Hardware Architecture for the Regular Interpolation Filter of the AV1 Video Coding Standard

Daiane Freitas*, Rafael da Silva†, Ícaro Siqueira*, Cláudio M. Diniz*, Ricardo A. L. Reis†, Mateus Grellert‡
*Graduate Program on Electronic Engineering and Computing, Catholic University of Pelotas (UCPel), Pelotas, Brazil
†Graduate Program on Microelectronics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
‡Embedded Computing Lab, Federal University of Santa Catarina (UFSC), Florianópolis, Brazil

***Abstract*—This work proposes a hardware architecture for fractional-pixel interpolation filter defined in the royalty-free AV1 video coding standard. Analysis conducted in this work shows that the AV1 Regular family of filters has the highest usage especially when considering high resolution videos. The proposed architecture implements the 15 interpolation filters of the AV1 Regular family and is capable to interpolate videos of up to 8K video resolution at 120 fps. The proposed architecture achieves the highest throughput compared to related works.**

***Index Terms*—Video coding, AV1, interpolation filter, hardware, architecture**

I. Introduction

Video technology is a fast-growing and data-intensive consumer technology market. It is estimated that by 2022 digital video will correspond to more than 80% of the total Internet traffic [1]. Such trend has driven the development of advanced video coding standards.

Among current video encoders, High Efficiency Video Coding (HEVC) is the latest ITU/ISO standard developed by the Joint Collaborative Team on Video Coding (JCT-VC) in 2013. It provides 25%−50% higher compression compared to the previous standard, H.264/AVC, with the same visual quality [2]. Recently, ISO announced the development of the new Versatile Video Coding (VVC). Expected to be released this year, the codec will be the successor of HEVC, with higher compression capacity.

Many technologies developed for the HEVC and VVC codecs are subject to royalties, which bring an undesired cost for companies that handle videos. An alternative solution is being developed by the Alliance for Open Media (AOMedia), a consortium co-funded by Google that comprises over 30 technology companies, including Netflix, Youtube, Facebook and Apple, created in 2015 to develop open and royalty-free codecs, in contrast to ITU/ISO standards. AOMedia Video 1 (AV1), released in 2018, in the first codec that incorporates tools from Google's VP10 (not released) [3]–[5], combined with tools from Daala [6] and Thor [7] codecs. AV1 achieves compression ratio reduction of 30% compared to VP9 and HEVC [8]. Considering adaptive streaming video distributions, up to 40% compression ratio efficiency is achieved [9].

Despite the advantages that the coding process adds, this improvement entails high computational cost. While emerging encoders offer ever-increasing compression rates, there is also a significant increase in encoder complexity, specifically in the motion estimation stage (ME) [10], [11]. ME exploits the temporal redundancy in videos by searching for similar blocks in reference frames then indicating their displacements using motion vectors (MV). It is usually performed in two steps: an integer search (called IME), followed by a fractional search (called FME). FME searches over samples that are not actually in the input frame, so it interpolates fractional pixels before any search can occur. This process is performed by filters whose coefficients are defined by the standard.

In the reconstruction step, the information processed by the ME is used in a process called motion compensation (MC) to generate a displayable frame once again. MC combines the residue blocks with the ones pointed by the motion vectors (from ME) to build a reconstructed block. Like in the FME, this component must realize an interpolation process to obtain fractional samples from integer ones when a fractional MV is used. Hence, interpolation filter is an important component of codec design, because it is used in both the encoder and decoder. Therefore, solutions that reduce the complexity and energy requirements of this step benefit both ends.

Several dedicated hardware solutions were proposed to enable performance/energy efficient FME computing on HEVC [12], [13] and VP9 [14] encoders. The work in [15] is the only one that presents a design for the AV1 interpolation filter focusing on MC, but the proposed architecture has a limited output throughput and is unsuited for FME acceleration.

This work proposes an interpolation filter hardware architecture targeting the AV1 encoder. The proposed architecture includes the 15 interpolation filters of the AV1 Regular family and uses Multiple Constant Multiplication (MCM) to optimize filter hardware [16]. It is capable to process up to 8k@120fps. This work also contributes with an analysis that shows which types of filter and which partition sizes are used more often in AV1, considering different QP ranges and resolutions.

Before presenting the contributions of the work (Sections III and IV), a description of the AV1 interpolation in given in Section II. Results and comparisons with related work are presented in Section IV. Section V concludes the work.

II. Background

AV1 standard defines in the specification a Superblock (SB) with a maximum size of 128×128 pixels, which can be subdivided into 2 vertical or horizontal rectangles, 4 vertical or horizontal rectangles, an integer square divided into 4 blocks

TABLE I
REGULAR FILTER COEFFICIENTS

Y	CbCr	Coefficients
-	**1/16**	{2, -6, 126, 8, -2, 0}
1/8	**2/16**	{2, -10, 122, 18, -4, 0}
-	**3/16**	{2, -12, 116, 28, -8, 2}
2/8	**4/16**	{2, -14, 110, 38, -10, 2}
-	**5/16**	{2, -14, 102, 48, -12, 2}
3/8	**6/16**	{2, -16, 94, 58, -12, 2}
-	**7/16**	{2, -14, 84, 66, -12, 2}
4/8	**8/16**	{2, -14, 76, 76, -14, 2}
-	**9/16**	{2, -12, 66, 84, -14, 2}
5/8	**10/16**	{2, -12, 58, 94, -16, 2}
-	**11/16**	{2, -12, 48, 102, -14, 2}
6/8	**12/16**	{2, -10, 38, 110, -14, 2}
-	**13/16**	{2, -8, 28, 116, -12, 2}
7/8	**14/16**	{0, -4, 18, 122, -10, 2}
-	**15/16**	{0, -2, 8, 126, -6, 2}

resulting in new partitions of recursive mode. In addition, it supports irregular subdivisions of blocks with 2 squares a rectangle in 4 combinations, totaling 10-ways partition tree. The lowest block level for luma prediction is 4×4, while for chroma it is able to reach level 2×2 in some cases [5], [17]. In the AV1 encoder's inter-frame prediction step the motion estimation (ME) and motion compensation (MC) are situated. ME has the purpose of removing the temporal redundancies existing between near frames by means of previously processed frames, called reference frames. Generally, the ME can be decomposed into two steps: integer motion estimation (IME) and fractional motion estimation (FME). In the IME, integer pixels positions of the current and reference frames are compared. That comparison is done between the block in the current frame to be encoded and the blocks in the reference frame that surrounds the same position of the current block with the help of a block matching algorithm. The best match occurs for the blocks with the least difference.

In the FME, the process is performed using fractional samples generated around the integer samples. The sub-samples are obtained from integer samples of the reference frame through interpolation, performed by Finite Impulse Response (FIR) filters defined by AV1. The samples from the area with the smallest difference from the reference block are subtracted from the current block, resulting in Motion Vectors (MV), used to construct a next frame at MC.

The AV1 encoder defines 90 interpolation filters of the FIR or Bilinear type, with up to 8-taps applied on samples of luma and chroma, with accuracy of $1/8$ and $1/16$ respectively. This accuracy leads to increased compression efficiency, but significantly increases the necessary computational effort.

The filters are divided into 4 families: *i)* Regular: 6-taps interpolation filters based on Lagrange; *ii)* Smooth: 6 tap smoothing filters designed with Hamming window; *iii)* Sharp: 8-taps interpolation filters based on discrete cosine transform; *iv)* Bi-linear: coding or fast decoding filters. In addition, AV1 standard defines approximations for the Regular and Smooth filter families for 4×4 or lower-sized blocks [15], [17].

Regular family uses 15 different 6-tap FIR filters for fractional pixel interpolations. The coefficients of these 15 FIR filters are shown in Table I. Note that, AV1 standard uses the same filters for luminance and chrominance samples, differing only in accuracy. In addition, the filters are symmetrical: the coefficients used for the calculation of $1/16$ chrominance samples are similar to those used for the accuracy of $15/16$ and $13/16$. Similarly, the coefficients used for the calculation of luminance samples $1/8$ and $2/8$ are similar to those used for accuracy $7/8$ and $6/8$, respectively.

Fig. 1 shows the fractional samples around the integer luminance samples. The figure shows gray squares that correspond to integer samples, and 63 fractional samples pixels for one integer pixel (14 half and 49 quarter) generated from the interpolation filters. An example of filter equation corresponding to Filter 2 (F2) is shown in (1). The other equations of the filters can be found in [17].

$$F2 = 2a_0 - 10a_1 + 122a_2 + 18a_3 - 4a_4 + 64 >> 7 \quad (1)$$

III. Analysis on AV1 Filter Families Usage

In order to evaluate the AV1 interpolation filter families usage, an analysis was performed using information obtained from the AV1 reference software [17]. The graphs in Fig. 2 and 3 illustrate the usage of AV1 horizontal and vertical interpolation filter families, respectively, when encoding video sequences of different spatial resolutions and quantization parameters (QP) ranges. In Fig. 2, that shows horizontal interpolation results, when considering 1920×1080 resolution videos, the Regular filter family is the most requested com-

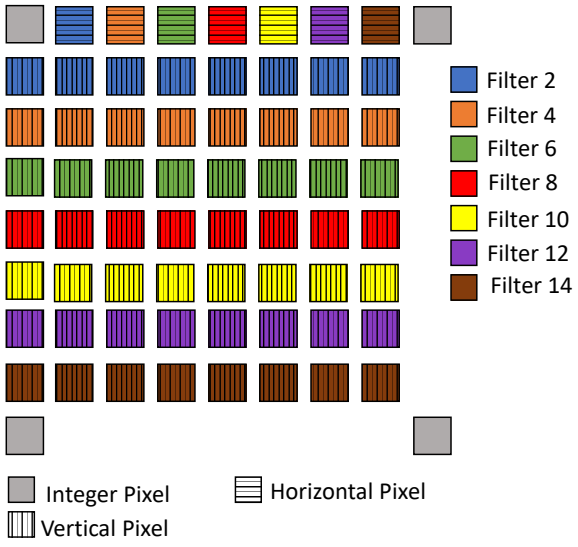

Fig. 1. Interpolation process of luminance samples in AV1 standard. (Source: adapted from [18])

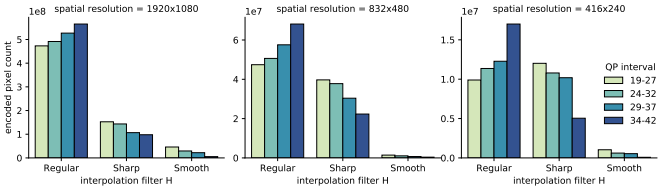

Fig. 2. Evaluation of the usage of AV1 horizontal interpolation filter.

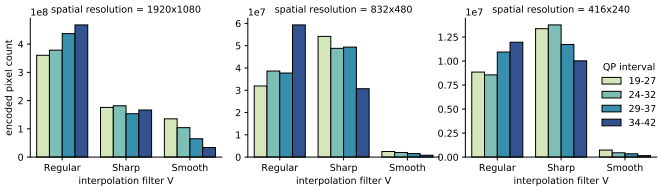

Fig. 3. Evaluation of the use AV1 vertical interpolation filter.

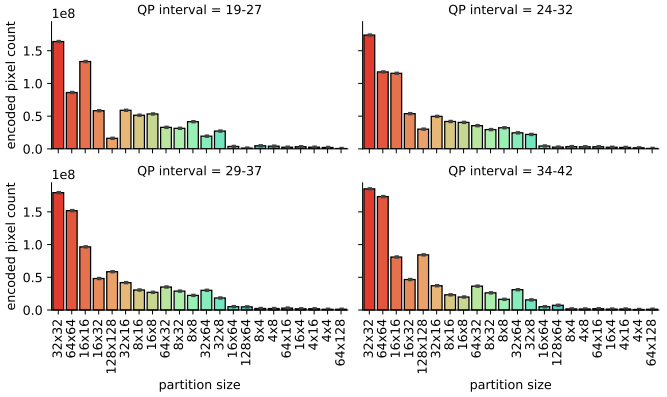

Fig. 4. Evaluation of the most requested block sizes for different QPs.

pared to Sharp and Smooth filters. This is verified for all observed QP ranges.

Considering a 832×480 resolution, the Regular filter is still the most used for all QPs, but the Sharp filter is employed more often compared to the previous test. Given the spatial resolution 416×240, Regular and Sharp show similar performance. Considering the lower QP values (range 19-27) the Sharp filter is more requested. For all others, Regular filter is more employed, especially for the 34-42 range. For the three resolutions tested, regarding horizontal interpolation, the Smooth filter has not shown significant contribution.

The usage of vertical interpolation filters for three spatial resolutions presented in Fig. 3 was also evaluated. For 1920×1080 resolution videos, the Regular filter was the most used for all QP ranges. In such case, Sharp and Smooth filters were responsible for lower interpolation plots, especially for the 19-27 and 24-32 QP ranges. Given 832×480 resolution, the Regular filter was the most used only for QP values ranging from 34-42. For other QPs, the Sharp filter was the most requested one. The Smooth filter showed no significant contribution. For tests performed with 416×240 resolution, the Regular filter was most used only for QPs in the 34-42 range. For the other QPs ranges, the Sharp filter was most requested. The Smooth filter showed no significant contribution.

AV1 supports 22 block sizes and 10 types of partitioning. These blocks are evaluated in the encoder process to define which is best. Fig. 4 shows the evaluation to observe the most relevant block sizes. The evaluation is done for different QP ranges. Considering QPs in the 19-27 range, pixels are most often encoded with square blocks of sizes 32×32, 16×16 and 64×64. For QPs in the range 24-32, the most frequently used block size is also 32×32, followed by the sizes 64×64 and 16×16, required with the same frequency. For the largest observed QPs (34-42), the most selected block size is 32×32, followed by 64×64, 128×128 (Superblock) and 16×16.

Therefore, based on the evaluations presented, it is stated that during the interpolation process of AV1, for all tested QP values, most pixels are coded by larger regular blocks. Also, according to the tests, the Regular filter family is responsible for most of the interpolation performed in the encoder.

IV. Designed Architecture

The architecture of the regular interpolation filter proposed in this work is presented in Fig. 5. The architecture supports all 15 interpolation filter types defined by the AV1 encoder for the Regular family, considering luminance and chrominance samples. In the diagram, the filters responsible for generating interpolated luminance samples are indicated in blue.

The proposed architecture has a configurable input that receives a line formed by $N+6$ pixel samples of 8-bit width, where N refers to the supported parallelism levels (4, 8, 16, 32, 64 or 128 integer pixels). The 6 complementary samples represent the 3-sample padding zones required on each end. The expressions necessary to describe the 6-tap FIR filters in transposed mode were obtained using the HCub MCM algorithm, reducing the size and number of additions required. Thus, each integer pixel is a multiple common operand of the set of constant coefficients. The output signals of the 15 filters were obtained using combinations of sum trees and shifts.

Considering, for example, the process of interpolation of luma samples, the architecture delivers each of the 7 horizontal filter samples at the output in a first data pass. Then, the outputs are used in a second pass to calculate the vertical samples. For this reason, the filters were designed with 10-bits depth input $(10(N+1))$ to support the second pass. Finally, the output interpolated is the number of fractional samples required to a $N \times N$ block size, which in the proposed solution, can have outputs with 10, 11 and 12-bits depth.

For a given level of parallelism (N), considering that the architecture processes one row per cycle, the number of cycles required to process an $N \times N$ size block is given by (2).

$$C_N = H + V + D = (3+N+3) + (N) + (7*(N+1)) \quad (2)$$

In (2), H represents the horizontal lines $(3+N+3)$, V the vertical lines (N) and D the diagonal lines, $7*(N+1)$ for luminance samples.

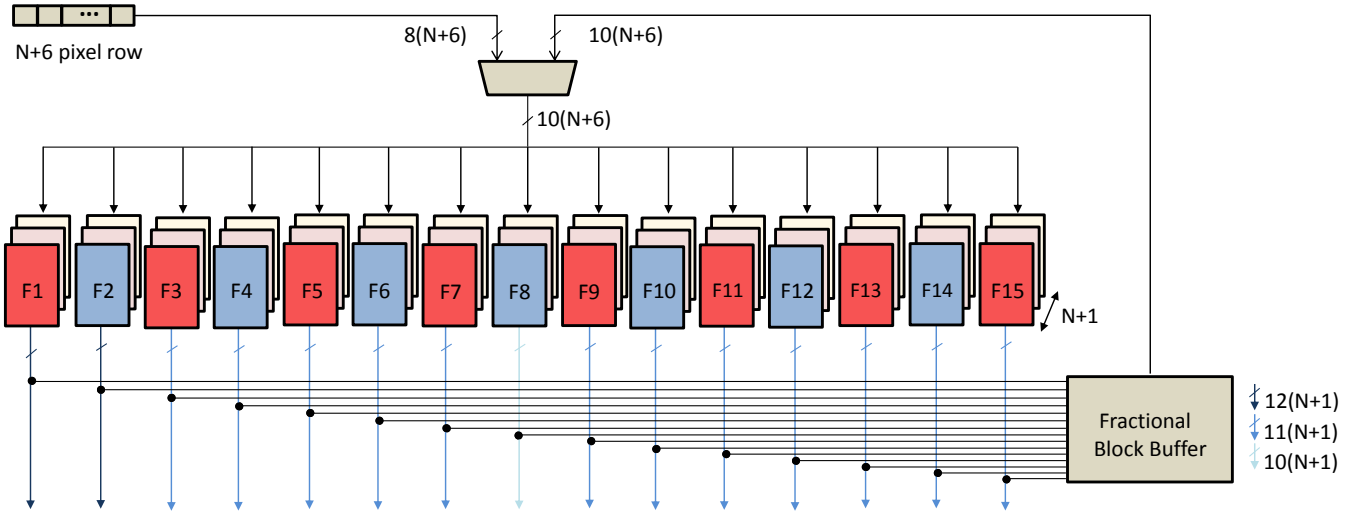

Fig. 5. Diagram of the proposed AV1 Regular interpolation filter architecture for an input size N integer samples.

TABLE II
REQUIRED FREQUENCY (MHZ) FOR DIFFERENT RESOLUTIONS AND LEVELS OF PARALLELISM (N= 4, 8, AND 16)

C_N / N	49 / 4	85 / 8	157 / 16
1920×1080@30	190.51	82.62	38.151
1920×1080@50	317.52	137.70	63.59
2560×1600@30	376.32	163.20	75.36
2560×1600@50	627.20	272.00	125.60
3840×160@30	762.05	330.48	152.61
3840×2160@50	1270.08	550.80	254.34
3840×2160@120	3048.19	1321.92	610.41

TABLE III
REQUIRED FREQUENCY (MHZ) FOR DIFFERENT RESOLUTIONS AND LEVELS OF PARALLELISM (N=32, 64, AND 128)

C_N / N	301 / 32	589 / 64	1165 / 128
1920x1080@30	18.29	8.95	4.42
1920x1080@50	30.48	14.91	7.37
2560x1600@30	36.12	17.67	8.74
2560x1600@50	60.20	29.45	14.56
3840×2160@30	73.14	35.78	17.69
3840×2160@50	121.91	59.64	29.49
3840×2160@120	292.57	143.13	70.77

Tables II and III show the number of cycles (C_N) required for processing the 6 different sizes of N integer luma samples, and the frequency (in MHz) required for interpolate videos of different spatial resolutions in real-time. Table II, given the 3 lowest parallelism levels supported (N = 4, 8, and 16), the minimum frequency required to process samples at high resolutions is extremely high. In the best case, for N=16 the minimum frequency is 610.41 MHz to interpolate 3840×2160 resolution videos at 120 fps. In the worst case (N=4), the minimum frequency for the same resolution should be approximately 5× higher. In Table III, considering the 3 highest levels of parallelism (N = 32, 64, and 128), the minimum frequency required remains at low levels. In the best case, for the processing of a superblock in the highest spatial resolution, the frequency should have the minimum value of 70.77 MHz. In the worst case (N=32), the processing of samples with the same resolution requires a minimum frequency of 2× less compared to the best case in the Table II, so larger a N should be used to encode higher resolutions.

V. RESULTS AND DISCUSSION

The proposed architectures were described in VHDL and synthesized with the Cadence Genus Synthesis Solution tool using the STMicro 65 nm standard cells at 1.35 VDD, a 100 ps slack, and worst-case conditions. The synthesis results obtained with the maximum frequency supported for each value of N are presented in the Table IV. The gate count (Kgates) was calculated based on 2-input NAND gates. Note that, as shown in Tables II and III, lower N values require higher operating frequencies to process low resolution samples, but with less power and area requirements.

Considering the most requested block size (32×32) according to the evaluation in Section III, the designed architecture is capable of processing the MC of 3840×2160 sequences in real time (120 fps). It is also possible to see that input sizes greater than 32 are not necessary for the designed architecture, unless even higher resolutions and frame rates are pursued. Table IV shows also the throughput of our architecture (in thousand samples per second).

Table V shows a comparison with related work. Since there are not many AV1 implementations published in the literature, HEVC solutions were also included. This also allows us to compare how the increased complexity of AV1 affects hardware for this codec compared to HEVC.

From Table V we can see that our solution consumes more power and area resources compared with a similar AV1

TABLE IV
SYNTHESIS RESULTS FOR THE REGULAR FILTERS AT THE MAXIMUM FREQUENCY (100PS SLACK)

N	Freq. (MHz)	Gates (k)	Power (mW)	Supported MC resol.	Through. (smp/s)
4	476	71.0	37.96	2560x1600 @30	16.7k
8	448	106.2	56.4	3840×2160 @30	28.3k
16	448	203.3	105.6	3840×2160 @50	53.4k
32	345	270.4	130.8	3840×2160 @120	79.7k
64	304	459.2	203.7	3840×2160 @120	138k
128	282	828.7	356.0	3840×2160 @120	254k

TABLE V
COMPARISON WITH RELATED WORKS.

Related works	[13]	[12]	[15]	Ours N=8	Ours N=32
Standard	HEVC	HEVC	AV1	AV1	AV1
Tech. (nm)	90	45	40	65	65
Freq. (MHz)	300	116.64	256.6	448.43	344.83
Gate Count (K)	12.8	87.9	40.70	106.17	270.44
Power (mW)	15.8	15.06	16.12	56.37	130.75
Supported MC resol.	1080p @49fps	2160p @60fps	2160p @30fps	2160p @30fps	2160p @120fps
Throughput (smp/s)	-	-	1.0k	28.3k	79.7k

implementation [15]. First, this is likely due to the fact that our technology is dated compared to a 40nm node. Second, area is also increased because our architecture was designed targeting a higher throughput to support the processing requirements of FME accelerators, whereas the related work focuses only on the MC throughput requirements [15]. This way, our architecture is capable of achieving higher throughput compared to other solutions. The same table shows that HEVC solutions are less power-consuming than all the AV1 implementations (similar power results, but older technologies in both HEVC solutions), showing that efficient interpolation architectures are even more important for this new codec.

VI. CONCLUSION

This paper presents a dedicated hardware architecture for the interpolation filter of AV1 standard, considering different levels of parallelism. The architecture support the 15 filters defined in the Regular family, implemented using the Hcub MCM algorithm. For the lowest level of parallelism architecture is able to process luma samples at 2560×1600@30 37.96mW with energy dissipation, while higher levels of parallelism, it is possible to process video 3840×2160@120 dissipating 130.75mW in the best case (N=32). Also, an evaluation of the interpolation of AV1 was presented, where according to analysis of the reference software, we can say that the filters of the Regular family are responsible for most of the interpolation process, and blocks of larger sizes, especially 32×32 are the most requested, considering different spatial resolutions and QPs.

ACKNOWLEDGEMENT

This work was supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - and also by FAPERGS and CNPq Brazilian research support agencies.

REFERENCES

[1] C. V. N. Index, "Forecast and trends, 2017–2022," *Cisco Systems*, pp. 1–7, 2018.
[2] V. Sze, M. Budagavi, and G. J. Sullivan, "High efficiency video coding (HEVC)," in *Integrated circuit and systems, algorithms and architectures*. Springer, 2014, vol. 39, p. 40.
[3] P. Akyazi and T. Ebrahimi, "Comparison of compression efficiency between hevc/h. 265, vp9 and av1 based on subjective quality assessments," in *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*, 2018, pp. 1–6.
[4] D. Grois, T. Nguyen, and D. Marpe, "Coding efficiency comparison of AV1/VP9, H.265/MPEG-HEVC, and H.264/MPEG-AVC encoders," in *2016 Picture Coding Symposium (PCS)*. IEEE, 2016, pp. 1–5.
[5] Y. Chen, D. Murherjee, J. Han, A. Grange, Y. Xu, Z. Liu, S. Parker, C. Chen, H. Su, U. Joshi *et al.*, "An overview of core coding tools in the av1 video codec," in *2018 Picture Coding Symposium (PCS)*. IEEE, 2018, pp. 41–45.
[6] Mozilla and Xiph.Org, "Next generation video: Introducing av1," https://people.xiph.org/ xiphmont/demo/av1/demo1.shtml, 2018.
[7] G. Bjøntegaard, T. Davies, A. Fuldseth, and S. Midtskogen, "The thor video codec," in *2016 Data Compression Conference (DCC)*, March 2016, pp. 476–485.
[8] AOMedia, "Av1," https://aomedia.org/av1-features/, 2018.
[9] C. Feldmann, "Best Video Codec: An Evaluation of AV1, AVC, HEVC and VP9," https://bitmovin.com/av1-multi-codec-dash-dataset/, 2018.
[10] B. Bing, *Next-generation video coding and streaming*. John Wiley & Sons, 2015.
[11] I. Chakrabarti, K. N. S. Batta, and S. K. Chatterjee, *Motion Estimation for Video Coding*. Springer, 2015.
[12] W. Penny, M. Ucker, I. Machado, L. Agostini, D. Palomino, M. Porto, and B. Zatt, "Power-efficient and memory-aware approximate hardware design for HEVC fme interpolator," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2018, pp. 237–240.
[13] E. Kalali and I. Hamzaoglu, "Approximate HEVC fractional interpolation filters and their hardware implementations," *IEEE Transactions on Consumer Electronics*, vol. 64, no. 3, pp. 285–291, 2018.
[14] G. Paim, W. Penny, J. Goebel, V. Afonso, A. Susin, M. Porto, B. Zatt, and L. Agostini, "An efficient sub-sample interpolator hardware for VP9-10 standards," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 2167–2171.
[15] R. Domanski, J. Goebel, W. Penny, M. Porto, D. Palomino, B. Zatt, and L. Agostini, "High-throughput multifilter interpolation architecture for AV1 motion compensation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 5, pp. 883–887, May 2019.
[16] Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," *ACM Transactions on Algorithms*, vol. 3, no. 2, p. 11, 2007.
[17] P. de Rivaz and J. Haughton, "AV1 bitstream & decoding process specification. version 1.0.0 with errata 1," *The Alliance for Open Media*, p. 681, 2018.
[18] A. CanMert, E. Kalali, and I. Hamzaoglu, "A low power versatile video coding (VVC) fractional interpolation hardware," in *2018 Conference on Design and Architectures for Signal and Image Processing (DASIP)*. IEEE, 2018, pp. 43–47.