	**67.67**	**67.77**	**66.97**	3.09%
Standard Deviation	Baseline	AAR - -1	AAR - 1	AAR - 2	AAR - 4	AAR - 6	Delta
Freedom	4.44	4.44	4.44	4.89	4.81	4.87	10.11%
Football	8.92	8.92	8.92	7.83	7.73	7.87	15.50%
Average	**6.68**	**6.68**	**6.68**	**6.36**	**6.27**	**6.37**	13.53%

Updated Command String

Auto-alt-ref

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -auto-alt-ref 1 -f
webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4 -auto-
alt-ref 1 output.webm
```

Lag In Frames

Issue: Best setting for lag-in-frames

Default: -1

Findings:

- Lag-in-frames -1 didn't work
- Lag-in-frames 1 speeds encode but delivers significant transient issue in football clip (see following)
- 25 delivers best overall VMAF, low frame, and standard deviation

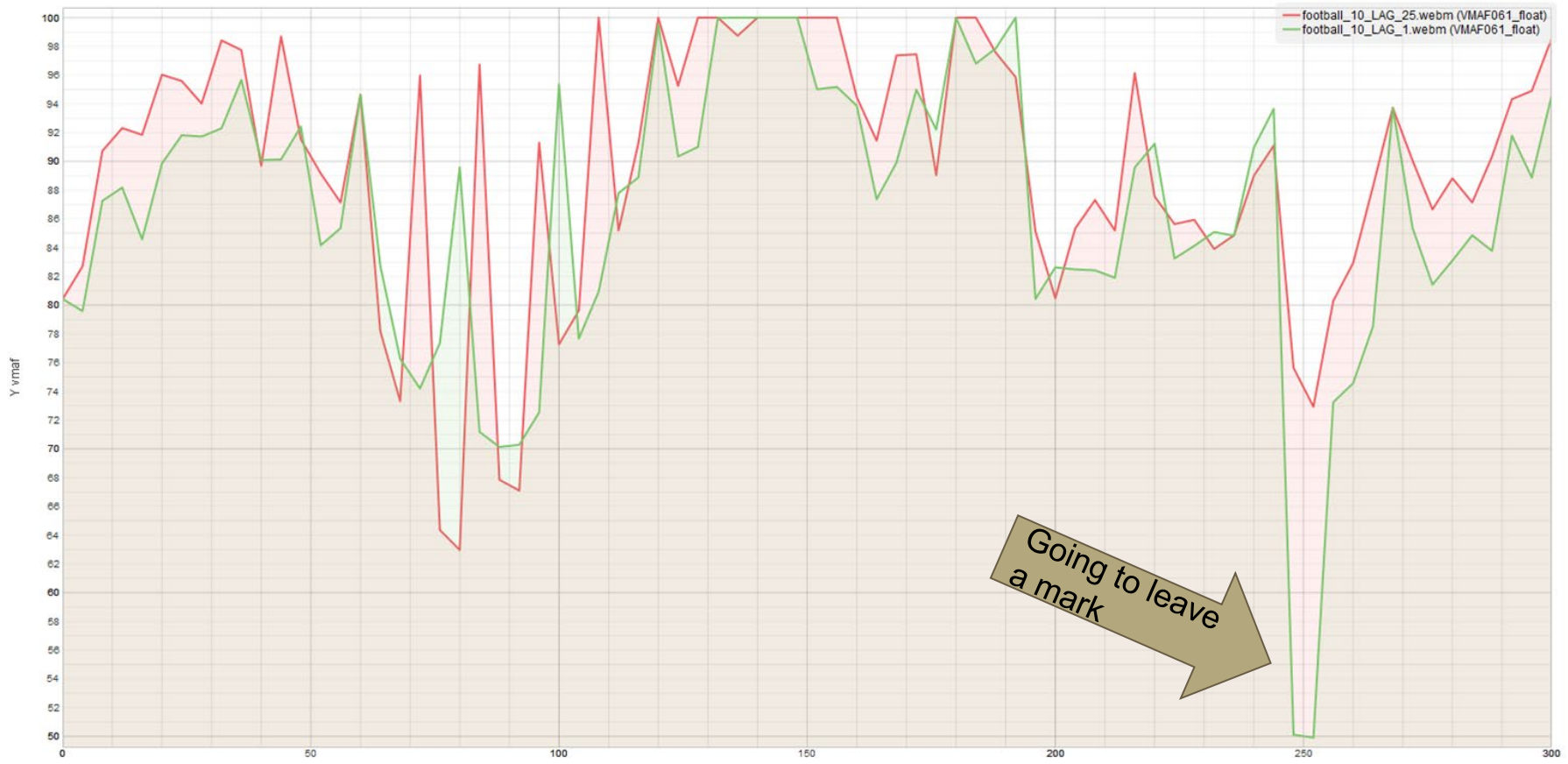
Decision:

- Got with lag-in-frames 25

Encoding Time	Baseline	LAG -1	LAG 1	LAG 8	LAG 16	LAG 25	Delta
Freedom	0:01:58		0:01:07	0:01:12	0:01:53	0:01:57	76.12%
Football	0:02:24		0:01:32	0:01:56	0:02:23	0:02:26	58.70%
Average	**0:02:11**		**0:01:20**	**0:01:34**	**0:02:08**	0:02:12	65.41%
Bitrate	**Baseline**	**LAG -1**	**LAG 1**	**LAG 8**	**LAG 16**	**LAG 25**	**Delta**
Freedom	2,398		2,409	2,390	2,398	2,398	0.79%
Football	2,484		2,356	2,368	2,485	2,484	5.48%
Average	2,441		2,383	2,379	2,442	2,441	2.63%
VMAF	**Baseline**	**LAG -1**	**LAG 1**	**LAG 8**	**LAG 16**	**LAG 25**	**Delta**
Freedom	86.96		86.74	86.19	86.70	86.96	0.90%
Football	88.50		86.08	85.82	87.66	88.50	3.12%
Average	**87.73**		**86.41**	**86.01**	**87.18**	**87.73**	2.01%
Low Frame	**Baseline**	**LAG -1**	**LAG 1**	**LAG 8**	**LAG 16**	**LAG 25**	**Delta**
Freedom	75.68		77.87	67.13	74.99	75.68	16.00%
Football	62.00		38.50	40.09	60.65	62.00	61.03%
Average	**68.84**		**58.18**	**53.61**	**67.82**	**68.84**	28.41%
Standard Deviation	**Baseline**	**LAG -1**	**LAG 1**	**LAG 8**	**LAG 16**	**LAG 25**	**Delta**
Freedom	4.44		4.11	5.31	5.06	4.44	28.98%
Football	8.92		10.38	11.06	9.39	8.92	23.91%
Average	**6.68**		**7.25**	**8.18**	**7.22**	**6.68**	22.41%

Lag-in-Frames 1 vs. 25

Red - LIF - 25
Green - LIF - 1




HOME TEAM MARKETING
HOME TEAM MARKETING
01;00;08;11
harmonic


LAG 25
HOME TEAM MARKETING
HOME TEAM MARKETING
01;00;08;11
harmonic


LAG 1
HOME TEAM MARKETING
HOME TEAM MARKETING
01;00;08;11
harmonic

Updated Command String

Lag-in-Frames

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -auto-alt-ref 1 -lag-
in-frames 25  -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4 -auto-
alt-ref 1 -lag-in-frames 25 output.webm
```

Frame Parallel

Issue: Best setting for frame parallel (improves decoding efficiency)

Default: auto

Findings:

- Frame parallel enabled appears to be default
- Disabling frame parallel improves quality very slightly

Decision:

- Enable frame parallel (in green because appears to be default)

Encoding Time	Baseline	FP 0	FP 1	Delta
Freedom	0:01:57	0:01:57	0:01:58	0.85%
Football	0:02:26	0:02:25	0:02:25	0.69%
Average	0:02:12	0:02:11	0:02:12	0.38%
Bitrate	**Baseline**	**FP 0**	**FP 1**	**Delta**
Freedom	2,398	2,398	2,398	0.00%
Football	2,484	2,493	2,484	0.36%
Average	2,441	2,446	2,441	0.18%
VMAF	**Baseline**	**FP 0**	**FP 1**	**Delta**
Freedom	86.96	87.18	86.96	0.25%
Football	88.50	88.68	88.50	0.20%
Average	**87.73**	**87.93**	**87.73**	0.22%
Low Frame	Baseline	FP 0	FP 1	**Delta**
Freedom	75.68	76.44	75.68	1.00%
Football	62.00	63.29	62.00	2.08%
Average	**68.84**	**69.86**	**68.84**	1.49%
Standard Deviation	**Baseline**	**FP 0**	**FP 1**	**Delta**
Freedom	4.44	4.37	4.44	1.75%
Football	8.92	8.71	8.92	2.45%
Average	6.68	6.54	6.68	2.21%

Updated Command String

Frame Parallel

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -auto-alt-ref 1 -lag-
in-frames 25 -frame-parallel 1 -f  webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4 -auto-
alt-ref 1 -lag-in-frames 25 -frame-parallel 1 output.webm
```

Row MT

Issue: Does enabling row-mt (row-based multithreading) increase encoding speed without significant quality reduction?

Default: auto

Findings:

- Significant speed increase
- Negligible quality change

Decision:

- Enable row-mt

Encoding Time	Baseline	RMT 0	RMT 1	Delta
Freedom	0:01:58	0:01:59	0:01:16	56.58%
Football	0:02:25	0:02:25	0:01:31	59.34%
Average	**0:02:12**	**0:02:12**	**0:01:24**	**58.08%**
Bitrate	**Baseline**	**RMT 0**	**RMT 1**	**Delta**
Freedom	2,398	2,398	2,398	0.00%
Football	2,484	2,484	2,494	0.40%
Average	**2,441**	**2,441**	**2,446**	**0.20%**
VMAF	**Baseline**	**RMT 0**	**RMT 1**	**Delta**
Freedom	86.96	86.96	87.08	0.13%
Football	88.50	88.50	88.60	0.11%
Average	**87.73**	**87.73**	**87.84**	**0.12%**
Low Frame	**Baseline**	**RMT 0**	**RMT 1**	**Delta**
Freedom	75.68	75.68	75.34	0.46%
Football	62.00	62.00	62.03	0.05%
Average	**68.84**	**68.84**	**68.68**	**0.23%**
Standard Deviation	**Baseline**	**RMT 0**	**RMT 1**	**Delta**
Freedom	4.44	4.44	4.40	0.90%
Football	8.92	8.92	8.90	0.27%
Average	**6.68**	**6.68**	**6.65**	**0.48%**

Updated Command String

Row MT

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -auto-alt-ref 1 -lag-
in-frames 25 -frame-parallel 1 -f -row-mt 1 webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4 -auto-
alt-ref 1 -lag-in-frames 25 -frame-parallel 1 -row-mt 1 output.webm
```

Tuning

Issue: Does tuning for psnr or ssim improve VMAF score?

Default: both off

Findings:

- Tuning for PSNR doesn't seem to work
- Tuning for SSIM slightly decreases VMAF

Decision:

- Don't tune

Encoding Time	Baseline	PSNR	SSIM	Delta
Freedom	0:01:16	0:01:20	0:01:27	14.47%
Football	0:01:31	0:01:32	0:01:33	2.20%
Average	**0:01:24**	**0:01:26**	**0:01:30**	**7.78%**
Bitrate	**Baseline**	**PSNR**	**SSIM**	**Delta**
Freedom	2,398	2,398	2,396	0.08%
Football	2,494	2,494	2,505	0.44%
Average	**2,446**	**2,446**	**2,451**	**0.18%**
VMAF	**Baseline**	**PSNR**	**SSIM**	**Delta**
Freedom	87.08	87.08	85.02	2.42%
Football	88.60	88.60	87.73	1.00%
Average	**87.84**	**87.84**	**86.37**	**1.70%**
Low Frame	**Baseline**	**PSNR**	**SSIM**	**Delta**
Freedom	75.34	75.34	73.55	2.44%
Football	62.03	62.03	62.35	0.51%
Average	**68.68**	**68.68**	**67.95**	**1.09%**
Standard Deviation	**Baseline**	**PSNR**	**SSIM**	**Delta**
Freedom	4.40	4.40	4.93	11.98%
Football	8.90	8.90	8.90	0.02%
Average	**6.65**	**6.65**	**6.91**	**3.95%**

Final Command String

```
ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 1 -b:v 2500K -g 60 -keyint_min
60 -speed 4 -threads 8 -quality good -tile-columns 4 -auto-alt-ref 1 -lag-
in-frames 25 -frame-parallel 1 -f -row-mt 1 webm NUL && \

ffmpeg -y -i input.mp4 -c:v libvpx-vp9 -pass 2 -b:v 2500K -maxrate 5000K  -g
60 -keyint_min 60 -speed 2 -threads 8 -quality good -tile-columns 4 -auto-
alt-ref 1 -lag-in-frames 25 -frame-parallel 1 -row-mt 1 output.webm
```

Google Recommendations - Concerns

```
ffmpeg -i input.mov -b:v 1800k -minrate 900k -maxrate 2610k -tile-
columns 2 -g 240 -threads 8 -quality good -crf 31  -pass 1 -speed 4
output.webm &&

ffmpeg -i input.mov -b:v 1800k -minrate 900k -maxrate 2610k -tile-
columns 3 -g 240 -threads 8 -quality good -crf 31 -c:v libvpx-vp9 -pass
2 -speed 4 -y output.webm
```

- Which controls quality, CRF or bitrate?
- Speed 4 in second pass (faster, but lower quality)
- GOP is 8 seconds for 30 fps (changed for our comparisons)

- No -row-mt (might make up quality deficit and reduce encoding time)
- Minrate caused data rate miss on our tests

https://developers.google.com/media/vp9/settings/vod

Google Recommendations

Issue: How does Google's recommendations compare?

Findings:

- Google encoding time faster (as expected)
- Bitrate about the same (no problem with minrate but encoding parameters are not intense)
- VMAF about the same
- Low frame - Baseline better
- Standard deviation - Baseline substantially better

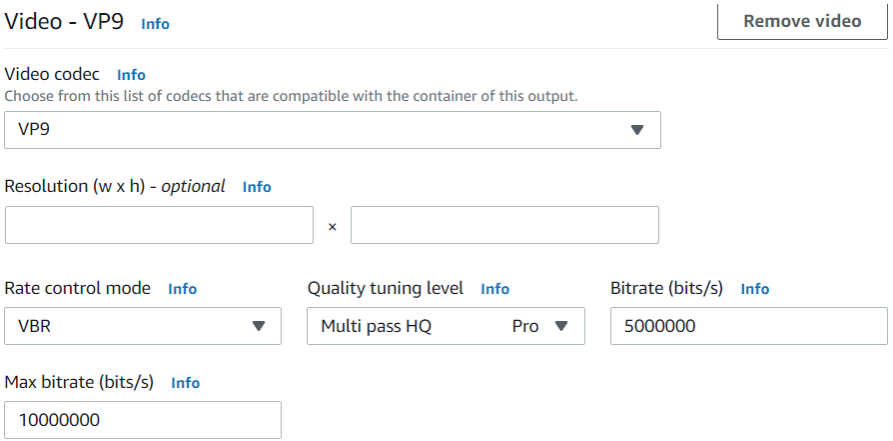
Decision:

- If you go with Google recs
  - Use speed 2 for second pass
  - Add -row-mt to speed encode (should offset)
  - Drop CRF

Encoding Time	Baseline	Google	Delta
Freedom	0:01:40	0:01:06	-34.00%
Football	0:01:27	0:01:13	-16.09%
Average	**0:01:34**	**0:01:10**	**-25.67%**
Bitrate	Baseline	Google	Delta
Freedom	4,809	4,976	3.47%
Football	4,949	4,694	-5.15%
Average	**4,879**	**4,835**	**-0.90%**
VMAF	Baseline	Google	Delta
Freedom	94.62	93.94	-0.71%
Football	96.44	94.29	-2.22%
Average	**95.53**	**94.12**	**-1.47%**
Low Frame	Baseline	Google	Delta
Freedom	88.83	88.05	-0.88%
Football	83.79	72.68	-13.25%
Average	**86.31**	**80.37**	**-6.89%**
Standard Deviation	Baseline	Google	Delta
Freedom	2.56	2.94	14.88%
Football	4.53	6.42	41.66%
Average	**3.55**	**4.68**	**31.99%**

String vs AWS Elemental MediaConvert

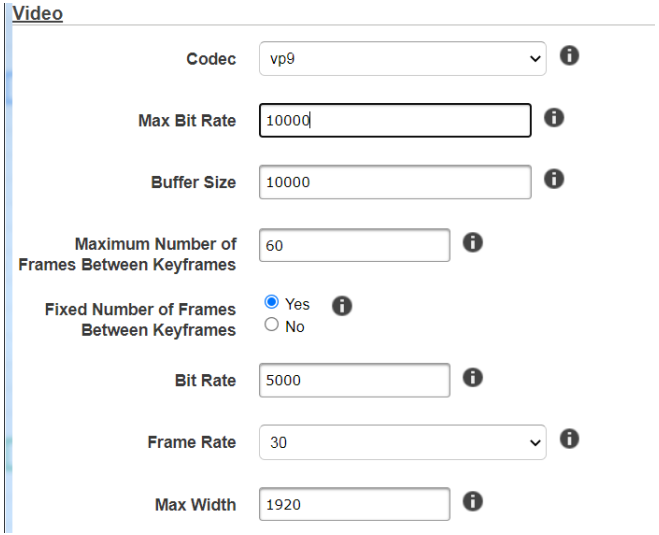
- Used default encoding parameters (not a lot of options - see below)
- Scores very similar



Bitrate	Baseline	AWS	Delta
Freedom	NA	NA	
Football	5006	4920	-1.71%
Average	5006	4920	**-1.71%**
VMAF	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	97.32	96.91	-0.42%
Average	97.32	96.91	**-0.42%**
Low Frame	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	82.77	84.07	1.56%
Average	82.77	84.07	**1.56%**
Standard Deviation	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	3.76	4.30	14.45%
Average	3.76	4.30	**14.45%**

String vs AWS Elastic Transcoder

- Used default encoding parameters (not a lot of options - see below)
- Scores very similar



Bitrate	Baseline	AWS	Delta
Freedom	NA	NA	
Football	4796	4640	-3.25%
Average	4796	4640	**-3.25%**
VMAF	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	95.93	95.80	-0.14%
Average	95.93	95.80	**-0.14%**
Low Frame	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	87.76	78.34	-10.73%
Average	87.76	78.34	**-10.73%**
Standard Deviation	**Baseline**	**AWS**	**Delta**
Freedom	NA	NA	
Football	3.50	3.41	-2.48%
Average	3.50	3.41	**-2.48%**

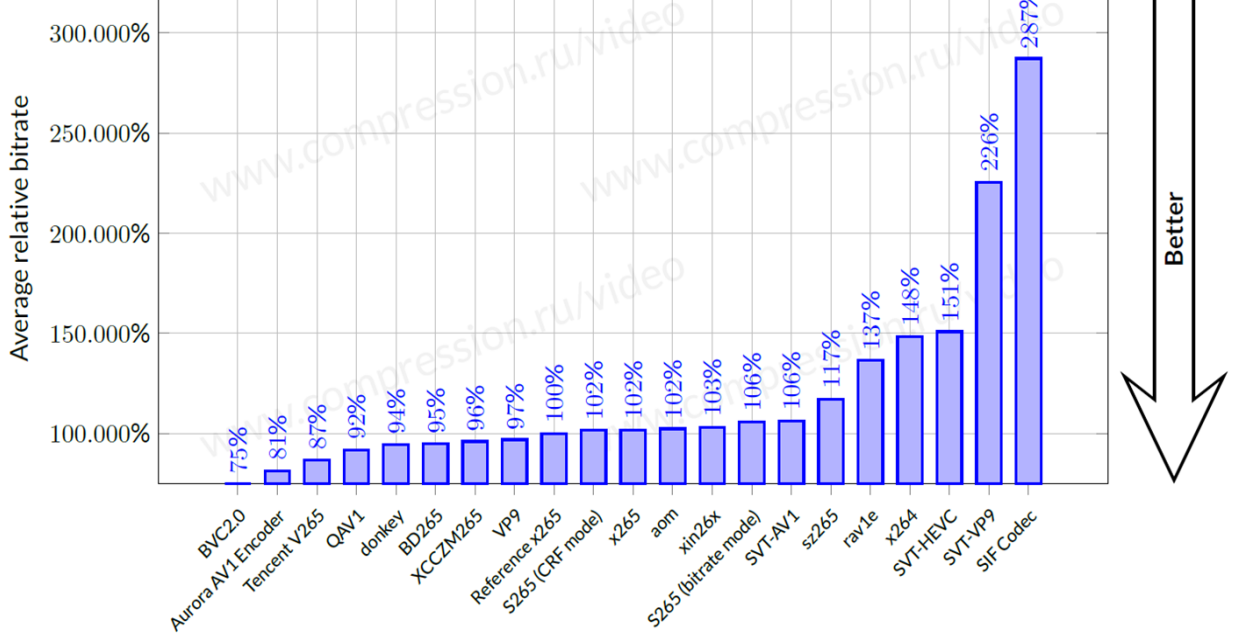
VOD Encoding

- There doesn’t appear to be a magic bullet with libvpx-vp9
  - Assuming that’s what Amazon is using but Elemental is known for creating their own codecs
- Maybe a different implementation like Twin Orioles would deliver a boost

Lesson 3: How VP9 Quality Compares to H.264, HEVC, and AV1

- Overview
- Command strings
- Encoding time
- Quality comparisons
  - Methodology
  - Comparisons
- Conclusion

Overview

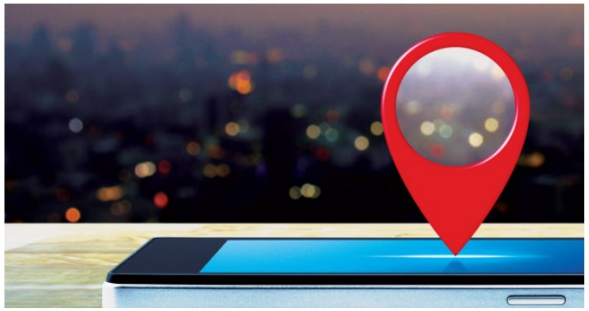


- Standards vs. codecs
  - Standards are specs any company can write to
  - Codecs are specific implementations of the standard
  - I'm testing libvpx-VP9 (in Ffmpeg)
- Codecs used in these tests all from FFmpeg
  - x264 - generally best of breed
  - x265 - middle of the road
  - libaom-av1 behind Aurora1
  - VP9 likely behind other implementations (like https://www.twoorioles.com/)

http://bit.ly/msu_reports_2021

Command Strings - AV1

AV1 Has Arrived: Comparing Codecs from AOMedia, Visionular, and Intel/Netflix



https://bit.ly/av1_comps

```
ffmpeg -y -i %1.mp4 -c:v libaom-av1 -strict -2  -row-mt 1 -tile-columns 1
-an -tile-rows 0 -threads 8 -cpu-used 8 -g 60 -keyint_min 60 -sc_threshold
0  -b:v 6000k -pass 1  -f mp4 NUL && \

ffmpeg    -i %1.mp4 -c:v libaom-av1 -strict -2  -row-mt 1 -tile-columns 1
-an -tile-rows 0 -threads 8 -cpu-used 3 -g 60 -keyint_min 60 -sc_threshold
0  -b:v 6000k -maxrate 12000k -bufsize 12000k -pass 2  %1_1080p_6000.mkv
```

Command Strings - x264/x265

```
ffmpeg -y -i input.mp4 -c:v libx264 -preset veryslow -threads 8 -tune psnr -g 60 -
keyint_min 60 -sc_threshold 0  -b:v 4900k -pass 1  -f mp4 NUL && \

ffmpeg    -i input.mp4 -c:v libx264  -preset veryslow -threads 8-tune psnr -g 60 -
keyint_min 60 -sc_threshold 0  -b:v 6000k -maxrate 12000k -bufsize 12000k -pass 2
output_1080p_6000.mp4
```

```
ffmpeg -y -i input.mp4 -c:v libx265 -tune ssim  -preset veryslow  -x265-params
keyint=60:min-keyint=60:scenecut=0:bitrate=2400k:vbv-maxrate=4800k:vbv-
bufsize=4800k:pass=1  -f mp4 NUL && \

ffmpeg    -i input.mp4 -c:v libx265 -tune ssim  -preset veryslow -x265-params
keyint=60:min-keyint=60:scenecut=0:bitrate=5000k:vbv-maxrate=10000k:vbv-
bufsize=10000k:pass=2  output_1080p_5000_x265_vslow.mp4
```

Encoding Time

	I7-6700HQ 2.60Gh	E3-1505M v5 2.8 Ghz Xeon	Dual E5-2687 3.1 Xeon	Average	Multiple x264
x264	0:01:07	0:00:38	0:01:04	0:00:56	
x265	0:12:14	0:06:19	0:15:41	0:11:25	12.15
VP9	0:01:54	0:01:32	0:02:25	0:01:57	2.08
AV1	0:13:46	0:13:04	0:23:36	0:16:49	17.91

- x265 looks very high, but ... ffmpeg-2021-06-02-git-071930de72-full_build.7z

x265 Encoding Speed Over Time

	8/31/18[1]	9/18/2020 [2]	Today
x265 veryslow	289	4:57	0:56
x264 veryslow	18	0:23	11:25
Multiple	16.05	12.87x	11.80x

- x265 encoding time has been relatively consistent
- Issue is VP9 encoding parameters
  - Usually tested at speed 0 (which adds 1 VMAF point but triples encoding time)
  - We tested at speed 2

- General perception that VP9 is very slow to encode
- Depends upon:
  - Settings used
  - What you're comparing it to

1 http://bit.ly/av1_firstlook

2 https://bit.ly/av1_comps

Overview

- Goal: Identify realistic bandwidth savings and quality improvement for each clip
- Procedure (Netflix Convex Hull analysis)
  - Encode clips (ten seconds) into multiple data rates and resolutions
  - Organize as shown; clip in green is highest quality resolution at that data rate
  - Start encoding ladder at first bitrate that exceeds 95 VMAF; choose highest quality resolution
  - Multiply data rate by .6 to identify next rung; choose highest quality resolution
  - Repeat until data rate at 300 kbps or lower

x264	1080p	720p	540p	360p	270p	180p	Max
6000	95.54						95.54
5800	95.31						95.31
5600	95.03						95.03
5400	94.61						94.61
5200	94.22						94.22
5000	93.89						93.89
4800	93.45						93.45
4600	92.93						92.93
4400	92.47						92.47
4200	91.95						91.95
4000	91.26	90.10					91.26
3800	90.66	89.68					90.66
3600	89.86	89.13					89.86
3400	89.07	88.56					89.07
3200	88.17	88.04					88.17
3000	87.07	87.37					87.37
2800	85.95	86.54					86.54
2600	84.66	85.67					85.67
2400	82.99	84.49	81.04				84.49
2200	81.41	83.32	79.98				83.32
2000	79.29	81.91	78.66				81.91
1800	76.70	80.07	77.12				80.07
1600	73.87	77.91	75.29	63.78			77.91
1400	70.08	75.33	73.00	61.98			75.33
1200	65.41	72.07	70.19	59.60			72.07
1000	59.30	67.62	66.52	56.73			67.62
900	55.78	64.92	64.20	54.87	38.69	11.45	64.92
800	51.27	61.76	61.44	52.73	37.03	10.64	61.76
700	46.10	57.93	58.30	50.15	35.23	9.50	58.30
600	39.10	53.54	54.23	47.11	32.94	8.15	54.23
500	29.52	47.91	49.48	43.51	30.17	6.41	49.48
400		40.94	43.33	38.83	26.68		43.33
300		30.92	35.39	32.60	21.93		35.39
200			22.37	23.33	15.12		23.33

Overview

- Then, repeat with each codec to produce optimal ladder for each codec
- Then, rinse and repeat with other codecs

x264	1080p	720p	540p	360p	270p	180p	Max
6000	95.54						95.54
5800	95.31						95.31
5600	95.03						95.03
5400	94.61						94.61
5200	94.22						94.22
5000	93.89						93.89
4800	93.45						93.45
4600	92.93						92.93
4400	92.47						92.47
4200	91.95						91.95
4000	91.26	90.10					91.26
3800	90.66	89.68					90.66
3600	89.86	89.13					89.86
3400	89.07	88.56					89.07
3200	88.17	88.04					88.17
3000	87.07	87.37					87.37
2800	85.95	86.54					86.54
2600	84.66	85.67					85.67
2400	82.99	84.49	81.04				84.49
2200	81.41	83.32	79.98				83.32
2000	79.29	81.91	78.66				81.91
1800	76.70	80.07	77.12				80.07
1600	73.87	77.91	75.29	63.78			77.91
1400	70.08	75.33	73.00	61.98			75.33
1200	65.41	72.07	70.19	59.60			72.07
1000	59.30	67.62	66.52	56.73			67.62
900	55.78	64.92	64.20	54.87	38.69	11.45	64.92
800	51.27	61.76	61.44	52.73	37.03	10.64	61.76
700	46.10	57.93	58.30	50.15	35.23	9.50	58.30
600	39.10	53.54	54.23	47.11	32.94	8.15	54.23
500	29.52	47.91	49.48	43.51	30.17	6.41	49.48
400		40.94	43.33	38.83	26.68		43.33
300		30.92	35.39	32.60	21.93		35.39
200			22.37	23.33	15.12		23.33

x264	1080p	720p	540p	360p	270p	180p	Max
6000	95.54						95.54
5800	95.31						95.31
5600	95.03						95.03
5400	94.61						94.61
5200	94.22						94.22
5000	93.89						93.89
4800	93.45						93.45
4600	92.93						92.93
4400	92.47						92.47
4200	91.95						91.95
4000	91.26	90.10					91.26
3800	90.66	89.68					90.66
3600	89.86	89.13					89.86
3400	89.07	88.56					89.07
3200	88.17	88.04					88.17
3000	87.07	87.37					87.37
2800	85.95	86.54					86.54
2600	84.66	85.67					85.67
2400	82.99	84.49	81.04				84.49
2200	81.41	83.32	79.98				83.32
2000	79.29	81.91	78.66				81.91
1800	76.70	80.07	77.12				80.07
1600	73.87	77.91	75.29	63.78			77.91
1400	70.08	75.33	73.00	61.98			75.33
1200	65.41	72.07	70.19	59.60			72.07
1000	59.30	67.62	66.52	56.73			67.62
900	55.78	64.92	64.20	54.87	38.69	11.45	64.92
800	51.27	61.76	61.44	52.73	37.03	10.64	61.76
700	46.10	57.93	58.30	50.15	35.23	9.50	58.30
600	39.10	53.54	54.23	47.11	32.94	8.15	54.23
500	29.52	47.91	49.48	43.51	30.17	6.41	49.48
400		40.94	43.33	38.83	26.68		43.33
300		30.92	35.39	32.60	21.93		35.39
200			22.37	23.33	15.12		23.33

x264

x265	1080p	720p	540p	360p	270p	180p	Max
4800	95.57						95.57
4600	95.27						95.27
4400	94.90						94.90
4200	94.55						94.55
4000	94.12						94.12
3800	93.67						93.67
3600	93.21						93.21
3400	92.53						92.53
3200	92.01						92.01
3000	91.17						91.17
2800	90.48						90.48
2600	89.47	87.31					89.47
2400	88.48	86.53					88.48
2200	87.20	85.51					87.20
2000	85.78	84.28					85.78
1800	84.05	82.87					84.05
1600	81.89	81.09	77.62				81.89
1400	79.20	79.11	75.72				79.20
1200	75.66	76.37	73.10				76.37
1000	70.90	72.78	70.05				72.78
900	67.90	70.44	67.91				70.44
800	64.09	67.58	65.51				67.58
700	59.57	64.18	62.61	52.33			64.18
600	53.72	59.65	58.74	49.32			59.65
500	45.54	53.93	54.05	45.53			54.05
400	35.2	46.10	47.45	40.33	31.50	4.97	47.45
300	28.4	34.34	37.81	33.12	27.80		37.81
200		15.46	17.83	20.67	22.69		22.69
100		11.95			14.55		14.55

x265

x264	1080p	720p	540p	360p	270p	180p	Max
6000	95.54						95.54
5800	95.31						95.31
5600	95.03						95.03
5400	94.61						94.61
5200	94.22						94.22
5000	93.89						93.89
4800	93.45						93.45
4600	92.93						92.93
4400	92.47						92.47
4200	91.95						91.95
4000	91.26	90.10					91.26
3800	90.66	89.68					90.66
3600	89.86	89.13					89.86
3400	89.07	88.56					89.07
3200	88.17	88.04					88.17
3000	87.07	87.37					87.37
2800	85.95	86.54					86.54
2600	84.66	85.67					85.67
2400	82.99	84.49	81.04				84.49
2200	81.41	83.32	79.98				83.32
2000	79.29	81.91	78.66				81.91
1800	76.70	80.07	77.12				80.07
1600	73.87	77.91	75.29	63.78			77.91
1400	70.08	75.33	73.00	61.98			75.33
1200	65.41	72.07	70.19	59.60			72.07
1000	59.30	67.62	66.52	56.73			67.62
900	55.78	64.92	64.20	54.87	38.69	11.45	64.92
800	51.27	61.76	61.44	52.73	37.03	10.64	61.76
700	46.10	57.93	58.30	50.15	35.23	9.50	58.30
600	39.10	53.54	54.23	47.11	32.94	8.15	54.23
500	29.52	47.91	49.48	43.51	30.17	6.41	49.48
400		40.94	43.33	38.83	26.68		43.33
300		30.92	35.39	32.60	21.93		35.39
200			22.37	23.33	15.12		23.33

x264

VP9	1080p	720p	540p	360p	270p	180p	Max
5600	95.53						95.53
5400	95.24						95.24
5200	94.88						94.88
5000	94.55						94.55
4800	94.20						94.20
4600	93.81						93.81
4400	93.37						93.37
4200	92.90						92.90
4000	92.40						92.40
3800	91.87						91.87
3600	91.27						91.27
3400	90.68						90.68
3200	89.92	87.83					89.92
3000	89.11	87.07					89.11
2800	88.29	86.29					88.29
2600	87.39	85.37					87.39
2400	86.32	84.37	80.55				86.32
2200	85.07	83.19	79.50				85.07
2000	83.72	81.81	78.21				83.72
1800	82.11	80.28	76.75				82.11
1600	80.31	78.50	75.00	63.31			80.31
1400	78.07	76.40	72.88	61.45			78.07
1200	75.26	73.81	70.34	59.16			75.26
1000	71.81	70.65	67.18	56.19			71.81
900	69.78	68.74	65.22	54.53	38.14	11.43	69.78
800	67.90	66.41	63.09	52.57	36.57	10.55	67.90
700	65.02	63.82	60.53	50.23	34.68	9.40	65.02
600	61.69	60.61	57.63	47.59	32.45	8.11	61.69
500	57.11	56.89	53.91	44.37	29.82	6.18	57.11
400	53.18	52.10	49.51	40.36	26.55		53.18
300	45.87	45.88	43.48	35.26	22.22		45.88
200			35.11	27.73	16.51		35.11
100			18.39	15.64	6.30		18.39

VP9

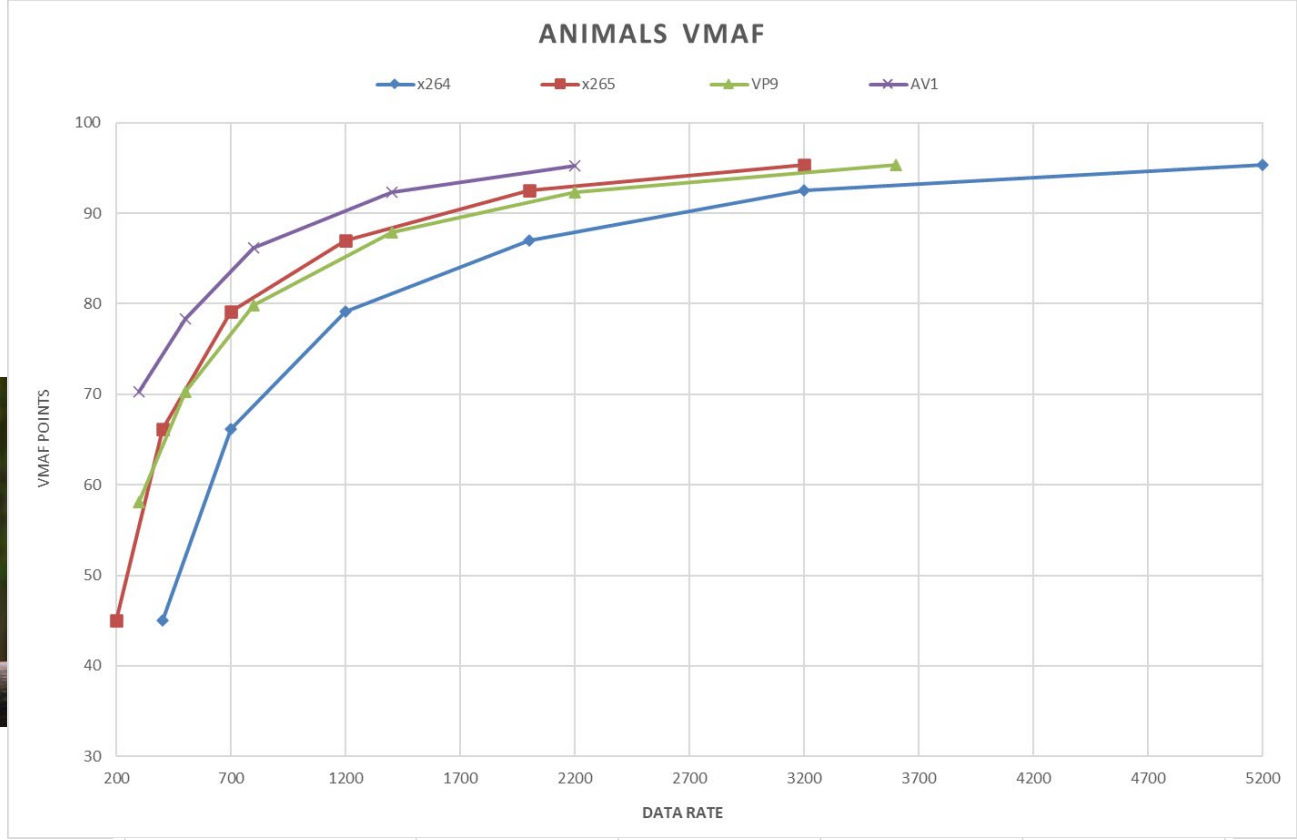
x264	1080p	720p	540p	360p	270p	180p	Max
6000	95.54						95.54
5800	95.31						95.31
5600	95.03						95.03
5400	94.61						94.61
5200	94.22						94.22
5000	93.89						93.89
4800	93.45						93.45
4600	92.93						92.93
4400	92.47						92.47
4200	91.95						91.95
4000	91.26	90.10					91.26
3800	90.66	89.68					90.66
3600	89.86	89.13					89.86
3400	89.07	88.56					89.07
3200	88.17	88.04					88.17
3000	87.07	87.37					87.37
2800	85.95	86.54					86.54
2600	84.66	85.67					85.67
2400	82.99	84.49	81.04				84.49
2200	81.41	83.32	79.98				83.32
2000	79.29	81.91	78.66				81.91
1800	76.70	80.07	77.12				80.07
1600	73.87	77.91	75.29	63.78			77.91
1400	70.08	75.33	73.00	61.98			75.33
1200	65.41	72.07	70.19	59.60			72.07
1000	59.30	67.62	66.52	56.73			67.62
900	55.78	64.92	64.20	54.87	38.69	11.45	64.92
800	51.27	61.76	61.44	52.73	37.03	10.64	61.76
700	46.10	57.93	58.30	50.15	35.23	9.50	58.30
600	39.10	53.54	54.23	47.11	32.94	8.15	54.23
500	29.52	47.91	49.48	43.51	30.17	6.41	49.48
400		40.94	43.33	38.83	26.68		43.33
300		30.92	35.39	32.60	21.93		35.39
200			22.37	23.33	15.12		23.33

x264

AV1	1080p	720p	540p	360p	270p	180p	Max
5000	96.89						96.89
4800	96.65						96.65
4600	96.40						96.40
4400	96.11						96.11
4200	95.79						95.79
4000	95.45						95.45
3800	95.07						95.07
3600	94.67						94.67
3400	94.20						94.20
3200	93.67	90.67					93.67
3000	93.12	90.19					93.12
2800	92.43	89.60					92.43
2600	91.78	89.05					91.78
2400	90.99	88.33					90.99
2200	90.06	87.53					90.06
2000	88.95	86.55					88.95
1800	87.65	85.36					87.65
1600	86.12	83.99		66.34			86.12
1400	84.31	82.27		65.00			84.31
1200	82.01	80.09		63.36			82.01
1000	79.09	77.28		61.22			79.09
900	77.25	75.60		59.75	41.34	12.80	77.25
800	75.05	73.50		58.30	40.15	12.11	75.05
700	72.48	71.17		56.31	38.68	11.30	72.48
600	69.41	68.18		53.92	36.90	10.40	69.41
500	66.09	64.48		50.90	34.57	9.10	66.09
400		59.65	57.70	47.18	31.67	7.47	59.65
300		53.98	51.88	42.12	27.68	5.19	53.98
200			43.20	34.33	22.11		43.20
100				24.32	10.90		24.32

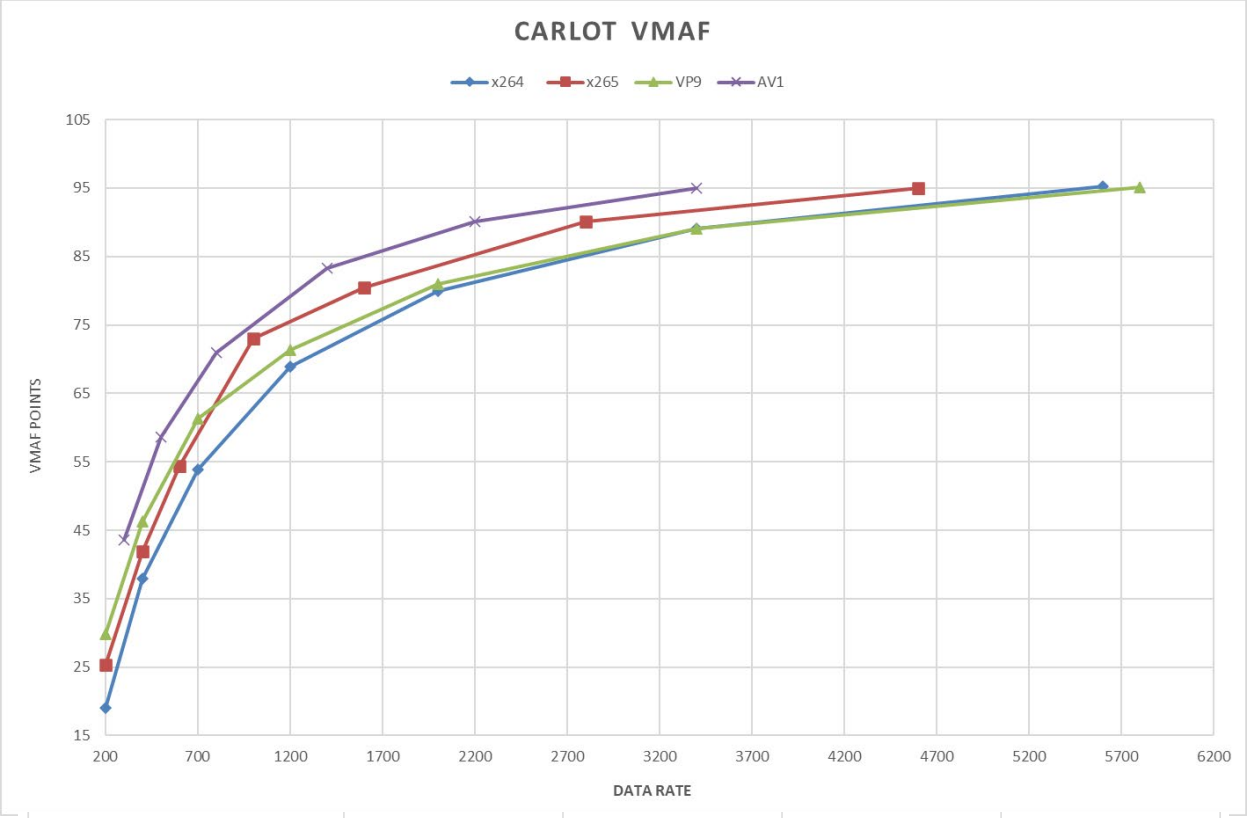
AV1

Analysis - Animals



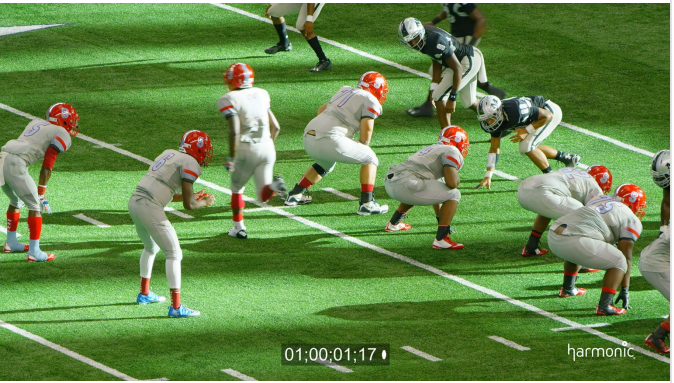
Overall - Sub	x264	x265	VP9	AV1
x264	X	29.66%	57.05%	142.65%
x265	-22.88%	X	18.77%	81.26%
VP9	-36.33%	-15.80%	X	53.31%
AV1	-58.79%	-44.83%	-34.77%	X

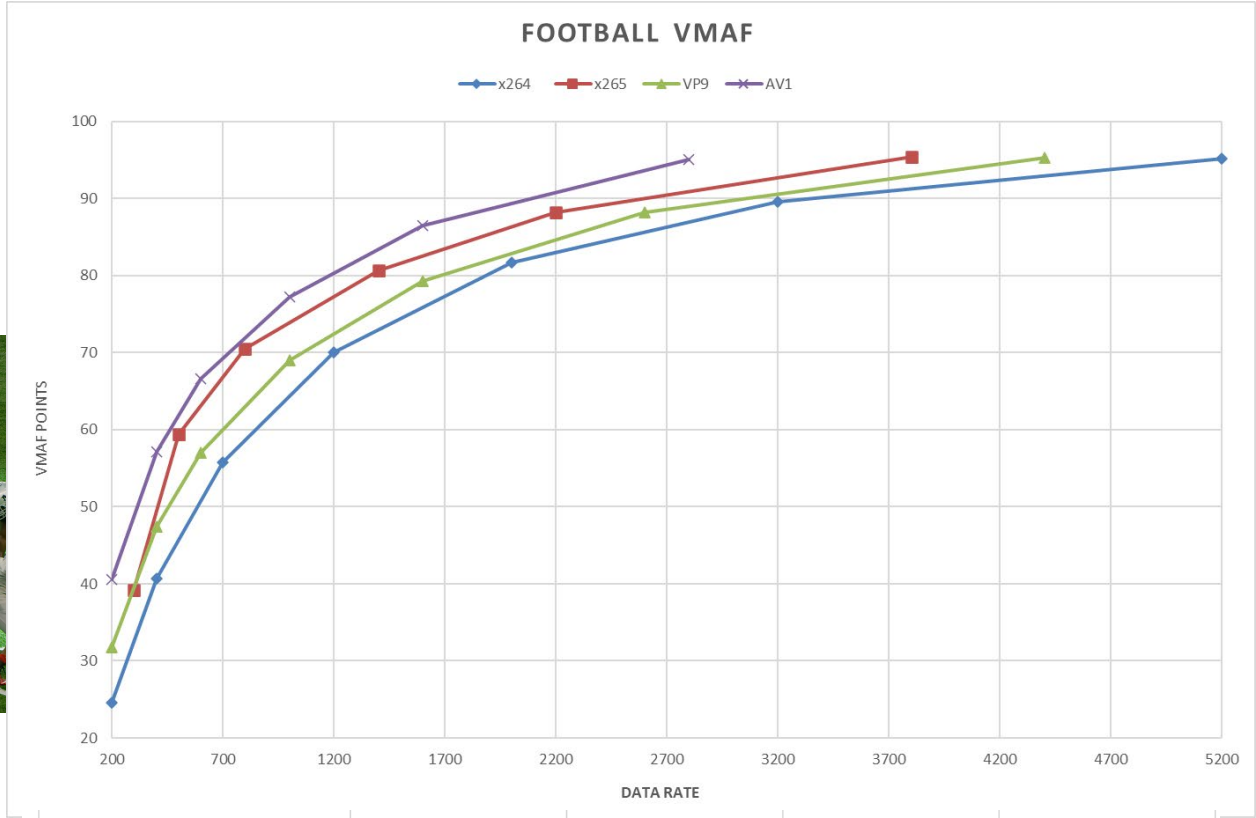
Analysis - Carlot



Carlot	x264	x265	VP9	AV1
x264	X	25.40%	21.39%	65.89%
x265	-20.26%	X	-3.25%	179.19%
VP9	-17.62%	3.36%	X	42.60%
AV1	-39.72%	-64.18%	-29.87%	X

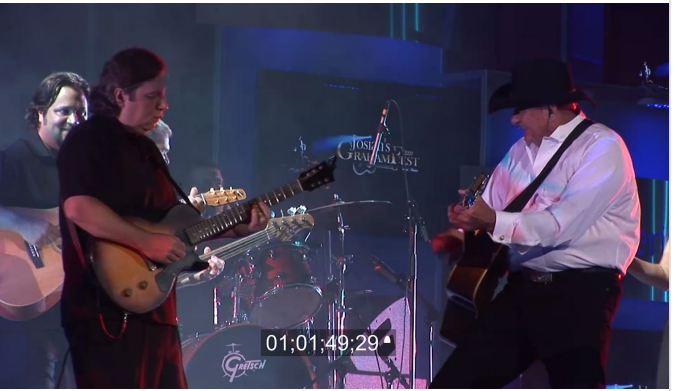
Analysis - Football

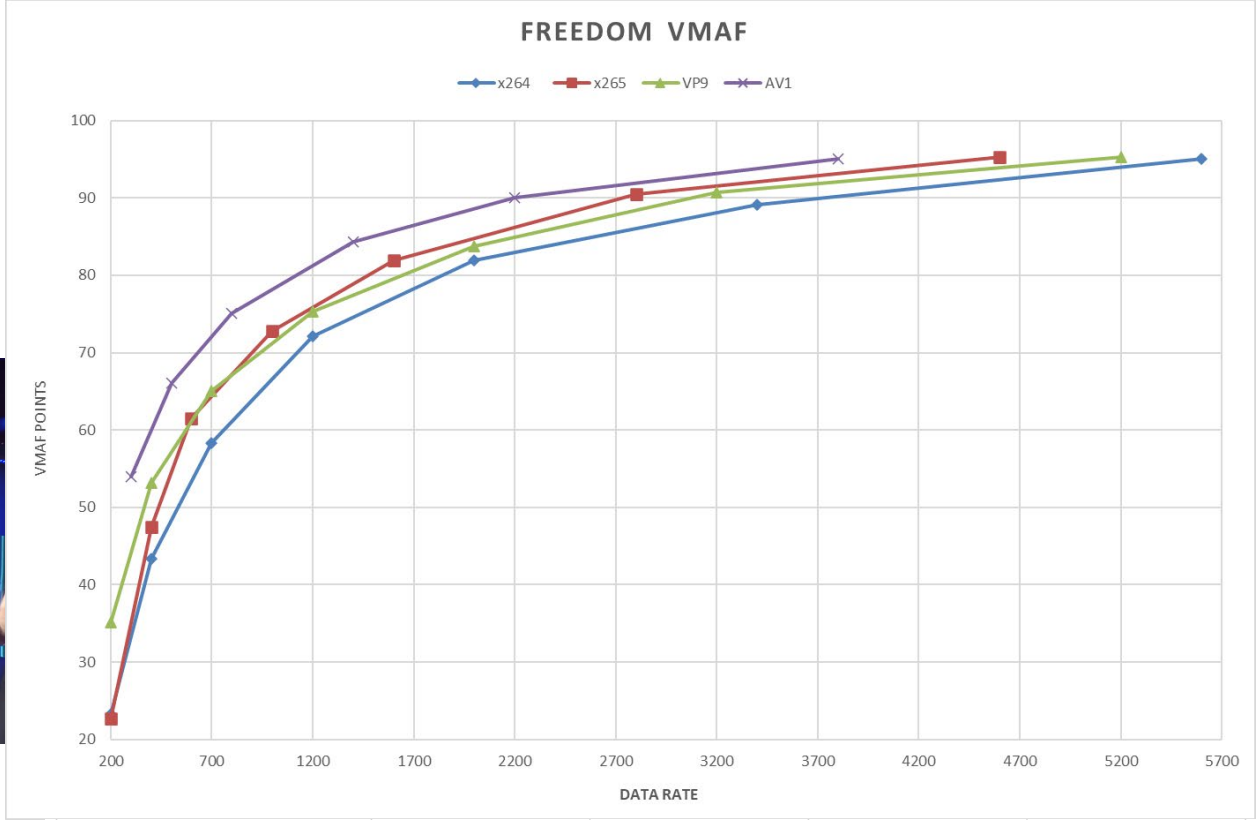




Football	x264	x265	VP9	AV1
x264	X	45.24%	21.09%	69.69%
x265	-31.15%	X	-17.97%	180.61%
VP9	-14.49%	25.39%	X	200.82%
AV1	-41.07%	-14.53%	-32.36%	X

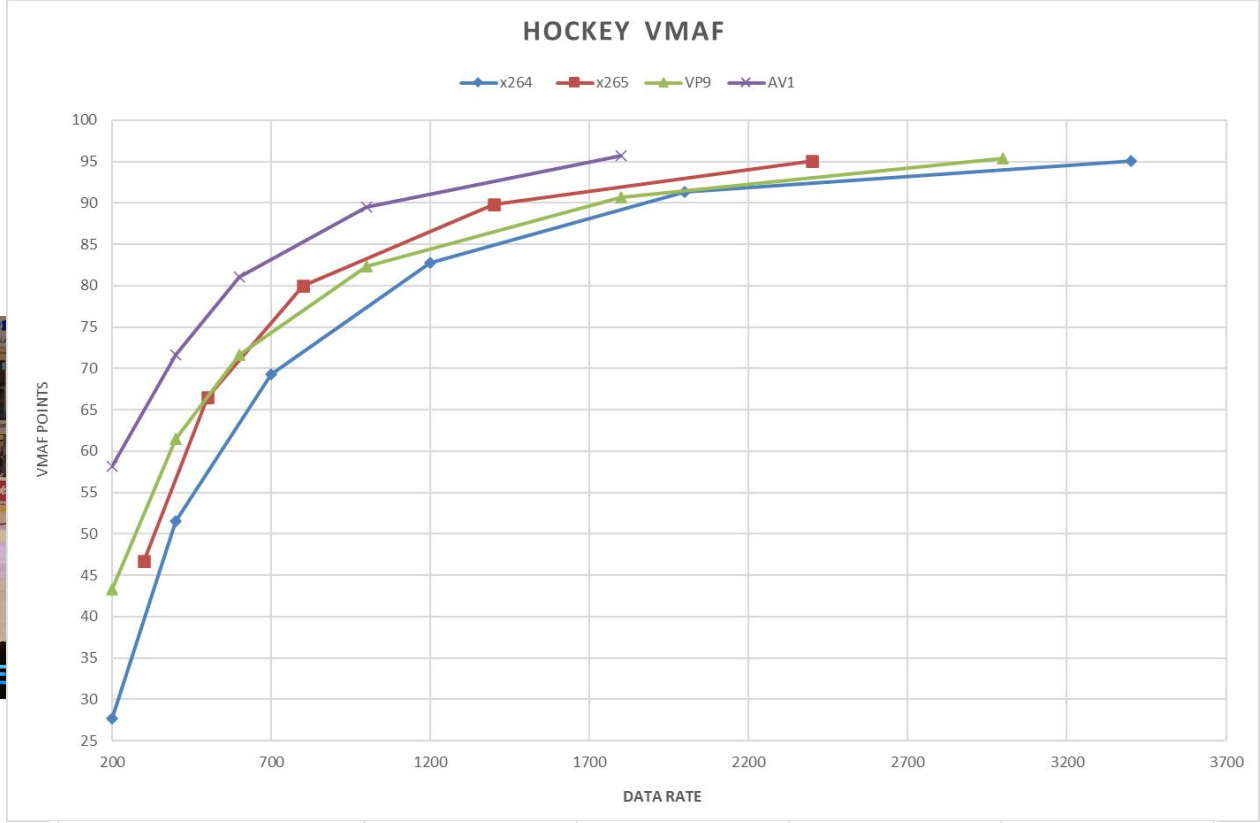
Analysis - Freedom





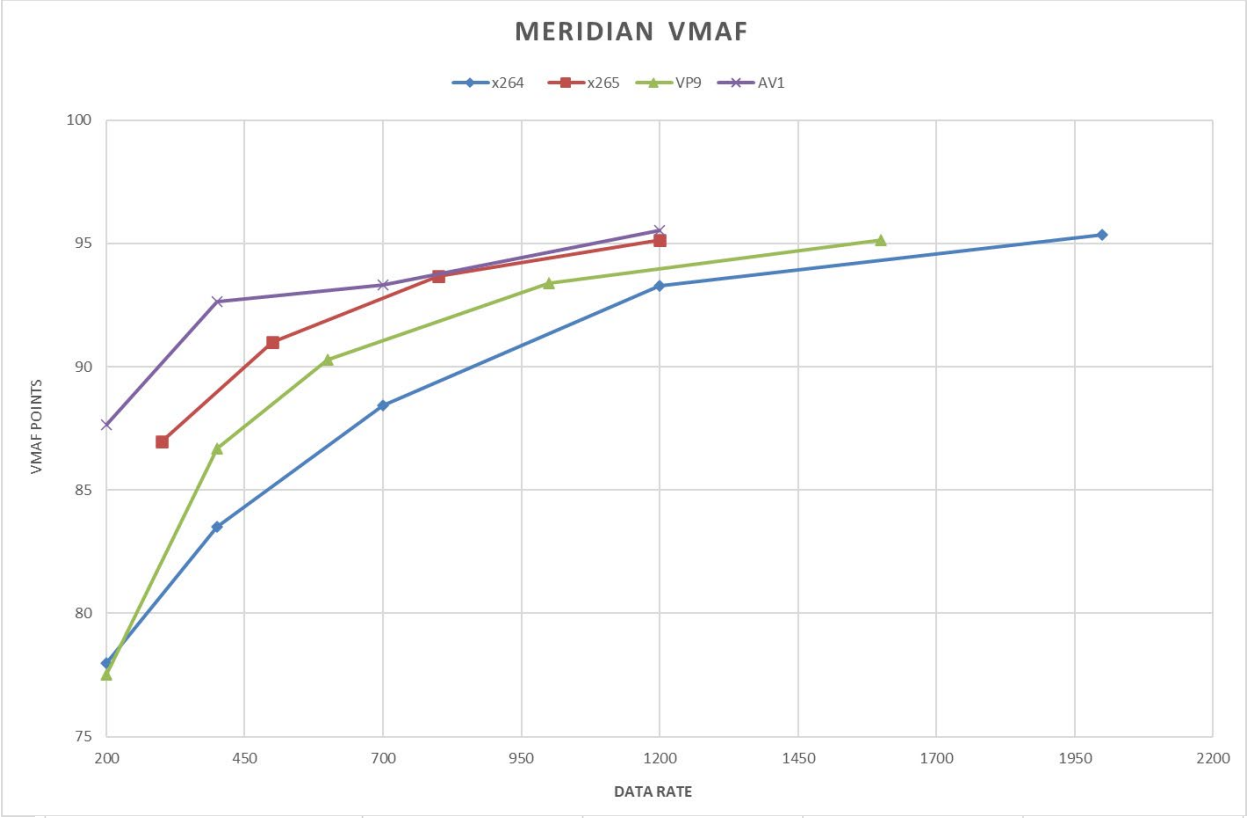
Freedom	x264	x265	VP9	AV1
x264	X	19.65%	29.99%	77.23%
x265	-16.43%	X	5.19%	38.24%
VP9	-23.07%	-4.93%	X	91.01%
AV1	-43.58%	-45.20%	-30.89%	X

Analysis - Hockey



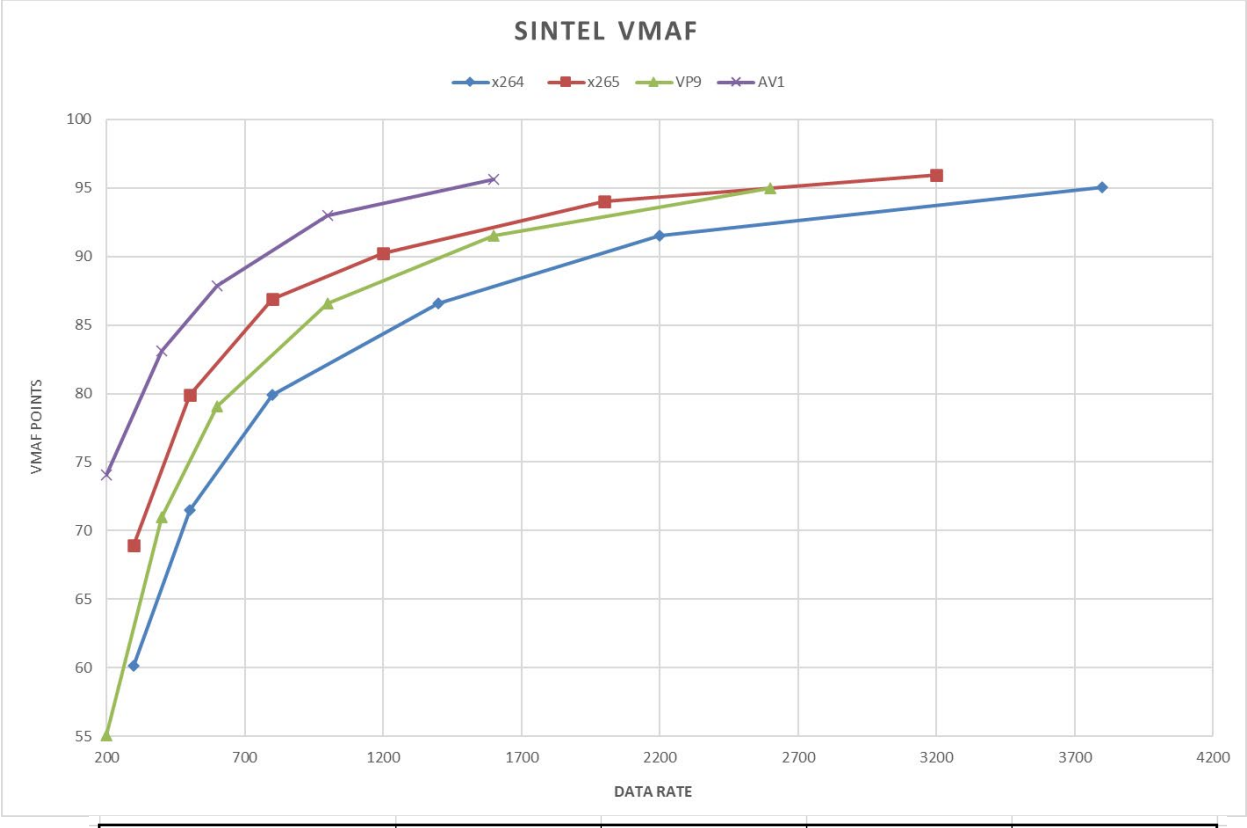
Hockey	x264	x265	VP9	AV1
x264	X	27.12%	28.56%	92.51%
x265	-21.34%	X	-0.19%	48.94%
VP9	-22.22%	0.19%	X	58.45%
AV1	-48.05%	-32.86%	-36.89%	X

Analysis - Meridian



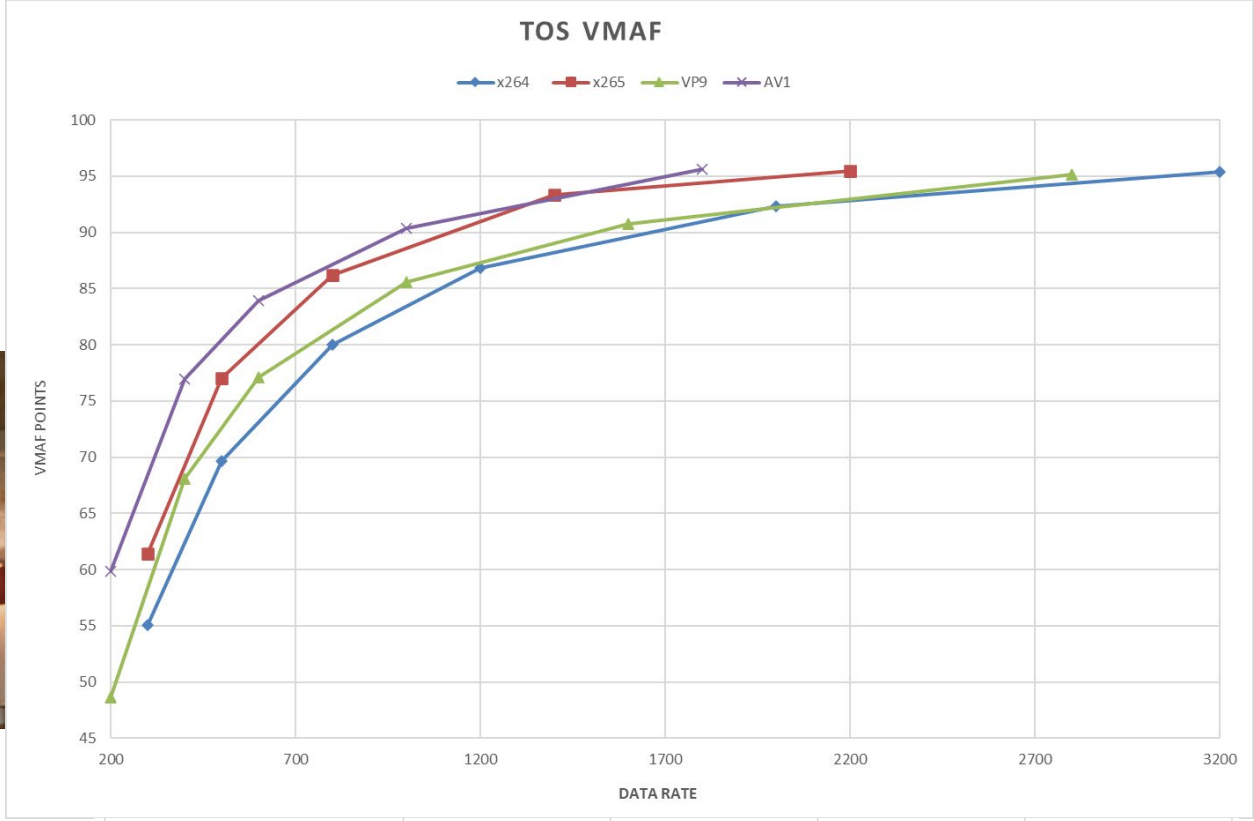
Meridian	x264	x265	VP9	AV1
x264	X	79.14%	29.00%	162.18%
x265	-44.18%	X	-25.08%	49.62%
VP9	-22.48%	33.48%	X	99.15%
AV1	-61.86%	-33.16%	-49.79%	X

Analysis - Sintel



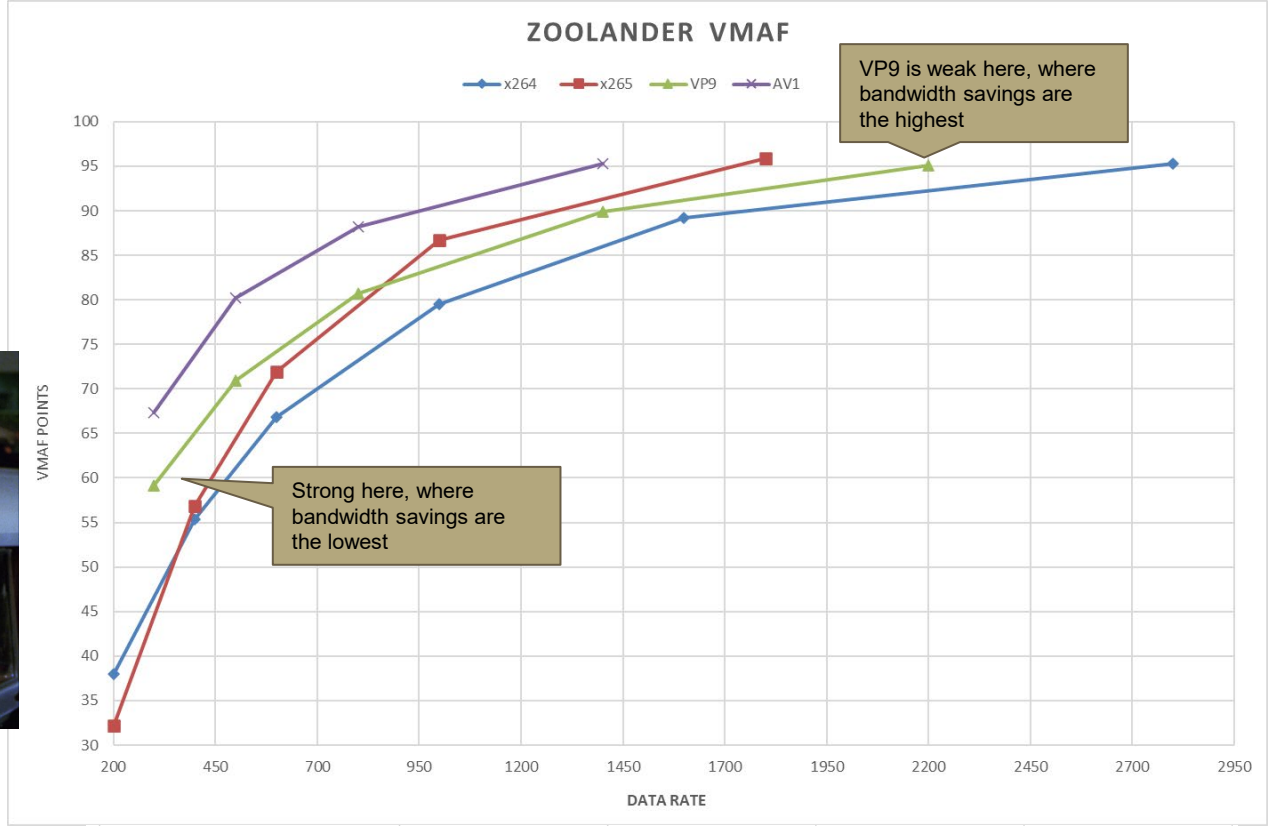
Sintel	x264	x265	VP9	AV1
x264	X	60.85%	28.34%	163.20%
x265	-37.83%	X	-18.68%	61.63%
VP9	-22.08%	22.97%	X	97.82%
AV1	-62.01%	-38.13%	-49.45%	X

Analysis - Tears of Steel



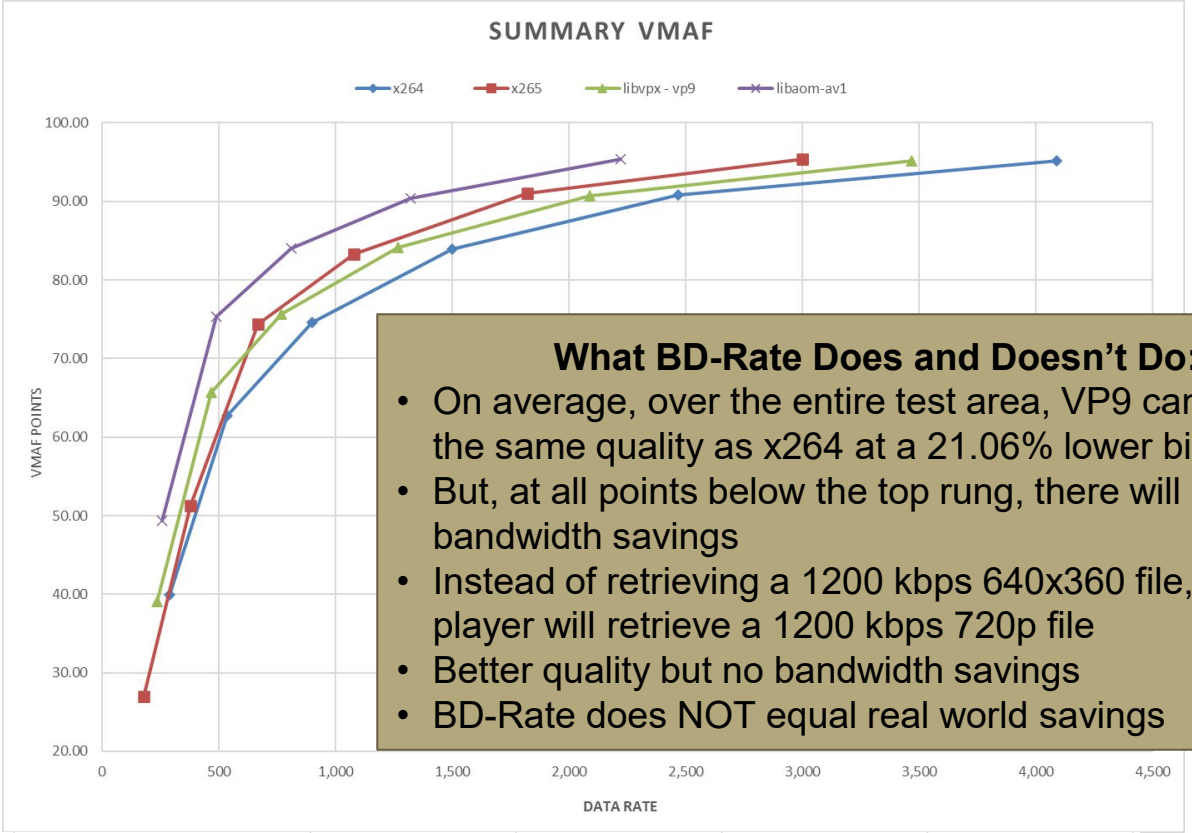
Tears of Steel	x264	x265	VP9	AV1
x264	X	39.32%	14.29%	70.45%
x265	-28.22%	X	-18.74%	22.21%
VP9	-12.50%	23.05%	X	50.20%
AV1	-41.33%	-18.17%	-33.42%	X

Analysis - Zoolander



Zoolander	x264	x265	VP9	AV1
x264	X	15.27%	34.13%	134.55%
x265	-13.25%	X	4.92%	50.18%
VP9	-25.45%	-4.69%	X	53.58%
AV1	-50.04%	-33.41%	-34.89%	X

Summary

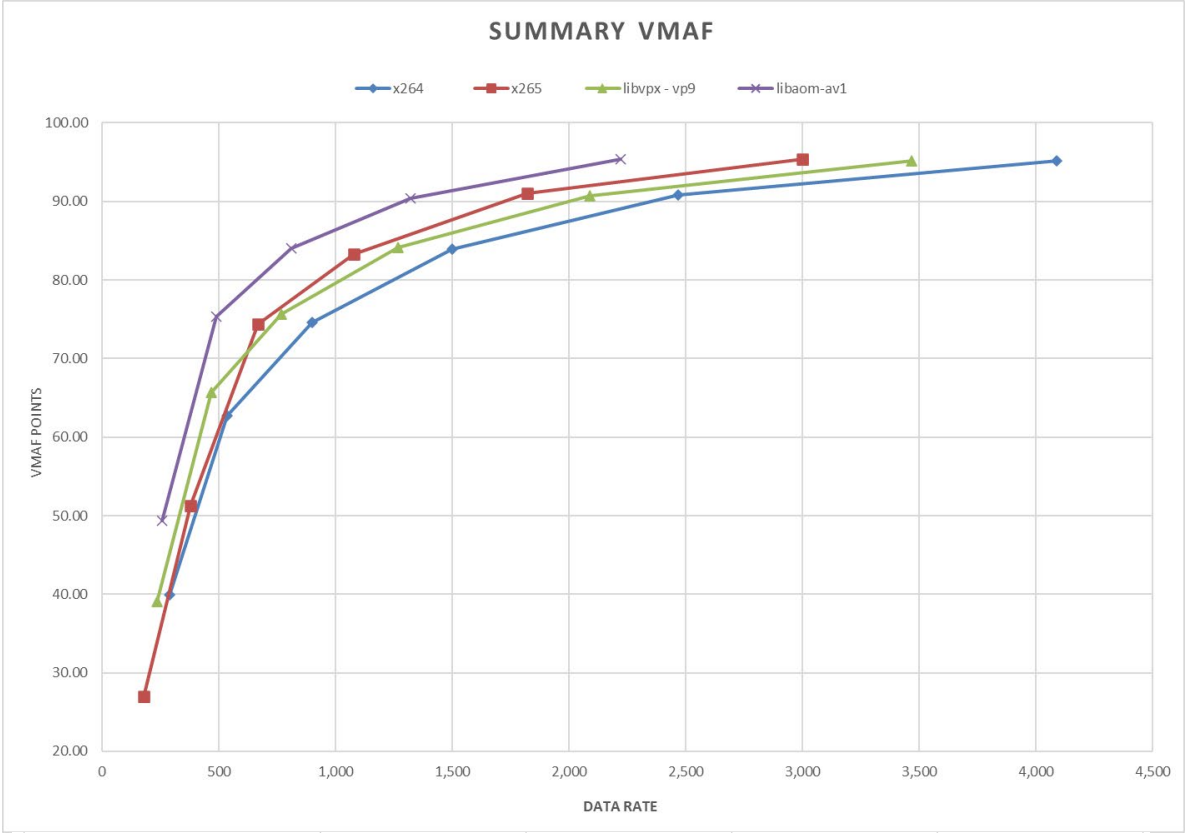


What BD-Rate Does and Doesn't Do:

- On average, over the entire test area, VP9 can produce the same quality as x264 at a 21.06% lower bitrate
- But, at all points below the top rung, there will be no bandwidth savings
- Instead of retrieving a 1200 kbps 640x360 file, the player will retrieve a 1200 kbps 720p file
- Better quality but no bandwidth savings
- BD-Rate does NOT equal real world savings

Summary	x264	x265	libvpx - vp9	libaom-av1
x264	X	40.13%	27.66%	106.15%
x265	-27.29%	X	-8.99%	74.46%
libvpx - vp9	-21.06%	11.83%	X	82.99%
libaom-av1	-49.12%	-34.22%	-36.92%	X

Summary



Summary	x264	x265	libvpx - vp9	libaom-av1
x264	X	40.13%	27.66%	106.15%
x265	-27.29%	X	-8.99%	74.46%
libvpx - vp9	-21.06%	11.83%	X	82.99%
libaom-av1	-49.12%	-34.22%	-36.92%	X

What's the Net/Net? Depends Upon Video Distribution - VP9

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	50.00%	2,045	47.61
2,467	90.79	25.00%	617	22.70
1,500	83.93	10.00%	150	8.39
900	74.61	8.00%	72	5.97
533	62.74	5.00%	27	3.14
289	39.89	2.00%	6	0.80
		100.00%	**2,916**	**88.60**

Bitrate	VMAF	%	Bitrate	VMAF
3,467	95.19	55.83%	1,935	53.14
2,089	90.73	21.58%	451	19.58
1,267	84.17	9.36%	119	7.88
767	75.66	7.13%	55	5.39
467	65.66	4.56%	21	2.99
233	39.12	1.54%	4	0.60
		100.00%	**2,585**	**89.59**
			331	0.99
			11.36%	**0.99**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	100.00%	4,089	95.22
2,467	90.79		0	0.00
1,500	83.93		0	0.00
900	74.61		0	0.00
533	62.74		0	0.00
289	39.89		0	0.00
		100.00%	**4,089**	**95.22**

Bitrate	VMAF	%	Bitrate	VMAF
3,467	95.19	100.00%	3,467	95.19
2,089	90.73	0.00%	0	0.00
1,267	84.17	0.00%	0	0.00
767	75.66	0.00%	0	0.00
467	65.66	0.00%	0	0.00
233	39.12	0.00%	0	0.00
		100.00%	**3,467**	**95.19**
			622	-0.03
			15.21%	**-0.03**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	15.00%	613	14.28
2,467	90.79	20.00%	493	18.16
1,500	83.93	40.00%	600	33.57
900	74.61	15.00%	135	11.19
533	62.74	6.00%	32	3.76
289	39.89	4.00%	12	1.60
		100.00%	**1,885**	**82.56**

Bitrate	VMAF	%	Bitrate	VMAF
3,467	95.19	19.66%	682	18.72
2,089	90.73	24.98%	522	22.66
1,267	84.17	33.69%	427	28.35
767	75.66	12.75%	98	9.65
467	65.66	5.84%	27	3.83
233	39.12	3.08%	7	1.21
		100.00%	**1,763**	**84.42**
			123	1.86
			6.51%	**1.86**

Bandwidth

VMAF Delta

What's the Net/Net? Depends Upon Video Distribution - HEVC

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	50.00%	2,045	47.61
2,467	90.79	25.00%	617	22.70
1,500	83.93	10.00%	150	8.39
900	74.61	8.00%	72	5.97
533	62.74	5.00%	27	3.14
289	39.89	2.00%	6	0.80
		100.00%	**2,916**	**88.60**
Bitrate	**VMAF**	**%**	**Bitrate**	**VMAF**
3,000	95.38	59.94%	1,798	57.17
1,822	90.99	19.42%	354	17.67
1,078	83.27	8.74%	94	7.28
667	74.41	7.01%	47	5.21
378	51.30	3.80%	14	1.95
178	26.95	1.09%	2	0.29
		100.00%	**2,309**	**89.58**
			606	0.98
			20.79%	**0.98**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	100.00%	4,089	95.22
2,467	90.79		0	0.00
1,500	83.93		0	0.00
900	74.61		0	0.00
533	62.74		0	0.00
289	39.89		0	0.00
		100.00%	**4,089**	**95.22**
Bitrate	**VMAF**	**%**	**Bitrate**	**VMAF**
3,000	95.38	100.00%	3,000	95.38
1,822	90.99	0.00%	0	0.00
1,078	83.27	0.00%	0	0.00
667	74.41	0.00%	0	0.00
378	51.30	0.00%	0	0.00
178	26.95	0.00%	0	0.00
		100.00%	**3,000**	**95.38**
			1,089	0.16
			26.63%	**0.16**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,089	95.22	15.00%	613	14.28
2,467	90.79	20.00%	493	18.16
1,500	83.93	40.00%	600	33.57
900	74.61	15.00%	135	11.19
533	62.74	6.00%	32	3.76
289	39.89	4.00%	12	1.60
		100.00%	**1,885**	**82.56**
Bitrate	**VMAF**	**%**	**Bitrate**	**VMAF**
3,000	95.38	19.66%	590	18.75
1,822	90.99	24.98%	455	22.73
1,078	83.27	33.69%	363	28.05
667	74.41	12.75%	85	9.49
378	51.30	5.84%	22	3.00
178	26.95	3.08%	5	0.83
		100.00%	**1,521**	**82.85**
			365	0.28
			19.34%	**0.28**

Bandwidth

VMAF Delta

What's the Net/Net? Depends Upon Video Distribution – AV1

Preliminary

	VMAF	%	Bitrate	VMAF
4,111	95.22	50.00%	2,056	47.61
2,489	90.79	25.00%	622	22.70
1,511	83.93	10.00%	151	8.39
911	74.61	8.00%	73	5.97
544	62.74	5.00%	27	3.14
289	39.89	2.00%	6	0.80
		100.00%	**2,935**	**88.60**

Bitrate	VMAF	%	Bitrate	VMAF
2,244	95.26	82.16%	1,844	78.26
1,322	90.36	8.47%	112	7.65
811	83.81	6.30%	51	5.28
489	75.86	1.08%	5	0.82
256	49.36	2.00%	5	0.99
		100.00%	**2,017**	**93.00**
			918	
			31.27%	**4.39**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,111	95.22	100.00%	4,111	95.22
2,489	90.79		0	0.00
1,511	83.93		0	0.00
911	74.61		0	0.00
544	62.74		0	0.00
289	39.89		0	0.00
		100.00%	**4,111**	**95.22**

Bitrate	VMAF	%	Bitrate	VMAF
2,244	95.26	100.00%	2,244	95.26
1,322	90.36	0.00%	0	0.00
811	83.81	0.00%	0	0.00
489	75.86	0.00%	0	0.00
256	49.36	0.00%	0	0.00
		0.00%	0	0.00
		100.00%	**2,244**	**95.26**
			1,867	0.04
			45.41%	**0.04**

Bandwidth

VMAF Delta

Bitrate	VMAF	%	Bitrate	VMAF
4,111	95.22	15.00%	617	14.28
2,489	90.79	20.00%	498	18.16
1,511	83.93	40.00%	604	33.57
911	74.61	15.00%	137	11.19
544	62.74	6.00%	33	3.76
289	39.89	4.00%	12	1.60
		100.00%	**1,900**	**82.56**

Bitrate	VMAF	%	Bitrate	VMAF
2,244	95.26	44.98%	1,009	42.85
1,322	90.36	33.62%	444	30.38
811	83.81	12.66%	103	10.61
489	75.86	4.74%	23	3.60
256	49.36	4.00%	10	1.97
		0.00%	0	0.00
		100.00%	**1,590**	**89.41**
			310	6.84
			16.31%	**6.84**

Bandwidth

VMAF Delta

Weighted Average Takes You From

Conclusions

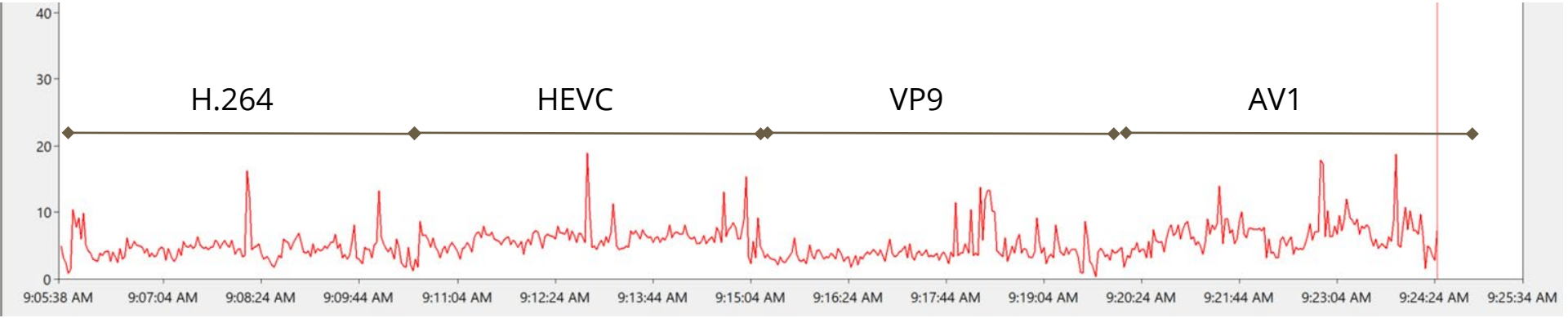
- VOD – doesn't deliver the real-world bandwidth savings needed to deploy unless you have massive scale
  - Encoding time is very competitive
  - If serious about VP9, check out alternative codecs like www.twoorioles.com
- AV1 looks very impressive for browser-based playback

Decoding Time

- Test conditions
  - I7-6700HQ 2.60Gh
  - From RAM drive
  - To RAM
  - No hardware acceleration
- Conclusion: VP9 is much better positioned than AV1 to play in software playback environments like mobile

Codec	Real Time
x264	15.4
x265	8.26
VP9	13.5
AV1	6.77

Playback CPU in I7



- Playback without hardware acceleration
  - Most platforms HEVC distributed to have hardware acceleration
- Playback in FFplay (may not be most optimized player)

- Observations
  - VP9 clearly much more efficient than AV1
  - HEVC clearly requires more CPU than VP9 but levels don't seem onerous
  - Even AV1 looks efficient, but mobile devices have less powerful CPUs

Lesson 4: Options for Live VP9

- Origination
- Transcoding

Origination

- Very few standalone hardware encoders
- Can achieve single stream from reasonable computer with FFmpeg
  - http://bit.ly/live_vp9
- Hardware support is available from NVIDIA and Intel

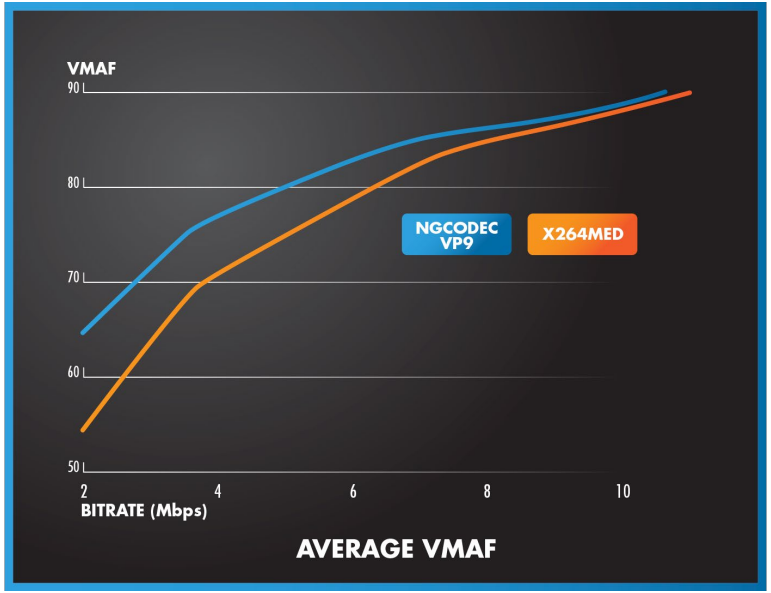
Transcoders

- NGCodec
- Intel
- NVIDIA
- FFmpeg

NGCodec - Works Great - Not for Sale

Bitrate (Mbps)	Rate savings (PSNR)	Rate savings (VMAF)
2	29.4%	36.5%
4	25.8%	27.1%
6	24.2%	24.9%
8	14.0%	14.4%

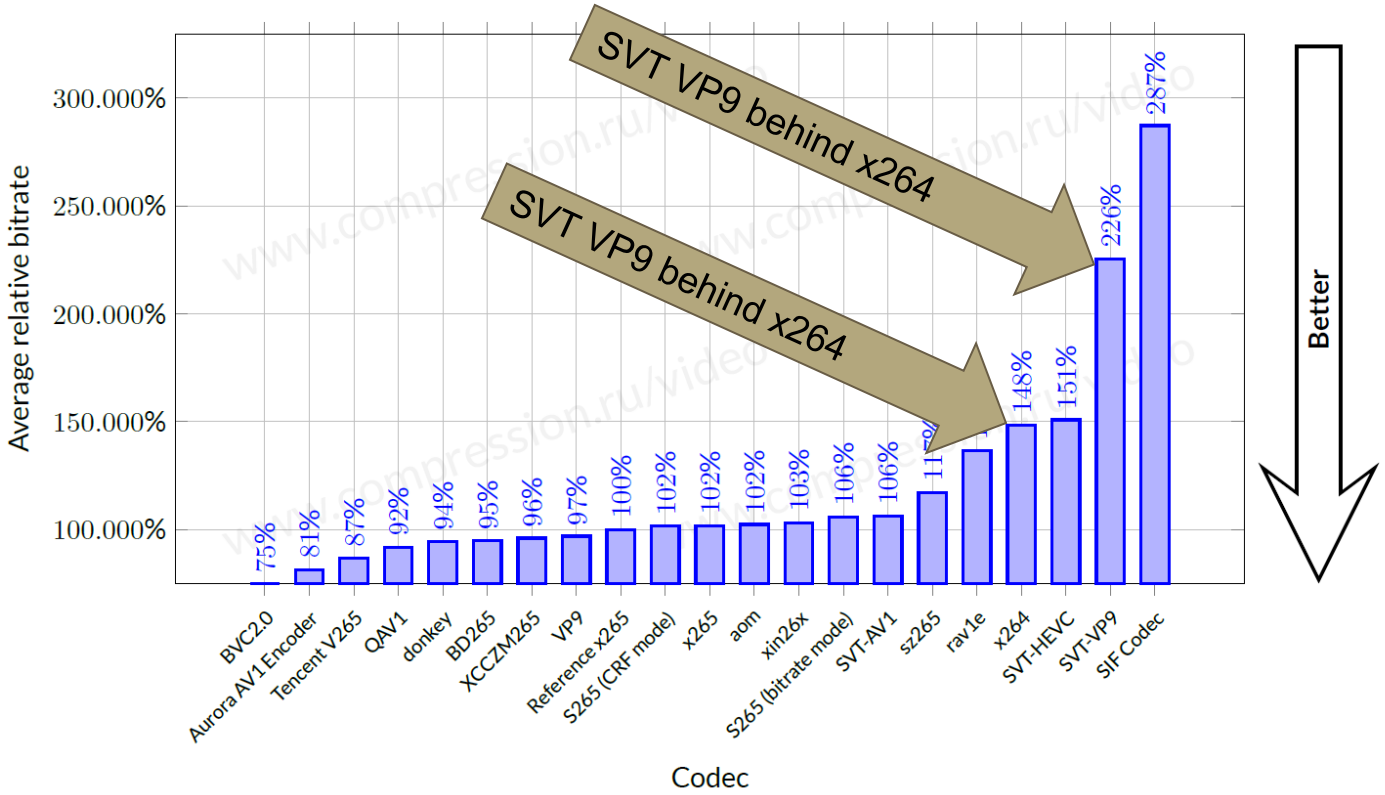
bit.ly/twitch_vp9



- NGCodec build FPGA-based transcoder
- Licensed to Twitch

- Xilinx bought NGCodec
- Xilinx only sells transcoder to Twitch

Intel - SVT VP9



Hardware Accelerated Transcoding

- Wowza
  - Software only (appears)
- Nimble Streamer
  - FFmpeg only
  - No hardware support (appears)

FFmpeg Transcoding

- Command strings
- Performance and accuracy
- Quality

FFmpeg Live Test String - VP9

```
ffmpeg -re -i football.mp4 ^
-y  -g 60 -quality realtime  -speed 8 -threads 8 -row-mt 1 -tile-columns 4 -frame-parallel 1 -auto-alt-ref 1 -
lag-in-frames 25  -b:v 3500k -maxrate 3500k  -c:v vp9 -c:a libopus -b:a 128k football_1080p_3500.webm ^
-y -g 60 -quality realtime -s 1280x720   -speed 8 -threads 8 -row-mt 1 -tile-columns 4 -frame-parallel 1 -
auto-alt-ref 1 -lag-in-frames 25  -b:v 2000k -maxrate 2000k  -c:v vp9 -c:a libopus -b:a 128k
football_720p_2000.webm ^
-y  -g 60 -s 960x540 -quality realtime  -speed 8 -threads 8 -row-mt 1 -tile-columns 4 -frame-parallel 1 -auto-
alt-ref 1 -lag-in-frames 25  -b:v 1200k -maxrate 1200k  -c:v vp9 -c:a libopus -b:a 128k
football_540p_1200.webm ^
-y  -g 60 -s 854x480 -quality realtime  -speed 8 -threads 8 -row-mt 1 -tile-columns 4 -frame-parallel 1 -auto-
alt-ref 1 -lag-in-frames 25  -b:v 800k -maxrate 800k  -c:v vp9 -c:a libopus -b:a 128k football_480p_800.webm ^
-y -s 640x360  -g 60 -quality realtime  -speed 8 -threads 8 -row-mt 1 -tile-columns 4 -frame-parallel 1 -auto-
alt-ref 1  -lag-in-frames 25  -b:v 400k -maxrate 400k  -c:v vp9 -c:a libopus -b:a 128k football_360p_400.webm
```

https://bit.ly/live_vp9

FFmpeg Live Test String - x264 (to match data rate)

```
ffmpeg -re -i football.mp4 ^

-c:v  libx264  -b:v 5250K -bufsize 5250K -maxrate 5250K  -g 60 -keyint_min 60 -sc_threshold 0 -preset veryfast
football_1080p_5250.mp4 ^
-c:v  libx264 -s 1280x720 -b:v 2800K -bufsize 2800K -maxrate 2800K  -g 60 -keyint_min 60 -sc_threshold 0 -
preset veryfast football_720p_2800.mp4  ^
-c:v  libx264 -s 960x540 -b:v 1650K -bufsize 1650K -maxrate 1650K  -g 60 -keyint_min 60 -sc_threshold 0 -
preset veryfast football_540p_1650.mp4  ^
-c:v  libx264 -s 854x480 -b:v 1100K -bufsize 1100K -maxrate 1100K  -g 60 -keyint_min 60 -sc_threshold 0 -
preset veryfast football_480p_1100.mp4  ^
-c:v  libx264 -s 640x360 -b:v 550K -bufsize 550K -maxrate 550K  -g 60 -keyint_min 60 -sc_threshold 0 -preset
veryfast football_640p_550.mp4
```

FFmpeg Transcoding - Data Rate Accuracy

Target	Actual	Overage
3,500	4,502	28.63%
2,000	2,603	30.15%
1,200	1,602	33.51%
800	1,139	42.41%
400	655	63.63%

- Requires 4x or greater resources of x264
- Very inaccurate data rate

- Recent consulting project missed the target data rate by an average of about 90%

Comparative Quality

Data Rate	VP9	x264	Delta
3,500	89.94	86.62	3.32
2,000	77.76	77.09	0.67
1,200	62.89	64.34	-1.45
800	52.58	53.82	-1.25
400	33.98	35.69	-1.71
Average			-0.42

- Nice bump at top of the ladder
- Overall, lower than x264

- Consulting project last winter involving 6 SD and 6 HD files yielded similar results

Live Conclusions

- Origination
  - Not a lot of options
  - Unlikely that there will be much new innovation given focus on AV1
  - Software encoding is an option, or NVIDIA
- Transcoding
  - Hardware works well but isn't available
  - Software is inefficient and has issues meeting the target data rate
  - Quality over entire ladder (single test fi
  - NVIDIA-based transcoding may be an option

VP9 Summary

- VOD – doesn't deliver the real-world bandwidth savings needed to deploy unless you have massive scale
  - Encoding time is very competitive
  - Decoding CPU is as well
  - If serious about VP9, check out alternative codecs like www.twoorioles.com
- AV1 looks very impressive for browser-based playback
- Live – Problematic
  - Few (if any) hardware encoders
  - Transcoding is inefficient, hardware alternatives unavailable or uncertain
  - Software transcoding is cost, inaccurate, and doesn't deliver substantially higher quality

Resources

- DASH resources
  - Instructions to playback Adaptive WebM using DASH (encoding and decoding)
    - https://bit.ly/webm_wiki_DASH
- General
  - Google VP9 Overview - https://developers.google.com/media/vp9
  - FFmpeg and VP9 Encoding Guide - https://trac.ffmpeg.org/wiki/Encode/VP9
  - Encoding VP9 in FFmpeg: An Update - http://bit.ly/vp9_done_right