Generalized deblocking filter for AVM

Andrey Norkin
Encoding Technologies
Netflix
Los Gatos, CA, USA
anorkin@netflix.com

***Abstract*—The AV1 deblocking filter did not sufficiently attenuate visual quality artifacts when used in the AOMedia video model (AVM), especially at lower bitrates. A generalized deblocking filter described in this paper uses one equation for any filter length. The filter brings PSNR-YUV BD-rate –0.17%, –0.90%, –1.07%, and –0.92% on All Intra, Random Access, Low-Delay, and Adaptive Streaming configurations in the AOMedia common test conditions. Improvement of visual quality is observed on a number of sequences, while the computational complexity is close to that of the AVM deblocking.**

Keywords—deblocking, AV1, AVM, codecs, video compression

I. Introduction

AV1 specification [1] was finalized by the Alliance for Open Media (AOMedia) in 2018. Recently, AOMedia has started exploration of video compression technologies beyond AV1. The exploration efforts are based on a joint software base, called the AOMedia Video Model (AVM) [2], which is publicly available online. The tools evaluation work is happening in stages, where experiments at each stage use the same version of the AOMedia software called anchor.

This paper describes the deblocking filter proposed in these exploration efforts and adopted in the AVM software. The deblocking filter is a video coding tool used in the majority of recent video coding standards. Since a typical video codec operates on rectangular blocks, the signal representation of each block at lower bitrate can deviate from the underlying signal. As transform and quantization is applied independently in each coding blocks, this may cause visually noticeable boundaries between the neighboring blocks, called block artifacts. The deblocking filtering is designed to mitigate these artifacts.

The deblocking is usually used in-loop, i.e. applied to the decoded pictures before storing them as reference pictures for further prediction. Applying deblocking to reconstructed pictures helps to not only improve the output video quality but also quality of reference pictures, thus resulting in further compression efficiency improvements. Different algorithms for in-loop deblocking filtering exist, such as deblocking in H.264 [3] and HEVC standards [4]. The deblocking filter used in VVC standard [5] is based on the HEVC deblocking filtering with some longer deblocking filters applied to larger blocks and short deblocking filters applied to 4x4 blocks.

Deblocking filters typically use information sent in the video bitstream (such as location of the transform and prediction block boundaries, block sizes, and quantization parameters) to determine picture locations where discontinuities may appear. Then, the deblocking filtering evaluates the signal at the sides of the block boundary to determine the strength of the filtering to be applied and whether the filtering should be applied at all. Typically, stronger filtering is applied when the signal is smooth on both sides of the boundary, which indicates that a discontinuity is likely caused by compression and may also be more visible in this area.

As other video codecs, AV1 uses a deblocking filter. During the AVM development, some changes have been made to the software, including a different quantization scheme that allowed coding at lower bitrates. The AV1 deblocking had no longer masked all block artifacts effectively in AVM. The deblocking approach described in this paper has been able to improve the subjective quality of AVM, especially at lower bitrates, while also improving the objective metrics.

The paper is organized as follows. Section II gives a brief overview of the AV1 deblocking filtering. Section III explains the proposed AVM deblocking. Section IV describes the results, including objective performance improvements and improvements in visual quality. Section V discusses the decoding complexity aspects, while Section VI concludes the paper.

II. AV1 deblocking filteirng

AV1 deblocking filter uses several filter lengths. For the luma color component, filters can modify 1, 2, 3, and 6 samples from the block boundary. One more sample is required for filtering decisions. For chroma components, 1 or 2-samples from the boundary can be changed. The choice of the maximum filter length is determined by the minimum size of adjacent blocks in the direction of filtering.

The deblocking can be applied if there are transform coefficients present in one of the adjacent blocks or the block boundary is a prediction block boundary.

The filters used in AV1 deblocking are low-pass filters. To avoid over-smoothing of textures, a boundary condition is checked. The boundary samples are classified as low and high variance by using the following equations (see Fig. 1):

$$| s[-2] - s[-1] | > T_0 \quad (1)$$

$$| s[1] - s[0] | > T_0 \quad (2)$$

$$2 | s[0] - s[-1] | + | s[-2] - s[1] | / 2 > T_1 \quad (3)$$

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

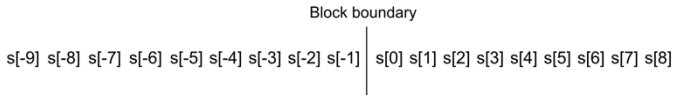

Fig. 1. Block boundary with adjacent samples.

Here $s[i]$ are values of reconstructed samples with $i = -1,\ldots, -N-1$ located on the left (top) and $i = 0, \ldots N$ on the right (bottom) of the block boundary. T_0 and T_1 are threshold values that can be adjusted at the frame or frame segment level. Also, the filtering strength and thresholds can be set separately for the vertical and horizontal block boundaries in the luma component. In chroma components, the same value of the deblocking strength is used for both vertical and horizontal boundaries.

When the AV1 deblocking filter can modify more than two samples from the block boundary, additional samples are checked by the following conditions:

$$| s[-i-2] - s[-i-1] | > T_0 \quad (4)$$

$$| s[\,i\,] - s[\,i-1\,] | > T_0 \quad (5)$$

III. Proposed deblocking filtering

The deblocking filtering consists of three main stages. The first stage determines locations of block boundaries based on the bitstream and reconstruction information. It also determines whether the deblocking can be applied to the boundary and finds the maximum number of samples that deblocking can modify. At the next stage, the samples on both sides of the boundary are examined, and the length/strength of the filtering may be adjusted based on the local content characteristics. Finally, the deblocking filtering operations are applied based on the decisions made in the first two stages.

A. Bitstream based deblocking filtering decisions

Bitstream-based decisions have not changed significantly from to AV1. The main difference is that for blocks of size 4 in the direction of filtering, only one sample from the block boundary is modified, and three samples accessed for the filtering decisions. This is done to avoid deblocking dependencies across the picture since the deblocking filtering decisions use at least 3 samples from the block boundary. The maximum length of the deblocking filtering has changed, which is described in subsection *D*.

B. Sample based deblocking filtering decisions

The actual length of the deblocking filtering is found based on the sample values at the sides of the block boundary. The filtering decisions are made for each line of the block boundary based on the sample values in this line. For each line, (see Fig. 1) the first derivative of the signal is calculated as

$$d1[\,i\,] = s[\,i+1\,] - s[\,i\,]. \quad (6)$$

Then, the absolute value of the second derivative is found as

$$d2[\,i\,] = |\, d1[\,i\,] - d1[\,i-1\,] \,|. \quad (7)$$

For modification of N samples from the block boundary, all of following conditions should be false. Eq. (8) should be evaluated to false for one sample from the block boundary to be modified. Comparisons in (8) and (9) should be false for modification of 2 samples from the block boundary.

$$d2[\,i\,] > thr1 \text{ for } i = 1, -2 \quad (8)$$

$$d2[\,0\,] + d2[\,-1\,] > thr2 \quad (9)$$

For modification of N samples, where N is greater or equal to 3, comparisons (10) and (11) are evaluated in addition to (8) and (9), which should also be false. The conditions are evaluated for values of N increasing from 0 to the maximum value allowed for the current block, and the evaluation stops once any evaluated condition holds true. The previous value of N is selected as the maximum length of the deblocking filtering.

$$| (s[\,0\,] - s[\,N\,]) - N\,(s[\,0\,] - s[\,1\,]) | > thr3, \quad (10)$$

$$| (s[-1] - s[-N-1]) - N\,(s[-1] - s[-2]) | > thr3 \quad (11)$$

Values of thresholds *thr*2 and *thr*3 depend on N with lower values assigned to higher N, so that longer deblocking filters are applied to smoother areas.

In the above, thresholds *thr*1 and *thr*3 are used for evaluating the texture smoothness on each side of the block boundary and depend on a so-called *side* threshold. The *thr*2 as well as *thr4* from the next sub-section are determined based on the quantization step size. Both the *side* threshold and *q* threshold values depend on the quantization index and can be modified by offsets sent in the bitstream.

C. Deblocking filtering operations

Deblocking filtering operations of AV1 are replaced by the generalized filtering operation. The same equation is used for modifying N samples at the block boundary. Provided the number of samples to be modified by the filter on each side of the block boundary is N, the following operations are applied.

$$\Delta = (\,3(s[0] - s[-1]) - (\,s[1] - s[-2]\,)\,) / 2 \quad (12)$$

$$\Delta' = \mathrm{clip}(\,\Delta, -thr4, thr4\,) \quad (13)$$

$$s'[\,i\,] = s[\,i\,] - \Delta\,(N-i)\,/\,(2N+1), \text{ for } i = 0,\ldots, N-1 \quad (14)$$

$$s'\,[-i-1] = s[-i-1] + \Delta(N-i)/(2N+1), \text{ for } i = 0,\ldots, N-1 \quad (15)$$

Here $s[i]$ are sample values before the deblocking filtering at position i and $s'[i]$ are sample values after the deblocking filtering. The $clip(v, v_{min}, v_{max})$ function limits the value of v to the interval $[v_{min}, v_{max}]$.

Parameter Δ has been derived to minimize the second derivative of the signal across the block boundary provided that the samples at the block boundary are modified in the way described above. The underlying assumption is that deltas for each sample position are inversely proportional to the distance from the block boundary to form a smooth transition between signals on both sides of the boundary (see Figs. 2 and 3).

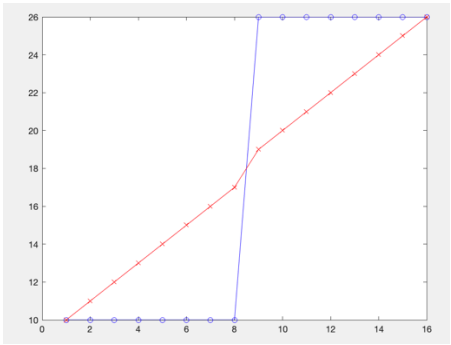

Fig. 2. Example of deblocking for modification of 6 samples from each side of the boundary. Original samples are blue, samples after deblocking are red. The remaining discontinuity in the middle is due to the integer representation.

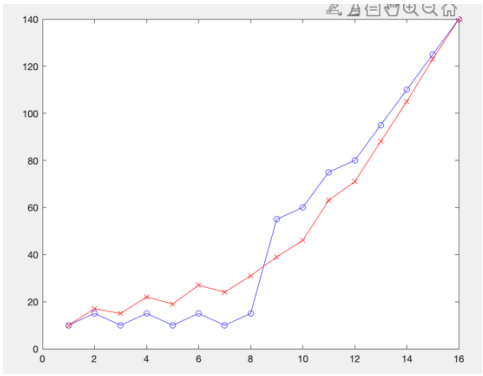

Fig. 3. Example of filtering operations. Note that variations in signal on the side of the block boundaries are preserved after the filtering is applied.

Note that the signal on the sides of the block boundaries in Fig. 3 has variations, it is not exactly smooth. Typically, a low-pass deblocking filter would remove such variations, which may be related to the boundary artifacts or may be part of the original signal. Since AVM has other in-loop filters applied after of deblocking, the proposed deblocking filter avoids modifying the texture on the side of the boundaries, leaving it to other loop filters if necessary.

Filtering operations in (14) and (15) have been implemented in integer arithmetic without use of divisions. Multiplications and shifts have been used instead.

D. Deblocking filtering length

The proposed deblocking AVM filter can modify the following number of samples from each side of the block boundary:

- 1, 2, 3, 4, 6, 8, or 10 samples from each side of the block boundary.
- 1 to 4 samples for chroma.

The number of samples modified by deblocking has been increased compared to AV1. Even though the proposed deblocking method has improved that objective and subjective quality, some artifacts at lower QPs and large blocks sizes (e.g. on 32x32 blocks) could not be completely removed with shorter filters. Longer filters were found to improve visual quality at higher QPs, especially in higher resolution sequences.

In order to keep the same line buffer size as in the AV1 deblocking, the maximum length of filters that could be applied at the horizontal superblock boundary was chosen to be:

- 6 samples for the luma component
- 2 samples for the chroma component

As mentioned previously, filtering decisions require one additional sample at each side of the block boundary, except the case of 4x4 blocks, where 1 sample is modified and additional 2 samples are required for the filtering decision (3 samples in total to find the second derivative).

E. Signaling of deblocking parameters

Signaling of the deblocking parameters has also been modified. In the proposed filter, a default index to the table with threshold values is based on the quantization index. An offset to the default index can be signaled in the bitstream on the frame or frame segment level. In addition, parameters for horizontal luma boundaries can be set equal to the parameters of the vertical luma boundaries to save bits. Otherwise, separate values for the vertical and horizontal parameter indices can be sent in the bitstream. Signaling of the deblocking parameters can thus be more efficient than in AV1, which always sends explicit parameters that determine deblocking thresholds.

F. Other modifications

Some modifications were applied to the offset-based intra prediction refinement (ORIP) tool [9]. For blocks larger than 16 samples in the direction of filtering, the refinement of prediction is switched off for the boundary across the direction of filtering since it may negatively interact with deblocking in some cases.

IV. Results

The algorithm has been implemented in the AVM software [6] version 2.0.0. The experiments have been done according to the AOMedia Common Test Conditions (CTC). The AOMedia CTC include All Intra (AI), Random Access (RA), Low Delay (LD), and Adaptive Streaming (AS) configurations. The Adaptive Streaming configuration uses multi-resolution encoding and construct a convex hull over obtained points to calculate the BD-rate [11]. The objective results are obtained on 60 mandatory CTC test sequences separated into several classes, according to their resolution and content. For example, class B1_SYN contains synthetic content primarily of 1080p resolution. All sequences are 130 frames long, with RA configuration using two closed GOP of 65 frames each. Only first 30 frames are encoded in the All Intra test configuration.

A. Objective results

The results are shown in Tables I-IV. One can observe the PSNR-YUV BD-rates [11] of –0.17%, –0.90%, –1.07%, and –0.92% for the AI, RA, LD, and AS configurations, respectively. The “N/A” in some cells of Table IV corresponding to chroma BD-rates are due to the convex hull optimization based on PSNR of the luma component. Consequently, this may cause non-monotonicity in chroma components of some sequences, for which the BD-rate cannot be calculated.

TABLE I. AVERAGE BD-RATES IN ALL INTRA (AI) CONFIGURATION IN AOMEDIA CTC

Seq. Class	Y	U	V	YUV
A1_4K	-0.14%	-0.59%	-0.65%	-0.20%
A1_2K	-0.12%	-0.49%	-0.59%	-0.16%
A3_720p	-0.16%	-0.74%	-0.64%	-0.21%
A4_360p	-0.24%	-0.86%	-0.75%	-0.27%
A5_270p	-0.23%	-0.51%	-0.25%	-0.24%
B1_SYN	-0.12%	-0.62%	-0.75%	-0.18%
Overall	**-0.15%**	**-0.60%**	**-0.63%**	**-0.19%**

TABLE II. AVERAGE BD-RATES IN RANDOM ACCESS (RA) CONFIGURATION IN AOMEDIA CTC

Seq. Class	Y	U	V	YUV
A1_4K	-1.41%	-1.74%	-1.91%	-1.46%
A1_2K	-0.97%	-1.47%	-1.51%	-1.01%
A3_720p	-0.81%	-1.60%	-1.16%	-0.85%
A4_360p	-0.63%	-1.14%	-1.17%	-0.67%
A5_270p	-0.40%	-0.21%	-1.10%	-0.42%
B1_SYN	-0.59%	-1.50%	-0.80%	-0.64%
Overall	**-0.86%**	**-1.41%**	**-1.31%**	**-0.90%**

TABLE III. AVERAGE BD-RATES IN LOW DELAY (LD) CONFIGURATION IN AOMEDIA CTC

Seq. Class	Y	U	V	YUV
A1_2K	-1.39%	-1.37%	-1.61%	-1.40%
A3_720p	-1.20%	-2.04%	-1.77%	-1.27%
A4_360p	-0.54%	-0.78%	-1.73%	-0.59%
A5_270p	-0.30%	0.15%	-1.44%	-0.32%
B1_SYN	-0.81%	-0.82%	-1.41%	-0.84%
Overall	**-1.05%**	**-1.16%**	**-1.59%**	**-1.07%**

TABLE IV. BD-RATES IN ADAPTIVE STREAMING (AS) CONFIGURATION IN AOMEDIA CTC

Sequence	Y	U	V	YUV
BoxingPractice	-1.49%	-1.21%	-1.37%	-1.47%
CrossWalk	-1.54%	-0.64%	-0.08%	-1.40%
FoodMarket2	-0.80%	-0.42%	-0.31%	-0.75%
Neon1224	-0.98%	-1.69%	-1.35%	-1.05%
NocturneDance	-0.50%	#N/A*	#N/A*	-0.56%
PierSeaside	-0.21%	#N/A*	#N/A*	-0.26%
Tango	-1.45%	-1.21%	-1.02%	-1.40%
Timelapse	-0.57%	#N/A*	#N/A*	-0.49%
Average	**-0.94%**	**#N/A***	**#N/A***	**-0.92%**

* #N/A in U and V cells correspond to non-monotonic *Quality*(*Rate*) curves caused by optimizing the convex hull for the Y component.

B. *Visual quality examples*

Visual quality examples can be seen in Figs. 4-6. The figures use higher values of QP settings (235 and 210) to emphasize the differences when printed, but visual quality improvements have also been observed at lower QPs, including base QP 160. It should be noted that the QP parameter in the AVM configuration settings corresponds to the highest frame QP in the prediction hierarchy, while frames of lower layers use lower QPs. The range of QPs in AVM is from 0 to 255.

C. *Expert viewing test*

An expert viewing test has been performed in the AOMedia Codec Working Group to study the effect of the proposed deblocking on visual quality [10]. The expert viewing was performed by volunteers at their facilities, and the results were collected through the online submission form. The test organizers provided test subjects with the scripts to play the decoded sequences and with scoring sheets. ABAB playback order was used, where A and B were decoded sequences of AVM 2.0.0 and the proposed method. The presentation order of AVM and the proposal was randomized for each pair. The following scale was used: "A is much better than B", "A is better than B", "A is equal to B", "B is better than A", "B is much better than A". The scores were de-randomized and converted to values from {-2, -1, 0, 1, 2} set. The 95% confidence intervals were calculated. Overall, 18 sequence/QP test points have been evaluated based on 9 sequences, 2 QPs per sequence, and a choice of LD and RA configuration encodings. Base QPs of 160, 185, 210, and 235 were used. Overall, 19 test subjects participated in the test.

(a) AVM v.2.0.0 (b) Proposed

Fig 4. Screen shot of sequence *Riverbed*, RA, QP 235.

(a) AVM v.2.0.0 (b) Proposed

Fig 5. Screen shot of sequence *Sol Levante Dragons*, RA, QP 210.

It was found that the proposal was better than the anchor (i.e. statistically significant difference was observed) in 11 of 18 test cases. In 7 test cases, there was no statistically significant difference. In particular, none of the four QP160 test cases showed statistically significant difference, while there were test cases with statistically significant differences among base QPs of 185, 210, and 235.

V. DECODING COMPLEXITY

The decoding complexity has been measured separately since timing on the computational cluster is not reliable. In particular, the decoding of AVM and the proposal was done on

(a) AVM v.2.0.0

(b) Proposed

Fig 6. Screen shot of sequence Tango, RA, QP 235

the same machine, using one GOP of RA configuration of all mandatory CTC sequences, and the total decoding time was obtained. MacBook Pro with 2.3 GHz 8-Core Intel Core i9 was used for the simulations. The decoder output was directed to /dev/null to exclude the effect of writing reconstructed files to the disk.

The first test reported in Table V used the decoder compiled with SIMD. Note that the AV1/AVM deblocking had complete SIMD coverage of sample-based deblocking decisions and filtering operations, while the proposed method had no SIMD optimizations. One can see the decoding time increase of 8.5% over AVM.

To compare the performance of the proposal without the SIMD optimization effect, another comparison was done with both decoders compiled without SIMD in any part of the code (see Table VI). One can observe that the increase in the decoding time is below 1% which indicates the decoding time similar to

TABLE V. DECODING TIME WITH SIMD. AVM'S DEBLOCKING IS FULLY SIMD OPTIMIZED, THE PROPOSAL HAS NO SIMD IN DEBLOCKING

Time AVM (s)	Time proposal (s)	Dec. time (%)
787.87	902.35	114.53%

TABLE VI. DECODING TIME COMPARISON WITH SIMD OFF

Time AVM (s)	Time proposal (s)	Dec. time (%)
3177.38	3199.04	100.68%

the decoding time of AVM deblocking. The increase in the encoding time has not been significant since the deblocking filter runtime is negligible compared to the test model encoding time.

VI. SUMMARY

The proposed deblocking has shown PSNR-YUV BD-rate (average bit rate reduction for the same quality) of –0.17%, –0.90%, –1.07%, and –0.92% for All Intra, Random Access, Low-Delay, and Adaptive Streaming configurations. The number of line buffers is the same as in AV1 deblocking, and the computation complexity is not significantly higher. Visual quality improvements have been observed on a number of test sequences at higher quantization settings.

ACKNOWLEDGMENT

The author would like to express gratitude to the AOMedia Testing sub-group co-chair Yeping Su (YouTube) for his help with the subjective evaluation of the proposal in the AOMedia Codec Working Group.

REFERENCES

[1] AV1 Bitstream & Decoding Process Specification, version 1.0.0 with errata, https://aomediacodec.github.io/av1-spec/av1-spec.pdf

[2] AVM reference software, https://gitlab.com/AOMediaCodec/avm

[3] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz, "Adaptive deblocking filter", in IEEE Trans. Circ. Syst. Video Technol., vol. 13, issue 7, pp. 614-619, July 2003.

[4] A. Norkin, G. Bjontegaard, A. Fuldseth, M. Narroschke, M. Ikeda, K. Andersson, M. Zhou, and G. Van der Auwera, "HEVC deblocking filter", in IEEE Trans. Circ. Syst. Video Technol., vol. 22, pp. 1746 – 1754, Dec. 2012.

[5] M. Karczewicz et al., "VVC In-Loop Filters," in IEEE Trans. Circ. Syst. Video Technol., vol. 31, no. 10, pp. 3907-3925, Oct. 2021.

[6] AOMedia Video Model (AVM) https://gitlab.com/AOMediaCodec/avm

[7] X. Zhao, Z. Lei, A. Norkin, T. Daede, and A. Tourapis, "AOM Common Test Conditions v2.0," CWG-B075o, 2021.

[8] A. Norkin, "Deblocking filter for AVM", AOMedia doc. CWG-C014, Feb. 2022.

[9] M. G. Sarwer and Y. Ye, "Offset based refinement for intra prediction (ORIP)", AOMedia doc. CWG-B019, Mar. 2021.

[10] A. Norkin and Y. Su, "Expert viewing of C014 Deblocking filter", AOMedia doc. CWG-C023, March 2022.

[11] G. Bjontegaard, "Calculation of Average PSNR Differences Between RD Curves", ITU-T-T SG16 document VCEG-M33, Joint Collaborative Team on Video Coding (JCTVC)