Adaptive Coding Tree for Complexity Control of High Efficiency Video Encoders

[1] Guilherme Corrêa, [2] Pedro Assuncao,
[1] Luis A. da Silva Cruz
Instituto de Telecomunicações
[1] DEEC – FCTUC – University of Coimbra
[2] DEE – ESTG – Polytechnic Institute of Leiria
Portugal
{guilherme.correa, amado, luis.cruz}@co.it.pt

Luciano Agostini
GACI – CDTEC – Federal University of Pelotas
Pelotas, Brazil
agostini@inf.ufpel.edu.br

***Abstract*—The emerging HEVC standard introduces several techniques which increase compression efficiency in comparison to its predecessors. However, such advances are accompanied by increases in computational complexity, limiting the encoder use in computational or power-constrained devices. This paper proposes a novel complexity control method for the future HEVC encoders based on a dynamic adjustment of the newly proposed coding tree structures. The relationship between coding tree depths and the encoding complexity is explored to selectively constrain encoding possibilities in order to not exceed a predefined complexity target. Experimental results show that the encoder computational complexity can be downscaled to 60% with a bit rate increase under 3.5% and a PSNR decrease under 0.1 dB.**

I. Introduction and Related Work

The recent large-scale popularization of multimedia systems has been enabled by the constant research and development of video/audio compression standards in the last decades. After the conclusion of H.264/AVC standardization in 2004, ISO/IEC and ITU-T kept exploring new methods and algorithms to increase compression efficiency, especially for high resolution video formats. Motivated by these investigations, the emerging High Efficiency Video Coding (HEVC) standard is currently being defined by the Joint Collaborative Team on Video Coding (JCT-VC) and is expected to be ready by 2013 [1]. The aim of HEVC is to decrease by 50% the bit rate in comparison with H.264/AVC High Profile at the same subjective image quality. However, to reach such a goal, several new functions and encoding modes have already been aggregated to the current version of the HEVC Test Model (HM) [2], effectively increasing its computational complexity in comparison with H.264/AVC.

The computational complexity of video encoders, on the other hand, has also been a constant concern in later years. Despite all recent evolution in portable devices, particularly in communications technology and computational power, the limited battery capacity still imposes major constraints in multimedia applications which demand high computational power, such as video coding and decoding. Besides, computational complexity has also become a concern in non-portable devices operating in real time with limited computational capability. With the introduction of HEVC accompanied by the use of high resolution video formats, such problems will surely become still more serious in multimedia-capable systems, limiting the user quality of experience despite all advances in terms of compression efficiency.

Several works have been published in the last few years aiming at adjusting the computational complexity of video encoding and decoding algorithms, generally focusing on the motion estimation (ME) and mode decision (MD) processes [3-6]. In [3, 5, 6] a complexity control algorithm is proposed to dynamically adjust the ME search method, either adopting a fast full search approach or a combination of different fast search techniques. In [4], complexity is managed through adjustments in parameters that jointly control both ME and MD, such as the search area in ME and the number of candidate coding modes in MD. Other encoder modules can also be controlled through some parametric adjustment, as proposed in [7], where eleven parameters are combined in twenty possible complexity operating points. In [8], the complexity of entropy coding, deblocking filter, ME and MD operations are controlled by four different parameters, which adjust the accuracy and complexity of each module operation separately. Even though these methods can be used to successfully control computational complexity of previous standards, most of them are not directly applicable to HEVC whose computational complexity is highly dependent on the newly proposed quadtree structures, as next section will show.

This paper proposes an algorithm which allows adjusting the computational complexity employed in the definition of coding tree structures according to the encoder resource availability. To the best of the authors' knowledge, the method introduces the first complexity scalability algorithm for HEVC, allowing the encoding computational complexity to be downscaled from 100% to 60% with negligible loss in terms of rate-distortion (RD) performance.

This work has been sponsored by the Instituto de Telecomunicações (IT), Portugal.

This paper is organized as follows: Section II gives a short overview on HEVC encoding structures and presents an analysis of the computational complexity demanded by defining the coding trees. Section III presents the method for complexity control proposed in this paper. Section IV presents experimental results and comparisons. Finally, section V concludes the paper.

II. Quadtree Structures in HEVC

One of the most important features introduced by HEVC is the use of quadtree-based structures. In the current HM, each video frame is divided into a number of square blocks of equal size called treeblocks, which are used as the roots for each coding quadtree, called simply as coding tree. The encoder decides the best coding tree configuration based on exhaustive iterations of the Rate-Distortion Optimization (RDO) process, which considers every possible solution and compares all of them in terms of bit rate and image quality. Each leaf of the coding tree is called a coding unit (CU) and its dimensions can vary from 8x8 to 64x64 pixels (or up to the treeblock size), depending on the tree depth. The smaller the coding tree depth, the larger is the coding unit.

For each CU, the intra-frame prediction or the inter-frames prediction can be independently applied. Each CU can be divided into two or four prediction units (PU), which are predicted separately. Even though PUs are not organized in a quadtree structure, the best PU division is also determined through exhaustive iterations of the RDO process taking into consideration the resulting bit rate and image quality for each PU division possibility.

Finally, during the transform coding of the prediction residual, each CU is assumed to be root of another quadtree-based structure called residual quadtree (RQT). Each leaf of the RQT is called a transform unit (TU), which sizes can vary from 4x4 to 32x32 pixels (or up to the CU size), depending on the RQT depth where it is located. Similarly to the coding tree, the RQT structure is also defined by exhaustive RDO iterations. Also, the smaller the RQT depth, the larger is the transform unit.

Fig. 1 shows an example of a coding tree divided down to the fourth depth level in which the gray nodes (leaves) represent the CUs. It is possible to notice that the number of possible trees tested in the RDO process increases exponentially with the maximum depth allowed, which means that the encoding complexity is tightly connected to quadtree definition. It is important to highlight that for each CU in each possible coding tree configuration, all possible PU divisions and all possible RQT configurations are tested in the RDO process, which includes the computation of bit rate and image distortion after performing intra/inter-prediction, direct and inverse transform and quantization, entropy coding and deblocking operations for all possibilities. The development of heuristics and methods which allow predicting quadtree structures for future frames is thus extremely necessary for complexity reduction and control targeting high efficiency video encoders.

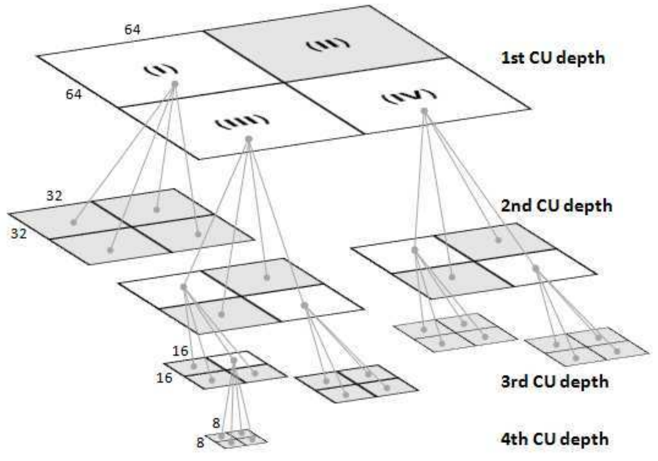

Figure 1. Example of a possible coding tree structure

III. Quadtree-Based Complexity Control

The complexity control algorithm proposed in this work was developed based on a previous analysis of the coding tree depth along the video temporal domain. The maximum coding tree depth used in random co-located treeblocks of neighboring frames was observed for three video sequences. Such observations led to the conclusion that the maximum coding tree depth remains constant during quite long periods of the video sequence, which was already expected due to the natural tendency of a video sequence to comprise several areas either static or with low motion between neighboring frames. Based on this, the proposed method assumes that maintaining the maximum coding tree depth for a relatively long period and skipping all the remaining tree possibilities tests would incur in a small decrease in terms of RD performance.

A. Proposed Method for Complexity Control

The proposed complexity control method is based on dynamic constraining of the maximum coding tree depth according to the depth used in the co-located area of a previously encoded frame. Two types of frames are thus defined in the proposed method: the unconstrained frames (*Fu*) and the constrained frames (*Fc*). *Fu* frames are those in which the full RDO process is applied and all possible coding tree structures are considered and compared in terms of bit rate and image distortion, i.e., maximum possible coding complexity is allowed. *Fc* are frames which have the maximum coding tree depths bound to those used in the most recently encoded *Fu* frame, i.e., coding complexity is constrained to be lower than the maximum possible.

The input frames are classified as unconstrained *Fu* frames while the encoder computational complexity does not exceed a certain pre-computed target computational complexity or after encoding *Nc* consecutive *Fc* frames, as depicted in the example shown in Fig. 2. As the computational complexity required to encode an *Fc* frame is smaller than the computational complexity demanded to encode an *Fu* frame, the number of *Fc* frames (*Nc*) is dynamically adjusted as a function of the available computational complexity. The larger the value of *Nc*, the smaller will be the computational complexity required to encode the video sequence.

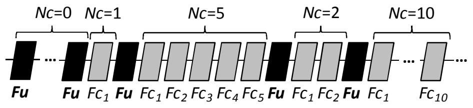

Figure 2. Example of operation of the complexity control algorithm

B. *Algorithm for Complexity Control*

Processing time is the metric used for computational complexity in this research. Fig. 3 presents a high-level diagram for the algorithm proposed, where the number of frames which compose the video segment (N) and the target computational complexity (C^T) are the two inputs. The value of C^T can vary from 0% to 100%, where 100% represents the maximum possible encoding complexity (i.e., when no complexity control is applied). As the actual maximum possible encoding complexity is still unknown at the beginning of the encoding process, its value is estimated as given by equation (1) after encoding the first three frames as *Fu*. In (1), C_E^M is the estimated maximum complexity, C_U^{Fi} is the computational complexity used to encode the i^{th} frame and N is the number of frames in a video segment.

$$C_E^M = \tfrac{1}{3}\sum_{i=1}^{3} C_U^{Fi} \cdot N \tag{1}$$

The value of C_E^M is then used in the calculation of the estimated target complexity as defined in (2), where C_E^T represents the estimated target complexity to encode the whole video segment and C^T is the input target complexity.

$$C_E^T = C^T \cdot C_E^M \tag{2}$$

After defining the estimated target complexity, the algorithm starts monitoring the computational complexity while encoding the video segment. Such monitoring is performed by computing a predicted overall computational complexity for the whole segment (C_P^N), as explained later. While C_P^N is within the limits imposed by C_E^T all frames will be encoded as *Fu* frames. Whenever C_P^N increases beyond C_E^T, the complexity control is activated and frames will be encoded as *Fu* or *Fc* according to the algorithm decisions.

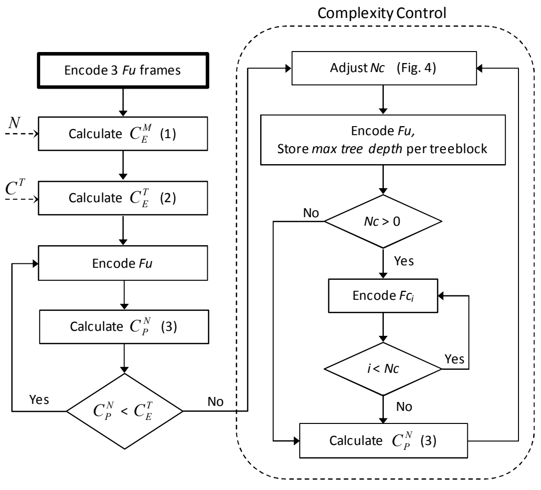

Figure 3. Complexity control algorithm proposed

The first step in the control part of the algorithm (portion inside the dashed line in Fig. 3) is the calculation of a new *Nc* value, which depends simply on Δ*C*, which is the ratio between C_P^N and C_E^T. Fig. 4 shows how *Nc* is adjusted as a non-linear function of Δ*C*. The larger the difference between C_P^N and C_E^T (horizontal axis), the larger is the increase or decrease of *Nc* (vertical axis). Once the positive or negative maximum *Nc* is achieved, the algorithm keeps it unchanged. When the new *Nc* value is defined, an *Fu* frame is encoded and the maximum coding tree depth used for each treeblock of that frame is stored for future use. For example, for the four treeblocks presented in Fig. 1 (I, II, III, and IV) the algorithm would store the values 2, 1, 4, and 3, respectively. After that, the next *Nc* constrained frames *Fc* are encoded with the RDO process limited to the maximum coding tree depths previously stored. The C_P^N value is then once again calculated and *Nc* is adjusted, if necessary.

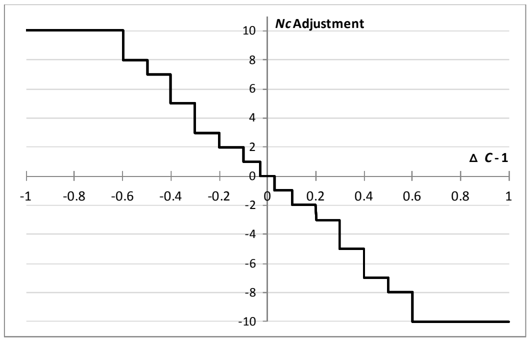

Figure 4. Adjustment of *Nc* value according to Δ*C*.

The C_P^N value is computed as shown in (3), where C_U^{Fi} is the computational complexity used to encode the i^{th} frame of the segment, C_U^{Fu} is the complexity used to encode the last *Fu* frame, $C_U^{Fc_j}$ is the computational complexity used to encode the j^{th} frame of the last group of constrained frames and *NE* is the number of frames already encoded in the video segment.

$$C_P^N = \sum_{i=1}^{NE} C_U^{Fi} + \frac{C_U^{Fu} + \sum_{j=1}^{Nc} C_U^{Fc_j}}{Nc+1} \cdot (N - NE) \tag{3}$$

IV. Experiments and Results

The proposed method was evaluated by measuring the complexity control accuracy under specific targets and its influence on the RD performance of the HM encoder (version 4.0). VTune Amplifier XE 2011 software profiler from Intel was used to accurately measure the processing time. Three 1280x768p video sequences composed by 500 frames (*BasketballDrive*, *BQTerrace*, *Cactus*) and a concatenation of two of them (*BasketballDrive* and *Cactus*, from now on referenced as *BasketballCactus*) were used in the experiments. A video segment is defined as a Group of Pictures (GOP) composed by 250 frames in the performed tests. The encoder performance was evaluated under five target complexities (from 60% to 100%) and four different quantization parameters (QP). An upper limit of 10 frames was used for *Nc* in all experiments. Even though this value can be increased or decreased depending on the system/user requirements, a large maximum *Nc* would incur in higher RD-efficiency losses.

Fig. 5 shows the relationship between the target complexities defined at the beginning of the encoding process and the actual complexities obtained after encoding the video sequences. The QP used for these simulations was 32, but other values were also tested, leading to similar results. The dashed line in the graph represents the ideal behavior of a complexity control method. As running complexity results for each tested sequence are similar to this ideal case, the proposed method is quite accurate and is capable of controlling computational complexity.

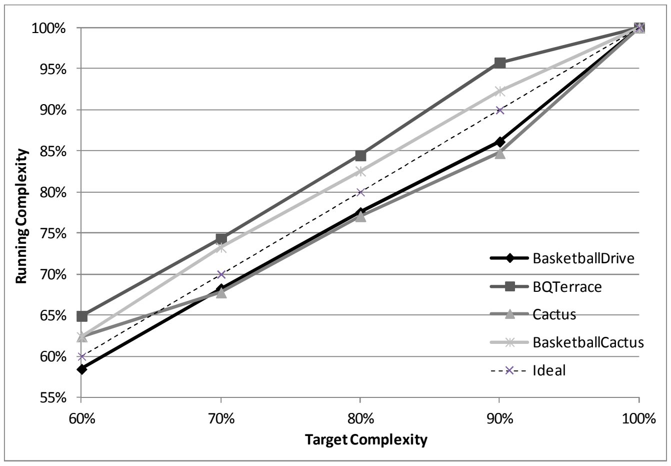

Figure 5. Accuracy of the complexity control method (target complexity versus running complexity)

The encoder rate-distortion efficiency as a function of the computational complexity under different bit rates is presented in Fig. 6. As expected, the best results are obtained when no complexity control is applied (i.e., C^T = 100%). However, Fig. 6 shows quite similar curves for all bit rates, which means that the complexity control algorithm maintains rate-distortion efficiency even when tighter target complexities are defined. The curves in Fig. 6 correspond to the average results for the four video sequences considering four different QPs.

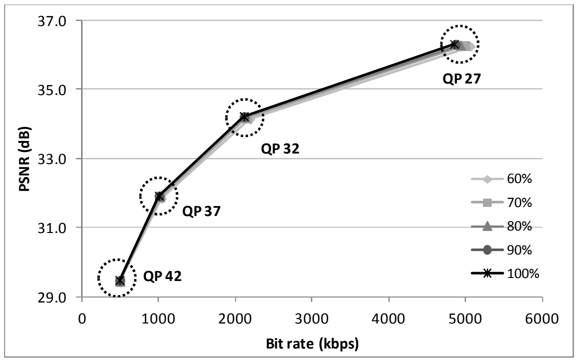

Figure 6. Average rate-distortion performance for each target complexity tested under four different QPs (27, 32, 37, 42)

Table I shows average performance results for the complexity control algorithm. The table shows, for each target complexity tested, the average running complexity, the average bit rate increase, the average luminance PSNR decrease, the BD-rate and the BD-PSNR values in comparison to the maximum complexity case (100%). A common increase of bit rate for smaller target complexities was noticed in all sequences, especially when C^T is set to 60%. This happens due to the fact that limiting the coding tree depth for complexity control leads to a smaller number of small size CUs than in the case of no complexity control, which results in larger residual prediction data to be encoded.

TABLE I. AVERAGE SIMULATION RESULTS (QPS 27, 32, 37, 42)

Target Complex.	Running Complex.	ΔBit rate (%)	ΔY-PSNR (dB)	BD-rate (%)	BD-PSNR (%)
100%	100%	—	—	—	—
90%	90%	+0.29	-0.01	+0.58	-0.017
80%	80%	+0.88	-0.02	+1.55	-0.044
70%	71%	+1.70	-0.04	+3.10	-0.090
60%	62%	+3.44	-0.07	+6.29	-0.181

V. CONCLUSION

A complexity control algorithm for High Efficiency Video Coding (HEVC) was described in this paper. The method relies on dynamically adjusting the newly introduced quadtree-based data structures depth according to computational or power resources availability. Computational complexity can thus be reduced to a predefined target at the cost of negligible loss of RD efficiency. Experimental results show that for a complexity reduction of up to 40% the average PSNR drop observed is lower than 0.1 dB and the average bit rate increase is under 3.5%. The method is especially useful in power-constrained portable multimedia devices to reduce energy consumption and to extend the battery life. It can also be applied to non-portable multimedia devices operating in real-time with limited computational resources.

REFERENCES

[1] ISO/IEC-JTC1/SC29/WG11, "Vision, Applications and Requirements for High-Performance Video Coding," Doc. N11096, ed. Kyoto, Japan: Video Coding Experts Group (VCEG), 39th Meeting, 2010.

[2] ISO/IEC-JTC1/SC29/WG11, "HEVC Reference Software Manual," Doc. JCTVC-E447, ed. Geneva, Switzerland, 2011.

[3] C. Man-Yau and S. Wan-Chi, "Computationally-scalable motion estimation algorithm for H.264/AVC video coding," *IEEE Transactions on Consumer Electronics,* vol. 56, pp. 895-903, 2010.

[4] S. Li, *et al.*, "Complexity-Constrained H.264 Video Encoding," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 19, pp. 477-490, 2009.

[5] Z. Li and G. Wen, "Reusable Architecture and Complexity-Controllable Algorithm for the Integer/Fractional Motion Estimation of H.264," *IEEE Transactions on Consumer Electronics,* vol. 53, pp. 749-756, 2007.

[6] S. Saponara, *et al.*, "Dynamic control of motion estimation search parameters for low complex H.264 video coding," *IEEE Transactions on Consumer Electronics,* vol. 52, pp. 232-239, 2006.

[7] I. R. Ismaeil, *et al.*, "A computation-distortion optimized framework for efficient DCT-based video coding," *IEEE Transactions on Multimedia,* vol. 3, pp. 298-310, 2001.

[8] C. Ming-Chen, *et al.*, "Complexity control for H.264 video encoding over power-scalable embedded systems," in *13th IEEE International Symposium on Consumer Electronics, 2009. ISCE'09*, 2009, pp.221-224.