Analysis of AV1 Coding Tools

Hsiao-Chiang Chuang[a], Zhijun Lei[a], Agata Opalach[b], Andrey Norkin[b]
[a]Meta Platforms, Inc., 1 Hacker Way, Menlo Park, CA 94025, USA
[b]Netflix, Inc., 100 Winchester Cir, Los Gatos, CA 95032, USA

ABSTRACT

AV1 is the first royalty-free video coding standard developed by the Alliance for Open Media (AOM), which was finalized in 2018. During its standardization process, coding tools were gradually adopted into the specification based on a tradeoff between multiple parameters, such as bitrate, quality, encoding and decoding implementation complexity. A fair comparison of the coding tools supported by this codec can be essential for encoder designers who seek to achieve a good balance among all these factors within their implementations. To this end, this paper compiles a tool-on/off analysis of several prominent coding tools supported by the AV1 specification. The analysis includes the impact of such tools on several objective quality metrics, i.e., PSNR, SSIM, and VMAF, when using the reference encoder libaom implementation, as well as the corresponding impact on the software (SW) runtime complexity of both the libaom encoder and decoder.
Keywords: AV1, video compression, coding tools

1. INTRODUCTION

The aim of this paper is to provide information about the relative performance of most of the coding tools supported in the AV1 [1] coding specification. This is achieved by evaluating such tools in the context of the reference SW of AV1, libaom, and by performing tool on/off tests. There are typically overlaps in coding gains among coding tools, and hence both tool on/off tests need to be performed to better understand the performance and behavior of a particular coding tool. A 'tool-on' test involves coding experiments where all AV1 coding tools to be evaluated are disabled in the encoder other than the tool under test. On the other hand, a 'tool-off' test involves performs coding tests by enabling all AV1 coding tools other than the tool under test. In both 'tool-on' or 'tool-off' test, encoder is configured to enable/disable individual coding tools at mode decision stage without completely removing or adjusting any corresponding normative syntax elements.

2. BRIEF DESCRIPTION OF AV1 CODING TOOLS

2.1 Partitioning

In the AV1 specification, partitioning of a super block (SB) follows a recursive quadtree, where each tree node is a square block with size of 2Nx2N ($N = 2^1, 2^2, ..., 2^6$). This includes up to 10 partitioning types: square partition (2Nx2N/NxN), horizontal partition (2NxN), vertical partition (Nx2N), AB partition, and 1-to-4 partition (4NxN/Nx4N). The AB partition involves a 3-way partitioning of a square block. On top of a horizontal or vertical partitioning, a further split is performed that is orthogonal to the direction of the first split in the first or the second partition. The 1-to-4 partition involves a 4-way partitioning of a square block with equal-sized partitioning vertically or horizontally. The maximum super block size is extended to 128x128 from 64x64 that is supported in VP9. In addition, the minimum block size with flexible inter-mode information is extended to 4x4 from 8x8 in VP9. Figure 1 shows all the 10 possible partitioning types within a square block.

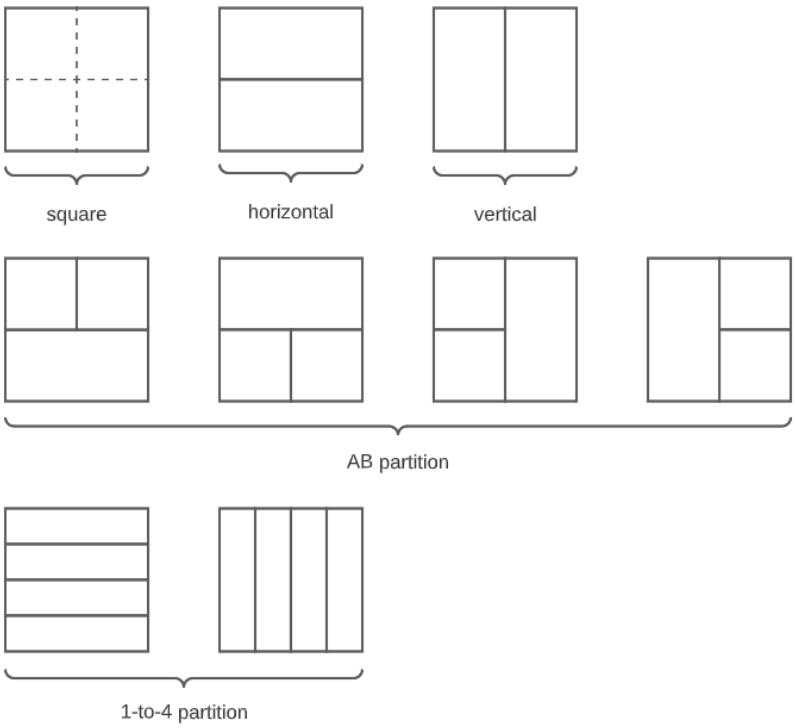

Figure 1. 10 partitioning types in AV1

2.2 Intra prediction

AV1 supports multiple intra prediction modes and tools, that include the following:

1. **Intra delta angles**: VP9 supports 8 nominal directional modes ranging from 45 to 207 degrees. In AV1, angular prediction modes are extended from the nominal modes supported in VP9 to include an extended set of angles with a finer granularity. Additional angles at +/- 3 degrees between two consecutive modes were introduced. Therefore, a total of 56 (8 nominal x 7 delta angles) angular modes are supported in AV1 [2].
2. **Intra edge filter**: for angular modes, a filter is applied to the neighboring reference samples based on the prediction mode used. In particular, the prediction mode can determine the filter strength, filter type, and filter taps that are applied on those reference samples prior to their use as predictors. An additional up-sampling filter is applied for blocks whose width is less than or equal to 16 [3].
3. **Paeth intra mode**: the TM_PRED mode in VP9 is replaced by the Paeth intra mode using the following formula to generate the prediction samples:

$$P(x,y) = \underset{p}{\operatorname{argmin}}(p - (T + L - TL)), \text{ where } p \in \{T, L, TL\} \quad (1)$$

   and T, L, and TL are the reference samples from the neighboring block at positions (x,-1), (-1, y), and (-1,-1), respectively, where $(0,0)$ is the top-left sample of the block.
4. **Smooth intra mode**: there are three types of smooth modes in AV1:

   SMOOTH_H_PRED:

$$P_H(x,y) = w(x) \cdot L + (1 - w(x)) \cdot TR \quad (2)$$

   where $w(x)$ is a weight value from a look-up table that corresponds to the x coordinate, and L and TR are the reference samples at positions $(\text{-}1, y)$ and $(\text{W-}1, \text{-}1)$, respectively. W represents the block width.

SMOOTH_V_PRED:

$$P_V(x,y) = w(y) \cdot T + (1 - w(y)) \cdot BL \tag{3}$$

where $w(y)$ is a weight value from a look-up table that corresponds to the y coordinate, and T and BL are the reference samples at positions $(x,\text{-}1)$ and $(\text{-}1, \text{H-}1)$, respectively. H represents the block height. Prediction sample generation with the third type takes the average sample values of the previous two types.

SMOOTH_PRED:

$$P(x,y) = (P_H(x,y) + P_V(x,y))/2 \tag{4}$$

5. **Filter-intra mode**: each block (smaller than or equal to 32x32) is partitioned into a set of 4x2 sub-blocks, and the prediction samples of the above, left, and above-left neighboring sub-blocks will be used as reference samples to generate the prediction of the subsequent sub-blocks. There are five prediction modes supported for the filter-intra mode: {DC, VER, HOR, D157, PAETH}. A set of predefined filter coefficients (7 neighboring samples x 5 modes x 8 positions) are used to generate the prediction samples. A bilinear interpolation filter is applied if the associated reference sample is located in between two integer reference samples.

6. **Chroma-from-Luma (CfL) mode**: in this mode, chroma prediction samples are generated by the collocated luma reconstructed samples. Figure 2 shows the process of prediction sample generation in CfL mode.

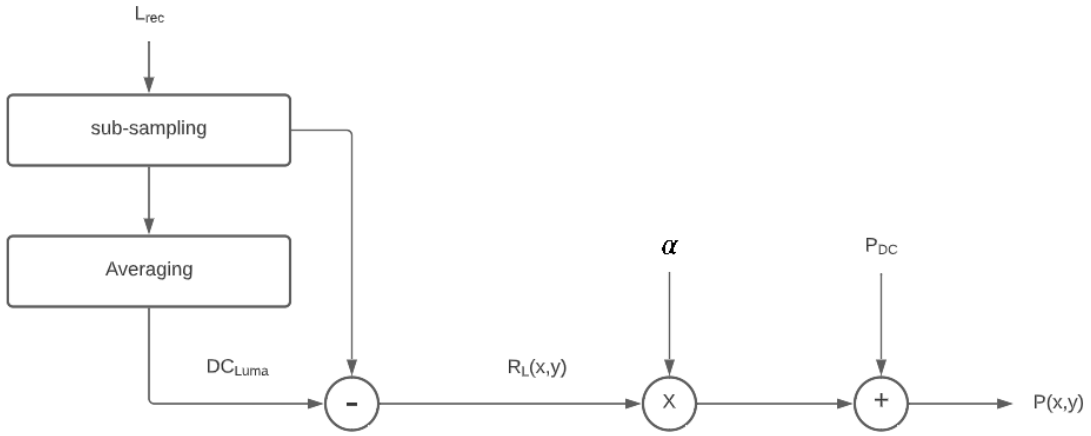

Figure 2. Illustration of prediction sample generation for CfL mode

Prediction samples are generated using the following formula:

$$P(x,y) = \alpha \cdot R_L(x,y) + P_{DC} \tag{5}$$

where α is a parameter signaled in the bitstream and P_{DC} is the DC prediction samples estimated using the reconstructed samples of the chroma components [4] (Note that there are separate values of α and P_{DC} for U and V components).

7. **Intra Block Copy (IBC) mode**: a special mode where the prediction samples are generated using reconstructed samples from within the same frame. An integer displacement vector is signaled in the bitstream to find the position of the reference block within the same frame. Reconstructed samples within 256 samples to the left of the current SB are unavailable for prediction to reduce the impact on and improve pipelining of hardware (HW) implementations.

8. **Palette mode**: prediction samples are generated by encoding an index map for every sample within the block. The indices refer to entries in a palette of sizes ranging from 2 to 8 (decided using an encoder search scheme), where the mapping from the indices to the sample intensity value is also signaled in the bitstream.

2.3 Inter prediction

AV1 supports multiple inter prediction modes and tools, that include the following:

1. **Overlapped Block Motion Compensation (OBMC)**: OBMC is one of the three motion modes available in AV1, where it uses neighboring motion information to create additional prediction samples which are combined with the original prediction samples. The final prediction samples are generated using the following formula:

$$P(x,y) = m(i) \cdot P_C(x,y) + (1 - m(i)) \cdot P_N(x,y) \tag{6}$$

where $m(i)$ is a masking function that considers the distance i to the block boundary as the input. P_C and P_N represent the prediction using the motion information of the current block and that of the neighboring block, respectively. The blending occurs for the top neighbors followed by the left neighbors, and the maximum distances for blending are 1⁄2 of the block height and 1⁄2 of the block width for the top and left neighbors, respectively [5].

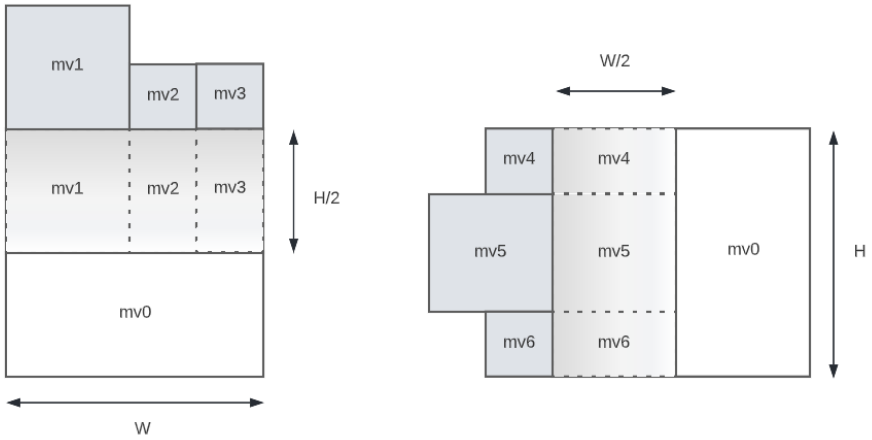

Figure 3. Illustration of OBMC. The solid dark blocks are neighboring blocks, and the prediction samples within the dashed lines are generated using a linear combination of the prediction samples associated with motion vectors of both neighboring and the current blocks.

2. **Local Warp**: Local Warp is another motion mode which derives 8x8 sub-block motion vectors to simulate affine motion for an inter block. The derivation of affine parameters is based on the motion information of the top and left blocks, as no additional syntax element is signaled for this mode [6].

3. **Global motion**: This coding tool provides a way to code videos that may be characterized by global warp motion (up to affine projection) more efficiently. A set of frame-level affine parameters is signaled in the frame header with respect to every reference frame. This replaces the ZERO_MV in VP9 as a more generic way of signaling global motion [6]. The global motion uses the following matrix of parameters [7]:

$$z\begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{7}$$

Global motion supports three types of projections: affine, rotation and scaling, and translation

For affine projection, $h_{31} = h_{32} = 0$, $h_{33} = 1$, and it preserves parallelism

For rotation and scaling only, $h_{31} = h_{32} = 0$, $h_{33} = 1$; $h_{11} = h_{22}$, $h_{12} = -h_{21}$

For translation, $h_{31} = h_{32} = 0$, $h_{33} = 1$; $h_{11} = h_{22} = 1$, $h_{12} = h_{21} = 0$

4. **Motion Field Motion Vectors (MFMV)**: In AV1, a set of temporal motion vectors are generated by scanning through the stored motion vectors from multiple reference frames. For each reference frame, a linear, temporal scaling is applied to project the stored motion vectors from the reference frame to an 8x8 sub-block in the current

frame. This forms a motion field, which can be used in the derivation process of the reference motion vector stack to find more effective motion prediction candidates [8].

5. **Dual filter**: In AV1, interpolation filter types can be signaled separately for the horizontal and vertical directions. There are three types of interpolation filters (Regular, Smooth, Sharp) which can be determined in the encoder for block sizes greater than or equal to 8x8. There are only two interpolation filter types (no Sharp type) for sub8x8 blocks.
6. **Extended reference set**: The reference frame types are extended from 3 in VP9 to 7 in AV1. This allows a more flexible selection of reference frames for different blocks within a frame [9].

In AV1, a set of extended compound modes is also adopted to improve the quality of compound prediction. Five extended compound modes are supported in AV1 [10]. For these extended compound modes, the final prediction samples are generated using the following equation:

$$P(x,y) = w \cdot P_0(x,y) + (1-w) \cdot P_1(x,y) \tag{8}$$

where w is a weight value from a look-up table, and can be either block-level (e.g., distance-weighted compound) or pixel-level (e.g., difference-weighted compound). P_0 and P_1 are the prediction samples from the first and the second reference frames, respectively. Note that the second reference frame here can refer to the current frame to support the inter-intra compound modes that are supported in AV1.

1. **Distance-weighted compound**: in this compound mode, prediction samples from both reference frames are weighted based on their relative distance (calculated by their associated OrderHints values) to the current frame. A fixed look-up table is provided to decide the block-level weights for both predictions.
2. **Inter-inter wedge**: each block is further partitioned into two parts based on a codebook of size 16, where each entry of the codebook defines a specific partitioning long oblique angle within a block. Each part is mainly generated by a motion vector from one of the reference frames, and soft-cliff-shaped masks are defined to smoothly blend the two predictions along the partitioning boundary.
3. **Inter-intra wedge**: this mode is similar to the inter-inter wedge mode and uses the same codebook of wedge shapes, with the main difference that, while one of the predictors is generated using motion information, the other predictor is generated with a specific intra prediction mode. Four intra prediction modes are supported: DC, SMOOTH, VER, HOR.
4. **Smooth inter-intra mode**: This mode accounts for a smoothly decaying pattern with direction of (intra) pixel extrapolation; the weights are a function of the intra modes, block sizes, and a primitive 64-tap, 1D decaying function. The weight maps are created such that the intra-prediction samples are weighted more heavily towards the block boundary to the top and left neighbors, and inter-prediction samples are weighted more heavily towards the bottom/right corner of the block.
5. **Difference-weighted compound**: the weight value for each pixel is calculated based on the absolute pixel difference between the two reference frames. A sign bit is signaled to indicate that either w or $1-w$ should be applied to the prediction from the first reference frame. This mode allows to choose prediction samples from one of the reference frames when the pixel difference is large.

2.4 Transform

Several transforms are supported for the coding of residuals in AV1 in addition to DCT/ADST supported by VP9. Additionally, AV1 supports:

1. **FlipADST and identity transform**: This extended set of transforms (along with DCT and ADST in VP9) provides more flexibility to AV1 to adapt to local residual signals for coding coefficients more efficiently. The identity transform, especially, is an important tool for coding screen content, where fine textual regions can benefit from transform-free operations to preserve their edge information [11][12].
2. **64x64 transform**: AV1 extends the maximum transform kernel from 32x32 (as in VP9) to 64x64. In addition, AV1 also allows rectangular transforms, recursive transform partitioning, as well as up to two levels of the transform split from the associated block. Only DCT is available for the 64x64 transform size.

2.5 In-loop filters

Two new in-loop filters are supported in AV1:

1. **Constrained Directional Enhancement Filter (CDEF)**: the CDEF allows the codec to apply a nonlinear deringing filter while preserving edges. It operates in 8x8 units, where non-separable 5x5 directional filters are applied to estimate the direction, with a selection from eight predefined directions combined with options for rotation and reflections.

   The pixel filtering operation involves primary and secondary filters where the primary filtering follows the direction d and the secondary filter is operated with both plus and minus 45-degree apart from d. In the frame header, up to eight groups of filter parameters can be signaled, and each 64x64 block region can select one group from the frame header to perform filtering with the associated filter parameters. The final filtered samples are generated using

   $$P_f(x,y) = P(x,y) + \sum_{m,n} w^P_{m,n} \cdot f(P(m,n) - P(x,y), S^P, D^P) + \sum_{m,n} w^S_{m,n} \cdot f(P(m,n) - P(x,y), S^S, D^S) \quad (9)$$

   where $P(x,y)$ represents the pre-filtered samples, $w^P_{m,n}$ and $w^S_{m,n}$ represent the filter coefficients for the primary and secondary filters, respectively. $f(\cdot)$ represents a nonlinear function which de-emphasizes neighboring samples with a large difference of sample values between the unfiltered samples with strengths $\{S^P, S^S\}$ and damping factors $\{D^P, D^S\}$ as control parameters for both the primary and secondary filters [13]. Note that it is allowed to signal a separate (from luma) set of strengths and damping factors for the combined chroma components.

2. **Loop restoration filters**: the loop restoration filter is applied to 64x64, 128x128, or 256x256 loop restoration units (LRU), which is the basic unit sharing the same signaled parameters. For each LRU, encoders can select to either skip filtering, or select one from the two filter types: Wiener filter and self-guided filter. The Wiener filter uses asymmetric and separable filter design with 7/5/3-tap filters, and hence the coefficients signaled in the bitstream can be reduced by half (3/2/1). For self-guided filters, two simple denoising filters with support of 3x3 and 5x5 samples are applied. This is done firstly to generate two denoised versions (namely R_1 and R_2) of the post-CDEF images (denoted as R), where the associated noise parameters are signaled in the bitstream. Finally, the filter output is generated by

   $$R + \alpha \cdot (R_1 - R) + \beta \cdot (R_2 - R) \quad (10)$$

   where α and β are signaled in the bitstream [14].

3. TEST RESULTS

3.1 Methodology

In the following results, testing is done on the objective-1-fast video set [15], a video test set that was selected during the development of the AV1 standard. This test set includes 30 sequences (eleven 1080p, seven 720p, seven 360p, and five synthetic videos). libaom-research is a branch which was forked off from the master branch in June 2020, intended to be the codebase for research purposes and for future development beyond the AV1 specification. The test conditions use single-pass encoding with a static sub-GOP structure of size of 16 frames. The libaom-research branch also includes other software optimizations, code clean-ups, encoder speed-ups, as well as software documentation. In this paper, an early version of libaom-research (commit# 3696422f1 [16][17]) was chosen as the software to test, which is AV1-compatible. QP values are selected to be equal to [23, 31, 39, 47, 55, 63]1 with 16-bit internal pipeline and Random-Access (RA) configuration. BD-rate values of frame-averaged PSNR, SSIM, and VMAF are reported as benchmarks to evaluate the coding performance of each tool. The quality metrics are calculated using libvmaf-v1.5.2 [18].

3.2 Tool-on tests

Figure *4* shows the encoder configuration of each test. In tool-on tests, one tool is enabled at a time while all other tools are disabled. Test 0 is a baseline test where all AV1 tools are disabled. Note that the inter tools are tested with 'OrderHint' enabled in the encoder configuration. 'OrderHint' represents an expected output order of the current frame. Bi-directional compound modes are enabled when 'OrderHint' is enabled in this version of the libaom software, in order to measure the coding performance of some inter tools correctly.

Test id	cli option	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1. 128x128 SB	sb-size	64	128	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64
2. sub 8x8 partition	max-partition-size	64	128	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64	64
3. Rectangular partition	min-partition-size	8	8	4	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
4. AB partition	enable-rect-partitions	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5. 1-to-4 partition	enable-ab-partitions	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6. Intra angle delta	enable-1to4-partitions	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7. Paeth intra	enable-angle-delta	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8. Smooth intra	enable-paeth-intra	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9. Intra edge filter	enable-smooth-intra	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10. Filter intra	enable-intra-edge-filter	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11. combined 6 - 10	enable-filter-intra	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12. CFL	enable-cfl-intra	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13. Intra BC	enable-intrabc	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14. Palette	enable-palette	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15. OBMC	enable-obmc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16. Local Warp	enable-warped-motion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17. Global Warp	enable-global-motion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18. MFMV	enable-ref-frame-mvs	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19. Dual filters	enable-dual-filter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20. Full reference set	reduced-reference-set	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	enable-order-hint	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
21. One-sided compound	enable-onesided-comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0	0
22. Diff-wtd compound	enable-diff-wtd-comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
23. Inter-inter wedge	enable-interinter-wedge	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
24. Dist-wtd compound	enable-dist-wtd-comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
25. All inter compound		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26. Inter-intra wedge	enable-interintra-wedge	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
27. Smooth inter intra	enable-smooth-interintra	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
28. Flip and IDTX	enable-flip-idtx	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
29. Full TX type set	reduced-tx-type-set	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
30. Transform size 64	enable-tx64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
31. CDEF	enable-cdef	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
32. Loop restoration	enable-restoration	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
33. CDEF + LR																																			

Figure 4. List of tests with their associated encoder configuration for tool-on tests

Table 1. Summary of tool-on test results

	Name	PSNR-Y	SSIM	VMAF	Enc Time	Dec Time
Partitioning	128x128 SB	-0.43%	-0.77%	-0.92%	117.07%	96.60%
	sub8x8 Partition	-1.89%	-0.32%	-1.06%	109.33%	95.95%
	Rect. Partition	-3.41%	-3.98%	-3.59%	153.44%	95.04%
	AB Partition	-0.43%	-0.35%	-0.48%	103.71%	99.57%
	1-to-4 Partition	-0.55%	-0.86%	-0.68%	112.75%	105.29%
Intra	Intra Angle Delta	-2.42%	-2.61%	-2.70%	103.74%	101.31%
	Paeth Intra	-0.14%	-0.18%	-0.37%	101.95%	100.56%
	Smooth Intra	-0.90%	-1.02%	-1.45%	104.63%	103.21%
	Intra Edge Filter	-0.36%	-0.42%	-0.41%	102.47%	102.62%
	Filter Intra	-0.77%	-1.07%	-1.91%	109.62%	103.36%

	Combined test of the above 5 tools	-3.50%	-4.01%	-4.68%	115.41%	105.98%
	CFL	-0.36%	-0.16%	-0.33%	101.04%	101.11%
SCC	Intra BC	-4.81%	-4.96%	-5.13%	102.75%	101.62%
	Palette	-6.47%	-4.06%	-4.93%	100.77%	103.18%
Inter	OBMC	-1.31%	-1.08%	-0.64%	118.47%	106.40%
	Local Warp	-1.21%	-1.26%	-1.44%	102.79%	101.44%
	Global Motion	-0.31%	-0.38%	-0.57%	135.68%	101.00%
	MFMV	-1.06%	-1.33%	-1.39%	118.06%	117.48%
	Dual Filter	-0.13%	0.09%	0.09%	102.68%	99.57%
	Full Reference Set	-0.83%	-0.68%	-0.59%	140.98%	99.14%
Extended compound	One-sided Compound	-0.38%	-0.17%	0.06%	104.87%	100.60%
	Difference-weighted Compound	-0.65%	-0.30%	0.25%	110.48%	95.82%
	Inter-inter Wedge	-1.64%	-0.77%	-0.51%	128.48%	99.47%
	Distance-weighted Compound	-0.58%	-0.29%	0.30%	107.13%	97.99%
	All Inter Compound	-1.96%	-1.00%	-0.45%	137.16%	96.84%
	Inter-intra Wedge	-1.20%	-0.84%	-0.47%	126.92%	101.36%
	Smooth Inter-intra	-1.12%	-0.90%	0.13%	110.99%	100.24%
Transform	Flip and IDTX	-1.30%	-0.10%	-0.13%	103.33%	99.42%
	Full TX Type Set	-1.24%	-0.76%	-1.46%	103.44%	98.30%
	Transform Size 64	-2.32%	-2.93%	-2.43%	106.15%	95.78%
In-loop Filters	CDEF	-2.23%	-1.22%	2.33%	102.92%	112.25%
	Loop Restoration Filter	-2.80%	-1.36%	-1.56%	112.12%	123.95%
	CDEF + LR	-3.90%	-2.16%	-2.65%	112.69%	127.52%

3.2.1 Partitioning tools

The baseline configuration in this category includes maximum SB size of 64x64, minimum-partition-size=8 (disallow blocks smaller than 8x8), and only the square partition is enabled. In VP9, rectangular partitions are supported as well as sub8x8 partitions but with a constraint that all sub8x8 partitions share the same reference frame type in case of inter blocks. Note that the AB-partition and the 1-to-4 partition are derivative partitioning types of the rectangular partition, and hence the rectangular partition is also enabled for the associated tests. The importance of the sub8x8 partition should be highlighted by additionally showing its impact on the synthetic videos. In the test of 'minimum-partition-size=4', the results show -7.14%/-1.41%/-4.38% coding gain on PSNR-Y/SSIM/VMAF on the five synthetic videos in the objective-1-fast test set.

3.2.2 Intra coding tools

The most significant tool in this category is the intra angle delta, which provides high coding gain with relatively low increase in runtime. For Chroma from Luma (CfL), although it only shows marginal coding gains on luma-based quality metrics, it provides high coding gain on chroma-based quality metrics (e.g., -10.03%/-9.87% on PSNR-U/PSNR-V). In addition, CfL shows significant coding gain on both natural and synthetic videos. From the test 'All Intra Tools' which

enables the five new intra coding tools, it can also be observed that around 73.5% (compared to the sum of coding gains from individual test) of the total coding gain is preserved, showing that the interaction among coding tools in this category is modest.

3.2.3 SCC tools

It should be noted that there is a screen content detection algorithm used to detect whether the to-be-encoded frames are screen content or not, which specifically determines the two frame-level syntax elements 'allow_screen_content_tools' (the top-level flag for both palette and intra block copy) and 'allow_intrabc' (flag for intra block copy) using separate threshold values. In these tests, only Minecraft and Wikipedia out of the five synthetic videos (along with Life, DOTA2, StarCraft) are classified to allow screen content tools, while only Wikipedia is detected with intra block copy enabled. For intra block copy, it provides -24.03%/-24.80%/-25.67% on PSNR-Y/SSIM/VMAF BD-rate savings on Wikipedia.

3.2.4 Inter tools

Note that the inter tools are tested with 'OrderHint' enabled in the encoder configuration. In the baseline, the number of allowed combinations for reference frame selection is limited to 4 single-directional and 7 bi-directional compound modes, which greatly reduces the total number of combinations the encoder needs to search. Hence, in the test of 'Full Reference Set', the increase in encoder runtime is significant as the encoder needs to search from a full set of bi-directional compound modes. For global motion, the encoder needs to invoke a RANSAC-based, frame-level algorithm to find the parameters of the affine model against each reference frame, and hence there is a significant amount of encoder runtime increase when global motion is enabled. The coding performance of global motion is highly sequence dependent. It is observed that -8.61%/-10.90%/-12.16% (PSNR-Y/SSIM/VMAF) coding gain is achieved for the sequence 'BlueSky', while its effect on other sequences is minor.

3.2.5 Inter tools: extended compound modes

For the extended compound modes, there are five coding tools tested in this category. The 'One-sided Compound' refers to the compound modes where both reference frames are from the same temporal direction. In this category, both inter-inter wedge and inter-intra wedge modes contribute higher coding gain than other coding tools. The smooth inter-intra mode also provides high coding gain given its relatively lower encoding runtime increase.

3.2.6 Transform tools

In AV1, for 32x32 transform blocks, only DCT and the identity transforms are employed for inter blocks while only DCT is used for intra blocks. In the baseline, the syntax element 'reduced-tx-type-set' is set to 1. This indicates that only the DCT and identity transform for inter blocks of other sizes are used, while the set of VP9 transforms (combination of ADST and DCT) plus identity transform are available for intra blocks of other sizes. 'Full TX Type Set' additionally tests the coding performance of allowing ADST for chroma blocks on top of the VP9 setup. The test of 'Transform Size 64' refers to enabling the transform block sizes of Nx64, and 64xN, where $N \in \{16, 32, 64\}$.

3.2.7 In-loop filters

CDEF and Loop Restoration filters provide -2.23%/-2.80% BD-rate saving on PSNR-Y, and -1.22%/-1.36% BD-rate saving on SSIM respectively. When both tools are enabled simultaneously, there is overlap in coding gain where -3.90% BD-rate saving on PSNR-Y and -2.16% BD-rate saving on SSIM is observed. It is also observed that CDEF endures BD-rate impact on VMAF, and it is reported that when CDEF is enabled compared with the baseline, the reconstructed frames tend to be blurrier, causing the VMAF score to be lower.

3.3 Tool-off tests

The software version used in the tool-off tests is the same as used in the tool-on testing, while the baseline of the tool-off test enables all AV1 coding tools but disables the tool to be tested individually in each test. In contrast to the tool-on tests, it is expected to observe BD-rate increases and runtime decreases in the tool-off tests. Figure *5* shows the list of tests and their associated encoder command line options.

Test id	cli option	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	super_block_size	128	64	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128
1. 128x128 SB	max_partition_size	128	64	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128	128
2. sub 8x8 partition	min_partition_size	4	4	8	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
3. Rectangular partition	disable_rect_partition_type	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4. AB partition	disable_ab_partition_type	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5. 1-to-4 partition	disable_1to4_partition_type	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6. Intra angle delta	disable_intra_angle_delta	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7. Paeth intra	disable_paeth_intra	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8. Smooth intra	disable_smooth_intra	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9. Intra edge filter	disable_intra_edge_filter	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10. Filter intra	disable_filter_intra	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11. CFL	disable_cfl	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12. Intra BC	disable_intrabc	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13. Palette	disable_palette	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14. OBMC	disable_obmc	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15. Local Warp	disable_warp_motion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16. Global Warp	disable_global_motion	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17. MFMV	disable_ref_frame_mv	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18. Dual filters	disable_dual_filter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19. Reduced reference set	reduced_reference_set	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20. One-sided compound	disable_one_sided_comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
21. Masked compound	disable_masked_comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
22. Diff-wtd compound	disable_diff_wtd_comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
23. Inter-inter wedge	disable_inter_inter_wedge	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
24. Dist-wtd compound	disable_dist_wtd_comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
25. Inter-intra compound	disable_inter_intra_comp	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
26. Inter-intra wedge	disable_inter_intra_wedge	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
27. Smooth inter intra	disable_smooth_inter_intra	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
28. Flip and IDTX	disable_flip_idtx	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
29. Full TX type set	reduced_tx_type_set	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
30. Transform size 64	disable_tx_64x64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
31. CDEF	disable_cdef	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
32. Loop restoration	disable_lr	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 5. List of tests with associated encoder configuration for tool-off tests

Table 2. Summary of tool-off test results

	Name	PSNR-Y	SSIM	VMAF	Enc Time	Dec Time
Partitioning	128x128 SB	0.72%	1.05%	1.27%	85.75%	102.66%
	sub8x8 Partition	1.36%	0.44%	0.75%	95.25%	98.52%
	Rect. Partition	5.11%	5.29%	5.06%	66.57%	101.99%
	AB Partition	0.30%	0.29%	0.45%	95.54%	99.76%
	1-to-4 Partition	0.75%	0.75%	0.64%	95.35%	99.09%
Intra	Intra Angle Delta	1.59%	1.81%	1.69%	99.74%	99.07%
	Paeth Intra	0.03%	-0.07%	0.01%	99.59%	99.10%
	Smooth Intra	0.23%	0.18%	0.12%	98.75%	99.57%
	Intra Edge Filter	0.23%	0.14%	0.21%	99.98%	100.03%
	Filter Intra	0.41%	0.39%	0.72%	98.87%	99.64%
	CFL	0.25%	-0.06%	-0.05%	100.40%	99.12%
SCC	Intra BC	5.93%	6.24%	6.80%	97.18%	99.17%
	Palette	1.92%	0.36%	0.89%	99.17%	100.52%
Inter	OBMC	0.44%	0.48%	0.30%	91.87%	98.71%

	Local Warp	0.86%	1.00%	0.89%	99.16%	98.51%
	Global Warp	0.06%	0.00%	-0.09%	93.76%	97.59%
	MFMV	0.77%	0.93%	0.87%	88.96%	94.43%
	Dual Filter	-0.02%	-0.13%	0.01%	97.77%	100.48%
	Reduced Reference Set	0.81%	0.63%	0.61%	77.59%	99.75%
Extended Compound	One-sided Compound	0.17%	0.09%	0.10%	95.01%	99.81%
	Masked Compound	0.34%	0.17%	0.11%	88.29%	101.25%
	Difference-weighted Compound	0.19%	0.10%	0.08%	96.23%	100.02%
	Inter-inter Wedge	0.22%	0.09%	0.12%	92.48%	98.79%
	Distance-weighted Compound	0.02%	0.02%	-0.09%	97.45%	99.76%
	Inter-intra Compound	0.33%	0.33%	0.01%	93.39%	99.60%
	Inter-intra Wedge	0.05%	0.04%	0.08%	96.22%	100.00%
	Smooth Inter-intra	0.24%	0.28%	-0.01%	100.21%	98.87%
Transform	Flip and IDTX	1.62%	0.42%	0.30%	95.12%	99.78%
	Reduced TX Type Set	2.61%	3.38%	3.36%	93.01%	100.53%
	Transform Size 64	2.34%	1.40%	1.68%	94.04%	100.57%
In-loop Filters	CDEF	0.60%	0.32%	0.66%	100.27%	96.23%
	Loop Restoration Filter	1.52%	1.00%	5.94%	97.96%	90.62%

3.3.1 Partitioning tools

Like the results from the tool-on tests on partitioning tools, rectangular partitioning contributes most of the coding gain. In addition, the effect of 'sub8x8 Partition' (i.e., all sub8x8 partitions are disabled in this test), shows 5.06%/3.29%/3.38% (PSNR-Y/SSIM/VMAF) BD-rate increase for the synthetic video class.

3.3.2 Intra coding tools

It can be observed that the coding gain of the tools is reduced compared to the results of tool-on tests. As RA is used as the baseline test configuration, it is expected that more inter blocks are chosen from the encoder perspective. For Paeth-intra prediction, it shows smaller changes in coding performance and runtime in this baseline. For filter intra, the coding gain is approximately half of the gain in the corresponding tool-on test, and for smooth intra, its coding performance is reduced to 0.1% ~ 0.2%. For CfL, the PSNR-U/V BD-rate performance is 12.11%/8.97%, respectively.

3.3.3 SCC tools

It can be observed that for IBC, there is a larger impact on the Wikipedia sequence, i.e., 30% ~ 33%, when compared to the tool-on test, mainly due to the sub8x8 partitions being enabled in this baseline. Like in the tool-on test, Palette modes are only effective on Minecraft and Wikipedia showing 4.7% BD-rate impact on average for PSNR. This implies that Palette mode may have larger overlap with IBC in this context, and it shows relatively lower coding gain in terms of SSIM and VMAF.

3.3.4 Inter tools

It can be observed that for global- motion and for the dual-filter, their coding impact is limited while they still contribute 6.24% and 2.23% of the encoding runtime, respectively. Also note that for OBMC and Local Warp, their coding performance is greatly reduced (from -4.24% and -2.55%, respectively) compared to the tool-on cases. The 'Reduced

Reference Set' allows a limited amount of combination of reference frames (4 single-reference, 4 inter-intra modes, 7 compound modes) which significantly simplifies the encoder search with only moderate increase in BD-rate.

3.3.5 Inter tools: extended compound modes

From the tool-off test results on the extended compound modes, it can be observed that the 'inter-intra wedge', 'difference-weighted compound', and 'smooth inter-intra' modes are the more effective coding tools in this category. Both the 'inter-intra wedge' and 'distance-weighted compound' modes show very limited coding gain. The test 'Masked Compound' is a combined test which includes both the 'Difference-weighted Compound' and the 'Inter-inter Wedge', and the test 'Inter-intra Compound' combines the 'Inter-intra Wedge' and 'Smooth Inter-intra'. Hence, the coding gains of these two tests appear to be higher than other tools which are tested individually.

3.3.6 Transform tools

In the test of the 'Flip and IDTX' transforms, majority of the coding impact comes from synthetic videos where the identity transforms on top of enabling the sub8x8 partitioning can be a very helpful combination to improve coding efficiency. 'Transform Size 64' is a helpful coding tool which contributes 3% ~ 4% coding gain on 720p and 1080p videos. 'Reduced TX Type Set' limits inter blocks to use only the DCT and identity transforms and limits the intra blocks to use ADST for block sizes less than 32x32. It eliminates the selection of a variety of transform kernels for blocks smaller than or equal to 16x16, showing significant increase in BD-rate with only 7% decrease in encoding runtime.

3.3.7 In-loop filters

Both CDEF and LR show relatively low coding gain in the tool-off testing. LR shows moderate coding gain of 1.52% on PSNR-Y BD-rate, while it shows 5.94% impact on VMAF BD-rate.

4. SUMMARY

This paper provides a brief description on a set of AV1 coding tools with test results using libaom as the reference software. It should be highlighted that the test is not exhaustive and only tools categorized as more critical or of interest were analyzed. In particular, the performance of some coding tools, such as reference motion vector derivation [19] and the M-ary symbol-based arithmetic coding [20], which also show improvement compared to their predecessor designs, were not included since it was found to be difficult or impossible to switch them off in the codebase and evaluate them. Other AV1 coding tools, such as frame super-resolution [21], film grain synthesis [22], reference frame re-sampling, and tile groups require specific test conditions that would reflect specific application scenarios to demonstrate their benefits. Tests on these tools are beyond the scope of this paper.

REFERENCES

[1] P. de Rivaz, J. Haughton, A. Grange, L. Quillio, "AV1 Bitstream & Decoding Process Specification," http://aomedia.org/av1/specification/

[2] Y. Chen et. al., "An Overview of Coding Tools in AV1: the First Video Codec from the Alliance for Open Media," APSIPA Transactions on Signal and Information Processing, vol. 9 (e6), pp. 1-15, 2020

[3] J. Han et. al., "A Technical Overview of AV1," Proceedings of the IEEE, vol. 109, no. 9, pp. 1435-1462, Sept. 2021

[4] Luc N. Trudeau, Nathan E. Egge and David Barr, "Predicting Chroma from Luma in AV1," Data Compression Conference (DCC), 2017.

[5] Y. Chen, D. Mukherjee, "Variable block-size overlapped block motion compensation in the next generation open-source video codec," in IEEE Int. Conf. on Image Processing (ICIP), September 2017.

[6] S. Parker, Y. Chen, D. Mukherjee, "Global and locally adaptive warped motion compensation in video compression," in IEEE Int. Conf. on Image Processing (ICIP), September 2017.

[7] https://gitlab.com/AOMediaCodec/SVT-AV1/-/tree/master/Docs

[8] J. Han, J. Feng, Y. Teng, Y. Xu, J. Bankoski, "A motion vector entropy coding scheme based on motion field referencing for video compression," in IEEE Int. Conf. on Image Processing (ICIP), September 2018.

[9] Lin W. et al., “Efficient AV1 video coding using a multi-layer framework,” Data Compression Conference (DCC), March 2018.
[10] J. Urvang et. al., “Novel inter and intra prediction tools under consideration for the emerging AV1 video codec,” in Proc. SPIE, Applications of Digital Image Processing XL, 2017.
[11] J. Han, Y. Xu, and D. Mukherjee, “A butterfly structured design of the hybrid transform coding scheme,” in Proc. Picture Coding Symp. (PCS), Dec. 2013.
[12] S. Parker et al., “On transform coding tools under development for VP10,” in Proc. SPIE, Applications of Digital Image Processing XXXIX, 2016.
[13] S. Midtskogen, J.-M. Valin, “The AV1 Constrained Directional Enhancement Filter,” in Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP), Apr. 2018.
[14] D. Mukherjee, S. Li, Y. Chen, A. Anis, S. Parker, and J. Bankoski, “A switchable loop-restoration with side-information framework for the emerging AV1 video codec,” in Proc. IEEE Int. Conf. Image Process. (ICIP), Sep. 2017.
[15] objective-1-fast video set, https://media.xiph.org/video/derf/objective-1-fast.tar.gz
[16] libaom-research, https://aomedia.googlesource.com/aom/+/3696422f122e78e0f803e2d809e1b8f21a7d1aea
[17] Allow disabling TX64 for AOM_IMG_FMT_I42016, https://aomedia-review.googlesource.com/c/aom/+/115802
[18] libvmaf-v1.5.2, https://gitlab.com/m-ab-s/vmaf/-/tags/v1.5.2
[19] J. Han, Y. Xu, J. Bankoski, “A dynamic motion vector referencing scheme for video coding,” in IEEE Int. Conf. on Image Processing (ICIP), September 2016.
[20] J.-M. Valin et al., “Daala: Building a next-generation video codec from unconventional technology,” in Proc. IEEE 18th Int. Workshop Multimedia Signal Process. (MMSP), Sep. 2016.
[21] U. Joshi, D. Mukherjee, Y. Chen, S. Parker, A. Grange, “In-loop frame super-resolution in AV1,” in Proc. Picture Coding Symp. (PCS), Nov. 2019.
[22] A. Norkin and N. Birkbeck, “Film grain synthesis for AV1 video codec,” in Proc. Data Compress. Conf. (DCC), Mar. 2018.