Abstract—With COVID-19, the interest for digital interaction has raised, putting in turn real-time or low-latency codecs into a new light. Most of the codec ecosystem, including AV1, has been focusing on coding efficiency which is the main sought after improvement for Video On Demand use case. Very little literature exist on real-time codecs. This work focuses on explaining the differences between the VOD and the interactive use cases from the codec point of view. It makes the difference between latency and throughput, and show that reducing the former to achieve interactive latency is orthogonal to achieving maximum coding efficiency. Measurements are made on encoding of Full HD video sequences from the literature to compare the respective performances of H.264, VP8, VP9 and AV1 all in real-time mode.

I. INTRODUCTION

A. Codec Use Cases

The term codec usually refer to algorithms that encode, respectively decode, binary representation of media. Currently there are arguably three major use cases for large-scale codec usage:

- encoding of original raw content, client side,
- trans-coding of content server-side,
- decoding of content on the receiving side.

Codecs are usually evaluated based on coding efficiency, a.k.a compression/quality ratio achieved against run time. Comparisons are made using the Bjøntegaard rate difference (BD-rate) [1] [2] and multiple documents exist to guide researchers choosing the most representative dataset, the most meaningful metric, and the best representation of results [3].

For example, the Xiph.org Foundation has developed the AreWeCompressedYet [4] automatic service to enable comparisons between different implementations of video codecs using various metrics.

With COVID-19 and the new normal, people have a new appetite for more interactive uses of streaming, to get the same experience and value they were enjoying in real-life. Correspondingly, it has increased the demand for faster-than-live streaming, ”live” being 5 seconds behind real-time, to reach the 500 ms level of latency where human interactions thrive. In parallel, the rise of cloud gaming, AR and VR have been pushing in the same direction, albeit with generated content.

In this paper we focus on the specific use case of real-time consumption of media for interactive applications. This use case puts a specific emphasis on latency, making coding efficiency a secondary metric.

B. Difference Between Pre-Recorded Content and Live Content Encoding

In VOD, latency is usually negligible for both encoder and decoder.

For encoders:
- Production time itself already accounts for weeks, sometimes months, and has already happened.
- the media does not need to be consumed right away,
- the encoded media is stored before delivery, making storage cost a concern,
- the media will be delivered over the public internet, making the bandwidth cost a concern,
- the media will be delivered a very large number of time, making any gain on the media size even more impacting on the eventual operational cost, very often outweighing the encoding cost by several orders of magnitude.

For decoders:
- Latency, or start-up delay (time to first media) is acceptable as long as once started the playback is smooth.
- For playback to be smooth, the throughput of the decoder needs to be equal or above real-time for any given public internet condition. That adaptation mechanism is then often more important than the already good decoding speed as reflected in all the recent research on “adaptive video streaming” [5] [6] [7].

As such, the delay or latency induced by the encoding and decoding processes are almost never taken in account, and the focus of almost all research and benchmarks in the literature has been on compression ratio, and throughput. The encoding throughput is the total number of frames of the input divided by the total duration of the encoding process. Latency is the time it takes for a single frame to go through the encoding process. The total run time used in coding efficiency computation includes both latency and processing time, diluting out latency. Latency should be reported separately.

Another difference between recorded content and real-time content is content availability and its impact on latency.

As described in [8] figure 1, the motion estimation is the dominant contributor to run time budget in encoding. Not only this, but motion estimation takes 73 times more time than reading the data in memory. Motion estimation works with a frame buffer as input, whose depth condition the latency.
Let’s take the use case where you have a 60 frames deep frame buffer. With pre-recorded content, the latency will be the times it takes from reading corresponding frames in memory, as the frames are already available. With live content, you will need to wait until those frames are generated before you can start any kind of processing. Obviously, the former is much faster than the later by several orders of magnitude. At a capture rate of 30 fps, one must wait 2 seconds before the buffer is ready to compute motion estimation, even if the encoder can then encode faster than 30 fps.

In **real-time or live event, the encoding rate will be limited by the capture rate, and the frame buffer depth**.

It has been proven that using b-frames, frames encoded using both past frames and future frames, yield great gains for codecs. Here again, for real-time content where future frame are not available, that means working with a frame buffer with a minimum length and an offset. However, I-frames and P-frames do not induce delay, even if P-frame will not be available right away. This has been noted for example by the author in "The H.264 Advanced Video Compression Standard, Second Edition” [9], in 6.4.7.1 on page 169 – “The prediction structure shown [...] uses only I and P slices. It is compatible with the Baseline Profile or Constrained Baseline Profile of H.264, which do not allow B slices, and would be suitable for an application requiring low delay and/or minimal storage memory at the decoder.”

If using the x264 implementation, one can use the “baseline profile” to remove b-frames, the "veryfast" speed preset in conjunction with the “zerolatency” tuning option to achieve real-time. The x264 encoder also provides “intra-refresh” settings which allows to better distribute the data rate over to several frames.

**C. Benchmarking Codecs for Interactive Use Cases**

We have shown that the run time is latency + encoding time, and the latency is a function of the depth of the frame buffer and the constant capture rate. Increasing the frame rate reduces latency but increases the work load. The easiest way to reduce the latency is to reduce the size of the frame buffer, or the need for a frame buffer altogether.

This not to be confused with the speed. Most codecs have a speed variable in the [0,8] range. It reflects the encoder complexity and resulting speed and NOT the latency setting.

In AV1 real-time compilation mode for example, some tools are not used, a single reference frame is used, no look ahead (B-frames in H.264) or lagging frame are used, it is always in constant bitrate mode, and the partition decision is based on variance distribution instead of being searched, among other heuristics used. Of course, the speed setting is also always in the [5,8] range, depending on the input size and the CPU capacity mainly.

Now, this comes at a cost in terms of coding efficiency, and makes the real-time codecs not directly comparable with traditional rate-compression graphs. This study is going to propose a way to evaluate codecs for the interactive use case, and try to replace the performances of real-time mode of some codecs in the bigger context of video codecs.

H.264, VP8, VP9 and AV1 in their real-time version are all used in webrtc.org code today. Separately, they are also used in VOD streaming, and a lot of benchmarks are provided for those cases. A recent study has measured the coding performance of libaom AV1 encoder against x265 and libvpx-vp9 using their best quality mode and two-pass compression [10]. In this paper, we will concentrate on the real-time mode of encoders. What kind of BD-rate improvement can we expect when going from one codec to another in real-time mode (i.e. in webrtc, or for interactive use cases in general)? How does it relate to the existing streaming benchmarks? This paper is going to address those questions.

**II. Methodology**

**A. Dataset**

For easier comparison of results, we have used video sequences having the same resolution, same frame rate, same color space and depth. Only the duration or the frame rate of video sequences is different from one video to another.

While compression can be objectively evaluated, quality metrics have in the past years shifted from objective, but poorly correlated with human perception metrics like PSNR (Peak Signal to Noise Ratio) [11], to subjective metrics like VMAF (Video Multimethod Assessment Fusion) [12,13]. It is known that PSNR does not correlate much with subjective human evaluation of image or video quality [14]. A recent study on the evaluation of objective video quality metrics has demonstrated a good correlation between subjective scores given by humans and VMAF scores [15]. We have followed the latest trend in codec research and used WMAF.

To give an interpretation of a VMAF score, one can relate it to the five scores of the Absolute Category Rating (ACR) methodology [16]: “bad”, “poor”, “fair”, “good” and “excellent.” VMAF gives a score in the range [0,100]. VMAF score 20 can be mapped to “bad,” score 40 to “poor,” score 60 to “fair,” score 80 to “good,” and score 100 to “excellent” [17].

We have focused on 1080p HD video. This is the resolution recommended to compute VMAF scores using the default model v0.6.1 [17]. Table I gives the list of the 12 videos used in our study. There are two groups of videos: a group of 7 videos having a frame rate of 25 fps, and a group of 5 videos with a frame rate of 50 fps. Figure 1 shows a snapshot of each video.

All the video sequences have a full HD resolution of 1920×1080 pixels, YUV format, 8 bits depth and they are not compressed. They have been selected from the publicly available Xiph.org Video Test Media [derf’s collection] dataset.$^1$

**B. Video Codecs**

We will compare the performance of six encoders, namely x265 and SvtAv1EncApp for AV1, vpxenc for VP8 and

---

1. Xiph.org Video Test Media [derf’s collection] https://media.xiph.org/video/derf/
VP9, h264enc and x264 for H.264, compiled in their real-time mode when available, and using speed 7 when applicable, for encoding the 12 video sequences of our dataset at various bitrates.

Table II gives for each codec the version we have used, where to get their source code and which options we selected to compile them.

To give an insight of encoding performance difference between real-time mode and non-real-time mode, only for AV1, we have compiled a second version of AOM encoder aomenc using the option --DCONFIG_REALTIME_ONLY=0. The non real-time version of aomenc can be used with speed setting in the range [0..6]. We have selected speed option --cpu-used=5.

For the real-time version of aomenc, possible speed setting should be in the [6..8]. We have selected speed option --cpu-used=7.

We will refer to aomenc in real-time mode as aomenc-rt (encoder called with option --rt), and to aomenc not in real-time mode as aomenc-good (encoder called with good quality option --good).

The options used at run time to launch each codec are given in Table III.

Compilation of encoders and encoding of videos have been performed on a Dell™ OptiPlex 5050 with processor Intel® Core™ i7-7700T 8 cores at 2.90 GHz and 16 GB memory running Ubuntu Desktop 20.04.1 64 bits operating system.

III. RESULTS AND ANALYSIS

We measured the VMAF scores for each encoder at six different target bitrates: 800, 1200, 2000, 3000, 5000 and 10000 kbps (see VMAF graphs in Figures 3a to 3l), and computed the BD-rates from the VMAF curves according to bitrate (see Tables IV and V).

A VMAF score can be interpreted as a score given by a human viewer on the 5 scale.

A. Interpretation of BD-rate and BD-VMAF

A BD-rate is a measure of the average percentage bitrate savings that can be obtained for the same visual quality level. This measure is computed over the range of quality levels that are common to two curves.

For example, let consider VMAF scores on Figure 3a for Blue sky video. We want to compute the bitrate savings at same VMAF level of aomenc-rt (orange curve, VMAF range from 83 to 100) as compared to vpxenc-vp9 (red curve, VMAF range from 74 to 100). The common VMAF range for these two curves is 83 to 100. Using that common quality range, we compute the average bitrate savings by calculating the area between the curves (to the left of the vpxenc-vp9 red curve and to the right of the aomenc-rt orange curve), and we divide it by the area to the left of the aomenc-rt orange curve up to the right of the y-axis. We get a BD-rate of -21.16 as shown in Table III.

Note: BD-rate for aomenc-good on video Station2 cannot be evaluated reliably because the intersection area between the two VMAF score curves of aomenc-good and aomenc-rt is too thin, see Figure 3e.

Note: BD-rate for aomenc-good on video Old-town-cross cannot be evaluated reliably because the intersection area between the two VMAF score curves of aomenc-good and aomenc-rt is too thin, see Figure 3k.
in Table IV. It is a negative value, which means that there is a reduction in bitrate for aomenc-rt as compared to vp9-vpx. The interpretation of this value is that the same VMAF score, we may expect that aomenc-rt gives in average a $1.97$ points higher VMAF score, which means that there is an increase in VMAF for aomenc-rt as compared to vp9-vpx. The interpretation of this value is that for the same bitrate, we may expect that aomenc-rt gives in average a $21.16\%$ bitrate savings as compared to vp9-vpx.

Similarly, we can compute the average visual quality improvement for the same bitrate between aomenc-rt and vp9-vpx by switching the variables. Using the same example, we look for the common bitrate range between aomenc-rt (orange curve, bitrate range from 795 to 9950 kbps) and vp9-vpx (red curve, bitrate range from 795 to 9940 kbps). The common bitrate range for these two curves is 799 to 9940 kbps. Using that common bitrate range, we compute the average VMAF improvement by calculating the area between the curves (to the bottom of aomenc-rt orange curve and to the top of vp9-vpx red curve), and we divide it by the area to the bottom of the vp9-vpx red curve down to the top of the x-axis. We get a BD-VMAF of $1.97$ as shown in Table V. It is a positive value, which means that there is an increase in VMAF for aomenc-rt as compared to vp9-vpx. The interpretation of this value is that for the same bitrate, we may expect that aomenc-rt gives in average a VMAF score $1.97$ points higher than vp9-vpx.

B. Discussion

It is much more challenging for all the encoders of this study to encode video clips at 50 fps than those at 25 fps. Not only it is necessary to have a higher bitrate at 50 fps to reach
the same image quality as can be achieved at 25 fps, but the real-time constraint means that, to generate frames at 50 fps, the encoder needs to encode each frame twice as fast as when generating frames at 25 fps.

In fact, only encoders openh264, x264 libvpx-vp8 are fast enough to encode a 1080p video at 25 frames per second or at 50 frames per second as shown in Fig. 2. Encoders libvpx-vp9 and aomenc-rt7 are able to produce 25 frames per second but only if the target bitrate is 2000 kbps or lower. Encoding a 1080p video at a rate of 50 frames per second is not achievable by libvpx-vp9 aomenc-rt7 on our test computer.

Generally, the video clips at 25 fps can be encoded with an excellent VMAF score of 90 or above at a bitrate of 5,000 kbps (with the exception of video clip Riverbed which proved very difficult for all the encoders to be encoded), while for video clips at 50 fps, even at a bitrate of 10,000 kbps, the resulting encoded videos reach usually a VMAF score of only 70 or below.

Among the two encoders studied for AV1, SVT-AV1 provides a much better coding efficiency than aomenc-rt7. However, SVT-AV1 is lacking an efficient real-time mode. It took SVT-AV1 about 7 times as long as aomenc-rt to encode the same video clips, although both were run using the same speed of 7.

Nevertheless, SVT-AV1 at speed 7 has a coding efficiency similar or better than aomenc-good at speed 5. It took SVT-AV1 about 3 times less time than aomenc-good to encode the same video clips.

Globally, we are showing results consistent with original assumptions:

- We find that the real-time modes of the codecs are performing relatively to each other as they would with their non real-time version, i.e AV1 better than VP9 better than VP8.
- The non-realtime version of aomenc has a much better coding efficiency than the real-time version. Actually, the coding efficiency of aomenc in real-time is so good that in all tested sequences but one (Riverbed), we could reach perfect visual reconstruction without ever needing to use the 10,000 kbps maximum.
- The coding efficiency of SVT-AV1 is excellent, even compared with aomenc-good. What is missing to SVT-
AV1 is a real-time mode.

The pendant is also interesting, most of the real-time codecs had problems keeping a good visual quality (VMAF score of 80 or above) with encoding 1080p content with target below than 2,000 kbps except VP9 and AV1. Riverbed is the exception being difficult for all encoders to encode with a good quality even with target of 5,000 kbps.

The x264 implementation of H.264 is more or less often in par with VP8, while theopenh264 implementation of H.264 exhibits generally lower coding efficiency on the 12 video clips of this dataset.

IV. CONCLUSION

Pre-recorded content provide encoders with the capacity to fill up buffers to increase the coding efficiency without increasing the latency too much. The same buffers which are filled at I/O speed for pre-recorded content need to wait for frame to be acquired in live, real-time and interactive use case, making any operation that requires frame buffers prohibitive.

Very often, benchmarks are only provided for pre-recorded content, and cannot be directly translated into the real-time configuration.

One may expect the real-time mode of aomenc for AV1 to be about 33% less efficient than its normal counterpart. While theoretically interesting, this has but little practical interest, since the default aomenc encoder, even at speed 6 will have too much latency to be used in interactive case.

More interestingly, we have shown from the 12 video sequences of our dataset that one can expect an average of 17% less bandwidth usage for the same video quality with aomenc-rt than with VP9-rt, 40% less than with VP8.

V. FUTURE WORK

Some of the work will be done and integrated for the camera-ready version of this paper, if accepted.

- more content (larger dataset)
  - impact of other bit depths and chroma sampling.
  - impact of resolutions
- more work on decoupling latency and throughput in measurements
  - to best compare directly codecs still using reference pre-recorded datasets, it would suffice to pace the reader to 25 fps or 50 fps.
  - modify code to report direct numbers for latency, and encoding time.
- test others AV1 implementations. including hardware encoders.
- impact of SVC?
- impact of 16bits pipeline?

REFERENCES

[1] G. Bjøntegaard, “Calculation of average PSNR differences between RD-curves,” VCEG-M33, ITU-T SG16/Q6, Tech. Rep., June 2001. [Online]. Available: http://wftp3.itu.int/av-arch/video-site/0104_Aus/

[2] ——, “Improvements of the BD-PSNR model,” VCEG-A111, ITU-T SG16/Q6, Tech. Rep., July 2008. [Online]. Available: http://wftp3.itu.int/av-arch/video-site/0807_Bet/

[3] T. Daede, A. Norkin, and L. Brailovskiy, “Video codec testing and quality measurement,” IETF, Tech. Rep., January 2020. [Online]. Available: https://tools.ietf.org/id/draft-ietf-netvc-testing-09.txt

[4] xiph.org. AreWeCompressedYet. [Online]. Available: https://wiki.xiph.org/AreWeCompressedYet

[5] A. V. Katsenou, F. Zhang, M. Afonso, and D. R. Bull, “A subjective comparison of AV1 and HEVC for adaptive video streaming,” in IEEE International Conference on Image Processing (ICIP), September 2019.

[6] C. G. Bampis, Z. Li, A. K. Moorthy, I. Katsavounidis, A. Aaron, and A. C. Bovik, “Study of temporal effects on subjective video quality of experience,” IEEE Transactions on Image Processing, vol. 26, pp. 5217–5231, November 2017.

[7] C. G. Bampis, Z. Li, I. Katsavounidis, T.-Y. Huang, C. Ekanadham, and A. C. Bovik, “Towards perceptually optimized end-to-end adaptive video streaming,” 2019. [Online]. Available: https://arxiv.org/abs/1908.03898

[8] K. Yu, J. Lu, J. Li, and S. Li, “Practical real-time video codec for mobile devices,” in International Conference on Multimedia and Expo (ICME). IEEE, July 2003, pp. 509–512.

[9] I. E. Richardson, The H.264 Advanced Video Compression Standard, 2nd ed. Wiley, 2011.

[10] Y. Chen, D. Mukherjee, J. Han, A. Grange, Y. Xu, S. Parker, C. Chen, H. Su, U. Joshi, C.-H. Chiang, Y. Wang, P. Wilkins, J. Bankoski, L. Trudeau, N. Egee, J.-M. Valin, T. Davies, S. Midtskogen, A. Norkin, P. de Rivaz, and Z. Liu, “An overview of coding tools in AV1: the first video codec from the Alliance for Open Media,” AFSIPA Transactions on Signal and Information Processing, vol. 9, February 2020.

[11] American National Standards Institute, Objective video quality measurement using a peak-signal-to-noise-ratio (PSNR) full reference technique, American National Standards Institute, Ad Hoe Group on Video Quality Metrics, 2001.

[12] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, “Toward a practical perceptual video quality metric,” Tech. Rep., June 2016. [Online]. Available: https://netflixtechblog.com/vmaf-virtual-method-assessment-fusion/

[13] Netflix. VMAF – Video Multi-Method Assessment Fusion. [Online]. Available: https://github.com/Netflix/vmaf

[14] H. R. Sheikh, M. F. Sabir, and A. C. Bovik, “A statistical evaluation of recent full reference image quality assessment algorithms,” IEEE Transactions on Image Processing, vol. 15, pp. 3440–3451, November 2006.

[15] C. Lee, S. Woo, S. Baek, J. Han, J. Chae, and J. Rim, “Comparison of objective quality models for adaptive bit-streaming services,” in International Conference on Information, Intelligence, Systems & Applications (IISA), June 2017.

[16] ITU, “Subjective video quality assessment methods for multimedia applications,” ITU-T. Tech. Rep. P910. Apr. 2008.

[17] Z. Li, C. Bampis, J. Novak, A. Aaron, K. Swanson, A. Moorthy, and J. D. Cock, “VMAF: The journey continues,” Tech. Rep., October 2018. [Online]. Available: https://netflixtechblog.com/vmaf-the-journey-continues-44b51ee9ed12
Fig. 3: VMAF scores according to bitrate