Enhancing SVT-AV1 with LCEVC to improve quality-cycles trade-offs and enhance sustainability of VOD transcoding

Guendalina Cobianchi[*a], Guido Meardi[a], Stergios Poularakis[a], Alexei Walisiewicz[a], Omran Abdelkafi[b], Foued Ben Amara[b], Faouzi Kossentini[b], Cosmin Stejerean[c], Hassene Tmar[c]

[a] V-Nova Ltd, 20 Eastbourne Terrace, London UK; [b] Intel Corporation, Suite 700, 450 SW Marine Drive, Vancouver, BC V5X 0C3 Canada; [c] Meta Platforms, 1 Hacker Way, Menlo Park, CA, USA

ABSTRACT

The trade-offs between compression performance and encoding complexity are key in software video encoding, even more so with increasing pressure on sustainability. Previous work "Towards much better SVT-AV1 quality-cycles trade-offs for VOD applications" [1] described three approaches of evaluating compression efficiency vs cycles trade-offs within a convex-hull framework using the Dynamic Optimizer (DO) algorithm developed in [2] [3] for VOD applications.

In parallel, the new video codec enhancer LCEVC (Low Complexity Enhancement Video Coding) [4], designed to provide gains in speed-quality trade-offs, has recently been standardized as MPEG-5 Part 2. The core idea of LCEVC is to use any video coding standard (such as AV1) as a base encoder at a lower resolution, and then reduce artifacts and reconstruct a full resolution output by combining the decoded low-resolution output with up to two low-complexity reconstruction enhancement sub-layers of the residual data.

This paper starts by applying LCEVC to SVT-AV1 [5], as well as x264 [6] and x265 [7], while using two of the approaches presented in [1] to evaluate the resulting compression efficiency vs cycles trade-offs. The paper then discusses the benefits of LCEVC towards higher playback speed and lower battery power consumption when using AV1 software decoding.

Results show that, with fast-encoding parameter selection using the discrete convex hull methodology, LCEVC improves the quality-cycles trade-offs for all the tested codecs and across the full complexity range. In the case of SVT-AV1, LCEVC yields a ~40% reduction in computations while achieving the same quality levels according to VMAF_NEG [8]. LCEVC also enlarges the set of mobile devices capable of playing HD as well as high-frame-rate content encoded with AV1 and extends mobile battery life by up to 50% with respect to state-of-the-art AV1 software decoding.

KEYWORDS

AV1, SVT-AV1, Low Complexity Enhancement Video Coding (LCEVC), MPEG-5 Part 2, video compression, video coding, standard, enhancement, software codec, multi-resolution video coding, multi-layer encoding, convex hull, dynamic optimizer, encoder complexity, coding efficiency, adaptive streaming

[*] guendalina.cobianchi@v-nova.com, phone: +44 7825 563336, www.v-nova.com

Applications of Digital Image Processing XLV, edited by Andrew G. Tescher,
Touradj Ebrahimi, Proc. of SPIE Vol. 12226, 122260S · © 2022
SPIE · 0277-786X · doi: 10.1117/12.2633882

1. INTRODUCTION

This paper presents extensive evaluation results of using MPEG-5 Part 2 Low Complexity Enhancement Video Coding (LCEVC), a new video coding standard developed by MPEG and published in October 2021 as ISO/IEC 23094-2, to enhance SVT-AV1 with respect to a) encoding quality-cycles trade-offs and b) AV1 decoding speed and battery life.
The first part of the paper targets the evaluation of LCEVC when used to improve the quality-cycles trade-offs of SVT-AV1, x264 and x265 in VOD use cases. The latest versions of the SVT-AV1, x264, and x265 encoders were tested within an LCEVC framework using different variants of the convex-hull Dynamic Optimizer (DO)-based evaluation methodologies [1] [2] [3]. The LCEVC-enhanced encoders are first evaluated using the combined DO approach and are compared against the native versions of the same encoders. For the combined DO approach, the encoders' BD-rates are computed for one clip representing the concatenation of all video clips in the test set. The evaluation then continues using a variation of the combined DO approach, referred to as the restricted discrete DO approach, where the range of quality values considered in the evaluation is reflective of the quality values common in adaptive streaming (AS) applications, and where each encoder's BD-rates are generated for a few points on the convex hull. Finally, to reduce complexity, a fast DO approach is evaluated, where the identification of optimal encoder parameters is performed based on encodings generated using a fast encoder. The optimal encoder parameters are then used to generate final encodings using the desired encoders. The resulting LCEVC BD-rates and encoding cycles are then compared to those of the native SVT-AV1, x264 and x265 encoders.

The second part of the paper analyzes LCEVC's impact on AV1 decoding performance. With LCEVC, the AV1 base layer is decoded at a quarter resolution (e.g., 540p for a 1080p video), and the LCEVC enhancement is decoded leveraging existing hardware blocks such as GPU shaders and scalers. Although the most power-efficient decoding option for typical LCEVC decoding leverages the GPU, and it was not optimized for CPU only decoding, we start with a CPU-only decoder test on an ARM-based server as a cheaper proxy of the relative compute complexity of LCEVC SVT-AV1 against SVT-AV1 bitstreams. We then assess the real-life use case of LCEVC AV1 decode and playback on a set of Android mobile devices, using an LCEVC-capable Android player in combination with the latest Dav1d [9] and GAV1 [10] AV1 software decoders.

Concluding remarks are presented in the last section of the paper.

2. THE SVT-AV1 ENCODER: LATEST IMPROVEMENTS

Over the past year, SVT-AV1 development has focused on both extending the compression efficiency vs complexity trade-offs with a focus on VOD applications and adding support for low-delay applications. A number of improvements in SVT-AV1 have been introduced since the evaluation work reported in [1]. The main changes included the following:

- Faster encoder presets: Four new fast presets M9-M12 were added in SVT-AV1 to extend the range of encoder complexity levels to those of the medium/fast presets of x264 while maintaining a BD-rate advantage.
- Improved quality complexity trade-offs for all presets: Please see Figure 4 for more details on the BD-rates versus cycles trade-offs compared to those of other open-source encoders in high-latency VOD applications.
- Subjectively-tuned encoder mode: Addresses some subjective quality artifacts observed when using the default mode. Subjective improvements include higher sharpness and less blockiness.
- Support for new tools: Super-resolution frames, overlay frames, S frames and low-delay prediction structures.
- Rate control improvements: Improvements in the one-pass Variable Bitrate (VBR) rate control using a lookahead window, and in the Constant Bitrate (CBR) rate control algorithm using low-delay prediction structures.
- Decoder friendly encoder optimizations: These optimizations cover the range of M0-M10 presets, and they aim to reduce the computational cost of decoding SVT-AV1 generated bitstreams, while maintaining a minimal impact on the encoder's BD-rate performance [11]. The optimizations include using lighter versions of the Motion Field Motion Vector prediction, Overlapped Block Motion Compensation, warped motion prediction, deblocking filter, Constrained Directional Enhancement Filter and the Self-guided filter with subspace projection. Further reduction in decoding time only (with a slight increase in cycles) could be realized using tiles in the encoding process.

3. INTRODUCTION TO LCEVC

Rather than being another video codec, LCEVC aims at enhancing existing coding standards (e.g., AVC, VP9, HEVC, AV1, EVC or VVC) to reduce the computational load and improve the compression efficiency at video streaming relevant bitrates. The core idea is to use a conventional video codec as base codec at a lower resolution and reconstruct a full resolution video by combining the decoded low-resolution video with up to two enhancement sub-layers of residuals encoded and reconstructed with specialized low-complexity coding tools.

The MPEG requirements [12] satisfied by LCEVC are the following:

- When enhancing an n-th generation MPEG codec (e.g., AVC), compression efficiency for the aggregate stream is appreciably higher than that of the n-th generation MPEG codec used at full resolution and as close as possible to that of the (n+1)-th generation MPEG codec (e.g., HEVC) used at full resolution, at bandwidths and operating conditions relevant to mass market distribution; and
- Encoding and decoding complexity for the aggregate full resolution video (i.e., base plus enhancements) is comparable to that of the base encoder or decoder, respectively, when used alone at full resolution.

3.1 Structure of the LCEVC encoder

The encoding process to create an LCEVC bitstream is shown in Figure 1 and includes three major steps.

Base codec: The input sequence is fed into two consecutive downscalers and is processed according to the chosen scaling modes (i.e., any combination of 2-dimensional scaling, 1-dimensional scaling in the horizontal direction or no scaling). The output then invokes the base codec which produces a base bitstream according to its own specification.

Enhancement sub-layer 1. The reconstructed base picture is upscaled and then subtracted from the first-order downscaled input sequence to generate the layer 1 residuals. These form the starting point for the encoding process of the first enhancement sub-layer. A number of coding tools process the input and generate entropy encoded quantized transform coefficients.

Enhancement sub-layer 2. To create layer 2 residuals, coefficients from layer 1 are processed by an in-loop decoder to achieve the reconstructed picture. Since layer 1 might have a different resolution vs. the input sequence, the reconstructed picture is processed by an upscaler. Finally, the residuals are calculated by a subtraction of the input sequence and the upscaled reconstruction. Similar to layer 1, the samples are processed by a few coding tools. Additionally, a temporal prediction can be applied to achieve a better removal of redundant information.

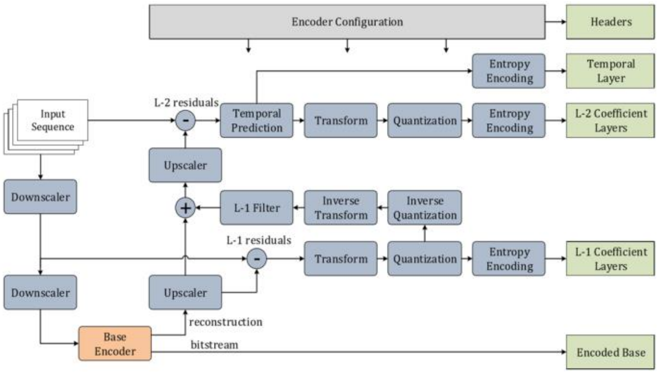


Figure 1 - Structure of the LCEVC encoder

3.2 Key technical features

LCEVC deploys a small number of very specialized coding tools that are well suited for the type of data it processes.

3.2.1 Sparse residual data processing. The coding scheme processes one or two layers of residual data. These residual data are produced by taking differences between a reference video frame (e.g., a source video) and a base-decoded up-sampled version of the video. The resulting residual data is sparse information, consisting typically of edges, dots and other details, which are efficiently processed using simple and small transforms designed to deal with sparse information.

3.2.2 Efficient use of existing codecs. The base codec is typically used at a lower resolution. Because of this, the base codec operates on a smaller number of pixels, thus allowing the more complex codec to use less power, operate at a lower quantization parameter (QP) and use tools in a more efficient manner.

3.2.3 Resilient and adaptive coding process. LCEVC allows the overall coding process to be resilient to the typical coding artifacts introduced by traditional DCT block-based codecs. The first enhancement sub-layer (L-1 residuals) corrects artifacts introduced by the base codec, whereas the second enhancement sub-layer (L-2 residuals) adds details and sharpness to the corrected up-sampled base for maximum fidelity (up to lossless coding).

3.2.4 Agnostic base enhancement. LCEVC can enhance any base codec, from existing ones (MPEG-2, VP8, AVC, HEVC, VP9, AV1, EVC, VVC, AVS3, etc.) to future ones. The enhancement operates on a decoded version of the base codec in the pixel domain, and therefore it is agnostic to how the base codec processed pictures have been encoded and/or decoded.

3.2.5 Parallelization. LCEVC does not use any inter block prediction and enhancement layers are independent allowing them to be processed in parallel. The sequence is processed by applying small (2x2 or 4x4) independent transform kernels over the layers of residual data. Since no prediction is made between blocks, each 2x2 or 4x4 block can be processed independently and in a parallel manner, using CPUs, GPUs, DSPs and/or dedicated hardware blocks.

3.2.6 Performance. Multiple independent studies validated the LCEVC performance gains. MPEG Verification Tests [13] published in April 2021 demonstrated, comparing via ITU-R formal MOS subjective tests [14] full-resolution LCEVC sequences with full-resolution single-layer anchors, average bit rate savings for LCEVC of 46% for UHD and 28% for HD when enhancing AVC and 31% for UHD and 24% for HD when enhancing HEVC. The tests also confirmed that LCEVC is a more efficient means of resolution enhancement of half-resolution anchors than unguided up-sampling.

For further information regarding LCEVC the following readings are recommended [15][16][17].

3.3 LCEVC-Enhanced SVT-AV1

When encoding in combination with LCEVC, the overall compression efficiency relies on the efficiency of both the base codec and LCEVC. Since LCEVC uses the base codec at a low resolution and relatively low QPs, the overall coding efficiency relies significantly on suitably calibrating the base codec at low-QP/low-resolution operating points. During the tests, we observed that the current SVT-AV1 encoder has not yet been optimally calibrated for low-QP/low-resolution operating points. The relative per-pixel coding efficiency of SVT-AV1 when used as a lower-resolution base layer codec for LCEVC is lower than that of SVT-AV1 when used natively at full resolution.

To illustrate the above issue, we compared the relative efficiency of SVT-AV1 (preset 11) vs. x264 (preset faster, which according to [1] had a comparable encode time of SVT-AV1 M11) at 1080p and 540p resolutions. At 1080p resolutions, SVT-AV1 outperforms x264 across the tested range, converging only at very high data rates. On the other hand, at 540p resolutions, SVT-AV1 outperforms x264 only at lower qualities, crossing over at around the Constant Rate Factor (CRF) value of 33 [18]. While the high-quality/low-resolution rate-distortion pairs are hardly selected by a convex-hull-optimized encoder, the LCEVC encoder requires that the base encoder operate at lower resolutions and at higher qualities (e.g., CRF values below 33), where SVT-AV1 does not even outperform x264. Therefore, the non-optimized SVT-AV1 base encoder penalizes LCEVC SVT-AV1 when compared to SVT-AV1 at full resolution.

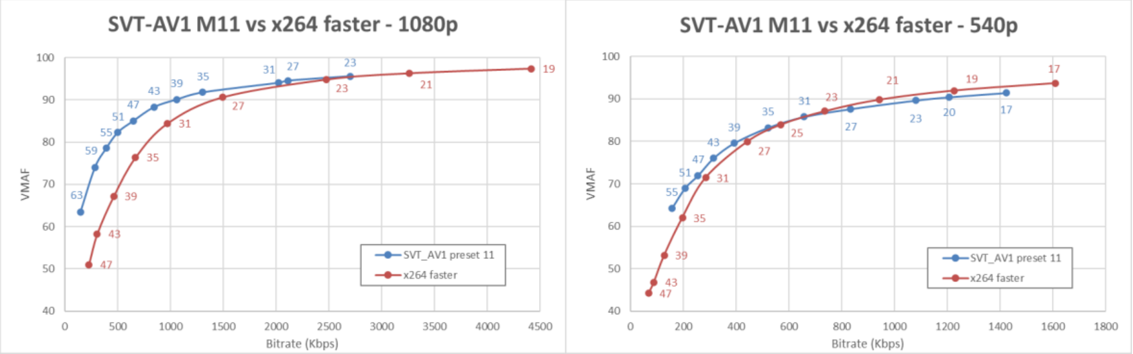


Figure 2 –VMAF rate-distortion curves for SVT-AV1 M11 vs. x264 faster at 1080p and 540p resolutions

Addressing this issue by improving SVT-AV1 at low-resolution/ high-quality operating points improves the performance of LCEVC SVT-AV1 and should be targeted as a future optimization opportunity. Figure 3 below shows the potential impact on LCEVC SVT-AV1 of a patched version [19] with some experimental changes (marked as ‘SPIE_patch’) to preset 11 that optimize low-QP/ low-resolution operating points.

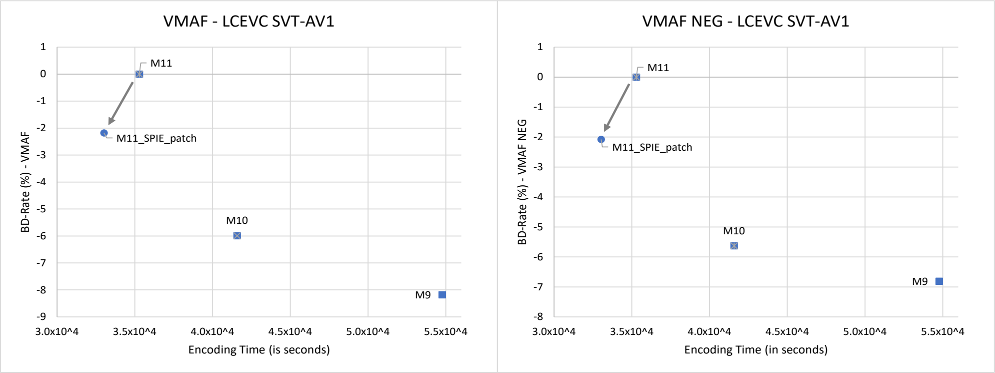


Figure 3 – Impact of the SPIE_patch on BD-rate-computations trade-offs for LCEVC- SVT-AV1 on M11

For the M11 preset, using the patched version generated a BD-rate improvement of over 3% while also reducing encoding time, resulting in a better BD-rate-computation trade-off. Similar improvements for presets below 11 should be possible but could not be completed in time for this paper. Therefore, we used the official release 1.1 of SVT-AV1 for all presets below 11, leaving further improvements of low-QP low-resolution operating points as a potential upside for future releases.

4. COMBINED CONVEX HULL APPROACH

This section presents the evaluation results of the LCEVC enhancements to SVT-AV1, x264 and x265 encoders using the combined (continuous, unrestricted) DO approach discussed in paragraphs 3.2 in [1]. Rather than simply averaging BD-rates across shots, we started from the combined approach to focus the evaluation on a scenario that better reflects expected gains in practical applications.

As described in Figure 4, the rest of this section presents a comparison of the quality vs. complexity trade-offs of different production grade encoder software implementations representing AVC, HEVC, and AV1 video coding standards against their respective LCEVC-enhanced versions.

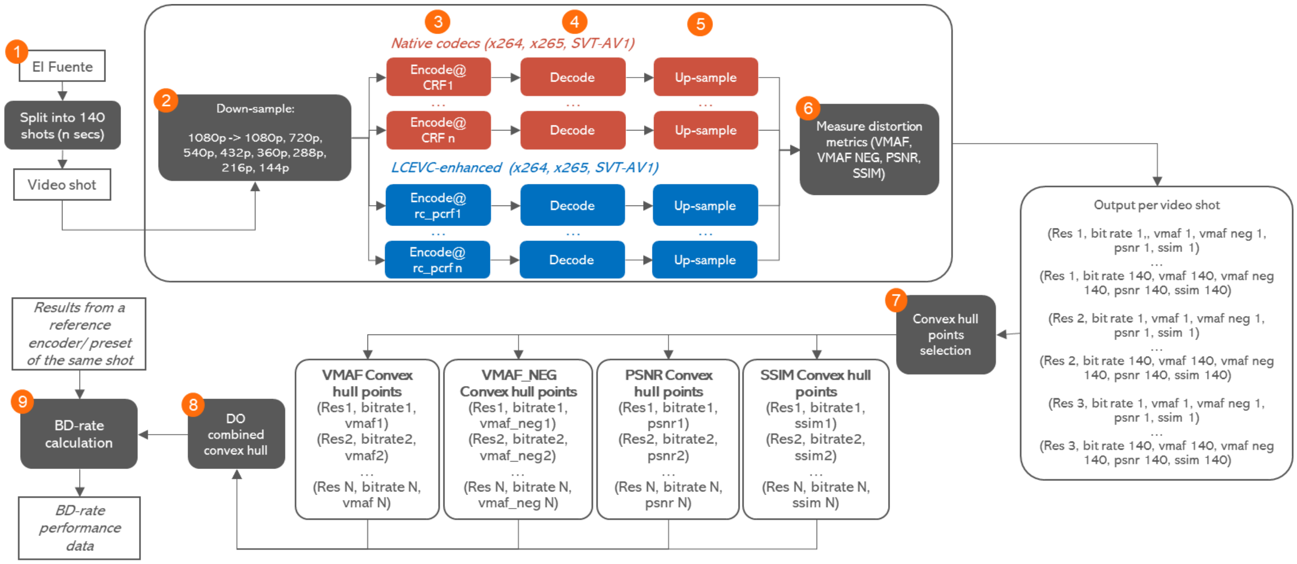

1
El Fuente
Split into 140 shots (n secs)
Video shot
2
Down-sample:
1080p -> 1080p, 720p, 540p, 432p, 360p, 288p, 216p, 144p
3
4
5
Native codecs (x264, x265, SVT-AV1)
Encode@ CRF 1
Decode
Up-sample
...
Encode@ CRF n
Decode
Up-sample
LCEVC-enhanced (x264, x265, SVT-AV1)
Encode@ rc_pcrf 1
Decode
Up-sample
...
Encode@ rc_pcrf n
Decode
Up-sample
6
Measure distortion metrics (VMAF, VMAF NEG, PSNR, SSIM)
Output per video shot
(Res 1, bit rate 1,, vmaf 1, vmaf neg 1, psnr 1, ssim 1)
...
(Res 1, bit rate 140, vmaf 140, vmaf neg 140, psnr 140, ssim 140)
(Res 2, bit rate 1, vmaf 1, vmaf neg 1, psnr 1, ssim 1)
...
(Res 2, bit rate 140, vmaf 140, vmaf neg 140, psnr 140, ssim 140)
(Res 3, bit rate 1, vmaf 1, vmaf neg 1, psnr 1, ssim 1)
...
(Res 3, bit rate 140, vmaf 140, vmaf neg 140, psnr 140, ssim 140)
7
Convex hull points selection
VMAF Convex hull points
(Res1, bitrate1, vmaf1)
(Res2, bitrate2, vmaf2)
...
(Res N, bitrate N, vmaf N)
VMAF_NEG Convex hull points
(Res1, bitrate1, vmaf_neg1)
(Res2, bitrate2, vmaf_neg2)
...
(Res N, bitrate N, vmaf_neg N)
PSNR Convex hull points
(Res1, bitrate1, psnr1)
(Res2, bitrate2, psnr2)
...
(Res N, bitrate N, psnr N)
SSIM Convex hull points
(Res1, bitrate1, ssim1)
(Res2, bitrate2, ssim2)
...
(Res N, bitrate N, ssim N)
8
DO combined convex hull
9
BD-rate calculation
Results from a reference encoder/ preset of the same shot
BD-rate performance data

Figure 4 - Combined (continuous, unrestricted) encoding workflow

4.1 Selection and preparation of the test video clips

To perform our experiments, we used the open source El Fuente clip [20], which contains 14296 frames with a frame rate of 29.97 fps corresponding to 7 min 52 sec of premium VOD content (at the spatial video resolution of 1920x1080). The partitioning of the video into 140 shots and spatial down-sampling are described in [1].

4.2 Encoders and encoding configurations (native codecs and LCEVC-enhanced)

To give a broader view of LCEVC capabilities, in addition to SVT-AV1, two other open-source encoders representing production grade software implementations of AVC and HEVC video coding standards have been used, both in their native and LCEVC implementations. The tested encoders were the following: x264 (0.163.3060), x265 (3.5), SVT-AV1 (1.1). V-Nova LCEVC SDK 3.7 was used for LCEVC-enhanced encoding. FFMPEG v5.0 was used for all native and enhanced codecs.

4.2.1 CRF values

The Constant Rate Factor (CRF) mode was used for all encoders, with selection of CRF values in line with [1]. The following CRF values were applied to the native codecs:

- SVT AV1: 11 CRF values ranging from 23 to 63 at equally spaced intermediate points, resulting in (23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63).
- x264 and x265: 11 CRF points between CRF 19 and 41, resulting in (19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 41).

The LCEVC encoder has an equivalent rate control mode to CRF, named ‘pCRF’ [21]. The selection of pCRF values was made to cover bitrates and qualities similar to those the native encoders. More specifically, the values are the following:

- LCEVC SVT-AV1: 11 pCRF values ranging from 21 to 42, resulting in (21, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42).
- LCEVC x264 and LCEVC x265: 11 pCRF values ranging between 21 and 44, resulting in (21, 22, 24, 26, 28, 30, 32, 34, 36, 38, 41, 44).

4.3 Presets

A representative selection of commonly used presets was tested, excluding the fastest presets (i.e., x264 ultrafast, superfast), and the slowest preset (i.e., SVT-AV1 M0), as they are less relevant for real-life applications:

- x264/x265 (native and LCEVC-enhanced): veryfast, faster, fast, medium, slow, slower, veryslow.
- SVT-AV1 (native and LCEVC-enhanced): M11, M10, M9, M8, M7, M6, M5, M4, M3, M2, M1.

Note: “lcevc_preset” is similar to the preset configuration of other codecs (i.e., x264, x265, AV1), and provides six discrete combinations of encoding parameters to optimize speed and video quality trade-offs. The options are from 0 to 5, where 0 is the slowest while achieving the maximum compression efficiency, and 5 is the fastest. The following mapping between LCEVC (base codec) preset and lcevc_preset has been applied.

Table 1 - List of lcevc_preset values corresponding to LCEVC base layer presets

x264/ x265 preset	**superfast**	**veryfast**	**faster**	**fast**	**medium**	**slow**	**slower**	**veryslow**
LCEVC x264	Not tested	4	2	2	2	1	1	0
LCEVC x265	4	1	1	1	1	0	0	0
SVT-AV1 preset	**M11**	**M10**	**M9**	**M8**	**M7**	**M6**	**M5**	**M4-M1**
LCEVC SVT-AV1	4	4	4	2	2	2	0	0

4.4 Command lines

The following command lines have been used for the native encoders:

```
x264: ./ffmpeg -y -i input.y4m -g <clip_length+1> -keyint_min <clip_length+1> -crf <crf> -c:v libx264 -threads 1 -preset
<preset> -tune psnr -x264-params scenecut=0 -f h264 output.bin

x265: ./ffmpeg -y -i input.y4m -g <clip_length+1> -keyint_min <clip_length+1> -crf <crf> -threads 1 -c:v libx265 -preset
<preset> -tune psnr -x265-params pools=1:wpp=0:scenecut=0:no-info=1 -f hevc output.bin
```

```
SVT-AV1: ./ffmpeg -y -i input.y4m -crf <QP> -preset <preset> -g <clip_length+1> -svtav1-params
fast-decode=1:lp=1 -threads 1 -c:v libsvtav1 -f webm  output.webm
```

For the LCEVC-enhanced encoders, we aligned the configurations to those of native encoders, by setting LCEVC-equivalent commands, resulting in the following:

```
LCEVC x264: ffmpeg -y -i input.y4m -g <clip_length+1> -b:v 0k -c:v lcevc_h264 -base_encoder x264
-eil_params "lcevc_tune=vmaf_neg;accel_num_worker_threads=0;min-
keyint=<clip_length+1>;rc_pcrf=<pCRF>;threads=1;preset=<preset>;lcevc_preset=<lcevc_preset>;scenecut=0" -f h264
output.bin

LCEVC x265: ffmpeg -y -i input.y4m -g <clip_length+1> -b:v 0k -c:v lcevc_hevc -base_encoder x265
-eil_params "lcevc_tune=vmaf_neg;accel_num_worker_threads=0;min-keyint=<clip_length+1>;rc_pcrf=<pCRF>;numa-
pools=1;wpp=0;preset=<preset>;lcevc_preset=<lcevc_preset>;scenecut=0" -f hevc output.bin

LCEVC SVT-AV1: ffmpeg -y -i input.y4m -b:v 0k -g <clip_length+1> -threads 1 -c:v lcevc_av1
-base_encoder svt_av1 -eil_params
"lcevc_tune=vmaf_neg;accel_num_worker_threads=0;rc_pcrf=<pCRF>;preset=<preset>;lcevc_preset=<lcevc_preset>;fast-
decode=1;logical-processors=1" -f webm output.webm
```

For all tests, only the first frame is encoded as an Intra (Key) frame and all other frames are encoded as Inter frames.

The LCEVC performance heavily relies on the quality of its lower resolution base layer, and it offers a number of scaling mode options for the resolution of the base layer with respect to full resolution. The LCEVC specification allows for the usage of different spatial relations between the base resolution and the full resolution. While LCEVC is typically used with quarter-resolution base layers for full resolutions at or above 720p, at SD and sub-SD resolutions, alternative scaling modes are advisable, such as half-resolution scaling or no scaling.

At particularly low resolutions, such as below 360p, LCEVC is recommended to be used in passthrough mode' LCEVC commands for the low resolutions are the following:

```
LCEVC x264 (<= 360p):  ffmpeg -y -i input.y4m -g <clip_length+1> -b:v 0k -c:v lcevc_h264 -base_encoder x264 -eil_params
"min-keyint=<clip_length+1>;threads=1;preset=<preset>;
scenecut=0;encoding_mode=native;scaling_mode_level0=0D;scaling_mode_level1=0D;tune=psnr;crf=<x264_crf>" -f h264
output.bin

LCEVC x265 (<= 540p):  ffmpeg -y -i input.y4m -g <clip_length+1> -b:v  0k  -c:v lcevc_hevc -base_encoder x265 -
eil_params "min-keyint=<clip_length+1>;numa-
pools=1;wpp=0;preset=<preset>;scenecut=0;encoding_mode=native;scaling_mode_level0=0D;scaling_mode_level1=0D;tune=psnr;cr
f=<x265_crf>" -f hevc output.bin

LCEVC SVT-AV1 (<= 540p): ffmpeg -y -i input.y4m -g <clip_length+1> -threads 1 -c:v lcevc_av1 -base_encoder svt_av1 -
eil_params "preset=<preset>;fast-decode=1;logical-processors=1;encoding_mode=native;
scaling_mode_level0=0D;scaling_mode_level1=0D;qp=<SVT-AV1_crf>" -f ivf output.bin
```

4.5 Platforms used for the experiments

The encodings for this experiment are performed on AWS EC2 instances, specifically, c6i.32xlarge. All instances ran Ubuntu Server 20.04 Linux OS on a Third Generation Intel® Xeon® Platinum 8375C. Hyper-threading and Turbo frequency are both enabled on these instances allowing the instance to access 128 logical cores (vCPU), with a maximum all-core turbo speed of 3.5 GHz. Instances ran from 3000 IOPS, GP3 EBS mounted disks. Running the list of commands is done by invoking the parallel [22] Linux tool and passing to it the command lines above. The parallel tool then schedules the command lines by executing newer commands when older ones retire, while maintaining 128 command lines running at a time. Each set of encodings per preset per encoder is run independently while capturing the run time using the GNU time command as follows:

```
/usr/bin/time --verbose parallel -j 128 < commands_encoder<x>_preset<n>.txt
```

4.6 Metrics

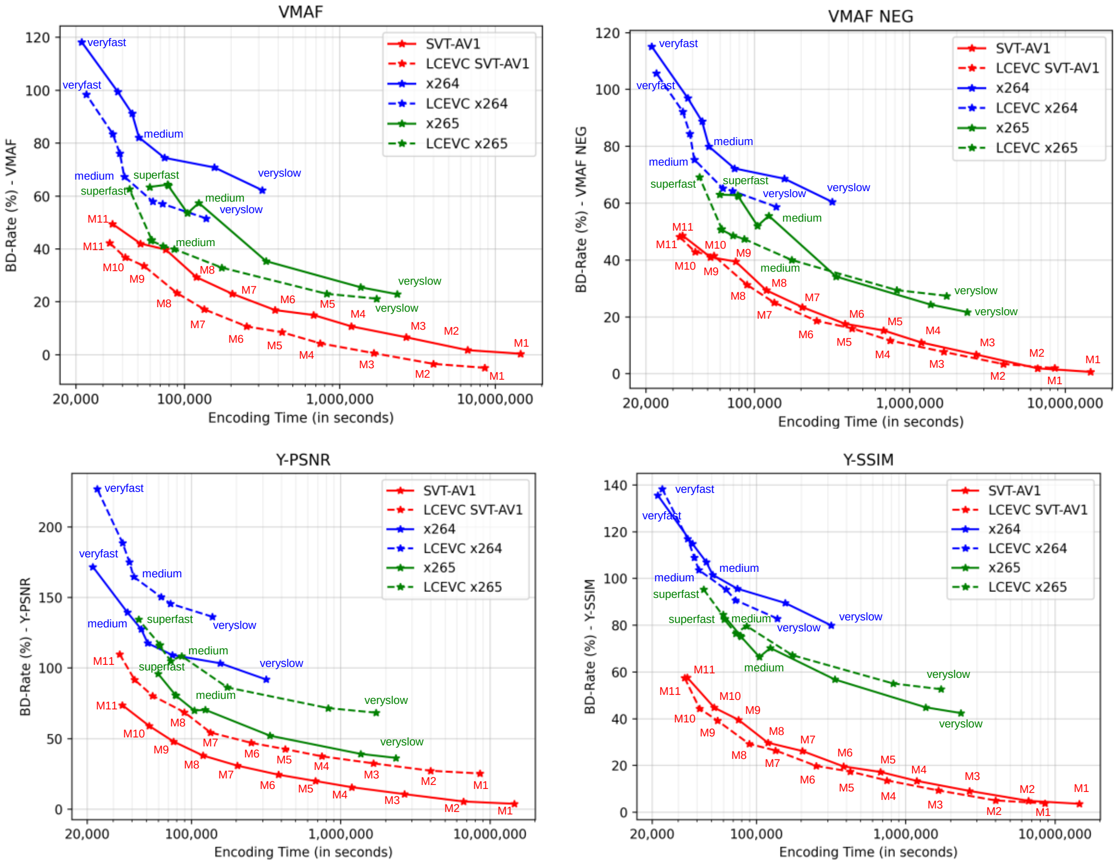
In this analysis, we computed visual quality measurements using four objective metrics: Video Multimethod Assessment Fusion (VMAF), Video Multimethod Assessment Fusion No Enhancement Gain (VMAF_NEG), Peak-Signal-to-Noise-Ratio (Y-PSNR), and Structural Similarity Index (Y-SSIM) [23]. Although perceptual-based objective metrics like VMAF were demonstrated to be good predictors of visual quality, they can be influenced by means of pre- or post-processing operations [24]. For this reason, with respect to [1], VMAF-NEG was added to the set of objective metrics.

4.7 Encoding results

Once all encodings are done, the resulting bitstreams are collected, decoded, and up-sampled to 1080p with the methodology described in [1]. Four performance metrics are then generated using the VMAF software (8b0782c6, based on v2.3.0). In order to select the RD points that would result in the best-possible trade-offs between quality, bit rate and resolution, all resulting elementary bitstream file sizes and quality metrics across all resolutions for each clip are passed to a C sample application [25] that determines the convex hull points. These points are chosen to allow the application to switch between encodings corresponding to different resolutions (based on the available bandwidth) while also maintaining the best possible video quality at a certain bit rate. With respect to the performed simulations, the input to this process is a set of 88 encoding results (8 resolutions * 11 CRF points) per video shot. The BD-rate [26] results for all encodings are generated by comparing the resulting convex hull points for each of the video clips to those generated using libaom AV1 at the slowest preset (i.e., -cpu-used=0, with encoding configurations in line with [1]), which represents the anchor encoder in this comparison.

After obtaining the convex hull for each “shot”, the multiple convex hulls are combined as described in [1] using the constant slope principle. This results in a single rate-distortion curve describing the coding performance over the entire combined sequence. Having two such combined rate-distortion curves, one for the anchor and another for a test encoder, allows the evaluation to be based on a single BD-rate figure that captures the performance over the entire ensemble.

Figure 5 shows the BD-rate vs complexity results of using the combined convex hull approach for VMAF, VMAF_NEG, PSNR, SSIM. Average of (VMAF, PSNR, SSIM), where average curves are calculated by averaging the BD-Rates of multiple metrics for each codec preset, has been added for consistency with [1]. However, in light of the considerations below regarding the reliability of using PSNR to compare single-layer schemes with multi-resolution schemes such as LCEVC-enhanced encoding, the average of (VMAF, PSNR, SSIM) will not be reported any further in the report.



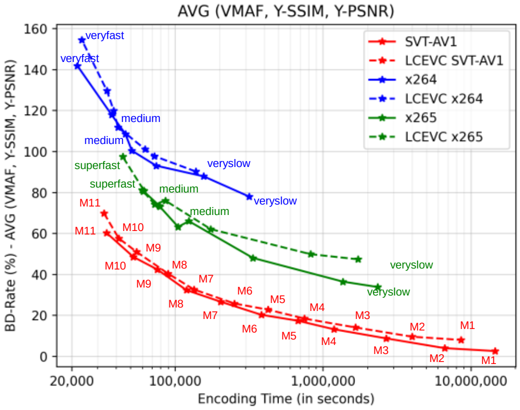


Figure 5 - AVC/HEVC/SVT-AV1 vs. LCEVC-enhanced BD-rate - Computations trade-offs across presets on El Fuente data set using the combined approach

While VMAF, VMAF_NEG and SSIM show an improved BD-rate, PSNR results are quite different from those of other objective metrics, with strong BD-rate penalties. It is important to note that objective metrics based on mean-squared error (MSE), such as PSNR, known to have some limitations in general ([27], [28] and [29]), are especially problematic when comparing conventional single-layer schemes with multi-resolution schemes such as LCEVC. Whilst PSNR may still be used to compare different LCEVC-enhanced codecs among themselves, when comparing a single-layer codec with LCEVC, the different error distribution profiles produce relevant structural disadvantages for the multi-resolution scheme in the range of scores between 35 and 45 dB ([30]). Therefore, BD-rates based on PSNR are in line with expectations and not representative of subjective visual quality. Subjective metrics (ITU-R BT.500 formal MOS performed during the course of MPEG standardization of LCEVC [31]) highlighted that comparisons of LCEVC vs. native coding performed with objective metrics (including VMAF) underestimated the formal MOS BD-rate benefits of LCEVC. In absence of formal subjective results, some objective results in this paper may thus underestimate the visual quality benefits provided by LCEVC.

5. RESTRICTED DISCRETE CONVEX HULL APPROACH

As outlined in [1], the combined single convex hull is very dense and requires a very big number of operating points, which is unrealistic for a practical video service that typically needs to create a so-called “bitrate ladder”, where each step represents a given quality/bitrate for an input video. The “restricted discrete convex hull” approach addresses this challenge and allows to focus on a range of quality points that are relevant for AS applications.

The restricted discrete convex hull approach shares the same first steps of the combined approach.

- After combining and obtaining a single Rate-VMAF curve for the combined sequence clips a series of discrete operating points are chosen in line with [1] at VMAF [30, 40, 50, 60, 70, 80, 90, 95].
- The resulting bitrates for these 8 quality targets are then used to obtain the corresponding points on the Rate-VMAF_NEG, Rate-PSNR and Rate-SSIM curves.
- These 8 points - where quality is one of VMAF_NEG, VMAF, SSIM, PSNR - for the anchor are then compared with the 8 points obtained by the same methodology for the test encoder

Figure 6 below shows the results of using the restricted discrete approach on the BD-rate vs computations results for El Fuente. The chart shows the BD-rate vs computation results for the four metrics (VMAF, VMAF_NEG, PSNR, SSIM).

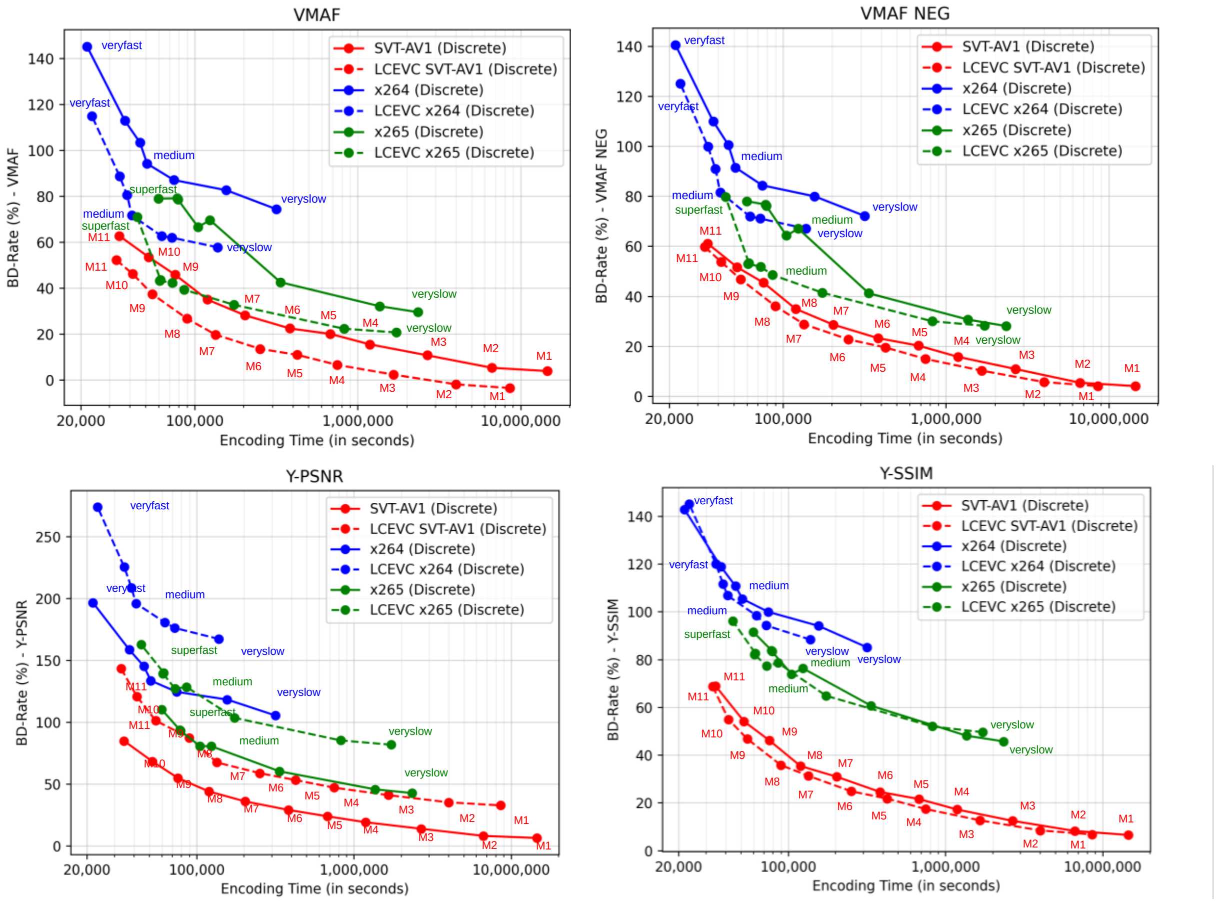


Figure 6 - AVC/HEVC/SVT-AV1 vs. LCEVC-enhanced BD-rate - Computations trade-offs with discrete DO

6. FAST ENCODING PARAMETER SELECTION FOR SVT-AV1 (NATIVE AND ENHANCED)

6.1 Assessing compute complexity

The actual encodings that are part of the convex hull for a given sequence are only a subset of the total encodings generated, which means we are using a subset of the encodings for BD-rate calculations, while we use all encodings for computational complexity calculations, as we don't know a priori which of the (resolution, QP) pairs would be suboptimal. The additional encodes that are not part of the convex hull can be considered as a compute tax to achieve optimality. Section 4 in [1] demonstrated an opportunity to greatly reduce computational complexity in the "restricted discrete" case, if we have a good way to generate estimates for the optimal (resolution, QP) pairs. In that case, the complexity of such an optimized bitrate ladder generation would be equal to the complexity of producing only those 8 (resolution, QP) encodings for each clip that correspond to the final 8 selected aggregate discrete points, plus the cost of the predictor of these encoding parameters.

As shown in [32, 33], one option is to use the same encoder in a faster setting to obtain the convex hull. As demonstrated in [1] for SVT-AV1, this introduces only a minor loss in coding efficiency with a very significant improvement in speed.

6.2 Fast encoding parameter selection

At the same target quality level, a faster and a slower encoder preset differ primarily in bitrate, where the bitrate difference is proportional to the content complexity, while distortion stays roughly the same. Thus, a constant multiplicative factor exists for the slopes in the RD domain between a faster and a slower encoder preset, and that enables a faster encoder to be used to encode the shots at multiple operating points and determine the optimal resolutions and target quality levels for each shot, which can then be used to encode the shot again with a slower preset.

The steps involved in fast encoding parameter selection approach are as follows:

- Compute the “combined convex hull” (as in Sec. 4), but with a faster preset A. This would give us X (rate, quality) operating points, with an associated list of (resolution, CRF/pCRF) parameter pairs to achieve each operating point.
- From the list of encoding parameters for each point, we can predict encoding parameters for a slower preset B.
- From the predicted encoding parameters, we can encode with the slower preset and obtain the 8 operating points for the bitrate ladder construction.

In [1], based on SVT-AV1 version (0.8.8-rc), the fastest available speed setting (M8) was roughly at the level of x264 at the “slower” preset, meanwhile the most recent SVT-AV1 releases unlocked faster presets (M9 to M12), with M12 achieving a comparable speed of x264 “veryfast”. In the following experiments, we perform the “analysis” step to determine the optimal encoding parameter pairs starting with M8 (for consistency with [1]) on SVT-AV1 and on the base layer of LCEVC-enhanced SVT-AV1, and then apply them to other slower speed settings from M7 to M1 (slowest).

Figure 7 shows the results of using the M8 preset in the analysis stage on the BD-rate vs complexity trade-offs of the rest of the SVT-AV1 presets. The total encoding time shown on the x-axis is the sum of the *encoder parameter estimation* ***cost***, plus the ***cost*** *of the selected* X *encodings per shot*.

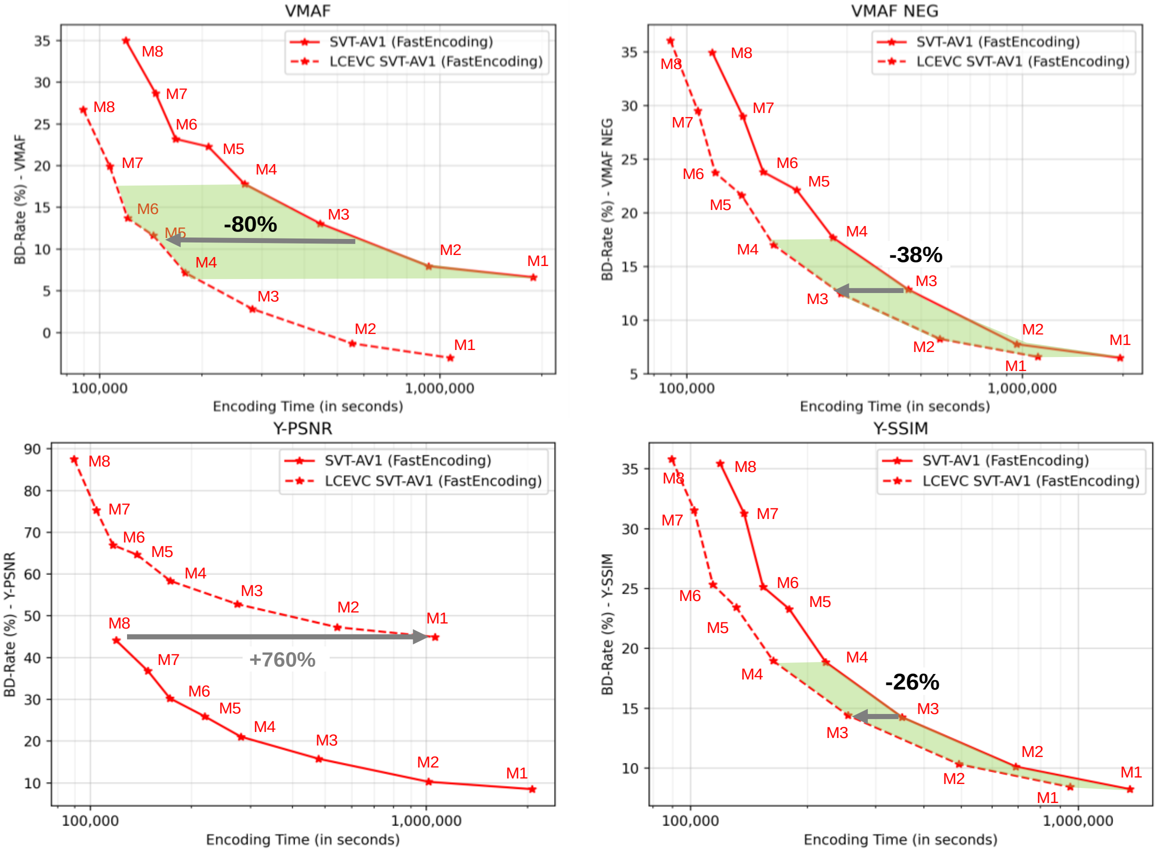


Figure 7 – Fast Encoding approach: BD-rate vs complexity trade-offs when M8 is used for the prediction

To better understand the effect of choosing various presets during the analysis phase, we also repeated the above experiment several times, each time choosing a different preset. In particular, we predicted the encoding parameters (CRF and resolution) of a target preset, M_x, using a reference preset, M_y. We denote the result as M_y | M_x. As the reference preset needs to be faster than the predicted preset, we require that y > x. We then computed the convex hull, constructed after considering all possible combinations M_y | M_x. Figure 7 shows the corresponding convex hull for SVT-AV1 and LCEVC SVT-AV1, as well as the optimal combinations M_y | M_x (from M3 to M11). Please note that, for some presets, the optimal combinations are not part of the convex hull, in which case we took the closest points to it.

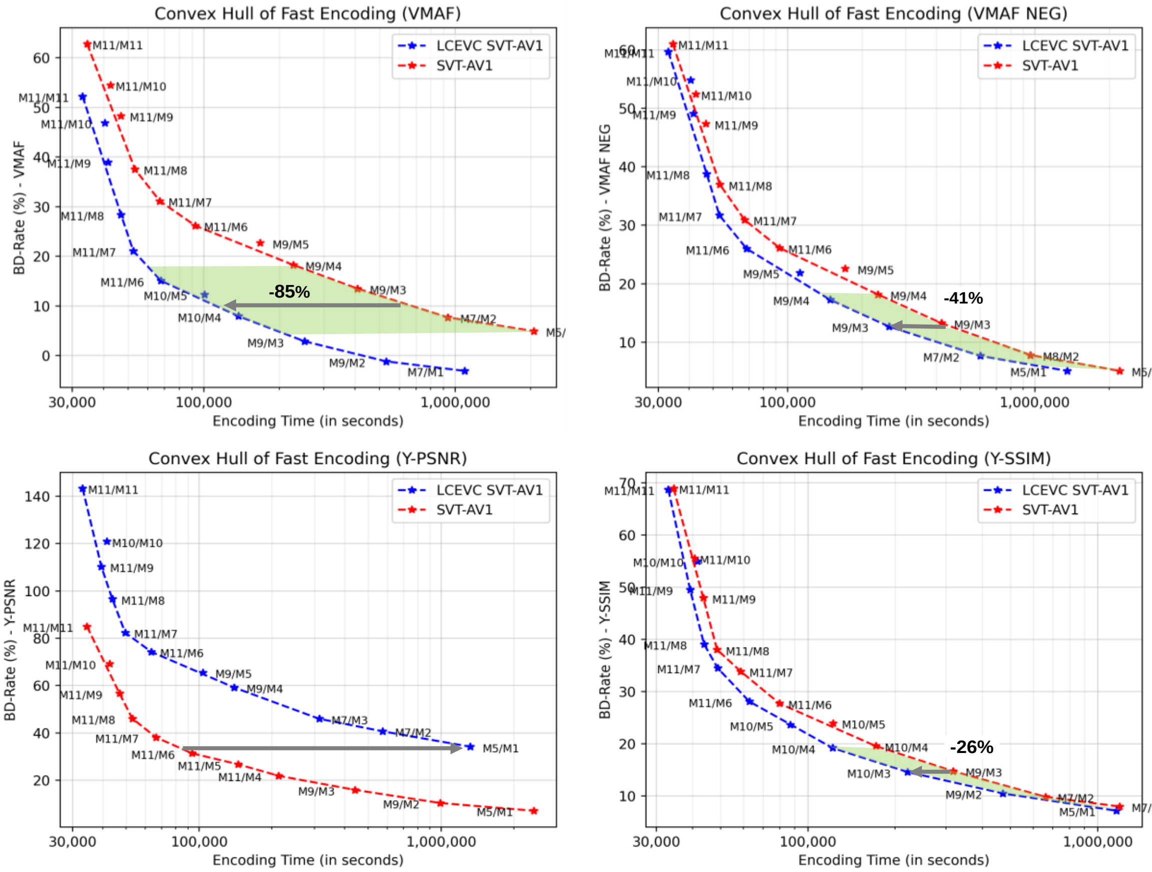


Figure 8 - Fast Encoding approach: BD-rate vs complexity trade-offs, with optimal M_y | M_x preset combinations

Figure 9 shows the comparisons between the discrete DO calculated in Sec. 5 and fast encoding DO for LCEVC SVT-AV1 and SVT-AV1 with M8 and M11 used for prediction. The fast encoding approach, consistently between the two tested codecs, significantly reduces the total encoding time needed while maintaining a limited BD-rate loss.

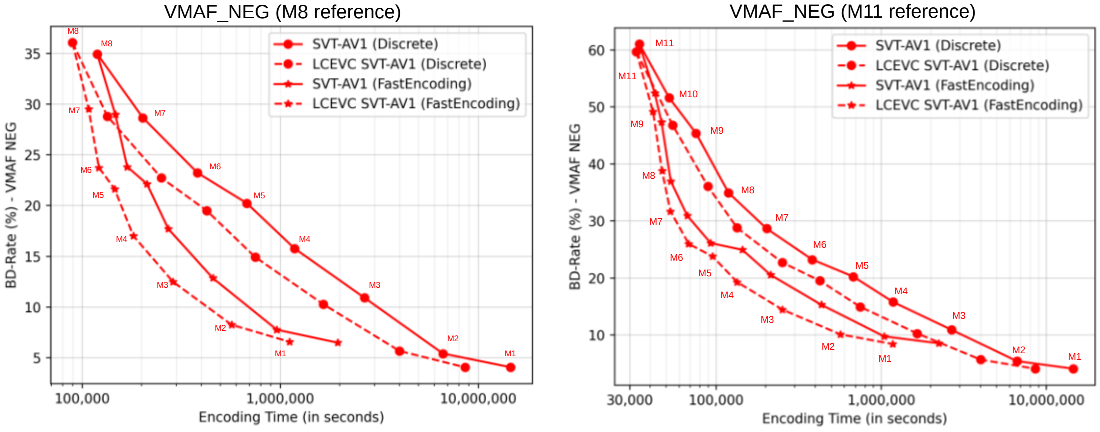


Figure 9 – Comparison of fast encoding approach vs discrete approach with M8 and M11 preset for prediction

In Table 2, we compared the fast encoding DO with optimal preset for prediction (as per Figure 7) against the discrete DO for each preset and computed the percentage saving of cycles and the corresponding BD-rate loss. The benefits of fast encoding are confirmed for LCEVC, it enables both AV1 and LCEVC AV1 to achieve a 75-85% saving in encoding time for the five slowest presets, with a limited BD-rate loss, between 1%-2.4% for both codecs.

Table 2 - Average BD-rate loss (VMAF_NEG) and cycle usage vs discrete convex hull

Native SVT-AV1 (Optimal convex hull reference)		
Preset	BD-rate cost of fast encoding	% of cycles used by fast encoding vs discrete CH
M5 | M1	1.0%	15%
M8 | M2	2.4%	14%
M9 | M3	2.3%	16%
M9 | M4	2.4%	20%
M9 | M5	2.3%	25%
…	…	…
M11 | M11	0%	100%

LCEVC SVT-AV1 (Optimal convex hull reference)		
Preset	BD-rate cost of fast encoding	% of cycles used by fast encoding vs discrete CH
M5 | M1	1.0%	16%
M7 | M2	1.9%	15%
M9 | M3	2.4%	15%
M9 | M4	2.3%	20%
M9 | M5	2.3%	26%
…	…	…
M11 | M11	0%	100%

7. LCEVC AV1 DECODING POWER AND BATTERY CONSUMPTION

7.1 Why efficient AV1 decoding in software is important

Increasingly efficient software (SW) and hardware (HW) AV1 decoders are becoming available for playback on mobile devices. However, despite increasing adoption of state-of-the-art SW decoder by streaming operators and the commitment of chipset manufacturers to add AV1 capabilities in their SoC, as of early 2022 there is only a very limited set of devices with hardware accelerated AV1 decoding. It could be many years until the majority of global consumers can decode high-quality HD AV1 content on their mobile devices. Even when this is possible, battery power consumption for SW AV1 decoders on mobile devices is often unsustainable, draining their battery faster than hardware decoding.

For this reason, in addition to the benefits in the encoding quality-computational cycles trade-off, LCEVC's decoding performance was assessed to determine its capabilities to extend the device reach of AV1 and to improve battery life, with specific focus on Android mobile devices. Due to the LCEVC hierarchical structure, the LCEVC-enabled decoder decodes the AV1 base layer at a quarter resolution (e.g., 540p for a full resolution HD stream) and leverages existing hardware blocks in the device, such as GPU shaders and scalers, to efficiently decode the LCEVC enhancement.

7.2 Decoder performance testing

Two approaches have been used to compare decoder performance of AV1 and LCEVC with AV1.

Method one measured CPU decoding time as a proxy for overall decode complexity, using an approach similar to that of the encoder performance evaluation. It should be noted that for LCEVC this test used CPU-only decoding libraries that – since they are not used in real-world scenarios – have not undergone significant code-optimization.

Method two measured actual playback capability and overall power usage on real-world mobile devices when they playback video content. For this test, AV1 decoding was carried out using the latest available software decoders (both Dav1d and GAV1) and LCEVC AV1 decoding used optimized CPU-GPU libraries, making results more indicative for real-world use cases. For this test, we used a video playback application capable of decoding and playing back both AV1 and LCEVC-enhanced AV1 content, based on ExoPlayer [34] with added support for the latest Dav1d (version 1.0) and GAV1 software decoders as provided by Android's MediaCodec interface (further details in 7.5). For native AV1 decoding we also performed sanity checks to verify that the performance of ExoPlayer+Dav1d was in line with VLC+Dav1d.

7.3 Decoder performance test content

The test content is a concatenation of the El Fuente shots encoded according to command lines and [resolution, CRF/pCRF] parameters generated using the discrete DO methodology described in Sec. 5, with specific focus on quality levels at 1080p and 720p. Importantly, the *fast-decode* SVT-AV1 flag was active for both AV1 and LCEVC AV1 encoding of all bitstreams below M11 (fast-decode is not currently supported by M11).

For method one (CPU decoding efficiency) we generated two sets of El Fuente shots concatenations, in both cases based on discrete DO selection according to VMAF_NEG-optimized convex hull, presets M3 to M11:

i) **1080p height match:** selecting matching sections for AV1 and LCEVC AV1 resulting in 1080p resolution, in order to show results for same-resolution contents. For this test we focused on concatenations at VMAF quality levels 95 and 90 from the discrete restricted analysis, for which 1080p was selected for over half of the shots.

ii) **Full set:** concatenating all 140 shots, at the resolution determined by the discrete DO. In order to focus the decoder analysis on high quality we limited the VMAF quality levels to 95, 90, 80, 70, 60.

For method two (playback test on mobile phones), we focused the analysis on high end qualities and resolutions, at VMAF 95 quality level. For the sake of efficiency, two representative presets have been chosen: M3 and M8 for both AV1 and LCEVC AV1. In line with method one, the tests have been executed on "1080p height match" concatenations and on "full set" concatenations, at resolutions selected by the DO and resulting bitrates illustrated in Table 3.

Table 3 – Average bitrate in Kbps of the El Fuente concatenations used for mobile tests

	LCEVC AV1		AV1	
Preset	**Full set**	**1080p Height match**	**Full set**	**1080p Height match**
M3	1856	1373	2158	1571
M8	2504	2185	2710	2310

For both codecs the bitrates of M3 concatenations are between 15% and 37% smaller than the equivalent M8 (for comparable VMAF 95 quality), hence M8 bitstreams are expected to be more challenging for the decoders.

In addition to the El Fuente content at 30 fps used for the encoder tests, in order to test the limits of AV1 playback and test LCEVC's potential to improve AV1 decoding capabilities, we expanded playback tests to El Fuente content at 60 fps. The 60 fps test sequence has been prepared using the [resolution, crf] points selected by DO at 30 fps (optimized according to VMAF_NEG), focusing on M3 preset and VMAF 95 quality level. The test focused on 1080p60, concatenating sequences for AV1 and LCEVC AV1 using the "1080p height matched" approach described above.

7.4 Decoder CPU efficiency testing

To measure decoder CPU efficiency, the following set of video decoder software was used:

- FFMPEG version 5.0, with support for both LCEVC and Dav1d AV1 decoder.
- Dav1d version 1.0.
- V-Nova LCEVC SDK 3.7, as the latest version of LCEVC used for encoder testing.

All target software has been built for Ubuntu and Android operating systems. Matching the encoder efficiency methodology, we simultaneously execute FFMPEG, single threaded, for each processor core of the host computer, using Gnu Parallel utility and using /usr/bin/time is used to capture the CPU execution time.
The following command lines have been used to execute the test runs.

- AV1 playback measurements:
  ```
  /sys/bin/time ffmpeg -loglevel error –thread_count 1 -base_decoder
  libdav1d -i content.webm -vf scale=1920x1080:flags=bilinear -f null /dev/null
  ```
- LCEVC with AV1 playback:
  ```
  /sys/bin/time ffmpeg -loglevel error -vcodec lcevc_av1_dpi -thread_count 1
  -allow_passthrough 1 -i content.webm -vf scale=1920x1080:flags=bilinear -f null /dev/null
  ```
-

CPU utilization measurements were performed on AWS C6gd.16xlarge instance with Ubuntu 20.04.

7.4.1 Decoder CPU efficiency results

CPU utilization results on the C6gd 16xlarge instance have been collected across all SVT-AV1 presets from M3 to M11. The CPU time measurement (user+sys decode time) has been recorded for both AV1 and LCEVC AV1, then the LCEVC gain is represented as *(CPU time LCEVC AV1/ CPU time AV1)-1*.

Table 4 illustrates the results for the "1080p height match" methodology on VMAF 90 and 95 qualities, where only 1080p content was used for the test from matching sections for the two tested codecs.

Table 4 - CPU time gain for LCEVC AV1 vs. AV1 decoding using the "1080p height match" approach

Preset	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**	**M9**	**M10**	**M11**
VMAF90	-19%	-20%	-21%	-19%	-20%	-24%	-26%	-25%	-28%
VMAF95	-25%	-25%	-29%	-26%	-24%	-29%	-30%	-28%	-27%

On 1080p contents, the LCEVC gain varies with presets and quality levels, oscillating between 19% and 30%. The stronger gains are focused on higher qualities combined with faster presets (e.g., VMAF 95 and presets above M8).

Table 5 illustrates results for the “full set”, using the concatenation of the full El Fuente sequence at the resolution selected by the DO and using bilinear upsampler to upscale to 1080p.

Table 5 - CPU time gain for LCEVC AV1 vs. AV1 decoding using the “full set” approach and bilinear upsampler

Preset	M3	M4	M5	M6	M7	M8	M9	M10	M11
VMAF60	-12%	-12%	-16%	-27%	-31%	-36%	-41%	-43%	-47%
VMAF70	-34%	-38%	-38%	-41%	-45%	-47%	-51%	-47%	-44%
VMAF80	-47%	-44%	-44%	-45%	-44%	-41%	-38%	-39%	-32%
VMAF90	-38%	-37%	-34%	-34%	-34%	-30%	-30%	-30%	-29%
VMAF95	-33%	-32%	-29%	-27%	-27%	-26%	-28%	-30%	-31%
Average VMAF	-33%	-33%	-32%	-35%	-36%	-36%	-38%	-38%	-37%

In this case, LCEVC gains on AV1 range between 32% and 38% depending on the preset.

7.5 Real-life device testing

To relate decoder CPU measurements to real life playback capabilities and battery usage, power measurements and rendered frame count have been captured using Android based mobile devices with a video playback application capable of decoding and playing back both AV1 and LCEVC AV1. Android and ExoPlayer provide several APIs to enable current drain and battery voltage data to be collected, along with information about the number of frames rendered/dropped during video playback. To enable objective comparisons between AV1 and LCEVC AV1 playback sessions, the following metrics are gathered per playback session: total number of frames, average milliwatts of battery power drain, total number of frames decoded during the total duration of play, number of frames dropped during the total duration of play. Each target playback device was put into the same condition before each test run, to mitigate external factors effecting test runs.

- Test video contents stored locally on device.
- Flight-mode enabled (i.e., WIFI, Cellular, Bluetooth and other radios off; location based services disabled.
- Screen brightness set to 50% and automatic brightness disabled.
- The device battery level charged to 100%; battery charger disconnected.

The following performance indicators are then calculated and used:

1. Playback continuity: Percentage of frames dropped throughout the full sequence, calculated as *[(total frames (tb) - RenderedBuffers (rb))/ Total frames].*
2. Power consumption: In situations where both AV1 and LCEVC AV1 were equally not dropping frames measurement of average power consumption in mWatts (product sampled of voltage and current).
3. Battery drain: 60 minutes playback tests on a subset of devices for which both AV1 and LCEVC AV1 were not dropping frames. Battery level has been measured at intervals of 5 minutes starting with fully loaded battery.

7.5.1 Test player and test framework

ExoPlayer 2.15.1 was used as the base player with additional extensions to add support for LCEVC SDK 3.7 using V-Nova’s standard LCEVC Extension (designed to work with all codecs supported by ExoPlayer) and an extension to enable AVCodec to add support for Dav1d (v 1.0) and LCEVC in the context of a media player. In all cases the same software build is used to run both AV1 and LCEVC AV1. To check the ExoPlayer integration of Dav1d was representative from a performance point of view, frame drop data was compared with VLC playing back the same AV1 encodings.

In the first set of tests, we compare Dav1d against LCEVC using ExoPlayer on both 30fps and 60fps test content. In the second set of tests, we compare Android’s native software implementation of AV1, GAV1, against LCEVC using V-Nova’s standard ExoPlayer extension integration. Since GAV1 was only made available for Android 10 devices onward, only data from a subset of devices could be collected (details in Table 6).

Test framework. To execute the mobile tests, a Appium based frameworks was used, devices were connected to the host PC using remote controlled USB switches so that USB and therefore power was removed from the mobile device whilst tests were executed. The test framework reduced human error and allowed testing to be carried out in parallel on multiple devices.

7.5.2 Test devices

Table 6 illustrates the tested Android mobile devices, representative of high- mid- to low-end specs devices.

Table 6 - Tested Android devices and related specifications (source: www.gsmarena.com)

						Test performed		
#	**Device model**	**Year**	**Android Version**	**Chipset**	**GPU**	**Dav1d (p30)**	**GAV1 (p30)**	**Dav1d (p60)**
1	Google Pixel 4	2015	12	Qualcomm SM8150 Snapdragon 855(7 nm)	Adreno 640	✓	✓	✓
2	Google Pixel 4a	2020	11	Qualcomm SDM730 Snapdragon 730G(8 nm)	Adreno 618	✓	✓	✓
3	Google Pixel 5	2020	12	Qualcomm SM7250 Snapdragon 765G(7 nm)	Adreno 620	✓	✓	✓
4	Huawei P8	2015	5	Kirin 930 (28 nm)	Mali-T628 MP4	✓		
5	Motorola Moto E5 Plus	2018	8	Qualcomm MSM8917 Snapdragon 425 (28 nm)	Adreno 308	✓		
6	Motorola Moto G4	2016	7	Qualcomm MSM8952 Snapdragon 617 (28 nm)	Adreno 405	✓		
7	Nokia 8.3 5G	2020	12	Qualcomm SM7250 Snapdragon 765G 5G (7 nm)	Adreno 620	✓	✓	✓
8	Nokia G50	2021	12	Qualcomm SM4350 Snapdragon 480 5G (8 nm)	Adreno 619	✓	✓	✓
9	Oppo A1k	2019	9	Mediatek MT6762 Helio P22 (12 nm)	PowerVR GE8320	✓		
10	Samsung Galaxy A13	2022	12	Exynos 850 (8nm)	Mali-G52			✓
11	Samsung Galaxy A20E	2019	11	Exynos 7884 (14 nm)	Mali-G71 MP2			✓
12	Samsung Galaxy A50	2019	11	Exynos 9610 (10nm)	Mali-G72 MP3	✓	✓	✓
13	Samsung Galaxy A8	2017	9	Exynos 7885 (14 nm)	Mali-G71			✓
14	Samsung Galaxy J8	2018	10	Qualcomm SDM450 Snapdragon 450(14 nm)	Adreno 506	✓		✓
15	Samsung Galaxy S5	2014	6.0.1	Qualcomm MSM8974AC Snapdragon 801(28 nm)	Adreno 330	✓		
16	Samsung Galaxy S9 (EU)	2018	10.0	Exynos 9810 (10 nm)	Mali-G72 MP18	✓	✓	✓
17	Sony Xperia 10III	2021	12	Qualcomm SM6350 Snapdragon 690 5G (8 nm)	Adreno 619	✓	✓	✓
18	Sony Xperia Z2	2014	6.0.1	Qualcomm MSM8974AB Snapdragon 801 (28 nm)	Adreno 330	✓		
19	Vivo V9	2018	9	Qualcomm MSM8953-Pro Snapdragon 626 (14 nm)	Adreno 506	✓		✓
20	Xiaomi Redmi 7	2019	10	Qualcomm SDM632 Snapdragon 632	Adreno 506	✓	✓	✓
21	Xiaomi Redmi 8A	2019	10	Qualcomm SDM439 Snapdragon 439 (12 nm)	Adreno 505	✓		✓
22	Xiaomi Redmi Go	2019	8.1.0	Qualcomm MSM8917 Snapdragon 425 (28 nm)	Adreno 308	✓		
23	Xiaomi Redmi N8T	2019	11	Qualcomm SDM665 Snapdragon 665 (11 nm)	Adreno 610	✓	✓	✓
24	Xiaomi Redmi Note 9	2020	11	MediaTek Helio G85 (12nm)	Mali-G52 MC2	✓		✓

7.5.3 Real life device results – El Fuente (30 fps), ExoPlayer + Dav1d

The chart below illustrates test result for the “1080p height match” approach in terms of playback continuity and power consumption, to decode AV1 and LCEVC AV1 1080p content (M3 and M8 presets).

The left side of Figure 9 shows the percentage of dropped frames vs. total across the full set of tested devices. On less powerful devices (left), the player struggled to playback 1080p content with native AV1 (more so with M8 due to higher bitrates vs. M3 AV1), resulting in stuttering on complex scenes, occasional freezes and in certain cases, such as with Huawei P8, Motorola G4, Sony Experia Z2 and Samsung Galaxy S5, in playback failures. With LCEVC AV1 playback was smooth, with less than 0.1% of dropped frames across all devices.

For devices that played back smoothly AV1 and LCEVC AV1, we recorded power consumption (in mWatts) and reported on the right side of Figure 10 the AV1 power overhead as a percentage increase vs. LCEVC AV1. The power overhead of AV1 vs. LCEVC AV1 indicates the percentage battery life extension of LCEVC AV1 vs. AV1.

AV1 playback consistently consumed more power than LCEVC AV1 playback, on average 21% more with M3 and 19% more with M8, resulting in faster battery drain. The overall power figures include not only video decoding, but also all other battery-draining features of the device (most notably the screen), which are common for both AV1 and LCEVC AV1 playback and thus dilute the relative power saving of LCEVC on decoding.

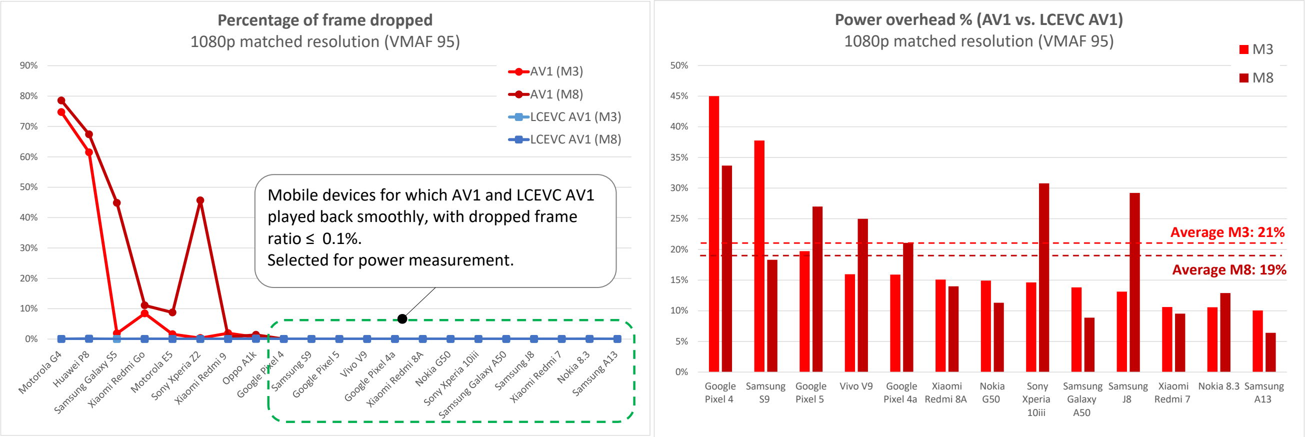


Figure 10 - "1080p matched" approach: percentage of frames dropped and power measurement (Dav1d)

Figure 11 shows the same indicators when the full El Fuente test sequence is played, each shot concatenated at the resolution determined by the DO. Notice that LCEVC played on average a higher resolution.

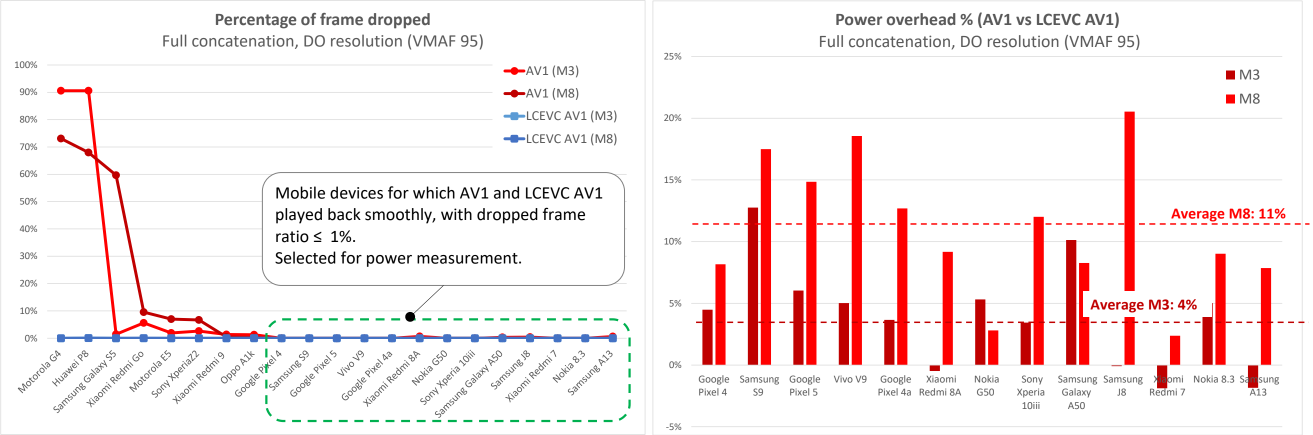


Figure 11 - "Full set" approach: percentage of frames dropped and power measurement (Dav1d)

While the frame dropped scenario is consistent with the 1080p-only results on less powerful devices, the LCEVC power saving is slightly diluted vs. the 1080p scenario. On average, native AV1 playback consumes 4% more power than LCEVC AV1 on M3 and 11% more with M8.

7.5.4 Real life device results – El Fuente (30 fps), ExoPlayer + GAV1

Figure 12 on the left side illustrates the playback test results for the "1080p height match" approach, using ExoPlayer with GAV1 software decoder on AV1 and LCEVC AV1 (M3 and M8 presets), for a subset of 10 devices capable of running GAV1.

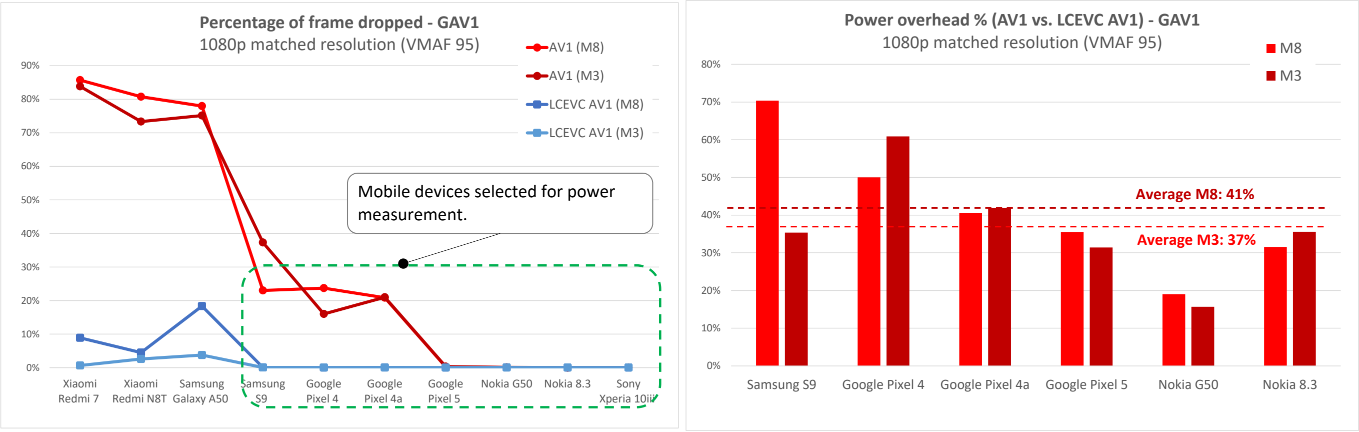


Figure 12 - Percentage of frames dropped with "1080p matched" approach (GAV1 decoder)

Using GAV1 decoder with ExoPlayer, native AV1 playback at 1080p drops frames to the point of not being watchable on the majority of tested devices. On Redmi 7, Redmi N8T and Galaxy A50 AV1 does not play at all, resulting in a frozen frame a few seconds after playback start; on Pixel 4, Samsung S9, Pixel 4a playback stutters, with over 20% dropped frames. LCEVC AV1 played on all devices, with occasional stuttering only on the three weakest devices and smooth playback on the others.

The right side of Figure 12 shows power figures on devices dropping less than 30% of the frames. On average LCEVC extends battery life by 41% vs. native AV1 on M8, and by 37% on M3, with reduced battery consumption even on devices where it is rendering more frames. LCEVC gains vs. native AV1 are greater with GAV1 than with Dav1d.

7.5.5 Real life device results – El Fuente (60 fps), ExoPlayer + Dav1d

Figure 13 illustrates test result for the "1080p height match" approach, using the ExoPlayer application combined with Dav1d software decoder on AV1 and LCEVC AV1 1080p60 content using M3 presets. As a 'sanity check' the same test (on AV1) has been performed using the VLC player combined with Dav1d decoder with similar results.

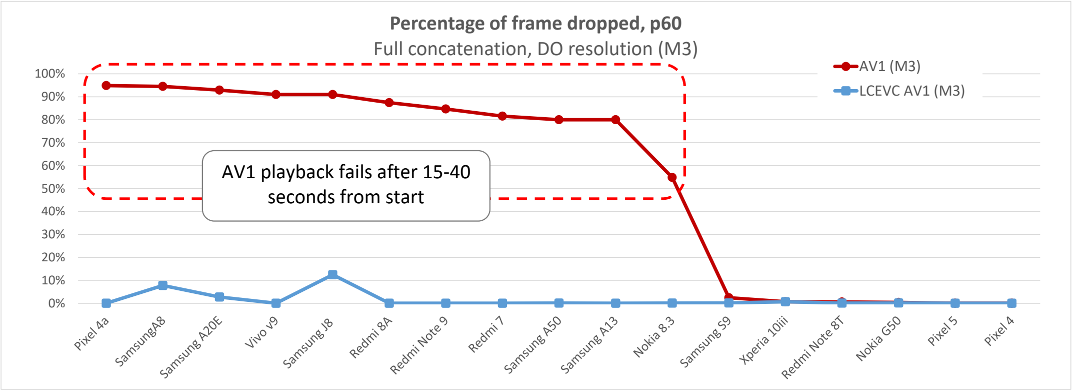


Figure 13 – On El Fuente p60 sequence, % frames dropped on full sequence using ExoPlayer and Dav1d decoder

As expected, ExoPlayer (and VLC + Dav1d, for which the sanity check provided consistent results to ExoPlayer + Dav1d) cannot playback 1080p60 AV1 content on most tested devices, resulting in frequent freezes or stuttering. With LCEVC AV1 video playback was smooth, with occasional stuttering only on older devices (Motorola G4 of 2016, and Samsung S5 of 2014).

7.5.6 Battery power drain test results – El Fuente (30 fps)

In addition to instantaneous power measurement, we performed battery drain tests, performing a real-life use case of playing one hour of content, encoded according to DO ("full set" scenario), on a small sample of devices which could playback smoothly both AV1 and LCEVC AV1. We started from device fully charged and measured the battery status at regular five-minute intervals during an hour of uninterrupted playback. Figure 14 illustrates test results using the ExoPlayer application combined with Dav1d and GAV1 respectively.

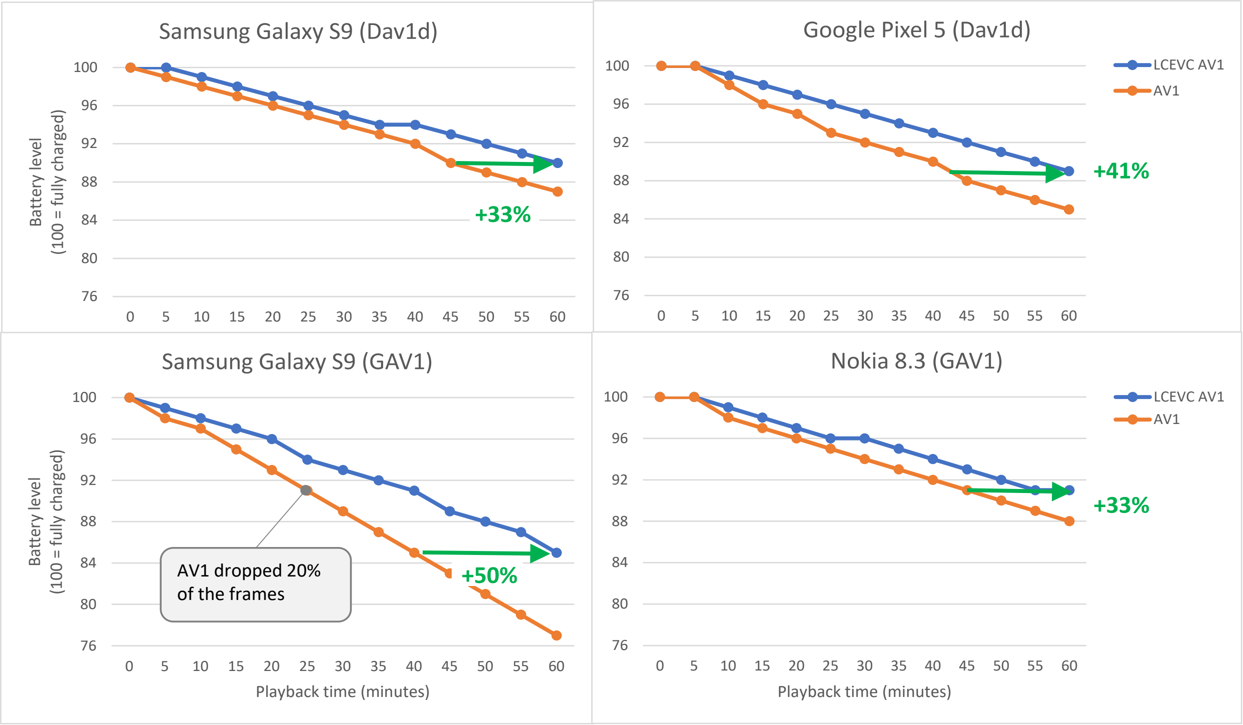


Figure 14 - One-hour battery drain test using ExoPLayer + Dav1d and ExoPLayer + GAV1 decoder

The test showed that native AV1 drained the battery 30% to 50% more quickly than LCEVC AV1, both in the case of Dav1d decoder (both phones rendered 100% of the frames), and GAV1 (where in the case of the S9 the phone rendered only 80% of the frames), demonstrating that the LCEVC power savings measured on a short 5-8 minutes sequence are reflected in even greater actual battery life improvements.

8. CONCLUSIONS

This paper evaluated the MPEG-5 LCEVC enhancement standard to assess its impact in improving the compression efficiency vs cycles trade-offs with reference to native SVT-AV1, x264 and x265 encoders for dynamically-optimized VOD applications, as well as its capability to improve playback and battery life of AV1 software decoding.

The extensive tests conducted for this work demonstrated that LCEVC is a valuable tool to improve the quality-cycles trade-offs across the full complexity range for the three tested standards – AVC, HEVC and AV1. In particular, in the case of SVT-AV1, LCEVC yields a ~40% reduction in computations while achieving the same quality levels according to VMAF_NEG.

Focusing on decoding, LCEVC is shown to enlarge the set of mobile devices capable of playing back high quality and high frame-rate content encoded with AV1, and to extend mobile battery life by up to 50% with respect to state-of-the-art AV1 software decoding.

The combination of LCEVC + SVT-AV1 builds on notable improvements of SVT-AV1 (improved speed-quality trade-off & fast decoding) and showcases the concrete possibility for widespread and energy efficient deployments.

ACKNOWLEDGMENTS

The authors are very thankful to Vasumathy Arumugam, Nicoló Bitetto, Lorenzo Cassina, Francesco Dessy, Andrew Jordan, Florian Maurer, Lewis Miller from V-Nova for their help in running the experiments, scripting the workflow and reviewing the drafts, and to Daniel Liu and Hassen Guermazi from Intel for their contribution to the development of the SVT-AV1 ‘SPIE_patch’.

REFERENCES

[1] Ping-Hao Wu, Ioannis Katsavounidis, Zhijun Lei, David Ronca, Hassene Tmar, Omran Abdelkafi, Colton Cheung, Foued Ben Amara, Faouzi Kossentini, "Towards much better SVT-AV1 quality-cycles tradeoffs for VOD applications," Proc. SPIE 11842, Applications of Digital Image Processing XLIV, 118420T (1 August 2021); doi: 10.1117/12.2595598.

[2] Katsavounidis, I., "The NETFLIX tech blog: Dynamic optimizer - A perceptual video encoding optimization framework," https://netflixtechblog.com/dynamic-optimizer-a-perceptual-video-encoding-optimization-frameworke19f1e3a277f (2018).

[3] Katsavounidis, I. and Guo, L., "Video codec comparison using the dynamic optimizer framework," Proc. SPIE 10752, Applications of Digital Image Processing XLI, 107520Q (17 September 2018); doi: 10.1117/12.2322118.

[4] See https://www.lcevc.org/lcevc-resources/ for MPEG specifications, performance evaluations, user guides, product documentation and other LCEVC-related resources.

[5] SVT-AV1, code repository - open-source SVT-AV1 encoder software, https://gitlab.com/AOMediaCodec/SVT-AV1.

[6] x264, code repository - open-source AVC encoder software, https://code.videolan.org/videolan/x264.

[7] x265, open source HEVC software encoder, https://www.videolan.org/developers/x265.html.

[8] Toward a Better Quality Metric for the Video Community, Netflix Technology Blog, https://netflixtechblog.com/toward-a-better-quality-metric-for-the-video-community-7ed94e752a30,

[9] Dav1d, source code repository – open-source AV1 decoder software, https://code.videolan.org/videolan/dav1d/ .

[10] GAV1, source code repository – open-source AV1 decoder software, https://chromium.googlesource.com/codecs/libgav1/.

[11] Kossentini, F., Ben Amara, F., Nouira, C., Tmar, H. “Faster AV1 Software Decoding Using SVT-AV1”, AOM Research Symposium, Redwood City, California, June 7-8, 2022.

[12] “Requirements for Low Complexity Video Coding Enhancements”, ISO/IEC JTC1/SC29/WG11 N18098, Macao, 2018.

[13] “Verification Test Report on the Compression Performance of Low Complexity Enhancement Video Coding”, International Organization For Standardization, ISO/IEC JTC 1/SC 29/WG 04 MPEG Video coding, April 2021.

[14] BT.500: Methodologies for the subjective assessment of the quality, https://www.itu.int/rec/R-REC-BT.500/en.

[15] Meardi G., Ferrara S., Ciccarelli L., Cobianchi G., Poularakis S., Maurer F., Battista S., Byagowi A., “MPEG-5 part 2: Low Complexity Enhancement Video Coding (LCEVC): Overview and performance evaluation”. Proc. SPIE 11510, Applications of Digital Image Processing XLIII, 115101C (21 August 2020). https://doi.org/10.1117/12.2569246.

[16] Battista S., Meardi G., Ferrara S., Ciccarelli L., Maurer F., Aachen University (Germany), Università Politecnica delle Marche (Italy), V-Nova (UK). “Overview of MPEG-5 Part 2 - Low complexity enhancement video coding (LCEVC) ITU:)”, ITU Journal, June 2020. https://www.itu.int/pub/S-JOURNAL-ICTS.V3I1-2020-12.

[17] Battista S., Meardi G., Ferrara S., Ciccarelli L., Maurer F., Conti M., Orcioni S., “Overview of the Low Complexity Enhancement Video Coding (LCEVC) Standard”, IEEE Transactions on Circuits and Systems for Video Technology, June 2022. https://ieeexplore.ieee.org/document/9795094.

[18] Constant Rate Factor, https://trac.ffmpeg.org/wiki/Encode/H.264#crf.

[19] SVT-AV1 SPIE_patch: https://gitlab.com/AOMediaCodec/SVT-AV1/-/tags/v1.1-SPIE2022 .

[20] Netflix El Fuente test video, https://opencontent.netflix.com/#h.goi1q35qkalw .

[21] pCRF and other LCEVC commands are documented in the user guide “FFmpeg with LCEVC”, https://docs.v-nova.com/v-nova/lcevc/reference-applications/ffmpeg.

[22] Tange, O. (2015). GNU Parallel 20150322 ('Hellwig'). The USENIX Magazine (Vol. 36, Number 1, pp. 42–47). Zenodo. https://doi.org/10.5281/zenodo.16303.

[23] Libvmaf, https://github.com/Netflix/vmaf/tree/master/libvmaf.

[24] A. Zvezdakova, S. Zvezdakov, D. Kulikov, D. Vatolin, Hacking VMAF with Video Color and Contrast Distortion, 2019. https://arxiv.org/abs/1907.04807.

[25] Convex hull test app, https://github.com/ikatsavounidisFB/convex_hull.

[26] Bjøntegaard, G., "Calculation of average PSNR differences between RD-Curves," ITU-T SG16/Q6, Doc. VCEGM33, Austin, Apr. 2001, http://wftp3.itu.int/av-arch/video-site/0104_Aus/.

[27] Z. Wang and A. Bovik, (2009) “Mean Squared Error: Love it or leave it?”, IEEE Signal Processing Magazine, p98-117, 2009,

[28] W. Lin, D. Li, and P. Xue, (2003) “Discriminative analysis of pixel difference towards picture quality prediction,” in Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429), vol. 3. IEEE, 2003, pp. III–193.

[29] “On learning based video quality metrics”, ISO/IEC JTC 1/SC 29/AG 5 m56636, April 2021.

[30] “[LCEVC] – Experimental Results of LCEVC versus conventional coding methods” ISO/IEC JTC1/SC 29/WG4 m53806, 2020

[31] “Summary of LCEVC test results” ISO/IEC JTC1/SC 29/WG4 N19238, May 2020, https://mpeg.chiariglione.org/standards/mpeg-5/low-complexity-enhancement-video-coding/summary-lcevc-test-results.

[32] Wu, P.-H., Kondratenko, V. and Katsavounidis, I., "Fast encoding parameter selection for convex hull video encoding," Proc. SPIE 11510, Applications of Digital Image Processing XLIII, 115100Z (21 August 2020); doi: 10.1117/12.2567502.

[33] Wu, P.-H., Kondratenko, V., Chaudhari, G. and Katsavounidis, I., "Encoding Parameters Prediction for Convex Hull Video Encoding," Picture Coding Symposium (PCS), Bristol, UK, (2021).

[34] ExoPlayer, source code repository – open-source video player, https://github.com/google/ExoPlayer.