Area and Energy Evaluation of an FME Hardware Architecture for HEVC and VVC Encoders

Nicole Citadin, Vanio Rodrigues Filho*, Ismael Seidel*, Marcio Monteiro*, Mateus Grellert*†, José Luis Güntzel*
Embedded Computing Lab. (ECL), *Computer Science Graduate Program (PPGCC),
Dept. of Informatics and Statistics (INE), Federal University of Santa Catarina (UFSC), Florianópolis, Brazil
†Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
nicole.citadin@grad.ufsc.br, {vanio.rodrigues, marcio.monteiro}@posgrad.ufsc.br,
{ismael.seidel, j.guntzel}@ufsc.br, mateus.grellert@inf.ufrgs.br

***Abstract*—Fractional Motion Estimation (FME) is used in most video coding standards to improve coding efficiency, but at the cost of high computational complexity, demanding dedicated hardware accelerators. In the High Efficiency Video Coding (HEVC) standard a 1/4-precision is adopted to signalize motion, while in Versatile Video Coding (VVC) the motion precision is adaptive, and a new alternative filter was introduced to interpolate samples when motion has 1/2-precision. This paper evaluates the impact of the alternative filter in the resulting area and power of a complete FME architecture that works for both HEVC and VVC standards. The synthesis results show an increase of 38% and 20% in area and power, respectively, in the filter module. Nevertheless, as this filter module is not predominant in the architecture, the overall area and power increase are only 8% and 4%, respectively. Even with the power increase, the FME architecture demands 35% less energy when operating using 1/2-precision in VVC.**

***Index Terms*—Video coding. Fractional Motion Estimation. Alternative Filter. Versatile Video Coding. VLSI Architecture.**

I. Introduction

The advancements in internet capacity and speed, along with the necessity imposed by modern daily activities, resulted in substantial growth in digital video consumption. Accentuated by the COVID-19 pandemic, video conferencing and streaming have become important to most people's routines. According to Ericsson [1], video is expected to represent 80% of the global mobile network traffic in 2028. This scenario motivates the development of new video coding standards, such as the Versatile Video Coding (VVC) [2], which achieves savings above 40% in bitrate [3] compared with its predecessor, the High Efficiency Video Coding (HEVC) [4]. However, because VVC packs a new series of tools, such savings come with a significant burden in encoding time. For instance, comparing the VVC Test Model (VTM) [5] and the HEVC Model (HM) [6], the former takes around 10× longer than the latter to encode a video sequence [7]. Therefore, hardware accelerators are often used to accomplish real-time and energy-efficient encoders, especially tools as demanding as the Fractional Motion Estimation (FME).

In HM, for instance, inter prediction represents more than 60% of the HM encoding time, from which 43% is used for the FME [8]. In the VTM, the FME algorithm is almost the same as in HM. The main difference in the algorithm is the inclusion of an alternative filter [9] that may be used depending on the Adaptive Motion Vector Resolution (AMVR) [10], [11], which is also a new VVC tool. Therefore, the high parallelism achieved by using a hardware accelerator for FME may greatly benefit a VVC encoder.

To our best knowledge, this is the first paper to evaluate area, power and energy efficiency of a complete FME hardware that targets the VVC, and therefore includes the new VVC alternative filter while keeping compatibility with HEVC. Moreover, we analyze the overheads caused by the inclusion of the alternative filter compared to a baseline state-of-the-art FME architecture without this filter. Also, our analysis includes a detailed view of each module in the architecture hierarchy, providing a unique view on the new filter impacts.

The remaining of this paper is organized as follows. Section II presents an overview of the HEVC and VVC FME. Section III discusses the baseline FME architecture and the modifications made to support VVC. The results obtained with a standard cell synthesis flow are presented and discussed in Section IV. Finally, Section V draws the conclusion.

II. Fractional Motion Estimation

Video compression is essential to enable activities such as streaming, recording, and watching videos on mobile devices. Also, in mobile devices, the energy efficiency of the video codec is of utmost importance to avoid exhausting the devices' battery. Thus, every encoding tool must be properly balanced in terms of coding and energy efficiency. In this work, we focus on the inter-frame prediction tool, which can be divided into three main steps [12]: 1) Motion Vector (MV) prediction, 2) Integer Motion Estimation (IME), and 3) Fractional Motion Estimation (FME).

The MV prediction defines the initial search point for the IME [13]. IME, in turn, seeks for a block to use as a reference ($\mathbf{B}^{\text{ref}}$) to predict the original block ($\mathbf{B}^{\text{ori}}$). Moreover, the Block

This work was supported in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and by Brazilian Council for Scientific and Technological Development (CNPq) through Edital Universal 423558/2021-8, PQ grant 316984/2021-3, and IC grant 168304/2022-8.

Matching Algorithm (BMA) [14] is used to find a $\mathbf{B}^{\text{ref}}$ from a set S of candidate blocks ($\mathbf{B}^{\text{can}}$), usually relying on the Lagrangian Rate-Distortion (RD) cost [15] (J-cost) to perform a Rate-Distortion Optimization (RDO). The IME results in a residual MV ($\overrightarrow{mv}^{\text{ime}}$), which is relative to the predicted MV pointing to the reference block ($\mathbf{B}^{\text{ref-ime}}$) that was selected.

Most modern video coding standards support MVs with fractional precision, obtained by the FME. The FME can be divided into two main steps: interpolation and search. The interpolation has as its objective to create samples in fractional positions around the $\overrightarrow{mv}^{\text{ime}}$, while the search applies the BMA similar to the IME, but using the interpolated samples to create candidate blocks. The interpolation of new samples is computed by standard-defined FIR filters. There are 48 possible candidate blocks in 1/4-precision MVs positions that surround a given integer MV position: 6 candidates from horizontal only displacement, formed with Horizontal Samples (HSs); 6 from vertical only displacement, from First-Order Vertical Samples (FOVSs); and 36 with both vertical and horizontal displacement, from Second-Order Vertical Samples (SOVSs). Fig. 1 presents the interpolation directions to obtain source values for the FIR filters in HEVC and VVC.

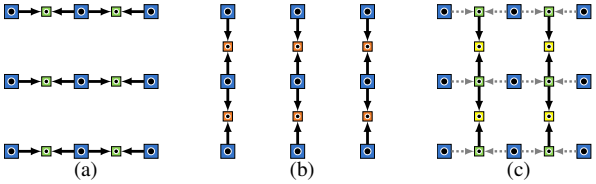


Fig. 1. (a) Horizontal Sample (HS); (b) First-Order Vertical Sample (FOVS); and (c) Second-Order Vertical Sample (SOVS). Adapted from [16].

A. HEVC and VVC Interpolation Filters

Usually, each interpolated position requires a specific filter depending on the MV displacement. Thus, the number of filters a standard defines depends on its adopted MV precision.

HEVC adopts 1/4-precision MVs, called quarter-pel (QPEL), for luma samples. Table I presents the coefficients defined in HEVC [17] for luma interpolation. In this table, i represents the nearest integer sample position with respect to the sample being interpolated. To interpolate samples with 1/4 (called up) or 3/4 (down) displacements relative to the integer MV, the q_i coefficients are adopted. When the displacement is 1/2 (middle, i.e. 2/4 displacement considering the 1/4-precision MV), the h_i coefficients are used.

TABLE I
FME INTERPOLATION COEFFICIENTS (Y-CHANNEL) [2].

Standard	i	-3	-2	-1	0	1	2	3	4
HEVC&VVC	q_i	-1	4	-10	58	17	-5	1	
HEVC&VVC	h_i	-1	4	-11	40	40	-11	4	1
VVC only	h_i^{alt}	0	3	9	20	20	9	3	0

By its turn, VVC introduced the Adaptive Motion Vector Resolution (AMVR) [10], [11], which allows the representation of variable precision MVs. With the AMVR, MVs in VVC may be encoded with 1/4-precision (QPEL), 1/2-precision, i.e. half-pel (HPEL), integer-precision, i.e. full-pel (FPEL) and in steps of four samples, called 4-pel (4-PEL). Thus, using AMVR allows the encoder to decide on the fly what is the best MV precision, as having higher resolutions for the MV may provide better prediction accuracy but also requires more bits to encode [18]. Therefore, due to the AMVR, the FME in VVC has two possible operating modes: QPEL and HPEL. For QPEL, the FME of VVC is the same as the FME of HEVC, and thus the same filters are used. On the other hand, in the case of HPEL, the VVC adopts an alternative 6-tap Gaussian filter instead of the 8-tap filter used in HEVC for 1/2 displacements. Table I shows the new VVC alternative filter coefficients, h_i^{alt}. When operating in HPEL, the FME will only use the alternative filter, and no interpolations are made in 1/4 and 3/4 positions as there is no way to represent the MVs of these positions.

III. HARDWARE ARCHITECTURE

Considering that for QPEL precision the FME of HEVC and VVC are virtually the same, we started from a baseline FME architecture for HEVC [13] and modified it to support the alternative VVC filter. The baseline FME architecture [13] was designed for 8×8 block size and interpolates and searches all 48 candidates. The architecture from [13] follows the design strategy from [19], which aims at both coding and energy efficiency. One key to energy efficiency is to adopt a regular dataflow that results in a high degree of parallelism and to perform no redundant filter operations [19]. Thus, considering the filter coefficients from Table I, the vector function in (1) can be used to represent all coefficients applied to a given integer sample (x) without redundancies, including the ones from the alternative VVC filter.

$$\overrightarrow{ss}(x) = [x; 3x; 4x; 9x; 10x; 5x; 11x; 20x; 40x; 58x; 17x] \quad (1)$$

Including the alternative VVC filter coefficients in (1) is the first step to adapt the baseline architecture so it can operate for HPEL precision as defined by the VVC standard. Moreover, as the coefficients are known beforehand, Multiplierless Multiple Constant Multiplication (MMCM) can be used to compute the multiplications in (1) with a series of sums and shifts (SS), as represented in Fig. 2a, that emphasizes the alternative filter paths. A group of SS modules (Fig. 2a), after passing through the proper routing and sums, provides the needed outputs of all filters, thus supporting the FME of both HEVC and VVC.

For the HEVC or the VVC with QPEL precision FME, the results from the multiplications obtained by the SS modules (Fig. 2a) are used as defined in (2) to (4) for computing the samples of up, middle, and down positions, respectively.

$$\begin{aligned}\overrightarrow{u}_n = &\big(\overrightarrow{ss}(\overrightarrow{x}_{n-3})_0 + \overrightarrow{ss}(\overrightarrow{x}_{n-2})_1 + \overrightarrow{ss}(\overrightarrow{x}_{n-1})_2 + \overrightarrow{ss}(\overrightarrow{x}_n)_6 \\ &+\overrightarrow{ss}(\overrightarrow{x}_{n+1})_7 + \overrightarrow{ss}(\overrightarrow{x}_{n+2})_3 + \overrightarrow{x}_{n+3}\big) \gg 6 \quad (2)\\ \overrightarrow{m}_n = &\big(\overrightarrow{ss}(\overrightarrow{x}_{n-3})_0 + \overrightarrow{ss}(\overrightarrow{x}_{n-2})_1 \\ &+\overrightarrow{ss}(\overrightarrow{x}_{n-1})_4 + \overrightarrow{ss}(\overrightarrow{x}_n)_5 + \overrightarrow{ss}(\overrightarrow{x}_{n+1})_5 + \overrightarrow{ss}(\overrightarrow{x}_{n+2})_4 \\ &+\overrightarrow{ss}(\overrightarrow{x}_{n+3})_1 + \overrightarrow{ss}(\overrightarrow{x}_{n+4})_0\big) \gg 6 \quad (3)\\ \overrightarrow{d}_n = &\big(\overrightarrow{x}_{n-2} + \overrightarrow{ss}(\overrightarrow{x}_{n-1})_3 + \overrightarrow{ss}(\overrightarrow{x}_n)_7 + \overrightarrow{ss}(\overrightarrow{x}_{n+1})_6 \\ &+\overrightarrow{ss}(\overrightarrow{x}_{n+2})_2 + \overrightarrow{ss}(\overrightarrow{x}_{n+3})_1 + \overrightarrow{ss}(\overrightarrow{x}_{n+4})_0\big) \gg 6 \quad (4)\end{aligned}$$

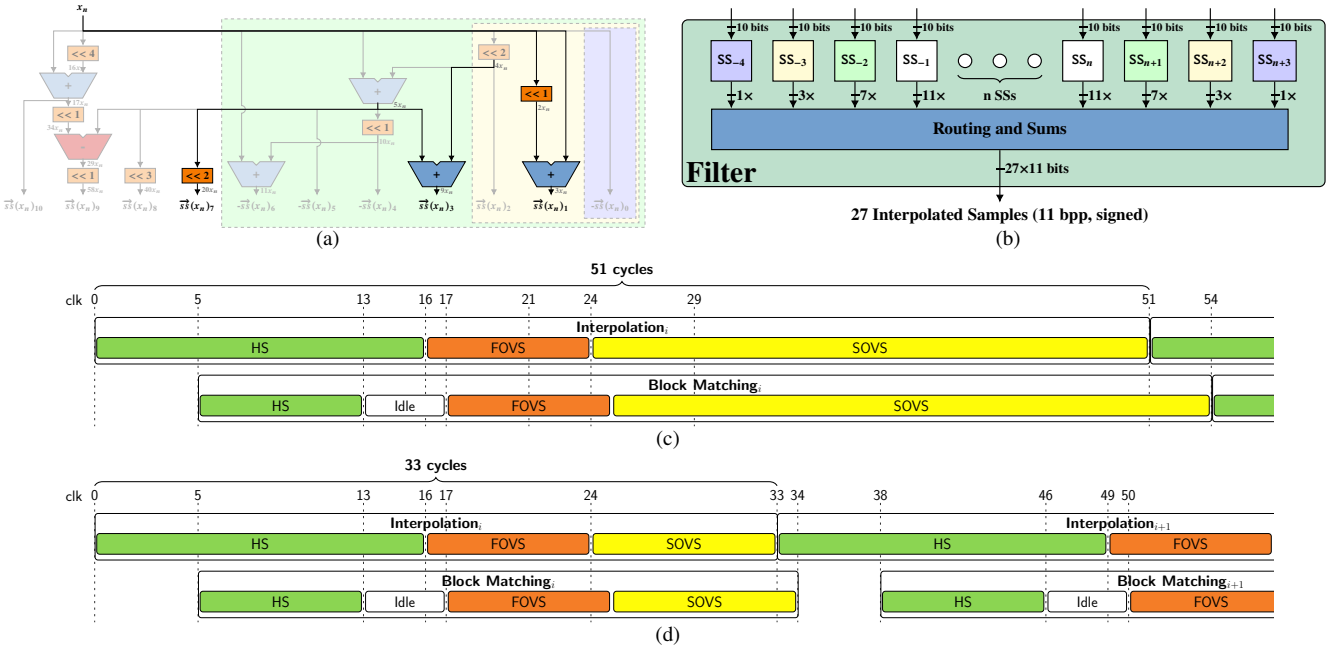


Fig. 2. (a) Multiplierless HEVC and VVC interpolation filter datapath, named as Sums and Shifts (SS), highlighting the paths exclusive to the alternative VVC filter. (b) Complete interpolation filter with SS modules. (a-b) Adapted from [19]. (c) Timing diagram for both FME architectures when operating with 1/4 MV precision taking 51 cycles/FME. (d) Timing diagram for the proposed FME architecture when operating with 1/2 MV precision taking 33 cycles/FME.

When operating with HPEL precision, only the alternative filter multiplications are obtained and added as shown in (5).

$$\begin{aligned}\overrightarrow{a}_n = &(\overrightarrow{ss}(\overrightarrow{x}_{n-2})_1 + \overrightarrow{ss}(\overrightarrow{x}_{n-1})_3 + \overrightarrow{ss}(\overrightarrow{x}_n)_7 \\ &+\overrightarrow{ss}(\overrightarrow{x}_{n+1})_7 + \overrightarrow{ss}(\overrightarrow{x}_{n+2})_3 + \overrightarrow{ss}(\overrightarrow{x}_{n+3})_1) \gg 6 \end{aligned} \quad (5)$$

Knowing that when operating with HPEL precision there are only 8 instead of 48 FME candidates (QPEL precision), the number of cycles for a HPEL FME may be reduced from 51 to 33 cycles/FME. The reduction in the number of cycles is not proportional to the reduction in the number of candidates due to a lower level of parallelism of the architecture when operating with HPEL, which could not be overcome due to the data dependency between the HSs and the SOVSs (Fig. 1d).

Apart from the filters and the control, no other modification in the architecture is required to support the alternative VVC filters. The baseline architecture contains three Transpose Buffers (TBs), namely the Horizontal Pel TB, the Integer Pel TB, and the Original TB. The latter TB is used by the Block Matching (BM) module to hold the $\mathbf{B}^{\text{ori}}$, used as input to the J Tree, which in turn computes the J-cost between the $\mathbf{B}^{\text{ori}}$ and each $\mathbf{B}^{\text{can}}$ generated by the filter (Fig. 2b), with a rate estimate provided by a dedicated LambdaRate module [16].

IV. Synthesis Results

Both the baseline and the proposed FME architecture with the alternative filter were described using Verilog and synthesized using the Synopsys® Design Compiler (DC®) [20] in Topographical mode, to obtain realistic estimates of area and power (considering the default switching activity from DC®). We used a 45 nm standard cell library from TSMC [21]. We determined five target throughputs and their respective periods to constrain the synthesis. For both architectures, the stricter period (1.25 ns) resulted in timing violations, so these results are not reported. Table II presents the area, power, and energy estimates for all eight successful synthesis cases.

When comparing the results for different periods in the same architecture, it is possible to notice a larger leap in the area of 2.5 ns syntheses, meaning a larger number of critical paths. Also, higher frequencies result in reduced energy consumption due to saving more time than the increase in dynamic power. The energy estimates also show how small the alternative filter's impact is, as there is only a small difference between the energy consumed by the baseline compared with the proposed architecture.

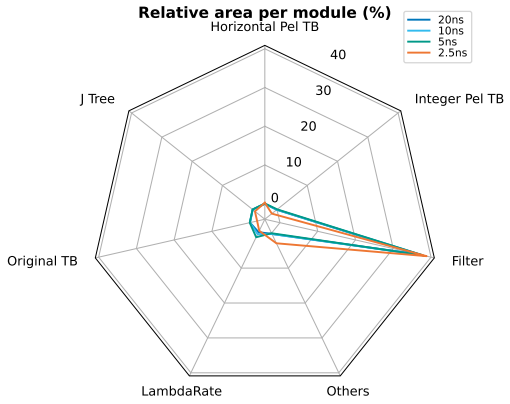
In Table II, one may also observe that the area overhead of including the alternative filter remains near 7%, increasing to about 8% for the lowest period. Such overhead is caused almost solely by the interpolation filters, as this is the only part of the architecture we changed. Fig. 3a highlights this by breaking down the overhead estimates of each major module in the FME architecture's hierarchy and shows that adopting the alternative filter increased the area of the filters by 37.78%, on average ($\sigma = 0.84\%$). The other modules had almost no change in terms of area ($\mu = -0.02\%, \sigma = 0.88\%$).

On the other hand, one may notice in Table II that the power overhead is about 4% for 20ns, getting less significant as the period is reduced, reaching 0.15% for 2.5ns. Although this behavior may seem unexpected at first glance, it occurs because the dynamic power plays a major role in the total power for higher frequencies, and the dynamic power estimates depend on the activity of the architecture. Given that, when operating in HPEL precision a large portion of the architecture stays idle, thus reducing the overall dynamic power increase. Fig. 3b makes this evident by showing there is even a decrease in the total power for the J Tree module of the proposed architecture, which helps to alleviate the filter overhead.

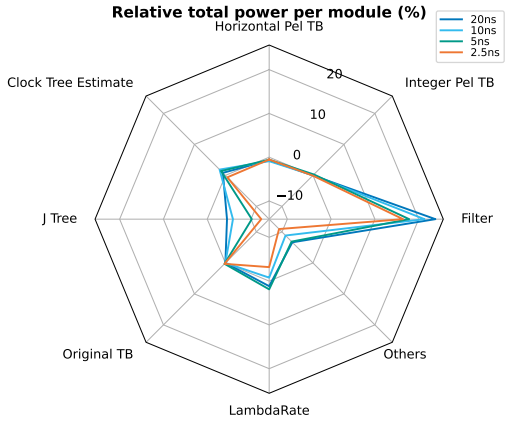
TABLE II
SYNTHESIS RESULTS FOR THE BASELINE AND THE PROPOSED ARCHITECTURES.

Architecture*	Baseline	Proposed	Relat. (%)	Baseline	Proposed	Relat. (%)	Baseline	Proposed	Relat. (%)	Baseline	Proposed	Relat. (%)
Target throughput	1080p@30fps			1080p@60fps			2160p@30fps			2160p@60fps		
Period (ns)	20			10			5			2.5		
Area (μm^2)	96453.23	103087.63	6.88	95718.87	102300.18	6.88	95425.52	102143.49	7.04	105532.89	113962.52	7.99
Dyn. power (μW)	2181.40	2203.30	1.00	4314.30	4350.80	0.85	8693.40	8767.20	0.85	17858.10	17755.60	-0.57
Static power (μW)	953.93	1053.70	10.46	958.59	1055.40	10.10	956.95	1061.20	10.89	1136.10	1266.20	11.45
Total power (μW)	3135.33	3257.00	3.88	5272.89	5406.20	2.53	9650.35	9828.40	1.85	18994.20	19021.80	0.15
Energy (nJ/FME)**	3.20	3.32/2.15	3.9/-32.8	2.69	2.76/1.78	2.5/-33.8	2.46	2.51/1.62	1.8/-34.2	2.42	2.43/1.57	0.1/-35.2

*Relative: $((\text{Proposed}/\text{Baseline}) - 1) \times 100\%$. **The proposed architecture has two energy estimates, considering QPEL and HPEL precision, respectively.



(a)



(b)

Fig. 3. (a) Relative area per module. (b) Relative power per module.

V. CONCLUSIONS

The goal of this paper was to evaluate the area, power, and energy overheads of supporting the alternative filter for VVC compatibility in an FME hardware architecture. Although the filter area increased by about 40%, considering the overall FME architecture area, the overhead was at most 8%. For power, we show that although the filter overhead was about 20%, other parts of the architecture reduced the total power consumption due to a lower activity caused by using HPEL precision, only available in VVC. Therefore, for the highest achieved target throughput of 2160p@60fps, the power overhead was as little as 0.15%. Also, we conclude that despite the area and power overheads imposed by the addition of the alternative filter, the energy efficiency is almost the same as the baseline architecture. When operating in HPEL precision, the proposed architecture becomes more energy-efficient than the baseline architecture without VVC support.

REFERENCES

[1] Ericsson, "Ericsson mobility report," Stockholm, Sweden, Document, Nov. 2022.

[2] ITU-T, "Recommendation ITU-T H.266: Versatile video coding," ITU, Geneva, CH, Recommendation H.265, Aug. 2020.

[3] F. Pakdaman *et al.*, "Complexity analysis of next-generation VVC encoding and decoding," in *ICIP'20*, IEEE, 2020.

[4] ITU-T, "Recommendation ITU-T H.265: High efficiency video coding," ITU, Geneva, CH, Recommendation H.265, Apr. 2013.

[5] F. Bossen *et al.*, *VVC Test Model (VTM) v6.2*, https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tags/VTM-6.2, 2020.

[6] K. Suehring and K. Sharman, *HM HEVC reference software*, https://vcgit.hhi.fraunhofer.de/jct-vc/HM, 2019.

[7] I. Siqueira *et al.*, "Rate-distortion and complexity comparison of HEVC and VVC video encoders," in *LASCAS'20*, Feb. 2020, pp. 1–4.

[8] M. Grellert *et al.*, "Complexity-scalable HEVC encoding," in *PCS'16*, Dec. 2016, pp. 1–5.

[9] A. Henkel *et al.*, "JVET-N0309-v3: Non-CE4: Switched half-pel interpolation filter," Fraunhofer HHI, Geneva, CH, Mar. 2019.

[10] H. Liu *et al.*, "JVET-M0255: AHG11: MMVD without fractional distances for SCC," Bytedance Inc., Marrakech, MA, Jan. 2019.

[11] W. Zhu *et al.*, "JVET-N0260-v1: Non-CE8: Adaptive fractional MVD search in DMVR for SCC," Bytedance Inc., Geneva, CH, Mar. 2019.

[12] Y. Zhang *et al.*, "Recent advances on HEVC inter-frame coding: From optimization to implementation and beyond," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 11, pp. 4321–4339, 2020.

[13] I. Seidel *et al.*, "SAD or SATD? how the distortion metric impacts a Fractional Motion Estimation VLSI architecture," in *MMSP'21*, Tampere, Finland: IEEE, Oct. 2021.

[14] I. Chakrabarti *et al.*, *Motion Estimation for Video Coding: Efficient Algorithms and Architectures* (Studies in Computational Intelligence). Springer International Publishing, 2015.

[15] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Process. Mag.*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[16] V. Rodrigues Filho *et al.*, "Standalone rate-distortion FME architecture," in *SBCCI'20*, 2020, pp. 1–6.

[17] ISO Central Secretary, "Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 2: High efficiency video coding," ISO, Geneva, CH, Standard ISO/IEC 23008-2, 2013.

[18] A. Henkel *et al.*, "Alternative half-sample interpolation filters for versatile video coding," in *ICASSP'20*, 2020, pp. 2053–2057.

[19] I. Seidel *et al.*, "Coding- and energy-efficient FME hardware design," in *ISCAS'18*, 2018.

[20] Synopsys, *Synopsys Design Compiler, v. S-2021.06-SP5*, 2021.

[21] *TSMC standard cell library tcbn45gsbwptc*, TSMC, 2011.