Alliance for Open Media, Codec Working Group Document: **CWG-B075o**

AOM Common Test Conditions v2.0

August 31, 2021

Status:	Output document
Purpose:	AOM Common Test Conditions
Author(s):	Xin Zhao, Zhijun (Ryan) Lei, Andrey Norkin, Thomas Daede, Alexis Tourapis
Email(s):	xinzzhao@tencent.com, ryanlei@fb.com, anorkin@netflix.com, thomas.daede@vimeo.com, atourapis@apple.com
Source:	Tencent, Facebook, Netflix, Vimeo, Apple
Contacts:	Andrey Norkin (anorkin@netflix.com), Ryan Lei (ryanlei@fb.com), Yeping Su (yeping@google.com)

Abstract

This document describes guidelines for evaluating a video coding specification. It covers subjective and objective video quality metrics, test sequences, test configurations, and test reports.

1 Introduction

When developing a video coding specification, changes to the coding specification need to be evaluated based on their performance tradeoffs, and measurements are needed to determine whether the video coding specification has met its performance goals. This document proposes a methodology on how to perform and report tests in the context of the development of a next-generation video coding specification beyond AV1. If changes to the test model are proposed to be included as part of the test model, proponents shall report the test results following the guidelines explained in this document.

2 Quality Measurement

Subjective testing is the important method of testing video codecs. Subjective testing can be used when objective metrics results contradict one another or when it is assumed that the evaluated tool has effects on the visual quality. When performing subjective tests, many factors should be taken into account, such as matching bitrates and creating appropriate test conditions.
Selection of a testing methodology depends on the feature being tested and the resources available. Test methodologies are presented in order of increasing accuracy and cost.

2.1 Subjective quality measurements

Selection of a testing methodology depends on the feature being tested and the resources available. Test methodologies are presented in order of increasing accuracy and cost. Testing relies on the resources of participants. For this reason, even if the group agrees that a particular test is important, if no one volunteers to do it, or if volunteers do not complete it in a timely fashion, then that test should be discarded. This ensures that only important tests are done, in particular, the tests that are important to participants. Subjective tests should use the same operating points as the objective tests unless decided otherwise at a Codec WG call.

2.1.1 Image Pair Comparison

One way to determine the superiority of one compressed image is to visually compare two compressed images, and have the viewer judge which one has a higher quality. For this test, the two compressed images should have similar compressed file sizes, with one image being no more than 3% larger than the other. In addition, at least 5 different images should be compared. Once testing is complete, a p-value is computed using the binomial test. A significant result should have a resulting p-value less than or equal to 0.5. For example:

$$p\_value = \text{binom\_test}(a, a+b),$$

where a is the number of votes for one video, b is the number of votes for the second video, and binom_test(x, y) returns the binomial probability mass function (PMF) with x observed tests, y total tests, and expected probability 0.5. If ties are allowed to be reported, then the equation is modified:

$$p\_value = \text{binom\_test}(a+\text{floor}(t/2), a+b+t),$$

where t is the number of tie votes.

Still image pair comparison is used for rapid comparisons during development - the viewer may be either a developer or user. As the results are only relative, it is effective even with an inconsistent viewing environment. Because this test only uses still images , it is more suitable for changes with similar or no effect on inter frames or when no effects from different encoding of previous frames are observed. If changes in inter frames are to be evaluated, the frames preceding them in the decoding order should preferably be the same in both bitstreams to exclude random effects from having different prediction pictures.

2.1.2 Video Pair Comparison

Video pair comparisons follow the same procedure as still images. It is preferable that videos used for testing are limited to 10 seconds in length, and can be viewed up to a limited number of times (e.g., three) to reduce the viewer's fatigue.

2.1.3 Subjective viewing test

The subjective test should be performed as either consecutively showing the video sequences on one screen or on two screens located side-by-side. The testing procedure should normally follow rules described in [1] and be performed with non-expert test subjects. The result of the test could be (depending on the test procedure) mean opinion scores (MOS) or differential mean opinion scores (DMOS). Normally, confidence intervals are also calculated to judge whether the difference between two encodings is statistically significant. In certain cases, a viewing test with expert test subjects can be performed, for example if a test should evaluate technologies with similar performance with respect to a particular artifact (e.g. loop filters or motion prediction). Depending on the setup of the test, the output could be a MOS, DMOS or a percentage of experts who preferred one or another technology. Unlike pair comparisons, a MOS test requires a consistent testing environment. This means that for large scale or distributed tests, pair comparisons are preferred.

2.2 Objective quality measurements

The following descriptions give an overview of the operation of each of the objective metrics. Implementations of metrics must directly support the input resolution, color representation, bit depth, and sampling format.

Unless otherwise specified, all of the metrics described below only apply to the luma plane, individually to each frame. When applied to the video, the scores of each frame are averaged to create the final score.

Codecs must output the same resolution, bit depth, and sampling format as the input. This is necessary to achieve an exact match when cross-verification is needed.

2.2.1 Overall PSNR

PSNR is a traditional signal quality metric, measured in decibels. It is derived from mean square error (MSE). The MSE formula is:

$$PSNR = 10 * \log10 (MAX^2 / MSE),$$

where the error is computed over all the pixels in the video. The MAX value is set equal to $255 * 2^{BitDepth - 8}$ to align PSNR of 8-bit content scaled to higher bit depth with PSNR of the content at a higher bit depth. In its turn, the MSE is defined as follows:

$$MSE = 1/(n * m) * \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} [I(i,j) - K(i,j)]^2 ,$$

where *I(i, j)* and *K(i, j)* are samples of a color component of the source and reconstructed pictures at positions *i* and j respectively, and *n* and *m* are spatial dimensions of the picture component.

This metric may be applied to all color planes, with all planes reported separately.

The overall PSNR corresponds to an arithmetic average of the frame MSE values. The overall PSNR is less sensitive to the characteristics of individual frames and may be less prone to influence from the outlier frames than the frame-averaged PSNR.

2.2.2 Frame-averaged PSNR

PSNR can also be calculated per-frame, and then the PSNR values are averaged together. This metric is reported in the same way as overall PSNR. This PSNR corresponds to a geometric average of the frame MSE values.

2.2.3 Overall combined PSNR (APSNR-YUV)

For calculating Overall combined PSNR, a weighted MSE is calculated using the following formula:

$$MSE_{YUV} = A_1 * MSE_Y + B_1 * MSE_U + C_1 * MSE_V$$

This MSE_{YUV} value is then used to calculate PSNR according to the previous section. The weights A_1, B_1 and C_1 are currently set to 2/3, 1/6, and 1/6, respectively. The weights might be updated in future versions of the CTC.

2.2.4 Frame-averaged combined PSNR (PSNR-YUV)

For Frame-averaged combined PSNR, the frame-averaged PSNR is calculated for each component separately and the combined according to the following formula:

$$PSNR\text{-}YUV = A_2*PSNR_Y + B_2*PSNR_U + C_2*PSNR_V$$

The weights A_2, B_2 and C_2 are currently set to 0.875(14/16), 0.0625(1/16), and 0.0625(1/16), respectively. The weights might be updated in future versions of the CTC.

2.2.5 PSNR-HVS-M

The PSNR-HVS-M metric performs a DCT transform of 8x8 blocks of the image, weights the coefficients, and then calculates the PSNR of those coefficients. Several different sets of weights have been considered [2]. The weights used by the dump_pnsrhvs.c tool in the Daala repository have been found to better match the real MOS scores in the previous experiments.

2.2.6 SSIM

Structural Similarity Image Metric (SSIM) is a still image quality metric introduced in 2004 [3]. It computes a score for each individual pixel, using a window of neighboring pixels. These scores are averaged to produce a global score for the entire image. The original paper produces scores ranging between 0 and 1. In the CTC results, for the metric to appear more linear on BD-rate curves, the score is converted into a nonlinear decibel scale as shown below:

$$SSIMdB = -10 * \log10 (1 - SSIM)$$

2.2.7 Multi-Scale SSIM

Multi-Scale SSIM is SSIM extended to multiple scales / resolutions of the content [4]. The metric score is converted to decibels in the same way as SSIM.

2.2.8 CIEDE2000

CIEDE2000 (also known as DE2000) is a metric based on CIEDE color distances [6]. It generates a single score taking into account all three color planes. It does not take into consideration structural similarity or other psychovisual effects.

2.2.9 VMAF

Video Multi-method Assessment Fusion (VMAF) is a full-reference perceptual video quality metric that aims to approximate human perception of video quality [7]. This metric is focused on quality degradation due to compression and rescaling. VMAF estimates the perceived quality

score by computing scores from multiple quality assessment algorithms and fusing them using a support vector machine (SVM). Currently, two image fidelity metrics and one temporal signal have been chosen as features to the SVM, namely Detail Loss Measure (DLM), Visual Information Fidelity (VIF) which includes 4 VIF signals collected at different scales, and the mean co-located pixel difference of a frame with respect to the previous frame.

Besides the default VMAF model, a VMAF NEG (“no enhancement gain”) model is also included [11]. The NEG model aims to suppress the effect of image enhancement operations (sharpening, contrasting, etc.) on the final score, such that the pure effect of compression can be measured. The quality score from VMAF is used directly to calculate BD-Rate [10], without converting it to decibels .

2.2.10 Metrics implementations

Metrics implementations are provided by the libvmaf-based metrics tool `vmaf`. The initial `--aom_ctc` preset release is libvmaf v2.2.1 (https://github.com/Netflix/vmaf/releases/tag/v2.2.1). Exact command line example is "vmaf -r <source.y4m> -d <distorted.y4m> --aom_ctc v1.0 -q -o <vmaf.log>". vmaf log contains the per frame quality metrics information as well as the aggregated result across the whole sequence. Full precision of the quality metrics (6 decimal points) shall be kept for post analysis.
To speed up vmaf run time, multithreading can be enabled by "--threads <number of threads>"
For `vmaf` usage as well as an up to date list of libvmaf releases and versioned `--aom_ctc` presets, see the libvmaf aom_cd.md (https://github.com/Netflix/vmaf/blob/master/resource/doc/aom_ctc.md) tracking document.

3. Test Sequences

Sources are divided into multiple categories to test different scenarios the codec will be required to operate in. For easier comparison, all videos in each set have the same color subsampling, same resolution, and the same number of frames. In addition, all test videos are publicly available [8] for testing use, to allow reproducibility of results. It is recommended to download the test sequences in whole rather than recreating them from original sources. The MD5sums can be used to check the correctness of the downloaded sequences.

The test set is categorized by content and resolution. The sequences are in YCbCr format with 4:2:0 chroma subsampling. The test sequences are available at the following link [8]: https://media.xiph.org/video/aomctc/test_set/

3.1 Natural video (Class A)

Table 1. Class A1, 4:2:0, 4K, 10 bit.

No.	Sequence	Resolution	Frame rate	Bit-depth	MD5 sum

1	BoxingPractice	3840x2160	59.94	10	bc68da64c0c2d88c7c5e666d2cb760eb
2	Crosswalk	3840x2160	59.94	10	f53dedacb86f16d24f49e2416db46aa2
3	FoodMarket2	3840x2160	59.94	10	1741e614b486679397b161bb9a5a584d
4	Neon1224	3840x2160	29.97	10	139d81dab11673da3a349b71e66680c0
5	NocturneDance	3840x2160	60	10	e0728b9e40cb3d53c98eab8fe22f6e1b
6	PierSeaside	3840x2160	29.97	10	b9152c4e6ae8d68a418f7f62f70d1f9c
7	Tango	3840x2160	59.94	10	98c6fe8a6cd30e3e337123fe164beda8
8	Timelapse	3840x2160	59.94	10	3049e77714307d81fc4c0ecb6ea437d3

Table 2: Class A2, 4:2:0, 1920x1080p, 8 and 10 bit.

No.	**Sequence**	**Resolution**	**Frame rate**	**Bit-depth**	**MD5 sum**
1	Aerial (view)	1920x1080	59.94	10	36a9047adadc01ebcdc8ec062fcb710d
2	Boat	1920x1080	59.94	10	38be73be3e9e00520dd5fb5fbc238c16
3	CrowdRun	1920x1080	50	8	cbabdc85baf9b50cb14dbdf83e432226
4	DinnerSceneCropped	1920x1080	29.97	10	c4e1ba92a9d289cb214355f291f0e1a2
5	FoodMarket	1920x1080	59.94	10	169791ee8f32b920e4151a8133b9a849
6	MeridianTalk_SDR	1920x1080	59.94	10	ca3e32036cc40cb99af8a70b17d126cf
7	Motorcycle	1920x1080	30	8	5cfdfe0fccc9392815a662839a01cdf4
8	MountainBike	1920x1080	30	8	0cee002a42b85abc5d3f54d79bed9d6c
9	OldTownCross	1920x1080	50	8	dc607b8a517cb031403c97f9ac642935
10	PedestrianArea	1920x1080	25	8	2ded1b2064ee0c32ec07aa8bf6b3abf1

11	Ritual Dance	1920x1080	59.94	10	6bb5835fcb421021b1dadd5384bac424
12	Riverbed	1920x1080	25	8	e4170a0ade450fb9b6d06d74c7656cf8
13	RushFieldCuts	1920x1080	29.97	8	0f652c54c6b5fb1fe77bb366f4eb4a55
14	Skater227	1920x1080	30	10	dce66dd6db8e9bf51782396f40e975a5
15	ToddlerFountain	1920x1080	29.97	10	79baf955d530afa396b0e8c298dcb627
16	TreesAndGrass	1920x1080	30	8	8e27a24fad8b0c60b3e1afd924c88d02
17	TunnelFlag	1920x1080	59.94	10	d216bc59bc4242512dff87227ad6d1ae
18	Vertical_Bees	1080x1920	29.97	8	14cc7326a6b201f838380868ff2b7ee7
19	Vertical_Carnaby	1080x1920	59.94	8	c25e250593bbe1bb054a0f2d25e4c05b
20	WalkingInStreet	1920x1080	30	8	5b118d38d7528f33a4d8df379a9c3b25
21	WorldCup	1920x1080	30	8	a001d0da92125138093dfb94931c1337
22	WorldCup_far	1920x1080	30	8	6d87c49d487c1479d524bdbd2e6549dd

Table 3. Class A3, 4:2:0, 1280x720p.

No.	**Sequence**	**Resolution**	**Frame rate**	**Bit-depth**	**MD5 sum**
1	ControlledBurn	1280x720	30	8	875d857d8ef1b7f1b653c4c530669005
2	DrivingPOV	1280x720	59.94	10	dca294b0f589f7bc40be921447f8c88a
3	Johnny	1280x720	60	8	1d6aab4003385c255262a88f4df7cccc
4	KristenAndSara	1280x720	60	8	43bb17b78086d0183642c7b74cc1c903

5	RollerCoaster	1280x720	59.94	10	502eb30ee5f771cb8b367c36bbdc27db
6	Vidyo3	1280x720	60	8	706f830c649a143f1e14ac91542286cd
7	Vidyo4	1280x720	60	8	c321577a882a70fb2a41f706ff8c921c
8	WestWindEasy	1280x720	30	8	da8afe9c91a260c835b39f58b13c99e7

Table 4: Class A4, 4:2:0, 640x360p, 8 bit.

No.	**Sequence**	**Resolution**	**Frame rate**	**Bit-depth**	**MD5 sum**
1	BlueSky	640x360	25	8	b3ab86eb59559ede1c28dbcc9e7683ca
2	RedKayak	640x360	29.97	8	4205131788b94668c4feef73ce44aec9
3	SnowMountain	640x360	29.97	8	63ae6b7fb786883c2144552e98d0c4dd
4	SpeedBag	640x360	29.97	8	8bfe53f0215b749744b3947e16ab09a4
5	Stockholm	640x360	59.94	8	62d126930a6c8f61f5267f0b21746cde
6	TouchdownPass	640x360	29.97	8	dd98868ad8286eb0fb0b8c00827098a1

Table 5. Class A5, 480x270p, 4:2:0.

No.	Sequence	Resolution	Frame rate	Bit-depth	MD5 sum
1	FourPeople	480x270	60	8	4d35db57a2377e4211134db345dea225
2	ParkJoy	480x270	50	8	1bda17e3839c46f1586bffcadac5ef42
3	SparksElevator	480x270	59.94	10	fffac95c021a2b9b8555591015e3cf9a
4	VerticalBayshore	270x480	29.97	8	6ec078264f2ae2c99e2fbe108b774847

3.2 Synthetic (Class B)

Table 6. Class B1, 4:2:0.

No.	Sequence	Resolution	Frame rate	Bit-depth	MD5 sum
1	CosmosTreeTrunk	2048x858	24	8	1595459d9d4c79eb14f90d7beb098940
2	DOTA2	1920x1080	60	8	689f67a7054ef125a56e5f8d5209702c
3	EuroTruckSimulator2	1920x1080	60	8	4b2cd1e6a87b465f616ada20bb6951cd
4	GlassHalf	1920x1080	24	8	c91eb60c3953dd7dee1bf04aa8fad3e6
5	Life	1920x1080	30	8	5cefef31082a22b57fd524f92459c7c7
6	MINECRAFT	1920x1080	60	8	a7299bdf79b8a679521044f70683ac86
7	MissionControl	1920x1080	60	8	9d173243d13610d2aa8a271571a4b4e7
8	Sniper	1920x1080	30	8	6202f6729d3a36e248684cff5e451ae4
9	SolLevanteDragons_SDR	1920x1080	24	10	8803aeec213f0a21cf238ecfa0a9314b
10	SolLevanteFace_SDR	1920x1080	24	10	ac8f1feec169413b1896e7e51089640a
11	Wikipedia	1920x1080	30	8	73f1a07a9205f4496f7dbd1b0cc3ab10
12	WITCHER3	1920x1080	60	8	cb150ab509b93a9a44080c416e00c9d7

3.3 HDR (Class G)

The HDR class contains sequences in BT.2100 color space with PQ transfer function.

Table 7. Class G1, 4K, 4:2:0, 10bit.

No.	Sequence	Resolution	Frame rate	Bit-depth	MD5 sum
1	MeridianRoad	3840x2160	59.94	10	7d01bce200635758ad305b6de4958e38
2	NocturneDance_HDR	3840x2160	60	10	e3197e844805ffe39f1053c17ef9cb1b
3	NocturneRoom	3840x2160	60	10	86919b9d0373ae80c805af9fe7216f10
4	SparksWelding	4096x2160	59.94	10	69539846fd18cfc372cbafe8e6d4843f

Table 8. Class G2, 2K, 4:2:0, 10bit.

No.	Sequence	Resolution	Frame rate	Bit-depth	MD5 sum
1	CosmosCaterpillar	2048x858	24	10	793c7f42b657ca2b5755a46c853ce4a8
2	CosmosTreeTrunk_HDR	2048x858	24	10	7bc5d9a7c8b56ef62dd8ea9afcc50110
3	MeridianShore	1920x1080	59.94	10	f63d7c82272e535b3c9cfe68b1c09bd2
4	MeridianTalk_HDR	1920x1080	59.94	10	2668b636221d2093269eccdc7006055a
5	SolLevanteDragons_HDR	1920x1080	24	10	b3d05fe7e6aa4793e6e34c005c2314e1
6	SolLevanteFace_HDR	1920x1080	24	10	72ed7e851367f133728623537c268ad2
7	SparksTruck	2048x1080	59.94	10	17cc73a36829fb51ae80e7562afd1fb6

3.4 Still Image Class (Class F)

Table 9. Class F1, high resolution images

No.	Sequence	Resolution	Bit-depth	MD5 sum
1	Animals_00	4032x3024	8	2c339451d7cd20081280ac02f05cb404
2	Animals_03	4032x3024	8	4cdc1425596df9cbdbf5c8cde9d4f3b3

3	Animals_09	4032x3024	8	a80cce1f86da9f1625dd4b623a8871da
4	Buildings_02	4032x3024	8	5a782c297990717b7b381e58fdc53595
5	Buildings_03	4032x3024	8	c03485c934429704b2f6bcf4a6fe7dce
6	Church	6000x4000	8	c3616d6f6d8084b46a6a577538e55ee2
7	Fireworks_01	4032x3024	8	ba332747b68be06b05171bafa874b464
8	Flowerfield	3696x2448	8	8623868841c19d9cbb7e5d56bc45161b
9	Flowers_08	4032x3024	8	a4f2761806c64ed12a1c07d028f7cf33
10	Flowers_12	4032x3024	8	7fe26b452e9c58655d26af090e4e860f
11	Food_02	4032x3024	8	70e8c63e0814650ce282274d36a626ec
12	Fountain	6406x4270	8	6703325ccc28ddbae4accd8b2861eaf8
13	Lady	4480x6720	8	eec379811d7d12dba8cbd348115034e4
14	Landscape_5	4032x3024	8	70029604fb540f27b3795c2df963e8c9
15	Landscape_15	4032x3024	8	6cc5a4baabcb34a8d62f73856761ced7
16	Landscape_16	4032x3024	8	cb229835ebcd2708e31af30afee9c8fb
17	Landscape_18	4032x3024	8	98fbfd2aee58f15639b5c0ecccba85d1
18	Landscape_25	4032x3024	8	8a141e8cdecc38bc8d948a84c01476f3
19	Landscape_26	4032x3024	8	928c9d92578dc7bb21966a4abd318eaa
20	Landscape_28	4032x3024	8	0615d12ce62051b711ae8226809b1b0c
21	Lodge	6000x4000	8	b2e696ee6a7a81517823f4c470fcca33

22	Party	6720x4480	8	12f276d395693487ec2a809daeee9850
23	River	5184x3456	8	59e8a29a37c279956a19b8eefa69c88b
24	Santa	5616x3744	8	6625b85b26863c00a2de7939262db65c
25	Seafood	5184x3456	8	bdb2b01cf4b791a95f098bc2284541ed
26	Snow_00	4032x3024	8	5124465a706ee37311eba3f2ac64cebf
27	Trees	4928x3264	8	b9daa8a094bd54316e0e58f496e1cdd6
28	Underwater_01	4032x3024	8	ea8b88c8700d76bad7ce29be468f0e85

Table 10. Class F2, medium resolution images

No.	**Sequence**	**Resolution**	**Bit-depth**	**MD5 sum**
1	Adventure_with_the_Windmills	896x1110	8	a7987e28e5c36643445cd133800f6e3c
2	Agapanthus_Postbloom2	1208x948	8	7ef70323cafad09806c7ecdc417cc276
3	Baruch	968x1188	8	a4ff2107d6e6734573c4217b8120c07a
4	Berlin-Fernsehturm	1290x856	8	4a62af5cfdfdd32802dbef3f68e52eed
5	Big_Easy_chair	1196x1008	8	f2ae02fd76e3d624e78446bf74032954
6	Butterfly	1420x918	8	e9490a099b202455f816684a149670b1
7	Cecret_Lake_Panorama	1586x752	8	a96b7f0c72ad3e0ca0b9aee650a24070
8	Claudette	900x1100	8	dfcd6d7089bea0887a6fd90931f02e75
9	Collage_Oppeln	904x1280	8	77cf46759acdfd857658fd64c3cdd3d3
10	Corona_Arch	1272x922	8	37a1ca08e621c817e6ff20885ae77c29

11	Correfocs_Festa_Major_del_Clot	1306x870	8	ee48b154ec432ab417be77058a1a5046
12	Crepuscular_rays_at_Sunset_near_Waterberg_Plateau	1402x934	8	2990fb767f8eecba441b83af37d343d1
13	Esquibien_Jean_Perrot	818x1228	8	456d6ab197fb53b8a4c2cbd81d4bb9d1
14	Florac-Le_Vibron-Source_du_Pecher	1080x874	8	9d77b25418381606311ffa235128da52
15	Fontaine_Place_Stanislas	1390x820	8	b1402c725febe01f92277e64123bb8b5
16	Genmaicha_Tea	1260x840	8	187fc1b1b5ea7d6d5b7bcfc739fac30b
17	Homestead_in_Montana	1326x826	8	d5479de1436c5ecd9c6146533dd87382
18	Madeira_151_Funchal_Mercado_dos_Lavradores	1228x816	8	b54dd1397c37b95e29155bac75092d4c
19	Madeira_159_Funchal_Mercado_dos_Lavradores	1228x816	8	29178a65d092262d91a1ef1860e434a7
20	MagicKindom	1000x1000	8	4208177684e3e48ff550c4afa58d2d3e
21	Michigan_Stadium	1400x934	8	49b3c1be066ab7207d3bed3f1d8e6a22
22	OperaLamps	1296x864	8	a466acf392b24de6ba971673ff92d854
23	Orion_Nebula	1200x840	8	e7ec873ee52287f5006e9ba69f57cccc
24	Saint_Catherine-Caravaggio	876x1140	8	8b57ae588ec7766202dc5fae1854ccaf
25	Streptopelia_orientalis	1404x936	8	6def88cbc02273bcdeb6765ccb780ded
26	Swallowtail	1300x900	8	efd2e87afad63f5f6954206de30cc1ba
27	Washington_Monument	1204x904	8	de162962a5d109da14fd3f83fe8f9910
28	Wasserfassstelle_von_1898_im_Schanerloch	816x1150	8	e62cfbd37531121a2a7fb1c33c2ebde8
29	Zoo_de_la_Barben	1296x864	8	8f1c710c21f123fb6ceacedfa623b628

3.5 Non-pristine Content, Class E

Class E sequences are user generated content (UGC) and other content with different technical content quality and noticeable compression artifacts, compared to typical pristine materials.

Table 11. Class E, User Generated Content

No.	Sequence	Resolution	Frame Rate	Bit-depth	MD5 sum
1	Artistic_Concert	1920x1080	25	8	932c06e8d91a88440a7b63007543fabc
2	Artistic_Intro	1920x1080	29.97	8	5630f7a362a3c21e78bab148482f1a30
3	MixedCoding_NewsIntroAnchor	1280x720	29.97	8	e86dc11f07b807d451507ec3d4dee04c
4	MixedCoding_NewsIntroOnly	1280x720	29.97	8	68aa1de52339410ec4dcd0d3617b8075
5	Noise_AnimationCrayon	1920x1080	23.98	8	3a74b8c9fbab0ffaeda78e5be8de3f82
6	Noise_Animation	1280x720	23.98	8	e68ab7c2dd679da8798ccdbfe5f2cdd2
7	Noise_Ocean	1920x1080	60	8	2a568ca797530a398ef7ed9261b52436
8	Noise_Soccer	1920x1080	50	8	867db8d23d1ed7e39c4ac350a7b48490
9	Shaky_Baseball	3840x2160	59.94	8	c1c3523304c1092772ad096bd8c48f55
10	Shaky_Fireworks	3840x2160	29.97	8	71c9f8a87f0da64d27fa18f2f5924eb7

11	Shaky_Quad	1920x1080	30	8	e2a124d442fbba49f3150 7add888434b
12	Shaky_Walk	1920x1080	25	8	069e589e05706720c28a 48a9b0c7b76c

4. Test Configuration

Four test configurations are defined. All Intra configuration is intended for evaluating intra coding methods. Random Access configuration is intended for on-demand streaming, one-to-many live streaming, and stored video. Low Delay configuration is intended for videoconferencing and remote access. Adaptive Streaming configuration reflects the use case of video streaming over the internet. Encoder only pre-filtering is disabled when running the tests.

Encoders should be configured to their best performing settings (i.e., --cpu-used=0), and single pass encoding (--passes=1) should be applied when being compared against each other. The exact QP values should be specified for each level of prediction hierarchy.

For video and images with 4K or higher resolution (width >= 3840 and height >=2160) in all classes (Class A1, F1), 2 column tiles and two threads should be used for encoding of 4K sequences, the related configuration is following:

```
--tile-columns=1 --threads=2 --row-mt=0
```

For other classes, single thread (--threads=1) should be used, one tile per picture (--tile-columns=0), the related setting is:

```
--tile-columns=0 --threads=1
```

The following four configurations described in this section are used to test incremental changes to a codec.

All simulations should use the .obu format as the bitstream output to compute the bitrate. Bitrate shall be calculated as:

Bitrate(kbps) = round(FilesizeInByte * 8 * fps_num/fps_denom/framenumber/1000, 6)

where the fps_num and fps_denom are the numerator and denominator used in the y4m header to specify frame rate. 6 decimal points are kept to maintain the same precision as quality metrics.

In all configurations, the codec shall use the internal bit depth equal to the input sequence bit depth. Note that the internal bit depth may be lower than the size of the data type used to store reference frames.

The following input qindex values shall be used by the configurations for the libaom codec.

Table 12. qindex values per configuration

Configuration	**Command line QP values**
Still image (Class F)	60, 85, 110, 135, 160, 185
All Intra (AI)	85, 110, 135, 160, 185, 210
Random Access (RA)	110, 135, 160, 185, 210, 235
Low Delay (LD)	110, 135, 160, 185, 210, 235
Adaptive streaming (AS)	110, 135, 160, 185, 210, 235

When calling the reference encoder, --qp shall be used to specify the qindex directly within the following valid range:

- 8 bit: [0, 255]
- 10 bit: [-48, 255]
- 12 bit: [-96, 255]

Encoder internally will add a proper offset (48 for 10 bit and 96 for 12 bit) to get the final qindex encoded in the bitstream.
Results for the following classes should be reported for each configuration.

Table 13. Sequence classes that should be reported for each configuration.

Class	**Configuration**			
	AI	**RA**	**LD**	**AS**
A1	Yes	Yes	No	Yes
A2	Yes	Yes	Yes	No
A3	Yes	Yes	Yes	No
A4	Yes	Yes	Yes	No
A5	Yes	Yes	Yes	No
B1	Yes	Yes	Yes	No

F1	Yes	No	No	No
F2	Yes	No	No	No
G1	Optional	Optional	No	No
G2	Optional	Optional	No	No
E	Optional	Optional	No	No

4.1 All Intra (AI) configuration

All intra configuration is used to encode frames from test sequences and still images (Class F). The test following this configuration uses a subset of the first 30 frames of the video sequences for all classes except the still image classes F1 and F2. All frames are encoded in intra-prediction mode. Frame QP modulation is not used in this configuration.
The still image classes F1 and F2 shall also be encoded in this configuration. In this case, each still image is encoded separately (and consists of one frame).

Still images and intra frames should be encoded using the following parameters:

--cpu-used=0 --passes=1 --end-usage=q --qp=x --kf-min-dist=0 --kf-max-dist=0
--use-fixed-qp-offsets=1 --deltaq-mode=0 --enable-tpl-model=0 --enable-keyframe-filtering=0
--obu

--qp is used to specify the qp value defined in Table 12. In addition, for video test data (Class A, Class B and Class G), “--limit=30” should be configured, for still images (Class F), “--limit=1” should be configured. Note that using the “--limit=30” parameter for still images would cause the encodes to use the full sequence header, which would result in incorrect results.

4.2 Random Access (RA) configuration

This coding test configuration uses non-zero structural delay. The number of total coded frames is 130, which includes two GOPs and one intra frame for each GOP. Closed-GOP configuration is used. QP modulation shall be explicitly selected for each frame type, as specified in the encoding config files accompanying this document. In total, 130 frames shall be coded (--limit=130).

--cpu-used=0 --passes=1 --lag-in-frames=19 --auto-alt-ref=1 --min-gf-interval=16
--max-gf-interval=16 --gf-min-pyr-height=4 --gf-max-pyr-height=4 --limit=130 --kf-min-dist=65
--kf-max-dist=65 --use-fixed-qp-offsets=1 --deltaq-mode=0 --enable-tpl-model=0 --end-usage=q
--qp=x --enable-keyframe-filtering=0 --obu

4.3 Low Delay (LD) configuration

This configuration requires the codec to operate in zero structural frame delay mode. One key frame (frame 0) in the beginning of the GOP is used. In total, 130 frames should be coded (--limit=130).

--cpu-used=0 --passes=1 --lag-in-frames=0 --min-gf-interval=16 --max-gf-interval=16
--gf-min-pyr-height=4 --gf-max-pyr-height=4 --limit=130 --kf-min-dist=9999 --kf-max-dist=9999
--use-fixed-qp-offsets=1 --deltaq-mode=0 --enable-tpl-model=0 --end-usage=q --qp=x
--subgop-config-str=ld --enable-keyframe-filtering=0 --obu

4.4 Adaptive Streaming (AS) configuration

The adaptive streaming configuration involves performing encodes of a number of sequences at several specified resolutions. Lower resolution sequences are obtained by downsampling the highest resolution sequences according to the downsampling procedure specified in this document. The quality metrics are obtained by upsampling the decoded sequences to the highest resolution according to the specified upsampling procedure and computing the video quality metrics against the source (the input video with the highest resolution). The BD-rate is computed by finding rate-quality convex hulls of both anchor and test and computing BD-rate based on these convex hulls.

The following resolutions are used for the adaptive streaming test conditions, with 3840x2160p resolution being the resolution of the original sequences, which is used for computing the quality metrics.

- 3840x2160p
- 2560x1440p
- 1920x1080p
- 1280x720p
- 960x540p
- 640x360p

Per-resolution BD-rate results shall also be reported for the adaptive streaming configuration.

4.4.1 Downsampling and upsampling

Downsampling and upsampling are performed directly between the original resolution and coding resolutions.

Conversions between resolutions should use Lanczos filter with parameter $a = 5$ for both luma and chroma components. When upsampling or downsampling a picture, the picture should be

padded by replicating a boundary sample. The alignment between the samples should use a so-called “centered phase” (the samples should be centered around the geometrical center of the picture). The filter coefficients should have 14-bit integer precision.

For video sequences in BT.709 format and other non-HDR formats, a vertical chroma sample position (Type 0) should be used. HDR sequences, if used, assume Type 2 chroma sample position (co-located with luma (0, 0) sample). The still images, if used, should use the “JPEG” chroma sample position (i.e. equal distance to the co-located luma samples).

4.4.2 Filters

The filters used for down- and up-sampling can be found in [12].

The implementation of the up- and downsampling filters is available in the HDRTools software available at the following link: https://gitlab.com/standards/HDRTools. Tag v0.22 shall be used in the CTC (https://gitlab.com/standards/HDRTools/-/tree/v0.22), commit b03868b27e5e34f5f7db80f0336910f9a29c3b35 .

For the configuration parameters and the config files that shall be used for down- and upsampling with HDRTools, please refer to the scripts in /aom/tools/convexhull_framework/src/VideoScaler.py in the libaom repository. In short, to enable the filters required by the CTC, the following parameters need to be set:

ScaleOnly=1
ScalingMode=12

4.4.3 Adaptive streaming command line

The following command line should be used for encodes in adaptive streaming configuration.

--cpu-used=0 --passes=1 --lag-in-frames=19 --auto-alt-ref=1 --min-gf-interval=16
--max-gf-interval=16 --gf-min-pyr-height=4 --gf-max-pyr-height=4 --limit=130 --kf-min-dist=65
--kf-max-dist=65 --use-fixed-qp-offsets=1 --deltaq-mode=0 --enable-tpl-model=0 --end-usage=q
--qp=x --enable-keyframe-filtering=0 --obu

4.4.4 Convex hull

The convex hull computation algorithm uses uniformly spaced interpolated points between the (rate, quality) points corresponding to the encodes. Convex hull computation algorithm and the software for the AS configuration can be found in the reference code (libaom) repository under /aom/tools/convexhull_framework/src/ConvexHullTest.py

After encoding tests are done, bit rate and quality metric information for all selected QPs (6 in total) within each resolution should be collected. In order to make sure there are enough data

points for each resolution before constructing a convex hull, (bitrate, quality metric) points for each resolution should be interpolated first. 7 interpolated points should be generated between each pair of adjacent QPs. The resulting (bitrate, quality) points shall contain the original 6 (bitrate, quality) points after encoding. The interpolated bitrate points shall be spaced uniformly between two simulated points in the log domain. Bilinear interpolation is used.

After interpolation, resulting (bitrate, quality) points for all resolutions shall be used to construct the convex hull.

4.4.5 Scripts

The following scripts can be used for calculating the results for adaptive streaming test conditions. The scripts are located in the libaom repository under [https://gitlab.com/AOMediaCodec/avm/-/tree/research-v2.0.0/tools/convexhull_framework]. The details on using the scripts can be found in the accompanied README file [https://gitlab.com/AOMediaCodec/avm/-/blob/main/tools/convexhull_framework/README.TXT]. Note that the scripts are located in the *research* libaom branch.

4.5 Encoding of HDR sequences

Encoding of HDR sequences should use these additional parameters:

--color-primaries=bt2020 --transfer-characteristics=smpte2084
--matrix-coefficients=bt2020ncl --chroma-sample-position=colocated

4.6 Encoding of synthetic contents sequences

In mandatory CTC configurations, use of screen content tools in the RDO process is decided for each frame based on the screen content detector output. For evaluation of screen content and synthetic video coding tools, it was found desirable to have an optional configuration in which screen content tools are turned on in all frames of class B1 sequences (synthetic content). For proposals on screen and other synthetic content coding tools, besides regular mandatory CTC results, it is required to provide additional test results with the command line parameter "--tune-content=screen" applied to encoding sequences in class B1 (synthetic video).

5. Test Report

New coding tools to be evaluated by the Codec WG shall report the test results described in this document. Focus Groups (FG) can modify or amend test conditions described in this document

in the cases when it is justified by the topic of the focus group (such as a different set of QPs in case of coefficient coding studies or extra sequences for the subjective tests and different QPs in case of loop filters evaluation, subject to approval by the Codec WG.

The coding performance and software runtime shall be reported. Both encoding and decoding runtime shall be measured and compared to those of the anchor.

To report the overall progress of the codec development and continuously track the tools performance, periodic tests should be performed on a regular time basis.

There are the following two options for measuring the encoding/decoding runtime:

1. using the built-in time utility available on Linux platform
    a. The following command line can be used to dump out the user time and system time into a text file.

        ```
        /usr/bin/time --verbose --output=time_log.txt <actual command>
        ```

    b. An example output log file is:

```
Command being timed: "aomenc-v1.0.0 ..."
User time (seconds): 263.93
System time (seconds): 0.67
Percent of CPU this job got: 99%
Elapsed (wall clock) time (h:mm:ss or m:ss): 4:24.78
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 1208152
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 1
Minor (reclaiming a frame) page faults: 302601
Voluntary context switches: 107
Involuntary context switches: 3001
Swaps: 0
File system inputs: 47392
File system outputs: 1352
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

    From the time log, user time can be extracted as the indicator for runtime.

2. using the built-in perf utility available on Linux platform.
    a. The following command line can be used to dump out the instruction count and cycle count into a text file.

        ```
        3>perf_log.txt perf stat --log-fd 3 <actual command>
        ```

b. An example output log file is:

```
Performance counter stats for 'aomdec-v1.0.0 ...':

          170.56 msec task-clock:u              #    0.954 CPUs utilized
               0      context-switches:u        #    0.000 K/sec
               0      cpu-migrations:u          #    0.000 K/sec
           6,343      page-faults:u             #    0.037 M/sec
     362,887,864      cycles:u                  #    2.128 GHz
     675,757,718      instructions:u            #    1.86  insn per cycle
      67,593,306      branches:u                #    396.292 M/sec
       2,749,467      branch-misses:u           #    4.07% of all branches

     0.178838897 seconds time elapsed

     0.150474000 seconds user
     0.021779000 seconds sys
```

From the perf log, instruction count, cycle count and user time can be extracted as indicators for complexity and runtime.

It is mandatory for proponents to provide the runtime information using method 1. When it is possible, detailed instruction count and cycle count acquired via method 2 can also be provided as optional supporting data.

5.1 Tool evaluation tests

Changes that are expected to affect the quality of encode or bitstream should run an objective performance test. The following data shall be reported:

- Identifying information for the codec version used, such as the git commit hash. Typically, the anchor (git tag) for the current codec development period shall be used
- Command line options to the encoder, configure script, and anything else necessary to replicate the experiment. Typically, the command lines specified in this document shall be used for the anchors
- For all encoding configurations, and for each objective metric:
    - The BD-Rate score, in percentage, for each test sequence
    - The average of all BD-Rate scores, equally weighted, for each sequence class in the test set
    - The average of all BD-Rate scores for all videos in all categories
    - Min and max BD-rates for all categories

5.2 Periodic tool tests

The performance of the adopted tools needs to be tracked during the codebase development. Tools adopted to the new codec model should be tested periodically, every time when the group is switching to the new anchor and after implementing the adopted tools in the new anchor. Both

tools on and tools off tests should be performed. The anchor for the tools off tests should be the anchor for the new codec development period. The anchor for the tools on test should be the first anchor (AV1 based) unless switching all tools from the tools on anchor is not possible or desirable. This activity is expected to be performed by the Testing sub-group. Test sequences specified in this document, including the optional test sets, should be used in this type of testing.

5.3 Periodic progress tests

Periodic tests are run on a wide range of QPs/bitrates in order to gauge progress over time, as well as detect potential regressions missed by other tests. The test sequences specified in the current document shall be used. The AV1 anchor should be used in these tests.

5.4 Current Anchor

For testing the coding tools at the current development period, the libaom research branch shall be used with tag research-v2.0.0.

5.5 Coding performance evaluation

The Bjontegaard rate difference, also known as BD-rate [9], allows the measurement of the bitrate reduction offered by a codec or codec feature, while maintaining the same quality as measured by objective quality measurements specified in Section 2.2. The rate change is computed as the average percent difference in rate over a range of qualities.
For each color component (Y, Cb, and Cr), as well as for APSNR-YUV and PSNR-YUV, the BD-rate value is calculated as follows:

- Given a selection of rate-distortion points, the rates are converted into log-rates.
- A piecewise cubic Hermite interpolating polynomial is fit to the points for each codec to produce functions of log-rate in terms of distortion.
- Metric scores are computed as described in Section 2.2.

Given the BD-rate of each color component, an overall BD-RateWeighted considering all color components is calculated as follows:

- $\text{BD-Rate}_{\text{Weighted}} = A*\text{BD-Rate}_{Y} + B*\text{BD-Rate}_{Cb} + B*\text{BD-Rate}_{Cr}$

The weighting factors A and B are adjustable based on the exact codec cost of coding luma and chroma components. These weighting factors are periodically updated and are currently set to A = 0.92 (23/25), B = 0.04 (1/25).

The BD-RateWeighted for both APSNR and PSNR are expected to be similar to the BD-rate calculated over APSNR-YUV and over PSNR-YUV. When there is significant deviation among these four metrics, further investigation is needed.

The reference codec used for reporting coding performance is the test model using the test configurations defined in Section 4.

Minimum and maximum of sequence BD-rate gains should also be reported in addition to the average BD-rates. All data (rate/metric) points should be available when reporting results (such as in the CTC document template) to make further analysis of the results possible.

5.6 Non-monotonic RD-curves

Occasionally, some sequences may have non-monotonic RD-curves. The following procedure should be used to handle these cases when they occur.

1. All non-monotonic cases/points should be flagged and reported in the results
2. When there is non-monotonicity in PSNR-Y on the CTC QPs, the PSNR-Y results cannot be reported. Note that there are two PSNRs reported, based on averaging frame PSNR and frame MSE values; if there is non-monotonicity in one of these PSNR curves and not in the other, further investigation may be needed.
    a. There could be non-monotonic cases in other objective metric results since encoder algorithms in the reference software optimize for SSD/MSE
3. The official CTC template and AWCY would not report averages for metrics where one or more sequences have non-monotonic RD-curves, except for the VMAF case explained in item 4.
4. To solve the problem with VMAF non-monotonicity in a flat (saturated) region of the curve, if VMAF non-monotonicity happens at VMAF value 99.5 or above, the non-monotonic value and the values corresponding to bitrates higher than the non-monotonic value are excluded from the BD-rate calculation. The VMAF BD-rate number is still reported and used in the VMAF metric average.

5.7 Encoding and decoding time measurement

Two types of the encoding and decoding time comparisons should be reported:

- The first type (in percent) is the geometric mean of ratios of the encoding (and decoding) sequence times of the test and the anchor
- The second reported type of the encoding/decoding time measurement is performed by adding up all encoding/decoding times for all QPs and sequences in the category. The ratio of the encoding time of the test to the encoding time of the anchor (in percent) is reported.

In addition to this data, the minimum and maximum of the encoding and decoding time ratios should be reported per class and per test set.

For the Adaptive Streaming test conditions, the encoding and decoding time shall be reported without time spent on downsampling and upsampling. The report for the Adaptive Streaming test conditions should include encoding and decoding times of all (resolution, QP) pairs, not only the times of the pairs that are selected to compute the convex hull.

It is recommended to use the simulation setup that allows for better runtime reliability. One of the following methods can be used for this purpose:

- Use the same type of instances and switch off turbo-boost mode on x86 CPUs
- Make sure the anchor and test for the same sequence and QP are run on the same instance in parallel to each other

5.8 Graphing

When displayed on a graph, bitrate is shown on the X axis, and the quality metric is on the Y axis. For publication, the X axis should be linear. The Y axis metric should be plotted in decibels. If the quality metric does not natively report quality in decibels but it is required to do so, it should be converted as described in Section 2.2.

7. Acknowledgements

This document has been drafted with thanks to the suggestions and comments received from the following experts:
Shan Liu, Leo Zhao (Tencent), Jill Boyce, Hassene Tmar, Faouzi Kossentini, Iole Moccagatta, James Holland, Samuel Wong (Intel), Yaowu Xu, Debargha Mukherjee, Urvang Joshi (Google), Krishna Rapaka (Apple), Ioannis Katsavounidis, Chia-Yang Tsai (Facebook)

8. References

[1] ITU-R, "Recommendation ITU-R BT.500-14", 2019.

[2] K. Egiazarian, J. Astola, N. Ponomarenko, V. Lukin, F. Battisti, and M. Carli, "A New Full-Reference Quality Metrics Based on HVS", 2002.

[3] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity", 2004.

[4] Z. Wang, E. Simoncelli, and A. Bovik, "Multi-Scale Structural Similarity for Image Quality Assessment", n.d.

[5] M. Chen, and A. Bovik, "Fast structural similarity index algorithm", 2010.

[6] Y. Yang, J. Ming, and N. Yu, "Color Image Quality Assessment Based on CIEDE2000", 2012.

[7] A. Aaron, Z. Li, M. Manohara, J. Lin, E. Wu, and C. Kuo, "Challenges in cloud based ingest and encoding for high quality streaming media", 2015.

[8] Test sequences: https://media.xiph.org/video/aomctc/test_set/

[9] G. Bjøntegaard, “Calculation of average PSNR differences between RD-Curves,” ITU-T SG16/Q6, Doc. VCEG-M33, Austin, Apr. 2001. [Online] Available: http://wftp3.itu.int/av-arch/video-site/0104_Aus/

[10] Codebase repository, https://aomedia.googlesource.com/aom

[11] Z. Li, “On VMAF’s property in the presence of image enhancement operations”, [Online] Available: https://tinyurl.com/y34mgafa

[12] Resampling filter coefficient specification. https://groups.aomedia.org/g/sg-codec-testing/files/ResamplingFilters.pdf