Performance and Computational Complexity Assessment of High-Efficiency Video Encoders

Guilherme Corrêa, *Student Member, IEEE,* Pedro Assunção, *Member, IEEE,* Luciano Agostini, *Senior Member, IEEE,* and Luis A. da Silva Cruz, *Member, IEEE*

***Abstract*—This paper presents a performance evaluation study of coding efficiency versus computational complexity for the forthcoming High Efficiency Video Coding (HEVC) standard. A thorough experimental investigation was carried out to identify the tools that most affect the encoding efficiency and computational complexity of the HEVC encoder. A set of 16 different encoding configurations was created to investigate the impact of each tool, varying the encoding parameter set and comparing the results with a baseline encoder. This paper shows that, even though the computational complexity increases monotonically from the baseline to the most complex configuration, the encoding efficiency saturates at some point. Moreover, the results of this paper provide relevant information for implementation of complexity-constrained encoders by taking into account the tradeoff between complexity and coding efficiency. It is shown that low-complexity encoding configurations, defined by careful selection of coding tools, achieve coding efficiency comparable to that of high-complexity configurations.**

***Index Terms*—Computational complexity, High Efficiency Video Coding (HEVC), video coding complexity.**

I. Introduction

HIGH EFFICIENCY Video Coding (HEVC) is currently under development by the Joint Collaborative Team on Video Coding (JCT-VC) and is expected to become a new video coding standard by 2013 [1]. Even though current H.264/AVC encoders are already capable of decreasing bit rates by 50% in comparison with previous standards [2], JCT-VC's plan for HEVC is to further decrease these rates, while still maintaining the same high video quality [3]. As high-resolution video applications and services are becoming increasingly popular, this worldwide research effort to achieve higher efficiency in video compression is undoubtedly for the benefit of the multimedia industry and users in general.

Similarly to H.264/AVC and other previous video coding standards, HEVC is a hybrid video coding scheme based on motion estimation/compensation and transform coding. HEVC has evolved from previous standards by adding more advanced features and higher-efficiency coding tools to improve compression ratios without affecting image quality. This is the case, for instance, of the adaptive loop filter, the sample adaptive offset, the motion merge technique, and the new coding structures introduced in HEVC. As expected, higher coding efficiency is obtained at the expense of a significant increase in computational complexity, mainly resulting from much more intensive processing requirements, nested data structures, and optimization algorithms dealing with larger amounts of data [4]. Therefore, research and development of new high-efficiency video encoders poses the challenge of not only improving coding efficiency, but also limiting the necessary additional complexity.

The problem of the high computational complexity required for achieving efficient video encoding directly affects the development of cost-effective multimedia systems; thus, it should be closely connected to the implementation of standards. Like previous video coding standards, HEVC incorporates a significant number of different coding tools, most of them using different parameters, to which several values can be assigned. Therefore, its overall complexity is actually the result of a cumulative contribution of all these coding tools and parameters. Different combinations of such tools and parameters give rise to specific encoding configurations, which necessarily result in quite different performance and complexity.

In this paper, the relationship between the compression efficiency of the new coding tools introduced by HEVC and the corresponding computational complexity is characterized and discussed in terms of the conflicting requirements of minimizing complexity and maximizing coding efficiency. The novel aspects highlighted by the complexity analysis presented in this paper provide a relevant insight into the standardization process of HEVC (including the definition of operating profiles) and also find useful application in the development and implementation of HEVC-based multimedia systems in the near future.

This paper is organized as follows. Section II presents a review of the most recent work on complexity analysis and profiling methodologies used in video encoding and decoding

Manuscript received April 15, 2012; revised July 19, 2012; accepted August 21, 2012. Date of publication October 9, 2012; date of current version January 8, 2013. This work was supported in part by the Portuguese Instituto de Telecomunicações and the Brazilian Agency for Scientific and Technological Development (CNPq, Brazil). This paper was recommended by Associate Editor B. Pesquet-Popescu.

G. Corrêa and L. A. da Silva Cruz are with the Department of Electrical and Computer Engineering, Faculty of Sciences and Technology, University of Coimbra, Coimbra 3030-290, Portugal, and with the Instituto de Telecomunicações, Polo II-Universidade de Coimbra, Pinhal de Marrocos, Coimbra 3030-290, Portugal (e-mail: guilherme.correa@co.it.pt; luis.cruz@co.it.pt).

P. Assunção is with the Polytechnic Institute of Leiria and the Instituto de Telecomunicações, Leiria 2411-901, Portugal (e-mail: amado@co.it.pt).

L. Agostini is with the Group of Architectures and Integrated Circuits, Federal University of Pelotas, Pelotas 96010-900, Brazil (e-mail: agostini@inf.ufpel.edu.br).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2012.2223411

systems. A brief description of the tools used in the HEVC video coding standard in its current state of development is presented in Section III. Section IV describes the methodology used in this paper for complexity and performance analysis and the testbench environment. Section V presents experimental results for a set of encoder configurations, analyzing the tradeoff between encoding performance and computational complexity. Section VI presents an analysis of encoding performance and complexity as a function of different QP values. Conclusions are presented in Section VII.

II. Recent Work on Complexity Analysis

Several works have been published in the last few years, addressing the problem of complexity analysis of video encoders [5]–[12] and decoders [10], [13]–[19]. Generally, these works evaluate the relationship between compression efficiency and complexity, mainly focusing on computational complexity (processing time or instruction-level profiling). Processing time has been largely used to measure computational complexity of video coding systems [5], [6], [9]–[13], [16]–[18], [20], but other approaches such as theoretical analysis [6], [7], [13], [15], direct analysis of reference code [8], [19], and power consumption evaluation [5], [11], [15] have also been frequently applied to measure complexity.

A. Reference Code Analysis

Lehtoranta and Hamalainen [8] analyzed the reference code of an MPEG-4 encoder, evaluating its processing modules by counting their equivalent number of RISC-like instructions. The overall complexity of the MPEG-4 encoder was then compared to that of H.263 and H.264/AVC encoders. Chang-Guo *et al.* [19] also analyzed the C code implementation of an MPEG decoder using the number of basic processor instructions to estimate the computational complexity measured in MOPS.

The code size and complex implementation structures of the reference software in modern video encoders/decoders make this kind of direct analysis of computer code extremely difficult and inaccurate, which encourages the use of other methods and tools.

B. Reported Runtime

Other works measure the computational complexity simply based on the encoding and decoding report generated by the reference software of each standard, which generally includes processing time information. As an example of this approach, Xiang *et al.* [20] report on a method called generalized Bjøntegaard delta PSNR (GBD-PSNR) for evaluating coding efficiency that takes into consideration the bit rate, the image distortion, and the processing time. The computational complexity for each of the main H.264/AVC encoding modules is also theoretically estimated and experimentally evaluated in [6], considering all coding modes possibilities. The experimental results in [6] and [20] are based on the runtime reported by the reference software. Although this type of information provides a rough indication about encoding complexity, to support valid conclusions an accurate analysis is necessary using a controlled test environment and specific tools.

C. Time Stamp Counter

The time stamp counter (TSC) is a register present in all x86 processors since Pentium, which counts the number of clock cycles from the last reset to the current instant. The instruction read time stamp counter (RDTSC) has been used in some works, such as [16] and [17], to estimate the computational complexity of each module and also that of the overall encoder/decoder. Until recently, the use of TSC has been accepted as an excellent way of getting high-resolution, low-overhead processing time information. However, the use of TSC is not accurate in modern processors that support out-of-order execution, since the RDTSC instruction may be executed earlier or later than expected, resulting in a wrong clock cycle count.

D. Energy Consumption Analysis and Estimation

In [5], the average power dissipation was measured using the *PTscalar* thermal and power microarchitecture simulator [21]. In [11], the complexity of the AVS encoder is analyzed taking into consideration the processing time in terms of clock cycles for the Pentium platforms. In this case, the energy consumption is estimated by the Intel Application Energy Toolkit. In [15], actual power dissipation data are extracted through direct measurements in a dynamic voltage and frequency scaling (DVFS) ARM processor employed for energy efficient video decoding.

E. Software Profilers

Profiling tools have been used to automate the complexity analysis of video codecs and generate more reliable results. In [9], the Atomium profiler was used to measure the computational complexity of the main coding functions in the H.264/AVC video codec. Comparisons between H.264/AVC and MPEG-4 Part 2 are also presented. Saponara *et al.* [10] analyzed 18 possible configurations of the H.264/AVC encoder, comparing all of them in terms of complexity versus PSNR in order to identify the best ones. The Atomium profiler was also used to obtain the complexity data by executing the instrumented code with representative test stimuli.

In [18], the computational complexity of a preliminary version of the HEVC decoder is analyzed for ARM and Intel processors under different encoding configurations. The processing time reported by the Oprofile software profiler [22] was used as the complexity measure. The performance application programming interface (PAPI) [23] software profiler was used in [5] to analyze the overall H.264/AVC encoder complexity under different configurations of instantaneous decoding refresh (IDR) period in terms of the average number of clock cycles per frame.

The main modules of the H.264/AVC decoder were also modeled and experimentally analyzed in [13] and [14], using the Intel VTune performance analyzer. This profiler was also used in [12] to measure processing times for the H.263, H.263+, and H.264/AVC encoders under seven different configurations.

The software profilers provide an accurate and easy-to-use method for measuring complexity in video encoders. Each

TABLE I
STRUCTURE OF TOOLS IN HM7 *Main* AND HE10 CONFIGURATIONS

Main	*High Efficiency 10* (HE10)
Rectangular Tiles	
–	Wavefront Parallel Processing
–	Entropy slices to support parallel processing
Coding Unit quadtree structure (CUs from 8×8 to 64×64 luma samples)	
Prediction Units from 4×4 to 64×64 (always square for Intra; also symmetric non-square for Inter)	Prediction Units from 4×4 to 64×64 (always square for Intra; also symmetric and asymmetric non-square for Inter)
Transform Unit tree structure (maximum of 3 levels)	
Transform block from 4×4 to 32×32 (always square)	Transform block from 4×4 to 32×32 (always square for Intra; also non-square for Inter)
Spatial intra prediction (35 angular directions and planar)	
–	Chroma intra prediction separate from or using luma samples
1/4-sample precision interpolation filter for luma motion compensation, 1/8-sample precision interpolation filter for chroma motion compensation	
Advanced Motion Vector Prediction with MV competition and merging	
Context adaptive binary arithmetic entropy coding (CABAC)	
8 bit-per-sample storage and output	10 bit-per-sample storage and output
DF	
SAO	
–	ALF

individual module can be evaluated separately, and batch processing can be used to run several tests in a row, using different predefined configurations.

In this paper, the Intel VTune performance analyzer was used to measure computational complexity, as explained in Section IV.

III. HEV ENCODERS

As previously mentioned, the future HEVC standard is based on the classic hybrid video coding scheme comprising motion estimation or compensation and transform coding. The reference software used in the development phase, called HEVC test model (HM), provides a common framework for research and development [24]. Six mandatory configurations for normalized experiments have been proposed by JCT-VC [25]. These are combinations of two possible complexity or efficiency settings, defined as high efficiency 10 (HE10) and Main,[1] and three basic coding or access settings, named as Intra Only, Random Access, and Low Delay. The HE10 setting is used for high coding efficiency (and high complexity), whereas Main defines a coding configuration with low computational complexity, which does not provide the best coding efficiency. The most relevant differences and similarities between Main and HE10 configurations are summarized in Table I (for more details, refer to [24]). The Intra Only mode uses no temporal prediction, Random Access allows the use of future frames as reference in temporal prediction, and Low Delay uses only previous frames as reference.

One of the most important features introduced by HEVC is the use of quadtree-based frame partitioning structures. In HEVC, each video slice is divided into a number of square blocks of equal size called treeblocks [or largest coding units (LCUs)], which are used as the roots of each coding quadtree (or coding tree). Each leaf of the coding tree is called a coding unit (CU) and its dimensions can vary from 8×8 to 64×64 pixels (or up to the treeblock size), depending on the tree depth. For each CU, intra- or inter-frame prediction can be independently applied. Each CU can be divided into two or four prediction units (PU), which are predicted separately. When transform coding of the prediction residue is used, each CU is assumed to be the root of another quadtree-based structure called residual quadtree (RQT). Each leaf of the RQT is known as a transform unit (TU), which can take on sizes from 4×4 to 32×32 pixels (or up to the CU size), including non-square formats (NSQT) such as 4×16, 16×4, 8×32 and 32×8 for inter-predicted areas.

HE10 encoder configurations allow the use of asymmetric PUs in a CU (e.g., 32×8 PU and 32×24 PU into a single 32×32 CU), known as asymmetric motion partitions (AMP), and nonsquare TUs (4×16, 16×4, 8×32 and 32×8), known as nonsquare transforms (NSQT), for inter-predicted areas. On the other hand, the Main encoder configurations limit PUs to symmetric formats and TUs to square sizes.

The encoding mode decision algorithm selects the best coding tree configuration, the best PU division, and the best RQT structure based on exhaustive iterations of the rate-distortion optimization (RDO) process, which considers every encoding possibility and compares all of them in terms of bit rate and image quality. Therefore, both the data structures and the encoding process are responsible for an enormous share of the overall computational complexity. A temporal analysis of computational complexity incurred by the use of such encoding structures in HEVC and a complexity control technique were recently presented in [26].

Intra-frame prediction in HEVC is an extension of intra-frame prediction in H.264/AVC, supporting larger block sizes (from 4×4 to 64×64) and allowing more prediction directions: 16 different luma angular modes for 4×4 PUs, 2 luma angular modes for 64×64 PUs, and 33 luma angular modes for the other PU sizes. Besides these angular modes, a DC prediction mode similar to DC in H.264/AVC and a planar prediction mode designed to reconstruct smooth image segments through bilinear interpolation from the PU borders may also be applied to any PU. The number of chroma intra prediction modes was also increased to six in HEVC: direct mode (DM), linear mode (LM), vertical (mode 0), horizontal (mode 1), DC (mode 2), and planar (mode 3). DM is used when the texture directionality of luma and chroma is similar; LM is used when the sample intensities of luma and chroma are highly correlated. In the first case, the same mode used for luma prediction is used for chroma, whereas in the second case, the chroma components are predicted from downsampled reconstructed luma values of the same PU.

[1]High-efficiency 10 (HE10) and main configurations are referred to as high efficiency and low complexity, respectively, in the common test conditions document [25] and in the HEVC test model versions released before HM6.

Similarly to H.264/AVC, HEVC inter-frame prediction relies on block-based motion compensation (MC). Motion estimation (ME) is based on advanced motion vector prediction, which includes a motion vector competition scheme [27] along with a motion merge [28] technique. The former defines a set of motion vector predictor candidates comprising motion vectors of spatially and temporally neighboring PUs. The best motion vector predictor is selected from the set of candidates through RD optimization. Motion merging is based on a similar principle, but in this case the motion parameters are not explicitly transmitted. They are instead derived at the decoder based on the prediction mode and merge mode parameters.

Transform and quantization are heavily based on H.264/AVC, even though larger block sizes (ranging from 4×4 to 32×32) are supported. For inter-predicted PUs, a discrete cosine transform (DCT) is applied to the prediction residue, whereas for intra-predicted PUs, the encoder can choose between applying the DCT or a mode-dependent discrete sine transform (DST).

Regarding the entropy coding, the current HEVC draft supports only context-adaptive binary arithmetic coding (CABAC), which was already used in H.264/AVC. Even though the CABAC structure is basically the same as it was in H.264/AVC, HEVC aims at decreasing the computational burden associated with binarization and context retrieving by reducing the number of syntax elements to binarize and the number of contexts, and by increasing throughput by enabling parallel processing of transform coefficients [29].

The HEVC deblocking filter (DF) is also similar to that used in H.264/AVC. However, instead of performing deblocking at the boundaries of 4×4 blocks, the DF is applied to boundaries belonging to CUs, PUs, and TUs larger than 4×4 pixels. Vertical and horizontal edges are filtered according to a border filtering strength, which varies from 0 (no filtering) to 4 (maximum filtering strength) and also depends on the border characteristics. In addition to the DF, HEVC employs two new filters: the sample adaptive offset (SAO) and the adaptive loop filter (ALF). SAO processes the entire picture as a hierarchical quadtree, classifying the reconstructed pixels into different categories and then reducing distortion by adding an offset to the pixels in each category. This classification is performed taking into consideration the pixel intensity and edge properties. ALF is used after the DF and the SAO to reduce the distortion between the original and the reconstructed frame caused by lossy compression. The filtering is performed by applying a 2-D Wiener filter at CU level depending on the difference between values of samples in the original and reconstructed frames.

Internal bit depth increase may also be used in some encoder configurations in HEVC for increased calculation accuracy. In such cases, a 2-b extension is applied to each sample for additional calculation precision, which means that each sample is multiplied by 4 before further calculations.

IV. Experimental Setup and Methodology

As highlighted in the previous section, the HEVC encoder includes a large number of coding tools, each one having a different contribution to the overall coding efficiency and complexity. The operation of these tools and their functional modes are determined by encoding configuration parameters, which may be set to several different values.

TABLE II
Test Sequences Characteristics

Name	Frame Count	Frame Rate (Hz)	Bit Depth	Spatial Resolution
BlowingBubbles	150	50	8	416×240
RaceHorses	150	30	8	416×240
BasketballDrillText	150	50	8	832×480
PartyScene	150	50	8	832×480
BQMall	150	60	8	832×480
SlideShow	150	20	8	1280×720
vidyo1	150	60	8	1280×720
vidyo4	150	60	8	1280×720
ParkScene	150	24	8	1920×1080
BasketballDrive	150	50	8	1920×1080
NebutaFestival	150	60	10	2560×1600
Traffic	150	30	8	2560×1600

The experimental study presented in this article was carried out in two steps. Firstly, the coding tools that have stronger impact on both coding efficiency and computational complexity were identified. To that end, the individual contribution of each and every encoding tool to such performance metrics was experimentally evaluated. Secondly, those tools identified in the first step and sorted according to encoding gain per complexity increase were selected for further analysis, which consisted in evaluating the impact of each tool in a cumulative enabling order.

To conduct the experiments, a subset of 12 video sequences was selected from the 24 sequences specified in the Common Test Conditions document from JCT-VC [25]. These sequences differ broadly from one another in terms of frame rate, bit depth, motion, and texture characteristics as well as spatial resolution. Table II presents the name, frame count, frame rate, bit depth, and spatial resolution of each sequence.

The reference software used in all tests was the HEVC Model—version 7.0 (HM7) [30], compiled using Microsoft Visual Studio C++ Compiler. Even though HM7 is currently the most recent version of the HEVC model, HEVC has not yet been finalized and some tools are still evolving.[2] Since the core of the draft specification is not likely to undergo significant changes, most conclusions will remain valid, although specific results might not exactly match the future standard. All tests were performed on a clustered computer based on Intel Xeon E5520 (2.27 GHz) processors running on Windows Server 2008 HPC operating system. Computational complexity was measured by processing time as reported by the Intel VTune Amplifier XE 2011 software profiler [31].

A. Identification of Relevant Configuration Parameters

As revealed by experiments carried out in the scope of this paper and in previous ones conducted by the JCT-VC group, the impact of different HEVC tools on encoding efficiency and computational complexity is highly variable [32]. Since

[2]Three coding tools (NSQT, ALF, and LM chroma) were removed from the standard draft in the 10th JCT-VC meeting (July 2012), after conclusion of this paper.

TABLE III
ENCODER CONFIGURATIONS FOR IDENTIFYING RELEVANT PARAMETERS

Tool	Test Case (*TEST*)																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Inter 4 × 4	D	**E**	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Search Range	64	64	**128**	64	64	64	64	64	64	64	64	64	64	64	64	64	64
Bi-prediction Refinement[a]	4	4	4	**8**	4	4	4	4	4	4	4	4	4	4	4	4	4
Hadamard ME	D	D	D	D	**E**	D	D	D	D	D	D	D	D	D	D	D	D
Fast Encoding	E	E	E	E	E	**D**	E	E	E	E	E	E	E	E	E	E	E
Fast Merge Decision	E	E	E	E	E	E	**D**	E	E	E	E	E	E	E	E	E	E
Deblocking Filter	D	D	D	D	D	D	D	**E**	D	D	D	D	D	D	D	D	D
Internal Bit Depth	8	8	8	8	8	8	8	8	**10**	8	8	8	8	8	8	8	8
Sample Adaptive Offset	D	D	D	D	D	D	D	D	D	**E**	D	D	D	D	D	D	D
Adaptive Loop Filter	D	D	D	D	D	D	D	D	D	D	**E**	D	D	D	D	D	D
LM Intra Prediction	D	D	D	D	D	D	D	D	D	D	D	**E**	D	D	D	D	D
NSQT	D	D	D	D	D	D	D	D	D	D	D	D	**E**	D	D	D	D
AMP	D	D	D	D	D	D	D	D	D	D	D	D	D	**E**	D	D	D
Transform Skipping	E	E	E	E	E	E	E	E	E	E	E	E	E	E	**D**	**D**	E
Fast Transform Skipping	E	E	E	E	E	E	E	E	E	E	E	E	E	E	E	**D**	E
PCM Mode	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	**E**

[a] Search range increase for Motion Estimation when bi-prediction is used.

testing all possible combinations of coding tools and functional modes for all video sequences would produce a huge amount of data and unnecessary processing, a preliminary exploration was conducted to identify the most relevant configuration parameters to this paper. Such exploratory experiments were performed by varying one parameter at a time in the multidimensional encoder configuration parameter set, such that the impact of enabling each tool could be separately analyzed. Starting with a baseline encoder configuration, each predefined tool was enabled, one after the other, and the resulting image quality, bit rate, and encoding computational complexity were recorded for comparison with the reference baseline configuration. In every case, including the baseline configuration, the encoding structure was set to support CUs of up to 64 × 64 pixels, coding tree depths of up to four levels (i.e., minimum CU size is 4 × 4), and TUs varying from 4 × 4 to 32 × 32 pixels.

Table III shows all test cases corresponding to 17 encoding configurations. The baseline encoder configuration is defined as *TEST 1*, while the other 16 configurations correspond to *TEST 2* through *TEST 17*. The table lists the parameter values used in active coding tools, while **D** and **E** represent a disabled or enabled tool/functional mode, respectively. Although a larger number of tests were performed using a broader spectrum of configuration parameters, only the 16 most representative ones in terms of PSNR, bit rate, and computational complexity were selected and included in Table III. Every test was performed using five different quantization parameters (QP): 22, 27, 32, 37, and 42.

Fig. 1 shows the computational complexity results obtained from encoding the 12 video sequences according to the 17 test cases listed in Table III. In Fig. 1, the computational complexity values were normalized with respect to *TEST 1* (reference configuration). For all cases, except *TEST 3*, one can notice a close similarity between the trends of lines in Fig. 1, which means that complexity varies fairly likewise for all video sequences when a specific tool is enabled. *TEST 3*, which evaluates the effect of increasing the ME search range from 64 to 128, shows different complexity values for each

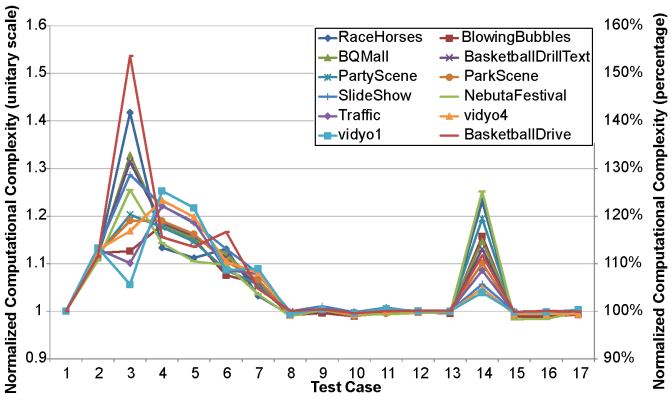

Fig. 1. Normalized computational complexity for encoding each video sequence under all 17 configurations (QP 32).

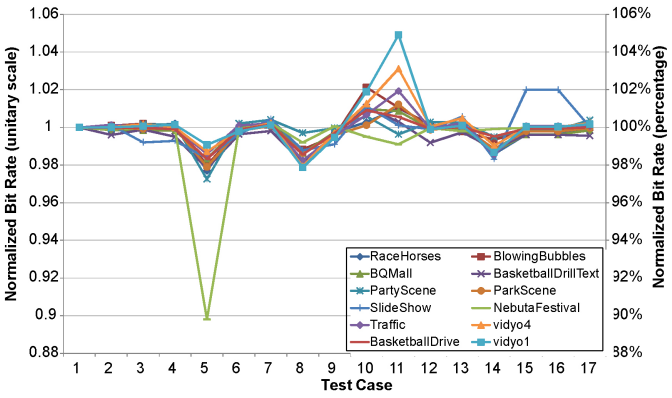

Fig. 2. Normalized bit rate for each video sequence encoded under all 17 configurations (QP 32).

sequence most likely due to their very different motion characteristics. It is quite evident that video sequences with large motion activity, such as *RaceHorses* and *BasketballDrive*, result in higher computational complexities than others with little or slow motion, such as *vidyo1*, *vidyo4*, and *Traffic*. The encoding efficiency, measured in terms of bit rate (also

TABLE IV
ENCODER CONFIGURATIONS USED FOR COMPLEXITY AND PERFORMANCE ANALYSIS

Tool	Configuration Case (*CFG*)															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Hadamard ME	D	**E**	E	E	E	E	E	E	E	E	E	E	E	E	E	**E**
Deblocking Filter	D	D	**E**	E	E	E	E	E	E	E	E	E	E	E	E	**E**
AMP	D	D	D	**E**	E	E	E	E	E	E	E	E	E	E	E	**E**
Search Range	64	64	64	64	**96**	**128**	128	128	128	128	128	128	128	128	128	**64**
Bi-prediction Refinement	4	4	4	4	4	4	**8**	8	8	8	8	8	8	8	8	**4**
Inter 4×4	D	D	D	D	D	D	D	**E**	E	E	E	E	E	E	E	**D**
Fast Encoding	E	E	E	E	E	E	E	E	**D**	D	D	D	D	D	D	**E**
Fast Merge Decision	E	E	E	E	E	E	E	E	E	**D**	D	D	D	D	D	**E**
SAO	D	D	D	D	D	D	D	D	D	D	**E**	E	E	E	E	**E**
ALF	D	D	D	D	D	D	D	D	D	D	D	**E**	E	E	E	**E**
Internal Bit Depth	8	8	8	8	8	8	8	8	8	8	8	8	**10**	10	10	**8**
LM Intra Prediction	D	D	D	D	D	D	D	D	D	D	D	D	D	**E**	E	**D**
NSQT	D	D	D	D	D	D	D	D	D	D	D	D	D	D	**E**	**D**

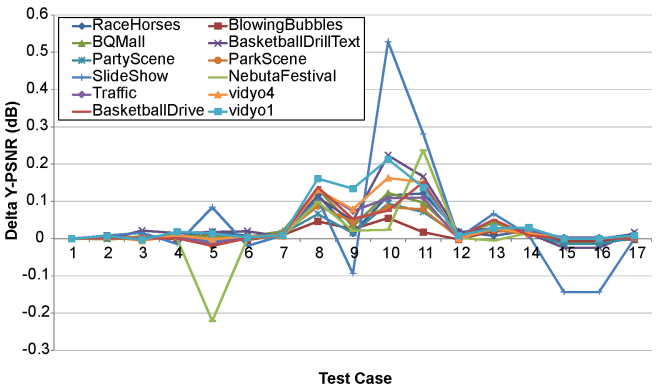

Fig. 3. Delta Y-PSNR for each video sequence encoded under all 17 configurations (QP 32).

normalized with respect to *TEST 1*) and delta Y-PSNR (using *TEST 1* as reference), was also evaluated for the same 12 video sequences and 17 test configurations. The results are shown in Figs. 2 and 3, respectively. Figs. 1–3 indicate that some encoder configurations do not influence significantly any of the three performance metrics. For example, choosing configuration *TEST 15* has a very small impact on the computational complexity and on bit rate savings in comparison to *TEST 1*, leading to a slight decrease in PSNR for most video sequences.

Based on these results, those coding tools that were shown to have the largest impact on performance and complexity were selected as the basis for the second step of the complexity analysis process, described in the next subsection.

B. Relevant Encoding Configurations

Most studies on complexity analysis of video encoders focus on testing each feature independently, by comparing the performance of a baseline configuration against the same baseline configuration with only one tool enabled at a time. However, current video coding algorithms are characterized by high levels of interdependency between coding tools, which means that any additional encoding gain obtained by enabling a particular coding option may be dependent on the enabled/disabled status of other coding tools. Since this is the case of HEVC, the complexity analysis performed in the next sections is based on a sequence of predefined encoder configurations where the coding tools are enabled in a cumulative order along such sequence. The sequence of relevant encoding configurations was constructed in two steps as follows.

In the first step, 13 configurations from Table III were identified as those having significant impact on PSNR, bit rate and overall computational complexity in comparison to the baseline test case (*TEST 1*). Here, significant impact was defined as PSNR variations of at least 0.1 dB, bit rate changes of at least 1.5%, and computational complexity increases of at least 5%. These 13 configurations correspond to seven coding tools (ME, DF, SAO, ALF, Internal Bit Depth, LM Intra Prediction, and NSQT), which were then used in the second step, along with several different parameters, to create a sequence of 15 relevant encoding configurations.

In the second step, the 15 configurations (*CFG 1* to *CFG 15*) presented in Table IV were created by defining the parameters for each of the seven above-mentioned tools, in a cumulative sequence. An additional configuration (*CFG 16*) was defined by choosing optimal tools and parameters as shall be explained in detail in the next section. The baseline configuration (*CFG 1*) is the same as *TEST 1* in Table III. Configurations 2, 4, 5, 6, 7, 8, 9, and 10 correspond to different functional modes of ME/MC; configurations 3, 11, and 12 vary filtering operations, enabling the DF, SAO, and ALF, respectively; *CFG 13* increases the internal bit depth from 8 to 10 bits; *CFG 14* enables the linear intra prediction mode (LM); and *CFG 15* enables non-square transforms (NSQT). As the tools were enabled in a cumulative order, *CFG 1* represents the simplest encoder configuration, while *CFG 15* is the most complex one with all coding tools enabled.

The enabling order of the tools, presented in Table IV, was defined based on the ratio between the bit rate reduction and the increase in encoding computational complexity associated with each tool. The ratio between the PSNR decrease and the complexity increase was also considered a tiebreak whenever the bit rate-complexity ratio was very similar between two tools. Those tools that presented the highest relative

TABLE V
COMPUTATIONAL COMPLEXITIES (IN SECONDS) FOR HEVC AND H.264/AVC CONFIGURATIONS UNDER QPS 22, 27, 32, 37, AND 42

Video (resolution)	Configuration	QP 22	QP 27	QP 32	QP 37	QP 42	Average Increase (%)
RaceHorses (416 × 240)	HEVC (1)	839.4	739.6	644.8	575.7	490.2	18.6
	HEVC (15)	2292.5	1969.2	1827.8	1599.3	1305.5	224.5
	H.264 HP	692.7	602.9	534.1	483.7	447.1	–
BasketballDrillText (832 × 480)	HEVC (1)	2493.0	2226.0	2023.0	1869.6	1627.6	9.1
	HEVC (15)	6365.8	5767.6	5197.3	4724.1	4084.5	178.4
	H.264 HP	2197.1	1903.8	1845.4	1741.7	1656.4	–
SlideShow (1280 × 720)	HEVC (1)	4264.1	4088.1	3939.9	3822.7	3541.3	46.7
	HEVC (15)	11550.7	10657.1	10092.5	9580.6	8663.0	276.7
	H.264 HP	2857.3	2717.8	2638.6	2610.1	2566.7	–
ParkScene (1920 × 1080)	HEVC (1)	12778.3	11199.3	10299.2	9676.0	8683.6	15.1
	HEVC (15)	29249.5	25798.4	23283.0	21463.1	19133.7	160.0
	H.264 HP	10310.3	8743.0	8384.7	9039.9	9293.4	–
NebutaFestival (2560 × 1600)	HEVC (1)	59417.6	57811.3	18065.4	49361.1	38294.9	103.6
	HEVC (15)	190443.0	182055.4	39127.9	142355.3	111125.9	502.2
	H.264 HP	27331.6	24146.9	21131.8	17903.1	19153.8	–

coding efficiency-complexity gains were enabled before the others, since they should have higher priority of activation in complexity-constrained video coding systems. In Figs. 1–3, it is noticeable that *TEST 8* (enabling Hadamard in fractional pixel ME) reduced the bit rate in a range from 0.3% to 2.2% (see Fig. 2) and increased the image quality in a range from 0.04 to 0.16 dB (see Fig. 3) in comparison to *TEST 1*, while the computational complexity was roughly unchanged (see Fig. 1). Since this was found as the case in which the highest bit rate savings and PSNR gains are obtained in regard to the computational complexity increase, the Hadamard ME tool is the first one to be enabled in the sequence of configurations in Table IV (*CFG 2*).

The set of 16 configurations presented in Table IV was used to encode all the test sequences listed in Table II. For each simulation, bit rate, Y-PSNR, and complexity results were recorded in order to allow the performance and complexity tradeoff analysis presented in the next section, which evaluates the activation of each HEVC tool/functional mode.

V. HEVC ENCODING PERFORMANCE AND COMPLEXITY TRADEOFF ANALYSIS

This section presents the performance results in terms of PSNR, bit rate, and computational complexity for the 16 test cases listed in Table IV. The results are summarized in Figs. 4–8 and Table V.

The computational complexity results for all video sequences under all encoding configurations are plotted in Fig. 4, normalized with respect to those of *CFG 1*, as shown in Fig. 1. In Fig. 4, all video sequences exhibit a similar monotonic increase of the encoding complexity from *CFG 1* to *CFG 15*. However, from *CFG 10* to *CFG 15*, the slope is very small and the normalized computational complexity is roughly constant. The largest computational complexity increases are observed in the transitions from *CFG 4* to *CFG 9*. As shown in Table IV, these configurations correspond to different functional modes of ME/MC according to the specified parameters. The results in Fig. 4 show that the choice of more accurate ME modes

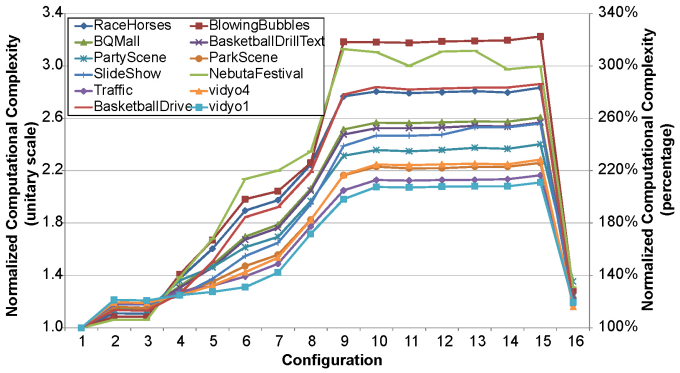

Fig. 4. Normalized computational complexity for encoding each video sequence under all 16 configurations (QP 32).

is accountable for most of the computational complexity increases observed in the results.

It is also noticeable that from *CFG 5* and especially from *CFG 6* onwards, the curves pertaining to the different video sequences start to be farther apart from each other. This happens due to the fact that these two configurations increase the ME search range from 64 to 96 and from 96 to 128, respectively. As explained in Section IV in regard to *TEST 3* in Fig. 1, the video sequences are quite distinct in terms of motion activity, and for this reason, they result in different encoder behavior when the motion estimation search range is increased. The spreading of the curves is even more pronounced in the case of the configurations with rank order above *CFG 6* due to the cumulative effect of the tool activations. This is observed in the last configuration, *CFG 15*, for which the computational complexity is up to 3.2 times larger than that of the baseline configuration. Even though Fig. 4 presents results for QP 32, other QP values were also tested and have shown similar behavior.

Table V presents absolute complexity values for the least and most complex HM configurations (*CFG 1* and *CFG 15*, respectively) encoded with QPs 22, 27, 32, 37, and 42. As a reference for comparison, the complexity obtained by encoding the same video sequences with an H.264/AVC High Profile encoder (JM software, version 18.3) is also presented

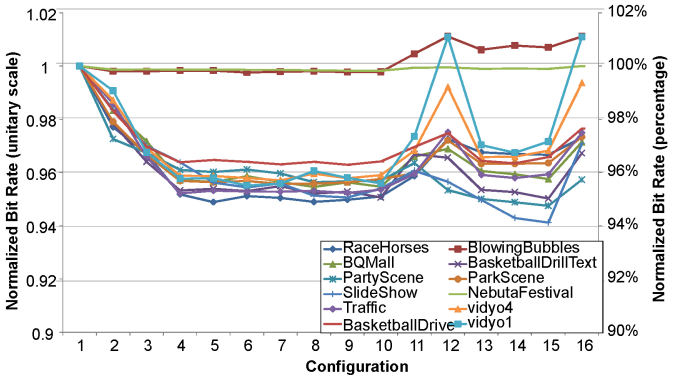

Fig. 5. Normalized bit rate for each video sequence encoded under all 16 configurations (QP 32).

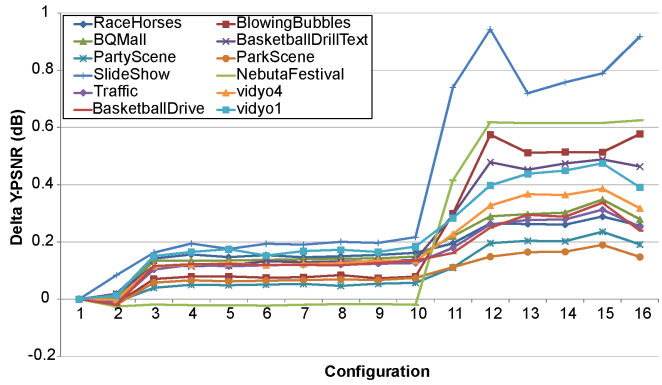

Fig. 6. Delta Y-PSNR for each video sequence encoded under all 16 configurations (QP 32).

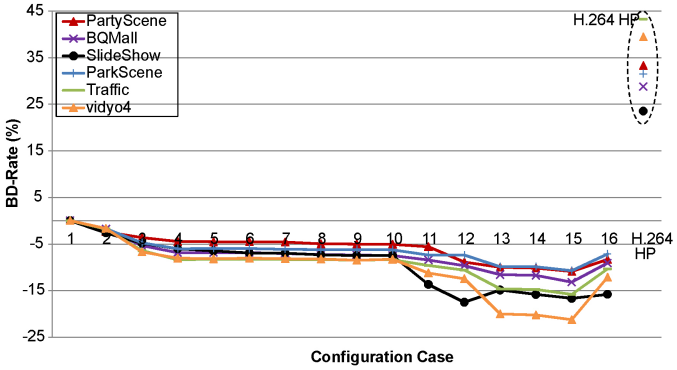

Fig. 7. BD-rate values for each configuration using configuration 1 as reference.

(*H.264 HP* lines). For brevity, Table V only shows the results for the sequences *RaceHorses*, *BasketballDrillText*, *SlideShow*, *ParkScene*, and *NebutaFestival*. The rightmost column shows the average computational complexity increase of *CFG 1* and *CFG 15* over the H.264/AVC HP encoder. It is noteworthy that even the least complex configuration in HEVC is still more complex than H.264/AVC HP (in a range from 9.1% up to 103.6%). Moreover, *CFG 15* is at least 1.6 times more complex than H.264/AVC HP, reaching an average computational complexity increase of up to 502.2% in the *NebutaFestival* video sequence.

Normalized bit rate results are presented in Fig. 5. It is clear that *CFG 2*, *CFG 3*, and *CFG 4*, which activate, respectively, the Hadamard transform in fractional pixel ME, the deblocking filter, and the use of asymmetric motion partitions, are responsible for the largest bit rate savings. On the other hand, the bit rate remains almost unchanged from *CFG 5* to *CFG 10* (different ME/MC configurations) and from *CFG 14* to *CFG 15* (LM intra mode and non-square transforms, respectively), showing however a considerable increase for configurations *CFG 11* and *CFG 12*, which enable SAO and ALF, respectively. These results imply that when considering only computational complexity increases and bit rate decreases, configurations *CFG 2-CFG 4* would be the most favorable ones, since they produce the highest bit rate reductions at the smallest costs in terms of computational complexity.

Fig. 6 shows the increase of Y-PSNR for each encoding configuration in comparison to the reference one. It is possible to notice that the image quality is roughly maintained from *CFG 4* to *CFG 10* and from *CFG 13* to *CFG 15*. Most of the image quality gains are obtained in *CFG 3*, *CFG 11*, and *CFG 12*, which enable the deblocking filter, the SAO filter, and the adaptive loop filter, respectively. These results lead to the conclusion that the three filters have significant impact on the objective image quality in HEVC. When the three filters are enabled (*CFG 12*), the PSNR is increased by up to 0.94 dB (*SlideShow* sequence). Since subjective quality does not always follow PSNR, a detailed subjective visual analysis of the video sequences encoded under different configurations was conducted, which revealed that the image quality was in fact enhanced when the three filters are enabled. Although this is not supported by formal subjective testing, the trend observed in PSNR was confirmed by different expert users.

Further results for a wider range of QP values (22, 27, 32, 37, and 42) are presented in Figs. 7 and 8 in terms of Bjøntegaard Delta rate (BD-rate) and Bjøntegaard Delta PSNR (BD-PSNR) [33], respectively. Six video sequences were used taking *CFG 1* as the reference. Both figures reveal large differences between H.264/AVC HP (non-connected points on the right side of these figures) and the remaining HM encoding configurations in terms of encoding efficiency. All HM configurations result in a significant decrease in the BD-rate value when more tools are added to the encoder, as shown in Fig. 7. The H.264/AVC HP configuration, however, leads to a BD-rate increase that varies from 23.5% to 43.3%, depending on the video sequence, in comparison to *CFG 1*. Similarly, BD-PSNR in Fig. 8 is improved when more tools are added to the baseline HM configuration. On the other hand, the H.264/AVC HP configuration presents BD-PSNR degradations in a range from 1.3 dB to 2.1 dB in comparison to HM *CFG 1*.

A detailed analysis of Figs. 4–8 provides relevant insight to select the best tradeoff between computational complexity and compression efficiency in HEVC. For example from *CFG 4* to *CFG 10* in Fig. 4, it is clear that even though the computational complexity increases when most of the functional modes are enabled in ME the compression rates and the Y-PSNR do not change significantly (see Figs. 5 and 6, respectively). This is confirmed by the BD-rate and BD-PSNR data in Fig. 7 and Fig. 8, which are fairly constant around these points. This leads to the relevant conclusion that the encoding efficiency

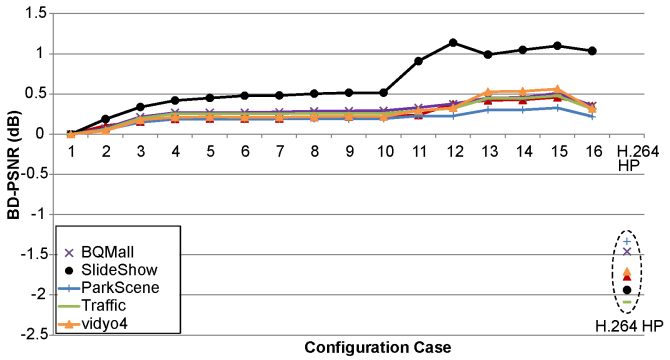

Fig. 8. BD-PSNR values for each encoding configuration using configuration 1 as reference.

saturates at some point and that these functional modes should not be enabled in a complexity-constrained encoder.

Another relevant conclusion is that the Hadamard ME, the asymmetric motion partitions and the filters (Deblocking Filter, Sample Adaptive Offset and Adaptive Loop Filter) should be first enabled in a complexity-constrained encoder, since they significantly increase the encoding efficiency at the cost of a low complexity increase, as previously shown. In fact, most of the gains in the BD-PSNR and BD-rate curves (Figs. 7 and 8) are accounted for by the tools enabled in *CFG 1-CFG 3* and from *CFG 11* onwards.

The evidence provided above can be taken into account when selecting the tools and functional modes of an efficient HEVC encoder configuration, in order to achieve high encoding performance under low computational complexity levels.

To exemplify how the proposed complexity versus performance analysis can be used for tuning a HEVC encoder, an optimized configuration was created. This configuration, which appears in Table IV labeled as *CFG 16,* does not use any tool or functional mode that is not profitable from the point of view of complexity versus performance. More specifically in *CFG 16,* the Hadamard ME, the filters, and the use of AMP are enabled, whil the remaining ME functional modes, the internal bit depth increase, the LM intra mode, and the use of NSQT were all either disabled or enabled in a low-complexity functional mode, as shown in the last column of Table IV. The results in Figs. 4–8 show that, even though the encoding efficiency of *CFG 16* is similar to that achieved by high-complexity configurations, its computational complexity is smaller. For instance, *CFG 16* provides roughly the same coding efficiency as *CFG 12* but its complexity is up to 2.5 times smaller (see Fig. 4). This optimization exercise shows that a wise selection of coding parameters can be used to define a low-complexity configuration that is capable of achieving roughly the same coding efficiency as a more complex one. In other words, higher levels of coding efficiency are not necessarily obtained using the most complex coding configurations.

VI. Performance and Complexity as a Function of QP

This section, presents results and analysis of coding performance and complexity for different bit rates. In order to decouple the results from the effects of rate control algorithms, a set of fixed quantization parameter (QP) values (22, 27, 32, 37, and 42) was used in each experiment.

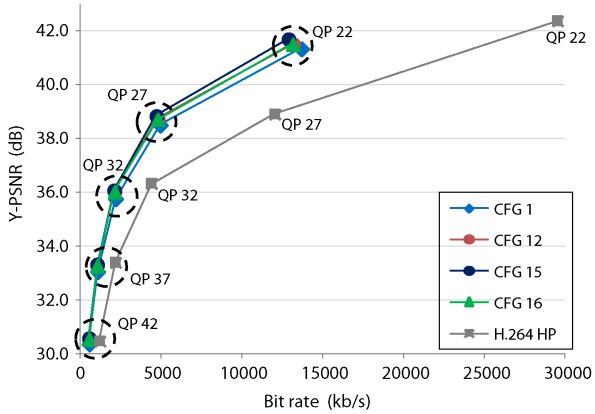

Fig. 9. Rate-distortion efficiency of HEVC under configurations 1, 12, 15, and 16 and H.264/AVC high profile for the *Traffic* video sequence (QPs 22, 27, 32, 37, and 42).

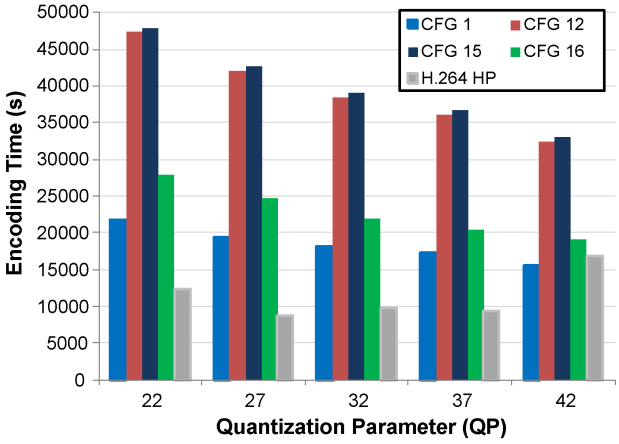

Fig. 10. Encoding time for HEVC under configurations 1, 12, 15, and 16 and H.264 HP (*Traffic* video sequence) for QPs 22, 27, 32, 37, and 42.

Fig. 9 shows the rate-distortion encoding efficiency of the HM encoder under different configurations for the *Traffic* video sequence. The H.264/AVC HP encoder was also included in this evaluation to provide an anchor for comparison. Even though all HM configurations presented in Section V were analyzed, only four of them (1, 12, 15 and 16) are presented in Fig. 9 for clarity. The results confirm that the encoding efficiency of HM configurations is much higher than the encoding efficiency of H.264/AVC HP, since the bit rates are reduced by 50% while maintaining roughly the same Y-PSNR. As *CFG 1* and *CFG 15* are the configurations which present the worst and the best rate-distortion performance results, respectively, the performance curves for the remaining HM configurations (omitted from Fig. 9) would fall between those of *CFG 1* and *CFG 15*. The curves for *CFG 12* and *CFG 16* confirm that the rate-distortion efficiency of the latter is almost as high as that of the former, i.e., they almost overlap in the figure.

Fig. 10 shows the encoding time as a function of QP for the same video sequence and the same configurations presented in Fig. 9. Fig. 10 shows that, although *CFG 16*

achieves rate-distortion efficiency close to that of *CFG 12* in Fig. 9, its computational complexity is much more similar to that of the baseline configuration (*CFG 1*) for all QPs. In the worst case (QP 37), the PSNR reduction obtained in *CFG 16* is below 0.02 dB when compared to *CFG 12* while in the remaining cases the reduction is below 0.009 dB. Regarding the bit rate, no significant differences are noticed when comparing *CFG 16* to *CFG 12*, apart from the fact that, for QP values below 37, there is a slight reduction of bit rate when *CFG 16* is used instead of *CFG 12*. Therefore, the previous conclusion for a single QP can be extended for a wide range of bit rates, meaning that coding complexity and efficiency are not necessarily correlated. Thus, in constrained-complexity encoder implementations it is worthwhile to take these findings into account.

Concerning the effect of QP on the encoding complexity, it was further observed that the overall complexity decreases slightly when the QP increases. This effect is more prominent in the cases with larger complexity values that allow such difference to be more easily noticed (*CFG 12* and *CFG 15*). This effect might result from the smaller amount of processed data in the case of higher QP, e.g., more zero coefficients.

VII. Conclusion

The tradeoff between computational complexity and encoding performance of the HEVC encoder was evaluated in this paper using a broad range of encoding configuration cases over a wide variety of video contents.

The experimental study analysis performed in this paper shows that maximum HEVC complexity can be reduced at practically no coding efficiency cost, if the coding tools are wisely combined and configured. The results show that when the number of tools and functional modes increase in a cumulative progression, the computational complexity grows in a similar way, even though the encoding performance does not increase at the same pace. Therefore, it is advisable to enable first those tools that provide most gain for the least cost. Such a strategy for enabling tools and choosing encoding parameter values leads to the best tradeoff between computational complexity and encoding efficiency, making it possible to achieve high encoding performance while still reducing the computational complexity in comparison to the case in which all tools are enabled.

This paper also demonstrated that high coding efficiency at low computational complexity can be achieved by enabling the Hadamard ME, the asymmetric motion partitions, and the filters (loop filter, sample adaptive offset, and adaptive loop filter) instead of enhancing the performance of other computationally demanding tools, such as motion estimation. Despite the known fact that subjective quality does not always match the objective metrics used in this paper, this paper has the merit of revealing the behavior of High Efficiency Video Coding tools regarding objective quality, bit rate, and complexity. The experimental study presented in this paper provides useful information for the definition of profiles in the HEVC standard. These results and insights might also be valuable to help tune an encoder configuration to a given complexity budget and to optimize encoding complexity control solutions.

References

[1] *Joint Collaborative Team on Video Coding (JCT-VC)* [Online]. Available: http://www.itu.int/en/ITU-T/studygroups/com16/video/Pages/jctvc.aspx

[2] A. Puri, "Video coding using the H.264/MPEG-4 AVC compression standard," *Elsevier Signal Process. Image Commun.*, vol. 1, pp. 793–849, Oct. 2004.

[3] *Joint Call for Proposals on Video Compression Technology*, VCEG-AM91, ISO/IEC-JTC1/SC29/WG11, Video Coding Experts Group (VCEG), 39th Meeting, Kyoto, Japan, 2010.

[4] *Comparison of Compression Performance of HEVC Working Draft 4 With AVC High Profile*, JCTVC-G399, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2011.

[5] L. Younghoon, K. Jungsoo, and K. Chong-Min, "Energy-aware video encoding for image quality improvement in battery-operated surveillance camera," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 2, pp. 310–318, Feb. 2012.

[6] W. Kim, J. You, and J. Jeong, "Complexity control strategy for real-time H.264/AVC encoder," *IEEE Trans. Consumer Electron.*, vol. 56, no. 2, pp. 1137–1143, May 2010.

[7] D. N. Kwon, P. F. Driessen, A. Basso, and P. Agathoklis, "Performance and computational complexity optimization in configurable hybrid video coding system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, no. 1, pp. 31–42, Jan. 2006.

[8] O. Lehtoranta and T. D. Hamalainen, "Complexity analysis of spatially scalable MPEG-4 encoder," in *Proc. Int. Symp. System-on-Chip*, 2003, pp. 57–60.

[9] J. Ostermann, *et al.*, "Video coding with H.264/AVC: Tools, performance, and complexity," *IEEE Circuits Syst. Mag.*, vol. 4, no. 1, pp. 7–28, Jan.–Mar. 2004.

[10] S. Saponara, K. Denolf, G. Lafruit, C. Blanch, and J. Bormans, "Performance and complexity co-evaluation of the advanced video coding standard for cost-effective multimedia communications," *EURASIP J. Appl. Signal Process.*, vol. 2004, pp. 220–235, Jan. 2004.

[11] P. Li, Y. Chen, and W. Ji, "Rate-distortion-complexity analysis on AVS encoder," presented at the *11th Pacific Rim Conf. Multimedia*, Shanghai, China, 2011.

[12] A. Hallapuro, V. Lappalainen, and T. D. Hamalainen, "Performance analysis of low bit rate H.26L video encoder," presented at the *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 2. 2001.

[13] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 704–716, Jul. 2003.

[14] L. Szu-Wei and C. C. J. Kuo, "Complexity modeling of spatial and temporal compensations in H.264/AVC decoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 5, pp. 706–720, May 2010.

[15] M. Zhan, H. Hao, and W. Yao, "On complexity modeling of H.264/AVC video decoding and its application for energy efficient decoding," *IEEE Trans. Multimedia*, vol. 13, no. 6, pp. 1240–1255, Dec. 2011.

[16] V. Lappalainen, A. Hallapuro, and T. D. Hamalainen, "Complexity of optimized H.26L video decoder implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 717–725, Jul. 2003.

[17] *On Software Complexity*, JCTVC-G757, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2011.

[18] *HM Decoder Complexity Assessment on ARM*, JCTVC-G262, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2011.

[19] Z. Chang-Guo *et al.*, "MPEG video decoding with the UltraSPARC visual instruction set," in *Proc. Compcon '95: Technol. Inform. Superhighway Dig. Papers*, 1995, pp. 470–477.

[20] L. Xiang, M. Wien, and J. R. Ohm, "Rate-complexity-distortion optimization for hybrid video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 957–970, Jul. 2011.

[21] *PTscalar V1.0* [Online]. Available: http://eda.ee.ucla.edu/PTscalar/

[22] *OProfile: A System Profiler for Linux* [Online]. Available: http://oprofile.sourceforge.net/news/

[23] *PAPI* [Online]. Available: http://icl.cs.utk.edu/papi/

[24] *High Efficiency Video Coding (HEVC) Test Model 7 (HM 7) Encoder Description*, JCTVC-I1002, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2012.

[25] *Common Test Conditions and Software Reference Configurations*, JCTVC-I1100, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2012.

[26] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Complexity control of high efficiency video encoders for power-constrained devices," *IEEE Trans. Consumer Electron.*, vol. 57, no. 4, pp. 1866–1874, Nov. 2011.
[27] G. Laroche, J. Jung, and B. Pesquet-Popescu, "RD optimized coding for motion vector predictor selection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 9, pp. 1247–1257, Sep. 2008.
[28] S. Oudin *et al.*, "Block merging for quadtree-based video coding," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2011, pp. 1–6.
[29] V. Sze and M. Budagavi, "Parallelization of CABAC transform coefficient coding for HEVC," in *Proc. PCS*, 2012, pp. 509–512.
[30] *HM7: High Efficiency Video Coding (HEVC) Test Model 7 Encoder Description*, JCTVC-I1002, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2012.
[31] *VTune^{TM} Amplifier XE 2011 From Intel* [Online]. Available: http://software.intel.com/en-us/articles/intel-vtune-amplifier-xe/
[32] *JCT-VC AHG Report: Complexity Assessment (AHG 12)*, JCTVC-G012, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2011.
[33] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," Austin, TX, 2001.

Guilherme Corrêa (S'08) received the B.S. degree in computer science from the Federal University of Pelotas, Pelotas, Brazil, in 2009, and the M.S. degree in computer science from the Federal University of Rio Grande do Sul, Porto Alegre, Brazil, in 2010. Currently, he is pursuing the Ph.D. degree in electrical and computer engineering with the University of Coimbra, Coimbra, Portugal.

He is currently a Research Assistant with the Institute for Telecommunications, Coimbra, where he works on video coding algorithms and digital systems design.

Pedro A. Assunção (M'98) received the Licenciado and M.Sc. degrees in electrical engineering from the University of Coimbra, Coimbra, Portugal, in 1988 and 1993, respectively, and the Ph.D. degree in electronic systems engineering from the University of Essex, Essex, U.K., in 1998.

He is currently a Professor of electrical engineering and multimedia communication systems with the Polytechnic Institute of Leiria, Leiria, Portugal, and is a Researcher with the Institute for Telecommunications, Coimbra, Portugal. He is an author or co-author of more than seventy scientific and technical papers in conferences and journals, and three book chapters. He holds three U.S. patents. His current research interests include 2-D and 3-D video coding, adaptation to diverse networking and user environments, multiple description coding, power-aware video coding, audiovisual error concealment, and perceptual quality evaluation.

Luciano Volcan Agostini (M'06–SM'11) received the B.S. degree in computer science from the Federal University of Pelotas, Pelotas, Brazil, in 1998, and the M.Sc. and Ph.D. degrees from the Federal University of Rio Grande do Sul, RS, Brazil, in 2002 and 2007, respectively.

Since 2002, he has been a Professor with the Center of Technological Development, Federal University of Pelotas, where he leads the Architectures and Integrated Circuits Group. He has more than 100 published papers in journals and conference proceedings. His current research interests include video coding, arithmetic circuits, FPGA based design, and microelectronics.

Dr. Agostini is a Member of the IEEE Circuits and System Society, the IEEE Signal Processing Society, the IEEE Consumer Electronics Society, the Brazilian Computer Society, and the Brazilian Microelectronics Society.

Luis A. da Silva Cruz (M'11) received the Licenciado and M.Sc. degrees in electrical engineering from the University of Coimbra, Coimbra, Portugal, in 1989 and 1993, respectively, and the M.Sc. degree in mathematics and the Ph.D. degree in electrical computer and systems engineering from the Rensselaer Polytechnic Institute, Troy, NY, in 1997 and 2000, respectively.

He has been with the Department of Electrical and Computer Engineering, University of Coimbra since 1990, first as a Teaching Assistant and then as an Assistant Professor since 2000. He is currently a Researcher with the Institute for Telecommunications of Coimbra, Coimbra, where he works on video processing and coding, and wireless communications.