HAL
open science

Quality-Driven Dynamic VVC Frame Partitioning for Efficient Parallel Processing

Thomas Amestoy, Wassim Hamidouche, Cyril Bergeron, Daniel Menard

To cite this version:

Thomas Amestoy, Wassim Hamidouche, Cyril Bergeron, Daniel Menard. Quality-Driven Dynamic VVC Frame Partitioning for Efficient Parallel Processing. 27th IEEE International Conference on Image Processing (ICIP 2020), Oct 2020, Abu Dhabi, United Arab Emirates. pp.3129-3133, 10.1109/ICIP40778.2020.9190928 . hal-03067262

HAL Id: hal-03067262

https://hal.science/hal-03067262v1

Submitted on 29 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

QUALITY-DRIVEN DYNAMIC VVC FRAME PARTITIONING FOR EFFICIENT PARALLEL PROCESSING

Thomas AMESTOY[⋆,†], *Wassim HAMIDOUCHE*[⋆], *Cyril BERGERON*[†] *and Daniel MENARD*[⋆]

[⋆] Univ Rennes, INSA Rennes, CNRS, IETR - UMR 6164, Rennes, France. Emails: firstname.lastname@insa-rennes.fr
[†] Thales SIX GTS France, HTE/STR/MMP Gennevilliers, France. Emails: firstname.lastname@thalesgroup.com

ABSTRACT

VVC is the next generation video coding standard, offering coding capability beyond HEVC standard. The high computational complexity of the latest video coding standards requires high-level parallelism techniques, in order to achieve real-time and low latency encoding and decoding. HEVC and VVC include tile grid partitioning that allows to process simultaneously rectangular regions of a frame with independent threads. The tile grid may be further partitioned into a horizontal sub-grid of Rectangular Slices (RSs), increasing the partitioning flexibility. The dynamic Tile and Rectangular Slice (TRS) partitioning solution proposed in this paper benefits from this flexibility. The TRS partitioning is carried-out at the frame level, taking into account both spatial texture of the content and encoding times of previously encoded frames. The proposed solution searches the best partitioning configuration that minimizes the trade-off between multi-thread encoding time and encoding quality loss. Experiments prove that the proposed solution, compared to uniform TRS partitioning, significantly decreases multi-thread encoding time, with slightly better encoding quality.

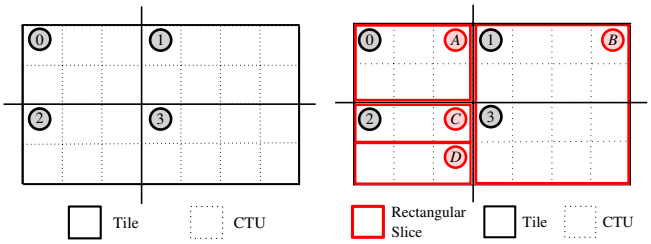
***Index Terms*—** Video Compression, VVC, High Level Parallelism, Rectangular Slices, VTM

1. INTRODUCTION

In recent years, the democratization of multimedia applications, coupled with the emergence of high resolution and new video formats (8K, 360°), has led to a drastic increase in the volume of exchanged video content [1]. This increasing need for higher compression rates prompted the Joint Video Exploration Team (JVET) to develop a new video coding standard called Versatile Video Coding (VVC) with coding capability beyond High Efficiency Video Coding (HEVC) [2]. The bit-rate savings brought by VVC [3] are however coupled with a considerable encoding computational complexity increase. This latter is estimated to 10 and 27 times HEVC computational complexity in Inter and Intra coding configuration, respectively [4]. In real-time implementations of VVC codec, intense parallel processing will therefore be mandatory to achieve real-time encoding and decoding.

Techniques of video parallel processing essentially operate at three levels of parallelism: data level, frame level and high-level. The data level parallelism techniques are applied on elementary operations, and no encoding quality is lost compared to sequential encoding. They include among other techniques relying on Single Instruction on Multiple Data (SIMD) architectures [5]. Frame level and high-level parallelism operate at thread level.The frame level techniques encode a group of frames in parallel where each thread is assigned to a single frame [6]. The encoding time of a single frame is not reduced with frame level techniques, i.e. the latency is not reduced. In high-level parallelism techniques, the threads operate on continuous regions of the frame, as tiles or slices [7]. Tiles and slices are independently encodable and decodable, allowing several threads to process simultaneously the same frame. These techniques improve equally both speed-up and latency. However, by enabling independent processing of frame regions, prediction dependencies across boundaries are broken and entropy encoding state is reinitialized for each region. These restrictions lead to an encoding quality loss compared to the encoding of the non-partitioned sequence. The encoding quality decreases with the number of independent regions of the frame, as has been measured in HEVC by *Chin et al.* [8].

In HEVC and VVC standards, only grid shaped tile partitioning is allowed, as shown by **Fig. 1a**. The tiles are delimited by the continuous black lines and the dashed lines correspond to the Coding Tree Unit (CTU) delimitation. The tile partitioning forms a 2x2 grid and tiles are labelled from 0 to 3. In order to increase the partitioning opportunities, VVC combines the tile partitioning with the new concept of Rectangular Slices (RSs). The partitioning combining tiles and RSs is further called Tile and Rectangular Slice (TRS) partitioning. **Fig. 1b** shows the TRS partitioning of a frame into the same 2x2 tile grid than **Fig. 1a**, combined with 4 RSs. The RSs are delimited by the continuous red lines and are labeled from A to D. The RS may contain one or several complete tiles, forming together a rectangular region of the frame. Moreover, as shown in the examples C and D, a RS may be a rectangular sub-region of the tile, composed of a number of complete and consecutive CTU rows of a tile. In this latter case, the RSs allow to further partition the tile grid into a horizontal sub-grid, improving greatly the tile grid partitioning flexibility.



(a) Grid of 4 tiles. Tiles labeled from 0 to 3

(b) Tiles combined with 4 RSs. RSs labeled from A to D.

Fig. 1: Illustration of tile partitioning in HEVC and TRS partitioning in VVC.

The partitioning of a frame into tiles and RSs raises two distinct optimization issues: on one side the multi-thread encoding time minimization (or speedup maximization), on the other side the minimization of encoding quality loss caused by the partitioning. In the literature, both issues have been addressed for HEVC tile partition-

This project has received funding from Bpifrance Financement under grant DOS0061463/00 (EFIGI FUI project).

ing. The multi-thread encoding time minimization is investigated by *Storch et al* [9] and *Koziri et al.* [10]. They observe that the encoding time does not vary significantly from a CTU to the co-located CTU in the closest temporal frame. Considering this temporal stability, the authors use the encoding times of previous frames to determine the tile partitioning that minimizes the multi-thread encoding time. In [11], the time estimator for each CTU is computed based on previously encoded frame CTU statistics (number of Skip, Inter, Intra blocks for instance). Authors in [12, 13] minimize the encoding quality loss induced by the tile partitioning by analyzing the CTU luminance variances of the frame. The technique proposed in [14] focuses on the particular case of variable number of available cores. The encoding loss is lowered in some cases by setting a number of tiles inferior to the number of available cores. However, the related works on HEVC tile partitioning only address independently minimization of encoding time and encoding quality loss.

In this work, we take advantage of the increased flexibility offered by the RSs in VVC, in order to propose a dynamic TRS partitioning solution under VVC Test Model (VTM)-6.2 software. Prior to the encoding of a frame, the TRS partitioning stage uses the spatial information and the times of previously encoded CTUs in order to optimize the TRS partitioning. The proposed solution minimizes a trade-off between encoding time and encoding video quality, which is a novel approach compared to related works. Moreover, to the best of our knowledge, this is the first work that implements a multi-thread VVC reference encoder, generating baseline results for future related works.

The rest of the paper is organized as follows. Section 2 describes the proposed solution, which establishes the trade-off between encoding time and encoding quality. Section 3 presents and analyses the experimental results on VTM-6.2. Finally, Section 4 concludes this paper.

2. DYNAMIC FRAME PARTITIONING FOR PARALLEL PROCESSING

As mentioned in Section 1, the proposed TRS partitioning solution addresses simultaneously the minimization of encoding time and the limitation of encoding quality loss. This section first describes the encoding time minimization of the current frame, using times of previously encoded co-located CTUs. The second subsection introduces the clustering of spatial information into the RSs to limit the encoding quality loss. The last subsection describes the proposed solution, that establishes a trade-off between encoding time and encoding quality.

2.1. Encoding Time Minimization

Let P be the partitioning of current frame into n RSs: $P = \{s_0, ..., s_{n-1}\}$. In the following, $T(P)$ is the encoding time of current frame partitioned with P, and simultaneously processed by N threads in parallel (each thread entirely dedicated to encode a single RS). In this case, $T(P)$ is equal to the time required by the slowest thread to encode his RS. Eq. 1 formally establishes $T(P)$, with $T(c_i)$ the encoding time of CTU c_i and $T(s_j)$ the encoding time of the RS s_j.

$$\begin{aligned} T(s_j) &= \sum_{c_i \in s_j} T(c_i), \\ T(P) &= \max_{s_j \in P}(T(s_j)). \end{aligned} \tag{1}$$

Eq. 1 shows that $T(P)$ is directly determined by the CTU encoding times $T(c_i)$. However, during the TRS partitioning stage, these values are not available, since the TRS partitioning stage takes place before the encoding of current frame. In order to overcome this lack of information, the values $T(c_i)$ are replaced during the TRS partitioning stage by estimated values noted $\tilde{T}(c_i)$.

Several related works [9, 10] define $\tilde{T}(c_i)$ as the encoding time of the co-located CTU (located at the same spatial coordinates) in the closest temporal frame previously encoded. This choice is motivated by the temporal continuity of the video sequences content. In Random Access (RA) configuration, authors in [15] have shown that $T(c_i)$ is more correlated with the times of the co-located CTU in co-Temporal Layer (TL) frame, compared to the co-located CTU of the closest temporal frame. The co-TL frame refers to the previously encoded frame belonging to same temporal layer. This is caused by the shared coding parameters of frames at similar temporal level in the group of pictures structure defined by the Common Test Conditions (CTC) [16]. Following the results of [15], the selected estimator $\tilde{T}(c_i)$ is defined as the encoding time of the co-located CTU in the co-TL frame. The encoding time minimization technique consists in the search of a TRS partitioning P that minimizes the estimated $\tilde{T}(P)$, computed with $\tilde{T}(c_i)$ values as an input.

2.2. Limitation of Encoding Quality Losses

Fig. 2: TRS partitioning of *BQTerrace* frame #4, computed with slice clustering.

As mentioned in Section 1, prediction dependencies across RSs boundaries are disabled and entropy coding state is reinitialized at each RS. In order to limit the encoding quality loss induced by these restrictions, the optimal TRS partitioning P^* gathers similar spatial information inside the same RSs. This corresponds to a K-mean clustering [17] of the spatial information into the RSs, further called RS clustering. The RS clustering searches the TRS partitioning P^* that minimizes the sum of luminance variance on all RSs. Eq. 2 computes the partitioning P^* where p_i is the value of luminance samples, and μ_j is the mean of RS s_j luminance samples.

$$P^* = \underset{P}{argmin} \left[\sum_{s_j \in P} \sum_{p_i \in s_j} (p_i - \mu_j)^2 \right]. \tag{2}$$

Fig. 2 shows the 8 RSs partitioning, obtained by solving Eq. 2 for frame #4 of sequence *BQTerrace*. In **Fig. 2**, regions of the frame with similar spatial information tend to be clustered into the same RSs. The dark water of the river is almost entirely contained in RSs 6 and 7, and the light homogeneous regions of the frame are mainly

included in RSs 0, 3 and 5. On the other hand, the RSs 1, 2 and 4 contain the regions with more complex spatial information.

2.3. Two Steps Slice Partitioning Search

The TRS partitioning in **Fig. 2** gathers similar spatial information inside the same RSs, but is far from optimal regarding the encoding time minimization. For instance, the encoding at $QP = 27$ of RS #1 is 12 times slower compared to the encoding of RS #3, due among others to the greater area and spatial complexity of RS #1 compared to RS #3. The encoding time of the considered frame is therefore sub-optimal due to the high encoding time of RS #1. In order to reduce such imbalances between RSs encoding times, the proposed solution combines the RS clustering (Section 2.2) with the encoding time minimization technique (Section 2.1).

The proposed solution is represented as a flowchart in **Fig. 3**. The TRS partitioning stage, enclosed in the blue dashed box, is applied prior to the parallel encoding of current frame F_{cur}, enclosed in the red dashed box. The TRS partitioning stage is divided into 2 distinct steps. The first step is called encoding time minimization step. This step computes the minimum estimated encoding time, defined by Equation 3 and noted $\tilde{T}_{min}$.

$$\tilde{T}_{min} = \min_{P}(\tilde{T}(P)) \tag{3}$$

The encoding time minimization step takes the CTU times of the co-TL frame F_{TL} as input.

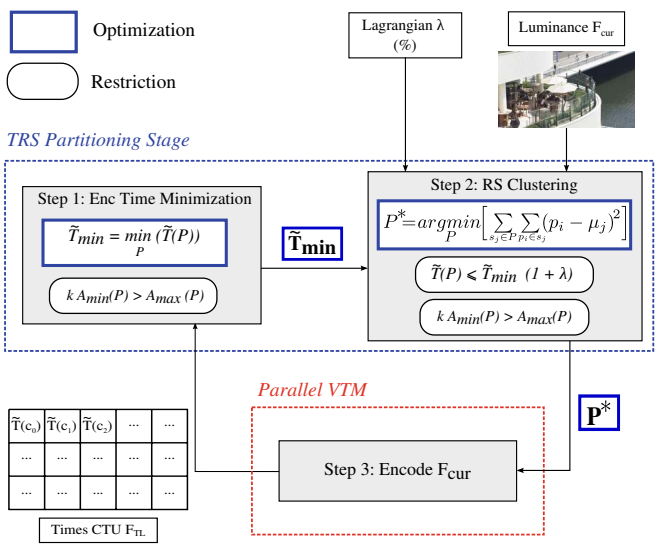


Fig. 3: Proposed solution flowchart.

The second step of the TRS partitioning stage computes the RS clustering of F_{cur}, under encoding time constraint. This step takes as inputs $\tilde{T}_{min}$ estimated during previous step, the luminance samples of F_{cur}, and a lagrangian parameter λ that manages the trade-off between encoding time and encoding quality. The possible values for $\tilde{T}(P)$ are bounded by Eq. 4.

$$\tilde{T}(P) \leq \tilde{T}_{min} \cdot (1 + \lambda) \tag{4}$$

When $\lambda = 0$, only the partitioning P that minimizes the estimated time is considered, since $\tilde{T}(P) = \tilde{T}_{min}$. When λ increases, more partitioning opportunities are offered to the RS clustering, and therefore higher weight is given to encoding quality compared to encoding time minimization. The parameter λ is therefore a means for the encoder to manage the trade-off, according to the requirement.

The aim of this paper is to show the relevance of a solution combining the 2 complementary steps previously presented. For this reason, a near exhaustive search is conducted to compute both $\tilde{T}_{min}$ and RS clustering. As shown in **Fig. 3**, the only constraint given to the search algorithm: $k \cdot A_{min}(P) > A_{max}(P)$, with A_{min} and A_{max} the area of the smallest and the largest RSs, respectively. The constant k is set to 3 in this work in order to contain search complexity. The choice of less complex heuristics for the TRS partitioning stage is a distinct issue, that will be part of future works. The global complexity overhead induced by the TRS partitioning stage is nonetheless measured and discussed further in this paper.

3. EXPERIMENTAL RESULTS

This section presents the experimental setup, as well as the performance of the proposed TRS partitioning solution.

3.1. Experimental Setup

The following experiments are conducted under VTM-6.2 software, built with gcc compiler version 7.4.0, under Linux version 4.15.0-74-generic as distributed in Ubuntu-18.04.1. The platform setup is composed of Central Processing Units (CPUs) Intel(R) Xeon(R) E5-2690 v3 clocked at 2.60 GHz, each of them disposing of 12 cores. The cores have each 768KB L1 cache, 3MB L2 cache and 30MB L3 cache.

The high-level parallelism structures included in VVC standard allow to tackle complexity increase on multi-core processors. This complexity increase raises a critical issue mainly for high resolution video sequences. For this reason, the test sequences selected in this work contain 4 Ultra High Definition (UHD) and 5 Full High Definition (FHD) sequences included in the CTC [16]: *CatRobot1*, *DaylightRoad2*, *FoodMarket4*, *Tango2* (UHD), and *BQTerrace*, *Cactus*, *MarketPlace*, *RitualDance* (FHD). The test sequences are encoded under RA configuration at four Quantization Parameter (QP) values: 22, 27, 32, 37. The performance of our TRS partitioning solution is assessed by measuring the trade-off between the encoding quality using the Bjøntegaard Delta BitRate (BD-BR) [18] and the multi-thread speed-up σ, defined by Eq. 5.

$$\sigma = \frac{1}{4} \sum_{QP_i \in \{22,27,32,37\}} \frac{T_O(QP_i)}{T_R(QP_i)} \tag{5}$$

$T_O(QP_i)$ and $T_R(QP_i)$ are the original time (encoded with 1 RS and 1 single thread) and reduced time (encoded with N RSs and N threads) spent to encode the video sequence with QP_i, respectively. The overhead induced by TRS partitioning stage is further noted θ and measured in percentage of T_R.

3.2. Performance of the Proposed Solution

The theoretical upper bound in terms of speed-up, noted σ_{max}, for the proposed solution is computed with the Amdahl law [19]. Let s be the sequential part (in %) of an application. The upper bound σ_{max} obtainable with n threads is expressed by Eq. 6.

$$\sigma_{max}(n) = \frac{1}{s + \frac{1-s}{n}} \tag{6}$$

In our case, the sequential portion of VTM-6.2 encoder contains the data initialization, entropy, in-loop filter and bitstream writing stages. All together, these stages represent 4% of the encoding time in average across test sequences and QP values. Therefore, Eq. 6 provides the following upper bounds: $\boldsymbol{\sigma_{max}(4) = 3.57}$, $\boldsymbol{\sigma_{max}(8) = 6.25}$ and $\boldsymbol{\sigma_{max}(12) = 8.33}$.

As mentioned in Section 2.3, the lagrangian parameter λ manages the trade-off between encoding quality and encoding time minimization induced by the TRS partitioning. Three values of parameter λ (0, 0.1 and 0.3) are tested, and the one offering the best trade-off is selected according to thread number and resolution. **Table 1** presents the average results obtained with the selected λ values, according to the resolution and number of threads n. Moreover, the results of the uniform TRS partitioning applied on the test sequences is also presented, in order to evaluate the performance of the proposed solution. The uniform TRS partitioning is an usual and straightforward technique that partitions the frame in a grid of the same RS dimension.

Table 1: Average speed-up σ, BD-BR and overhead θ obtained by both uniform and proposed TRS partitioning, according to the resolution and number of threads n.

		FHD		**UHD**	
		Unif	**Proposed**	**Unif**	**Proposed**
			$\underline{\lambda = 0}$		$\underline{\lambda = 0}$
$n = 4$	BD-BR (%)	1.62	**1.57**	1.31	**1.27**
	Speed-up σ	2.68	**3.10**	2.91	**3.27**
	$\theta(\%)$		0.0		0.0
			$\underline{\lambda = 0.3}$		$\underline{\lambda = 0.1}$
$n = 8$	BD-BR (%)	2.69	2.80	2.39	**2.33**
	Speed-up σ	4.27	**5.07**	4.55	**5.34**
	$\theta(\%)$		0.01		0.08
			$\underline{\lambda = 0.1}$		$\underline{\lambda = 0.1}$
$n = 12$	BD-BR (%)	4.31	**3.90**	3.26	**3.20**
	Speed-up σ	5.57	**6.44**	6.13	**7.09**
	$\theta(\%)$		0.54		1.84

Table 1 shows that the proposed TRS partitioning solution enables better results compared to uniform TRS partitioning in term of σ, regardless the resolution and number of threads n. The σ increase ranges from 0.36 to 0.94, for UHD content with $n = 4$ and $n = 12$, respectively. The proposed TRS partitioning solution therefore reduces significantly the distance to the upper bounds σ_{max} computed by Amdahl law, compared to uniform TRS partitioning. This significant σ increase proves the efficiency of the encoding time minimization step, presented in Section 2.1. It is important to note that the encoding time of every frame is reduced. Therefore both speed-up and latency are improved equally by the proposed solution.

In term of BD-BR, the results of the proposed solution with the selected λ values are slightly better (around -0.05%) compared to uniform TRS partitioning. Two exceptions are however noticeable. The BD-BR decrease is substantial (-0.41%) for FHD content with $n = 12$, and the only case for which the BD-BR is slightly higher is for FHD content with $n = 8$ ($+0.11\%$). The related works in HEVC minimizing the BD-BR reported 0.16% [12] and 0.10% [15] average BD-BR decrease with 8 threads on FHD and UHD content. Our results in term of BD-BR are therefore close to the results of previously mentioned works, even though these works minimize the BD-BR without taking into consideration the speed-up optimization.

The conclusion of **Table 1** is that the proposed solution is able to maintain the BD-BR increase to values close to uniform RS partitioning. The variation of λ value is however not sufficient to decrease significantly the BD-BR, except for FHD content with $n = 12$. On the other hand, the proposed solution is highly effective to increase the speed-up offered by the TRS partitioning in VVC. Regarding the overhead θ, the values are half induced by the encoding time minimization step, and half by the encoding quality loss limitation step. The values are negligible when $n = 4$ and $n = 8$. For $n = 12$, θ is greater than 0.5% due to the almost exhaustive search implemented (see Section 2.3). We are confident that the investigation of simple heuristics in future works will reduce greatly θ, without degrading the results presented in **Table 1**.

Table 2: Proposed solution with $\lambda = 0$ and $\lambda = 0.1$, encoded with 8 threads, according to UHD sequence.

8 Threads, UHD Sequences

	Proposed Solution $\boldsymbol{\lambda = 0}$		**Proposed Solution $\boldsymbol{\lambda = 0.1}$**	
Sequence	BD-BR (in %)	σ	BD-BR (in %)	σ
CatRobot1	1.38	5.24	1.14	5.19
DaylightRoad	1.82	5.79	1.70	5.70
FoodMarket	4.09	5.16	3.85	5.10
Tango2	2.67	5.54	2.61	5.40
Average	**2.49**	**5.43**	**2.33**	**5.34**

Table 2 shows the performance of the proposed solution with $\lambda = 0$ and $\lambda = 0.1$ running with 8 threads, according to the UHD sequence. As explained in Section 2.3, the higher λ, the more importance is given to encoding quality with regard to the speed-up. The results of **Table 2** are coherent with this explanation. Indeed, for every sequence the proposed solution with $\lambda = 0.1$ enables better BD-BR but lower σ compared to the proposed solution with $\lambda = 0$. In average, the BD-BR is 0.16% better when selecting $\lambda = 0.1$, without degrading significantly σ (-0.09). The results are particularly noticeable for sequence *FoodMarket*. For this sequence, the BD-BR is 0.24% better and σ only decreases by 0.06% when selecting $\lambda = 0.1$, compared to the proposed solution with $\lambda = 0$.

4. CONCLUSION

In this paper, a dynamic TRS partitioning is proposed for next generation video standard VVC. The proposed solution combines two techniques to minimize multi-thread encoding time and encoding quality loss, respectively. A lagrangian parameter λ is applied, allowing to select a trade-off between encoding time and encoding quality. The experiments show that the proposed solution decreases significantly multi-thread encoding time, with slightly better encoding quality, compared to uniform RS partitioning. Future works will focus among other points on the improvement of the CTU time estimator, used in the encoding time minimization step. Instead of simply relying on the co-located CTU times of the co-TL frame, future solutions will rely on CTU deduced by motion information. The investigation of lightweight heuristics for the TRS partitioning stage will also be part of future works. We are confident they will reduce drastically the overhead, especially for 12 threads encodings of UHD content.

5. REFERENCES

[1] CISCO, "Global_2021_forecast_highlights," p. 6, 2016.

[2] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[3] Naty Sidaty, Wassim Hamidouche, Olivier Deforges, Pierrick Philippe, and Jerome Fournier, "Compression Performance of the Versatile Video Coding: HD and UHD Visual Quality Monitoring," in *2019 Picture Coding Symposium (PCS)*, Ningbo, China, Nov. 2019, pp. 1–5, IEEE.

[4] Frank Bossen, Karsten Suehring, and Xiang Li, "JVET-P0003: AHG report: Test model software development (AHG3)," 2019.

[5] Benjamin Bross, Mauricio Alvarez-Mesa, Valeri George, Chi Ching Chi, Tobias Mayer, Ben Juurlink, and Thomas Schierl, "HEVC real-time decoding," San Diego, California, United States, Sept. 2013, p. 88561R.

[6] Wassim Hamidouche, Mickael Raulet, and Olivier Deforges, "4K Real-Time and Parallel Software Video Decoder for Multilayer HEVC Extensions," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 169–180, Jan. 2016.

[7] Kiran Misra, Andrew Segall, Michael Horowitz, Shilin Xu, Arild Fuldseth, and Minhua Zhou, "An Overview of Tiles in HEVC," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 969–977, Dec. 2013.

[8] Chi Ching Chi, M. Alvarez-Mesa, B. Juurlink, G. Clare, F. Henry, S. Pateux, and T. Schierl, "Parallel Scalability and Efficiency of HEVC Parallelization Approaches," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1827–1838, Dec. 2012.

[9] Iago Storch, Daniel Palomino, Bruno Zatt, and Luciano Agostini, "Speedup-aware history-based tiling algorithm for the HEVC standard," in *2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, Sept. 2016, pp. 824–828, IEEE.

[10] Maria Koziri, Panos K. Papadopoulos, Nikos Tziritas, Nikos Giachoudis, Thanasis Loukopoulos, Samee U. Khan, and Georgios I. Stamoulis, "Heuristics for tile parallelism in HEVC," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Kos, Greece, Aug. 2017, pp. 1514–1518, IEEE.

[11] Yong-Jo Ahn, Tae-Jin Hwang, Dong-Gyu Sim, and Woo-Jin Han, "Complexity model based load-balancing algorithm for parallel tools of HEVC," in *2013 Visual Communications and Image Processing (VCIP)*, Kuching, Malaysia, Nov. 2013, pp. 1–5, IEEE.

[12] Cauane Blumenberg, Daniel Palomino, Sergio Bampi, and Bruno Zatt, "Adaptive content-based Tile partitioning algorithm for the HEVC standard," in *2013 Picture Coding Symposium (PCS)*, San Jose, CA, USA, Dec. 2013, pp. 185–188, IEEE.

[13] Xin Jin and Qionghai Dai, "Clustering-Based Content Adaptive Tiles Under On-chip Memory Constraints," *IEEE Transactions on Multimedia*, vol. 18, no. 12, pp. 2331–2344, Dec. 2016.

[14] Giovani Malossi, Daniel Palomino, Claudio Diniz, Altamiro Susin, and Sergio Bampi, "Adjusting video tiling to available resources in a per-frame basis in High Efficiency Video Coding," in *2016 14th IEEE International New Circuits and Systems Conference (NEWCAS)*, Vancouver, BC, Canada, June 2016, pp. 1–4, IEEE.

[15] Chia-Hsin Chan, Chun-Chuan Tu, and Wen-Jiin Tsai, "Improve load balancing and coding efficiency of tiles in high efficiency video coding by adaptive tile boundary," *Journal of Electronic Imaging*, vol. 26, no. 1, pp. 013006, Jan. 2017.

[16] Jill Boyce, Karsten Suehring, and Xiang Li, "JVET-J1010: JVET common test conditions and software reference configurations," 2018.

[17] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A K-Means Clustering Algorithm," *Applied Statistics*, vol. 28, no. 1, pp. 100, 1979.

[18] Gisle Bjontegaard, "Calculation of average PSNR differences between RD-Curves," Apr. 2001, VCEG-M33 ITU-T.

[19] Mark D Hill and Michael R Marty, "Amdahl's Law in the Multicore Era," p. 6.