Fast HEVC Encoding Decisions Using Data Mining

Guilherme Correa, *Member, IEEE*, Pedro A. Assuncao, *Senior Member, IEEE*,
Luciano Volcan Agostini, *Senior Member, IEEE*,
and Luis A. da Silva Cruz, *Member, IEEE*

***Abstract*— The High Efficiency Video Coding standard provides improved compression ratio in comparison with its predecessors at the cost of large increases in the encoding computational complexity. An important share of this increase is due to the new flexible partitioning structures, namely the coding trees, the prediction units, and the residual quadtrees, with the best configurations decided through an exhaustive rate-distortion optimization (RDO) process. In this paper, we propose a set of procedures for deciding whether the partition structure optimization algorithm should be terminated early or run to the end of an exhaustive search for the best configuration. The proposed schemes are based on decision trees obtained through data mining techniques. By extracting intermediate data, such as encoding variables from a training set of video sequences, three sets of decision trees are built and implemented to avoid running the RDO algorithm to its full extent. When separately implemented, these schemes achieve average computational complexity reductions (CCRs) of up to 50% at a negligible cost of 0.56% in terms of Bjontegaard Delta (BD) rate increase. When the schemes are jointly implemented, an average CCR of up to 65% is achieved, with a small BD-rate increase of 1.36%. Extensive experiments and comparisons with similar works demonstrate that the proposed early termination schemes achieve the best rate-distortion-complexity tradeoffs among all the compared works.**

***Index Terms*— Computational complexity, data mining (DM), decision trees, early termination, High Efficiency Video Coding (HEVC).**

I. Introduction

THE *High Efficiency Video Coding* (HEVC) standard has been recently launched by the *Joint Collaborative Team on Video Coding* (JCT-VC), a joint activity of *ITU-T Video Coding Experts Group* and *ISO/IEC Moving Picture Experts Group* [1], [2]. The standard became the state of the art in video compression and is expected to gradually substitute its predecessor, the H.264/AVC standard.

When compared with H.264/Advanced Video Coding (AVC) *High* profile (HP), HEVC *Main* profile reduces bit rates in 40% on average, maintaining a similar objective image quality [3]. However, the improved compression efficiency of HEVC is obtained at the expense of a significant increase in computational complexity, mainly resulting from much more intensive processing tools, nested partitioning structures and optimization algorithms dealing with larger amounts of data [4]. According to [5], the encoding computational complexity of HEVC is higher than H.264/AVC HP from 9% to 502%, depending on the *HEVC test Model* (HM) [6] configuration used. Such wide variation in computational complexity is due to the high number of optional coding tools that can be used within the standard compliance bounds. In [3], the computational complexity of HEVC is said to be 40% higher than that of H.264/AVC HP when only the essential coding tools are enabled.

The introduction of more flexible partitioning structures has been claimed as the main responsible for the compression gains achieved by HEVC, but has also been charged as the standard's most computationally demanding inclusion in recent works that analyze the encoder complexity [5]–[7]. To achieve the best rate-distortion (R-D) performance, the HM encoder employs an exhaustive search process that tests every possible combination of partitioning structures and chooses the one which results in the lowest R-D cost [8]. Although the most recent HM versions include some speedup features, this process still greatly increases the encoder's computational complexity, limiting its use in computationally and energy-constrained environments.

Several authors have recently proposed heuristics for simplifying the partitioning decision for *coding units* (CUs), *prediction units* (PUs), and *transform units* (TUs), aiming at decreasing the computational complexity of HEVC without hurting its compression efficiency [9]–[22]. Even though these works succeed in their goal of decreasing the encoding complexity up to a certain extent, most of them only deal with a single type of partitioning structures (either CUs [9]–[13], PUs [14]–[17], or TUs [21], [22] separately), which limits the achievable complexity reductions. Besides, the majority of them result in nonnegligible losses in R-D efficiency.

To minimize R-D efficiency losses, intelligent approaches that apply machine learning techniques to extract and

Manuscript received May 6, 2014; revised July 15, 2014 and September 30, 2014; accepted October 1, 2014. Date of publication October 16, 2014; date of current version April 2, 2015. This work was supported in part by the Brazilian Agency for Scientific and Technological Development, Brazil, in part by the National Council for the Improvement of Higher Education, Brazil, in part by the Foundation for Science and Technology under Project FCT/1909/27/2/2014/S, and in part by the Instituto de Telecomunicações. This paper was recommended by Associate Editor R. C. Lancini.

G. Correa and L. A. da Silva Cruz are with the Department of Electrical and Computer Engineering, Faculty of Sciences and Technology, University of Coimbra, Coimbra 3004-531, Portugal, and also with Instituto de Telecomunicações, Polo II-Universidade de Coimbra, Coimbra 3030-290, Portugal (e-mail: guilherme.correa@co.it.pt; luis.cruz@co.it.pt).

P. Assuncao is with Polytechnic Institute of Leiria, Leiria 2411-901, Portugal, and also with Instituto de Telecomunicações, Morro do Lena-Alto do Vieiro, Leiria 2411-901, Portugal (e-mail: amado@co.it.pt).

L. V. Agostini is with the Group of Architectures and Integrated Circuits, Federal University of Pelotas, Pelotas 96160-000, 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.2014.2363753

analyze image characteristics and intermediate encoding results have been proposed by some authors, especially for transcoding [23], [24], transrating [25], and even computational complexity reduction (CCR) [26]–[28]. However, such works are still rare and do not achieve expressive complexity reductions, since they have not been fully developed and applied to multiple types of partitioning structures.

The method proposed in this paper uses *data mining* (DM) as a tool to build a set of decision trees that allow terminating early the decision processes that find the best configurations for CUs, PUs, and TUs, relieving the encoder of the complex task of testing all encoding structure partitioning possibilities. Through the use of such trees, the partitioning structure decision is seen as a data classification problem, which is efficiently solved by a set of tests. The decision trees designed using DM were implemented in the HM encoder and lead to CCRs that range from 46% to 88% at the cost of an average bit rate increase of 1.35%. These results outperform previous works both in terms of complexity savings and compression efficiency, showing that the method is an effective solution for reducing the computational complexity of HEVC encoders.

The rest of this paper is organized as follows. Section II presents an overview of the HEVC partitioning structures and an analysis of their R-D-complexity (R-D-C) efficiency. Section III presents a review of techniques recently proposed for reducing the video coding computational complexity. The methodology used in this paper to build the decision trees and the proposed early terminations is presented in Section IV. Section V presents and discusses experimental results. Section VI presents comparisons with previous works, and Section VII concludes this paper.

II. HEVC Partitioning Structures

In HEVC, each frame may be divided into a set of *slices*, which are parts of the frame that can be independently decoded. A slice is composed of equal-sized *coding tree units* (CTUs), which are composed of one luminance *coding tree block* (CTB) and two chrominance CTBs. As the same partitioning structure decisions are applied to both the luminance and chrominance CTBs, from now on this paper will only refer to CTUs and their subdivisions.

A. Coding Units, Prediction Units, and Residual Quadtrees

Each CTU can be divided into smaller blocks, called CUs, following a recursive quadtree structure, the *coding tree*. The *largest CU* (LCU) size and the *smallest CU* (SCU) size allowed in HEVC are 64 × 64 and 8 × 8, respectively, so that up to four *coding tree* depths are possible. In the HM encoder, the *coding tree* structure is defined by an iterative splitting process, which evaluates all possibilities in a R-D optimization (RDO) scheme, until the SCU depth is reached. Fig. 1 shows an example of a 64 × 64 CTU divided into several CUs, where the tree leaves (gray blocks) are the final CUs encoded.

For intra- and inter-frame prediction, each CU may be divided into two or four PUs, which are separately predicted.

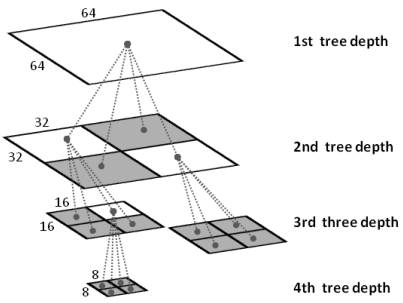

Fig. 1. *Coding tree* structure of a CTU divided into CUs.

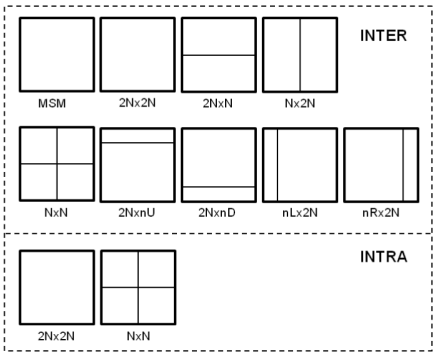

Fig. 2. Inter-frame and intra-frame PU splitting modes available in HEVC.

All PUs in a CU are predicted with either inter-frame or intra-frame prediction. Fig. 2 shows all possible PU partitioning modes that can be used in a CU. From now on, these partitioning possibilities will be called PU splitting modes to distinguish them from the prediction modes.

In HM, the best PU splitting mode is chosen through the RDO process, which evaluates all possibilities. The number of possible PU splitting modes varies according to the CU size. Inter-predicted $N \times N$ PUs are only tested if the CU is a LCU, since in the remaining cases the same area is tested as a $2N \times 2N$ PU in the upper depth of the *coding tree*. Inter-predicted $N \times N$ PUs are not tested in 8 × 8 CUs to reduce memory bandwidth requirements. *Asymmetric motion partitions* (AMPs), i.e., $2N \times nU$, $2N \times nD$, $nL \times 2N$, and $nR \times 2N$ [1], are not tested in 8 × 8 CUs to prevent the use of PUs with dimensions smaller than 4. Finally, only the *Square* modes (i.e., $2N \times 2N$ and $N \times N$) are used in intra-predicted PUs. The *Merge/SKIP mode* (MSM) is available for all inter-predicted CU sizes and is conceptually similar to the *SKIP* mode of H.264/AVC. Although considered a different PU splitting mode, MSM is applied to $2N \times 2N$ PUs, so that it can be seen as a submode of inter $2N \times 2N$. When MSM is used, the motion information from spatially and temporally neighboring PUs is inherited by the current PU, forming a larger merged region.

When transform coding the prediction residual, each CU is assumed to be the root of another quadtree-based structure called *residual quadtree* (RQT). Each CU is recursively partitioned into TU, which are the basic units to which transform and quantization operations are applied. The encoder configuration used in this paper allows up to three RQT depths, and TU sizes of 32 × 32, 16 × 16, and 8 × 8, since it uses the HEVC *Main* profile.

TABLE I
CONFIGURATIONS TESTED IN THE COMPLEXITY ANALYSIS

Partitioning parameter	Encoder configuration (*CFG*)						
	1	2	3	4	5	6	7
Max CU Depth	4	3	2	1	4	4	4
Max TU Depth	3	3	3	3	2	1	3
AMP	*E*	*E*	*E*	*E*	*E*	*E*	*D*

B. Rate-Distortion-Complexity Analysis

This section presents an R-D-C analysis of the HEVC encoder when varying the level of partitioning of the described structures. The analysis was performed to identify if and which structures have any impact in the overall encoding computational complexity and in the RD efficiency.

There are three encoding parameters in HM that control directly the HEVC partitioning structures: *Max CU Depth*, *Max TU Depth*, and *AMP*, which define, respectively, the maximum quadtree depth allowed for a CU in each CTU, the maximum quadtree depth allowed for a TU in each RQT, and the possibility of using AMP. Starting from a baseline configuration defined in the *common test conditions* (CTCs) document from JCT-VC [29] as the *random access* (RA) temporal configuration, which specifies groups of pictures of eight bidirectional prediction (B) frames and an intra-period of 32 frames, seven new configurations were created by modifying the value of each of the three parameters, one at a time. The seven configurations tested are presented in Table I. The baseline encoder configuration is defined as CFG 1, while the other six configurations correspond to CFG 2 through CFG 7. Signs *D* and *E* represent the disabled and enabled states of AMP, respectively.

Table II presents the characteristics of the video sequences used in the R-D-C analysis, which are marked with the *A* label in the rightmost column. The remaining labels (*T* and *V*) identify videos used in Sections IV and V. The resulting image quality, bit rate, and encoding computational complexity were recorded for comparison with the reference baseline configuration. These values were measured in terms of *Bjontegaard-Delta* rate (BD-rate), *Bjontegaard-Delta Peak Signal-to-Noise Ratio* (BD-PSNR) [30], and encoding time, which was measured with the *Intel VTune Amplifier XE* software profiler [31].

Table III shows the BD-rate increase, the CCR and the ratio between these two values per configuration tested (multiplied by 100), considering QPs 22, 27, 32, and 37. Both BD-rate and CCR values were calculated using the baseline configuration (CFG 1) as reference. An analysis of the results shows that computational complexity can be reduced by adjusting the partitioning structures, but some configurations result in much larger costs in terms of compression efficiency than others. By changing AMP (CFG 7) and *Max TU Depth* (CFG 5 and CFG 6), the encoding complexity can be reduced with very small BD-rate/CCR ratios (on average, between 3.2 and 8.0), while changing *Max CU Depth* (CFG 1–CFG 4) incurs in much more expressive costs (between 21.9 and 44.2). However, the computational complexity decrease achieved when modifying AMP (CFG 7) or *Max TU Depth* (CFG 5 and CFG 6) varies between 9.4% and 16.6%, which is still not enough when considering the large complexity of HEVC.

TABLE II
TEST SEQUENCES

Name	Frame Rate (Hz)	Bit Depth	Spatial Resolution	Use [a]
BasketballPass	50	8	416×240	*V*
BlowingBubbles	50	8	416×240	*A*, *T*
BQSquare	60	8	416×240	*V*
RaceHorses	30	8	416×240	*A*, *T*
BasketballDrill	50	8	832×480	*V*
PartyScene	50	8	832×480	*A*, *T*
BQMall	60	8	832×480	*A*, *T*
ChinaSpeed	30	8	1024x768	*V*
Kimono1	24	8	1024x768	*V*
SlideEditing	30	8	1280×720	*V*
SlideShow	20	8	1280×720	*A*, *T*
vidyo1	60	8	1280×720	*A, T*
BasketballDrive	50	8	1920×1080	*A*, *T*
BQTerrace	60	8	1920×1080	*V*
Cactus	50	8	1920×1080	*V*
ParkScene	24	8	1920×1080	*A*, *T*
NebutaFestival	60	10	2560×1600	*A*, *T*
PeopleOnStreet	30	8	2560×1600	*V*
SteamLocomotive	60	10	2560×1600	*V*
Traffic	30	8	2560×1600	*A*, *T*

[a] *A*: Analysis, *T*: Training, *V*: Validation.

To achieve a better tradeoff between computational complexity and compression efficiency, instead of applying restrictions to the coding structures at video segment-level or frame-level, it would be desirable to allow the encoder to choose the best combination of partitioning parameters for each image region and video segment according to local characteristics. Indeed, instead of simply removing indiscriminately the possibility of using certain partitioning structures in the whole frame or video segment, as done in the experiments described in this section, the encoder should be able to decide which partitioning structure to use at a finer scale, such as per frame or per CU, adapting itself to the video changing characteristics. This is the goal of the work presented in the next sections.

III. RELATED WORKS

Even though HEVC is a very recent standard, several works with proposals for decreasing its encoding computational complexity have been already published. Most of these works have the objective of decreasing the computational complexity involved in the definition of the new partitioning structures, especially the *coding trees*, PUs, and RQTs, and do so by applying different techniques to determine the best configuration without testing all possibilities through RDO.

Fast algorithms for determining the *coding tree* structure are proposed in [9]–[13]. In [9], a low-complexity R-D cost calculation is used to decide whether or not the CU splitting and pruning processes must be performed. However, the method is only applicable to intra-predicted CUs, which are a minority in the cases of encoding using configurations that use inter-frame prediction. The methods proposed in [10] and [11] use information obtained from intermediate encoding steps to determine if a CU is split into smaller CUs. In [10],

TABLE III
BD-RATE, CCR, AND RATIO BETWEEN THE TWO VALUES FOR DIFFERENT PARTITIONING CONFIGURATIONS

Video Sequence	BD-rate (%)						CCR (%)						Ratio BD-rate/CCR (×100)					
	CFG 2	*CFG* 3	*CFG* 4	*CFG* 5	*CFG* 6	*CFG* 7	*CFG* 2	*CFG* 3	*CFG* 4	*CFG* 5	*CFG* 6	*CFG* 7	*CFG* 2	*CFG* 3	*CFG* 4	*CFG* 5	*CFG* 6	*CFG* 7
BlowingBubbles	5.2	21.9	41.3	0.3	1.2	0.8	25.2	49.1	70.3	10.5	18.7	12.7	20.7	44.5	58.7	2.6	6.5	5.9
RaceHorses	8.3	28.0	49.4	0.4	1.0	1.3	26.4	51.6	71.9	10.7	19.2	17.7	31.5	54.4	68.7	3.4	5.2	7.5
PartyScene	6.9	19.1	35.9	0.3	1.1	0.5	23.4	54.7	74.3	9.9	17.6	9.7	29.5	35.0	48.3	2.8	6.3	5.5
BQMall	7.2	23.0	45.3	0.3	1.1	1.2	23.2	55.1	73.4	10.0	17.1	11.8	30.9	41.8	61.7	2.9	6.4	10.2
SlideShow	12.8	29.5	52.1	0.5	1.2	1.2	22.4	52.6	73.1	7.8	13.8	5.2	57.1	56.1	71.3	6.9	8.9	23.7
vidyo1	2.6	11.4	27.3	0.3	0.6	0.6	21.7	52.4	73.4	7.3	13.2	3.3	12.1	21.8	37.1	3.9	4.9	18.0
BasketballDrive	1.4	8.3	21.0	0.3	1.1	0.6	21.6	54.1	75.9	8.7	15.3	11.3	6.3	15.3	27.7	3.3	7.5	5.5
ParkScene	3.5	11.4	24.4	0.4	1.1	0.8	20.1	52.5	75.0	8.5	15.0	8.5	17.2	21.6	32.6	5.1	7.4	9.7
NebutaFestival	0.2	1.2	4.1	0.0	0.6	0.3	27.4	59.7	80.2	12.1	21.4	15.7	0.6	2.0	5.2	0.3	2.9	1.9
Traffic	3.1	11.9	27.3	0.2	0.7	0.9	22.1	53.7	75.6	8.3	14.7	7.4	14.0	22.2	36.2	2.4	5.0	12.1
Average	**5.1**	**16.6**	**32.8**	**0.3**	**1.0**	**0.8**	**23.3**	**53.5**	**74.3**	**9.4**	**16.6**	**10.3**	**21.9**	**31.0**	**44.2**	**3.2**	**6.0**	**8.0**

if the best prediction mode found for a determined CU is the *SKIP*, the splitting process is terminated. In [11], the ratio between the R-D costs in the current and upper-depth CUs are used in the decision. A method based on motion divergence is proposed in [12] to early terminate the splitting process of CUs. In [13], a complexity control method is proposed that reduces the number of *coding tree* possibilities evaluated through RDO by limiting the maximum depth according to the colocated CTU in the previous frame.

Other approaches seek to reduce the computational complexity of choosing the best PU mode [14]–[17]. Vanne *et al.* [14] propose optimized schemes that conditionally evaluate certain sets of modes according to intermediate encoding decisions. In [15], a complexity control scheme allows adjusting the number of evaluated PU modes for inter-predicted CUs according to a target complexity. In [16], a preprocessing stage analyzes the texture of down-sampled CTUs, leading to the exclusion of unnecessary PU modes, while in [17] small PUs are combined into larger PUs depending on the image characteristics, skipping the evaluation of certain modes.

References [18]–[20] optimize both the *coding tree* and PU mode decision processes. In [18], a complexity control scheme based on [13] introduces a set of conditions that rely on spatial and temporal correlations to early terminate the *coding tree* splitting process. Simultaneously, the PU modes tested for a given CU are chosen according to the size of neighboring CUs. Liquan *et al.* [19] determine the CU depth range and the PU modes tested according to image characteristics and intermediate encoding results, such as motion homogeneity, R-D costs of neighboring CUs, and the PU mode chosen in the upper *coding tree* depth. In [20], information like the variance of prediction errors and the motion vector magnitude are used to avoid exhaustive RDO searches over all possible *coding tree* depths and PU modes.

In [21] and [22], the computational complexity required to decide the RQT structure is reduced by early terminating the recursive TU splitting based on the number of nonzero transformed coefficients. However, as the CCRs achieved when constraining RQTs is very limited, not many works exploit simplifications of the procedures used to determine these structures. Even though some of the methods proposed in [9]–[20] achieve substantial complexity reductions, most of them still incur in nonnegligible compression efficiency losses, as shown later in Section VI. Besides, none of them achieve complexity reduction rates beyond 50%.

Methods based on machine learning have been proposed for use in different video coding-related applications, such as transcoding [23], [24], transrating [25], and even encoding complexity reduction [26]–[28]. However, only a few works, such as [26] and [27], have been published, which employ this type of methods for decreasing the computational complexity of the HEVC encoding process. Garcia *et al.* [26] apply DM techniques to create a linear model that estimates the PSNR and encoding time savings according to different encoding configurations. In [27], a CU splitting early termination algorithm based on support vector machines is described, which makes use of features like the correlation between CU depths in spatial and temporal domains, prediction error, motion activity, and R-D costs. However, this method only reduces the computational complexity involved in the *coding tree* definition process. Further reductions could be achieved if this scheme also included the remaining partitioning structures used in HEVC.

An alternative solution that can decrease R-D inefficiencies while still further reducing computational complexity is proposed in this paper based on machine learning techniques to find data patterns from intermediate encoding results.

IV. DATA MINING FOR HEVC COMPLEXITY REDUCTION

Predictive DM techniques are used to determine the value of dependent variables by looking at the value of some attributes in the data set, identifying regularities and building generalization rules that are expressed as models. There are several methods of predictive DM currently available, which vary broadly from one another in terms of efficiency, complexity, and applicability. Decision trees [32] are models built through predictive DM commonly used because of the following characteristics: 1) they usually achieve high prediction accuracy after proper training; 2) they are easily understood by human beings and therefore simple to be implemented; 3) many

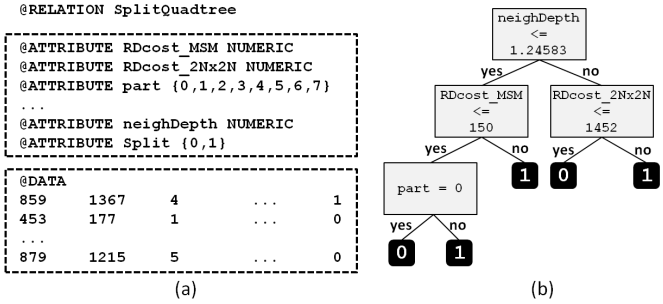

Fig. 3. Example of (a) ARFF file and (b) resulting decision tree.

efficient algorithms to build them from raw data exist; 4) they can deal with both categorical and numerical values; and 5) once implemented, they execute very fast. The first and the fifth characteristics are extremely important for the work presented in this paper, since it aims at reducing the HEVC encoding complexity without harming the R-D efficiency and increasing the complexity.

The next sections explain how a set of decision trees is built from data collected during offline encodings and used to control the early termination of the recursive search for the best *coding tree*, PU, and RQT structures.

A. Methodology

The *Waikato Environment for Knowledge Analysis* (WEKA) [33], version 3.6, was used to aid the DM process performed in this paper. WEKA is a free, open-source DM tool that includes several machine learning algorithms and tools for preprocessing, classifying, clustering, and visualizing the data to be explored. The HM software (version 12), was used to encode the video sequences with QPs 22, 27, 32, 37, and 42, using the RA configuration [29] and all video sequences of Table II assigned with the *T* label were used in the collecting of data that were processed using DM to train the decision trees.

The input for WEKA is an *Attribute-Relation File Format* (ARFF) file, which is exemplified in Fig. 3(a). The first section of an ARFF file consists of a header with the attributes declaration, and the second section contains the raw data with one instance per line and one attribute value per column. In the specific case of building decision trees, the last line of the first section identifies the class attribute, for which the machine learning algorithms try to find a general (prediction) rule. In the example of Fig. 3(a), *Split* is the class attribute.

For each early termination scheme proposed in this paper, several variables were recorded during execution of the HM encoder, such as the sum and the variance of luminance samples in a CU, the sum and variance of prediction residues in a PU, the horizontal and vertical gradients in a possible PU edge, and the R-D cost of each PU splitting mode. Some instructions were added to the HM encoder to calculate the values needed and not already computed by the original software. The usefulness of each of these variables for the decision tree was assessed through the *information gain attribute evaluation* (IGAE) method in WEKA, which measures the information gain [34] achievable using a variable to classify the data into the different classes represented in the data. This gain equates to the difference between the number of bits per data item necessary to convey its class identity before and after classification of the data set using decision rules based on the variable in question [32]. Therefore, the information gain of a variable indicates how relevant it is for the process of constructing a decision tree that correctly decides to which class each data item belongs. In the case of the WEKA software, this information gain is measured by the *Kullback–Leibler divergence* (KLD) [35] of the preclassification and postclassification probability distributions. Based on this measure (IGAE), a manual analysis procedure was followed to identify the most useful variables for the (tree) decision processes. The variables with higher information gain were selected as attributes for the tree training processes, as explained in Section IV-B to IV-D.

A decision tree is composed of a set of nodes and arcs, as shown in the example given in Fig. 3(b) where the nodes represent tests performed on the attributes and the arcs are the possible results of such tests. The leaves of decision trees are the values that the class attribute can assume and represent the possible outputs of the decision process. In the example of Fig. 3(b), these outputs are 0 and 1, which represent *do not split* and *split* decisions, respectively.

The training of the decision trees was performed with the *C4.5* algorithm [32], which, as explained before, uses KLD to choose the best attribute for each decision step and the thresholds corresponding to each decision step. The *C4.5* algorithm starts by taking all instances fed to it as inputs and calculates the information gain of using each attribute to perform the classification using a determined threshold. By iterating among all attributes and adjusting the thresholds, *C4.5* measures the information gain of each variable and threshold pair. Then, the attribute (and its corresponding threshold) with the largest information gain is chosen to divide the training data into two subsets. The same process is applied recursively to the two subsets.

After trained, the accuracy of all obtained trees was measured with WEKA by applying a 10-fold cross-validation process. The level of accuracy was measured by the percentage of correct decisions in the total amount of instances used in the training process. Finally, all the decision trees that resulted from the training procedure were implemented in HM and the RD efficiency of the low-complexity encoders was measured by encoding video sequences different from those used in the training phase.

B. Early Termination for Determining Coding Trees

Among the three partitioning structures dealt in this paper, *coding trees* have a central role due to their interdependence with the remaining partitioning structures, as explained in Section II. The early termination consists in deciding whether the splitting of CUs into four smaller CUs should be tested. If the decision tree outcome is *yes* (i.e., $Split = 1$), the next *coding tree* depth is tested. Otherwise, the current CU is not

TABLE IV
IGAE FOR THE CODING TREE EARLY TERMINATION

Attribute	64×64	32×32	16×16
Partition	0.352	0.336	0.269
Neigh_Depth	0.311	0.262	0.249
Ratio(2N×2N, MSM)	0.112	0.168	0.255
relRatio(2N×2N, MSM)	0.109	0.163	0.249
RD(2N×2N)	0.035	0.042	0.053
RD(MSM)	0.034	0.061	0.108
RD(2N×N)	0.033	0.036	0.044
RD(N×2N)	0.031	0.032	0.042
SkipMergeFlag	0.046	0.066	0.065
MergeFlag	0.020	0.035	0.046

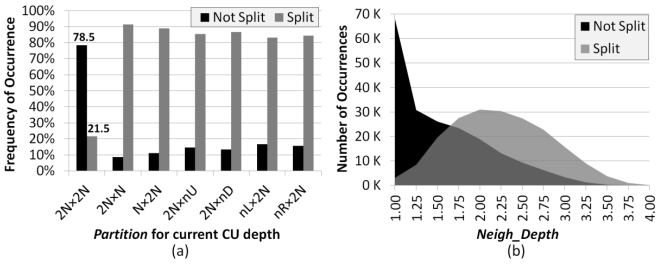

Fig. 4. Occurrence of CUs split and not split into smaller CUs according to (a) PU splitting mode chosen for the current CU and (b) average of CU depths in neighboring CTUs.

split and their sub-CUs are not considered in the RDO-based decision process.

To reduce the problem of the class data imbalance that occurs when there are significantly more training instances belonging to one class than to the other(s), following common practice [36], the ARFF files used in the evaluation are composed of a data set with 50% of CUs that have been split into smaller CUs and 50% of CUs that have not been split into smaller CUs. This was accomplished by having WEKA resample the training data instances when building the final ARFF files, which are composed of equal numbers of random samples of the data collected during the encoding of the 10 video sequences listed in Table II using all the QPs enumerated before (Section IV-A). The data were saved from runs of the HM encoder, which was modified to compute and output potentially useful variables to be selected as attributes for the decision trees.

As the HEVC standard allows up to four *coding tree* depths, three different decision trees were created for the sizes that allow splitting into smaller CUs: 64 × 64, 32 × 32, and 16 × 16. Table IV shows the HM variables that provided best performance as measured by the information gain with respect to the decision of splitting or not splitting a CU (i.e., those that were selected as attributes for the decision trees). Information gain values are presented separately for each CU size in the table and the attributes are described in detail in the following paragraphs.

Partition represents the PU splitting mode that was chosen for the current CU among those listed in Section II-A (i.e., $2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$, $2N \times nU$, $2N \times nD$, $nL \times 2N$, or $nR \times 2N$), independently of whether inter- or intra-frame prediction was applied. This partition mode information is important as it is likely that a CU predicted with large-size PUs (e.g., as a single $2N \times 2N$ PU) does not need being split into smaller CUs. Statistics that support this claim are shown in Fig. 4(a) for 32 × 32 CUs. The chart shows that 78.5% of the CUs encoded as a $2N \times 2N$ PU were not split into sub-CUs. Conversely, an average of 86.7% of CUs encoded with the remaining PU splitting modes were divided into sub-CUs. The remaining CU sizes also showed the behavior shown in Fig. 4(a).

The *Neigh_Depth* attribute is related with the *coding tree* depths used in neighboring CTUs already encoded. The rationale for considering using this attribute in the decision process is that there exists a correlation among maximum depths of spatially neighboring CTUs, as we have shown in [37]. The value of *Neigh_Depth* is calculated as follows. First, for each neighboring CTU, the average depth of all CUs is computed. Then, the average of averages is computed among all neighbors available for the current CTU. The top, left, top-left, and top-right CTUs in the current frame, as well as the colocated CTUs in the first frames of both reference lists (*list* 0 and *list* 1) are considered as neighbors. Fig. 4(b) shows the distribution of *Neigh_Depth* for 32 × 32 CUs. The curves show that there is a clear relationship between the distribution of *Neigh_Depth* and the CU splitting decision. CUs that are not split into smaller CUs have *Neigh_Depth* values that cluster toward low magnitudes, while for CUs that are split into smaller CUs the opposite occurs. Since the two distributions do not fully overlap it is possible to determine an optimal decision threshold that minimizes the classification error rate. WEKA computes these thresholds for each attribute during the process of training the decision trees.

The *Ratio(2N × 2N, MSM)* shown in Table IV is calculated as a simple division between the R-D costs of encoding the current CU as an inter-predicted $2N \times 2N$ PU and as a MSM PU. The *relRatio(2N × 2N, MSM)* value is the normalized difference between the *RD(2Nx2N)* and *RD(MSM)* costs, calculated as per (1). The reason for considering these values in the IGAE analysis is that when a compression gain (i.e., a drop in R-D cost) is observed due to the use of motion compensated prediction in a CU instead of encoding it with MSM, the block probably belongs to a medium/high-motion or complex-textured image region and usually in this type of situation it is advisable to split a CU into smaller CUs.

$$relRatio(2N \times 2N, MSM) = \left| \frac{RN(2N \times 2N) - RD(MSM)}{RD(MSM)} \right| \quad (1)$$

The R-D costs for inter $2N \times 2N$, MSM, $2N \times N$, and $N \times 2N$ PU splitting modes were also separately considered in the IGAE analysis and are represented in Table IV. Also present in the table are the *MergeFlag* and *SkipMergeFlag*, which are binary variables used by the encoder to identify CUs that have been predicted with MSM and *SKIP* mode (special case for MSM), respectively. These flags were included in the analysis because CUs encoded with MSM and *SKIP* mode generally belong to low motion or very homogeneous image regions, which are rarely encoded with small CUs.

TABLE V
CHARACTERISTICS OF THE DECISION TREES FOR THE CODING TREE EARLY TERMINATION

CU size	Decision Accuracy	Incorrect Depth	Depth	Test Nodes	Leaves
64×64	84.2%	7.1%	5	6	19
32×32	84.5%	7.5%	8	20	33
16×16	84.6%	6.9%	9	23	44

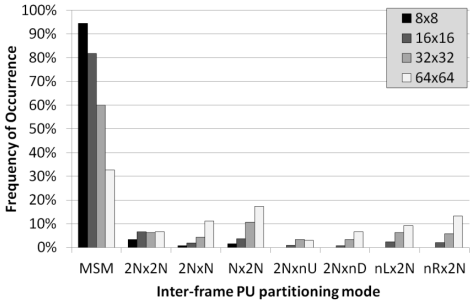

Fig. 5. Frequency of occurrence of each inter-PU splitting mode.

Table V lists some performance indicators and other characteristics of the decision trees constructed and trained with the attributes just described. The table presents the accuracy of each tree, as well as their depth (i.e., number of sequential tests), the number of test nodes, and the number of leaves. As shown, the designed trees achieve very good decision accuracy, with values slightly larger than 84%. However, these results account the case of splitting a CU that should not be split into smaller CUs as an inaccuracy, even though it does not harm the encoding R-D efficiency. The R-D efficiency could only be harmed when a CU that should be split is not split due to the early termination provided by the decision trees. The third column of Table V shows the number of incorrect early terminations caused by inaccuracies in the decision trees, which are the cases that could actually cause R-D efficiency losses.

Regarding the topological characteristics of the decision trees, it is important to notice that all of them are composed of less than 10 decision levels (depth), which means that the computational complexity added to the encoder associated with the decision trees is negligible.

C. Early Termination for Determining the PU Structure

The second early termination scheme is motivated by the statistics shown in Fig. 5, which presents the average occurrence probability of each PU splitting mode in inter-predicted CUs. The statistics were compiled for the video sequences marked with the *T* label in Table II. It is clear that most inter-predicted CUs are encoded without being split into smaller PUs and employ mainly the MSM mode. For example, around 95% and 82% of 8 × 8 and 16 × 16 CBs are encoded as MSM, respectively. However, even though the remaining modes are rarely used, they are still tested for every CU, which is not ideal when trading off compression efficiency and computational complexity.

TABLE VI
IGAE FOR THE PU EARLY TERMIATION

Attribute	64×64	32×32	16×16	8×8
Ratio(2N×2N, MSM)	0.245	0.390	0.475	0.572
relRATIO(2N×2N, MSM)	0.129	0.245	0.341	0.433
RD(MSM)	0.224	0.306	0.383	0.422
RD(2N×2N)	0.165	0.139	0.101	0.135
RD(best)	0.208	0.284	0.364	0.\407
UpperCU_div	—	0.223	0.297	0.240
Ratio(best, MSM)	0.195	0.266	0.229	0.208
relRatio(best, MSM)	0.146	0.207	0.203	0.186

As in the early termination for *coding trees*, the decision trees for early terminating the PU mode decision were designed to decide if the search for the best PU structure should continue after some modes have been tested. As most inter-predicted CUs are encoded as a single PU, the PU-related decision trees are applied after the MSM and $2N \times 2N$ modes are tested. Initially, the MSM and the $2N \times 2N$ modes are always tested for every CU. After that, the decision trees are used to determine if an early termination should occur. In case of early termination, a choice is made between the MSM and the $2N \times 2N$ modes and the one with the smallest R-D cost is chosen for the CU. Otherwise, if the decision is not to terminate the PU mode choice procedure, the remaining modes are tested.

As in the case presented in Section IV-B, 50% of the training data come from inter-predicted CUs that have been split into PUs smaller than $2N \times 2N$ and 50% come from inter-predicted CUs that have not been split into PUs smaller than $2N \times 2N$. Four different decision trees were created, one for each CU size (64 × 64, 32 × 32, 16 × 16, and 8 × 8).

The class attribute is the information of whether or not a CU should be predicted using a PU splitting mode smaller than $2N \times 2N$. After an extensive observation on the collected data, the IGAE analysis concluded that the HM variables presented in Table VI returned the largest partitioning information gains and should be used as attributes for training the decision trees. In Table VI, the information gain results are presented separately for each CU size and the attributes *Ratio(2N × 2N, MSM)*, *relRATIO(2N × 2N, MSM)*, *RD(MSM)*, and *RD(2N × 2N)* have the same meaning as the homonym attributes of Section IV-B. Attribute *RD(best)* is the lowest R-D cost among the MSM and the $2N \times 2N$ modes and *UpperCU_div* is the information of whether or not the CU in the upper *coding tree* depth was split into PUs smaller than $2N \times 2N$. Finally, the attributes *Ratio(best, MSM)* and *relRatio(best, MSM)* correspond to the ratio and normalized difference between *RD(best)* and *RD(MSM)*, respectively, as

$$relRatio(best, MSM) = \left| \frac{RD(best) - RD(MSM)}{RD(MSM)} \right|. \quad (2)$$

Fig. 6 shows statistical results for two attributes in the case of inter-predicted 16 × 16 CUs in the *BasketballDrive* sequence (QP 32). Even though the figure refers to one specific case, results for the remaining sequences and CU sizes presented similar characteristics. Fig. 6(a) shows that the ratio

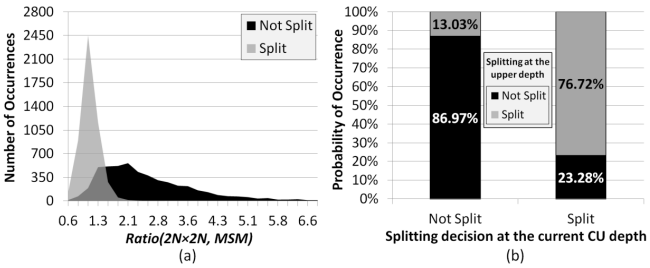

Fig. 6. Occurrence of CUs split and not split into PUs smaller than $2N \times 2N$ according to (a) ratio between $2N \times 2N$ and MSM R-D costs and (b) splitting decision in the upper *coding tree* depth.

TABLE VII
CHARACTERISTICS OF THE DECISION TREES FOR THE PU EARLY TERMINATION

CU size	Decision Accuracy	Incorrect Mode	Depth	Test Nodes	Leaves
64×64	79.6%	8.6%	5	8	9
32×32	86.0%	5.0%	7	9	10
16×16	89.2%	4.0%	5	9	10
8×8	91.2%	2.1%	4	5	6

TABLE VIII
IGAE FOR THE RQT EARLY TERMINATION

Attribute	32×32	16×16
AbsSumY	0.342	0.279
AbsSumU	0.057	0.033
AbsSumV	0.055	0.031
SingleCost	0.145	0.140
nonZeroCoeffY	0.348	0.284
nonZeroCoeffU	0.342	0.280
nonZeroCoeffV	0.342	0.280

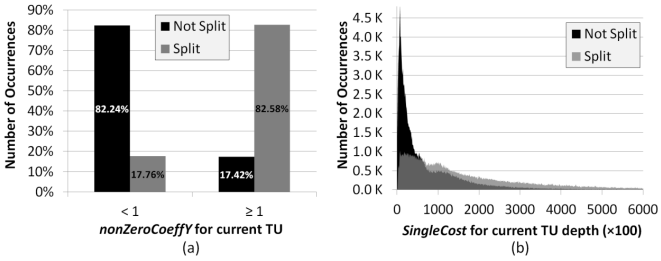

Fig. 7. Occurrence of TUs split and not split into smaller TUs according to (a) number of nonzero transformed luminance coefficients in the current TU depth and (b) R-D cost for the current TU depth.

between the $2N \times 2N$ and MSM R-D costs is a relevant indicator of the necessity of testing the remaining modes. It is possible to see that most CUs with a small ratio are split into PUs smaller than $2N \times 2N$, while CUs with larger ratio values are mostly encoded with $2N \times 2N$ or MSM. Fig. 6(b) shows that in 76.72% of the cases when a 16×16 CU was split into smaller PUs, its upper CU was also split. Analogously, in 86.97% of the cases when a 16×16 CU was not split into smaller PUs, its corresponding upper depth CU was not split.

The decision trees obtained for the PU early termination process were trained with the attributes shown in Table VI and their characteristics are shown in detail in Table VII. The table shows that the decision trees obtained after training achieve a decision accuracy that varies from 79.6% to 91.2%. However, it is important to realize that wrong decisions only result in R-D efficiency losses in those cases where a CU should have been predicted with PUs smaller that $2N \times 2N$ but a decision was taken not to pursue the partitioning, thus terminating early the search for the best PU splitting mode. In the remaining wrong decisions (i.e., when the CU should be predicted with MSM or $2N \times 2N$ and the decision process is not early terminated), the computational complexity is not reduced and the encoder chooses an optimal mode through RDO, since all modes are still tested. The third column of Table VII shows statistics only for the case that incurs in R-D losses, i.e., the rate of incorrect mode decisions caused by wrong early terminations, which varies between 2.1% and 8.6%.

D. Early Termination for Determining the RQT Structure

As previously shown in Table III, even though restricting the maximum TU size does not provide substantial CCRs, it only affects the encoding R-D efficiency marginally. For this reason, a third early termination scheme that halts the process for determining the best RQT structure is proposed in this section.

Similarly to the *coding tree* early termination scheme (Section IV-B), the RQT early termination consists in deciding whether or not the splitting of TUs into four smaller TUs should be tried. If the early termination algorithm decides *yes*, the next RQT depth must be tested. Otherwise, the splitting process is halted for the current TU and its sub-TUs are not considered in the RDO-based decision process.

As in the cases of previous schemes, resampling to solve class data imbalance was also used to train the decision trees [36]. Given that TUs can assume three different dimensions in the *Main* encoder configuration (32×32, 16×16, and 8×8), only two different decision trees were designed (32×32 and 16×16). The selected attributes are presented in Table VIII together with the respective results in terms of information gain.

The attributes *AbsSumY*, *AbsSumU*, and *AbsSumV* are the sum of absolute values from the luminance, blue chrominance, and red chrominance residues, respectively. These attributes are used because as the residue samples are the inputs to the transform modules, the probability of they being encoded with small-sized TUs increases with their absolute sum. The attributes *nonZeroCoeffY*, *nonZeroCoeffU*, and *nonZeroCoeffV* represent the number of nonzero coefficients obtained after transforming the TU as a whole block (i.e., before splitting it). Fig. 7(a) shows that when *nonZeroCoeffY* is zero, 82.24% of the TUs are not split into smaller TUs. The remaining 17.76% are split because one or both associated chrominance TUs present nonzero coefficients. Alternatively, when *nonZeroCoeffY* is nonzero, 82.58% of the TUs are split.

The *SingleCost* attribute is the R-D cost of encoding a TU as a whole block. Statistics for this attribute are shown in Fig. 7(b) for the specific case of 32×32 TUs and

TABLE IX
CHARACTERISTICS OF THE DECISION TREES FOR THE RQT EARLY TERMINATION

CU size	Decision Accuracy	Incorrect Depth	Depth	Test Nodes	Leaves
32×32	83.4%	8.9%	10	15	16
16×16	80.7%	12.1%	8	17	18

considering QP 32. The figure shows that there is a correlation between the TU splitting decision and the *SingleCost* value, since most of the nonsplit cases present very small R-D costs. Oppositely, for larger *SingleCost* values, the number of nonsplit TUs decreases and the split cases become the majority, although by a small amount.

The decision trees for 32 × 32 and 16 × 16 TUs are detailed in Table IX, which shows that their decision accuracy are 83.4% and 80.7%, respectively. As explained in the previous early terminations, R-D efficiency is not harmed when splitting a TU that should not be split, since the best splitting option will be chosen through RDO in the end. Incorrect TU depth decisions occur only in the cases when the TU should be split into smaller TUs but the process is early terminated after the whole TU is evaluated. The third column of Table IX shows statistics regarding this type of incorrect depth decisions caused by wrong early termination (8.9% and 12.1%).

V. EXPERIMENTS, RESULTS, AND DISCUSSION

To evaluate the performance of all early termination schemes proposed in this paper, the decision trees were implemented in the HM encoder, which was run on a computer equipped with a 2.27-GHz processor. The 10 video sequences labeled with *V* in Table II were used to validate the proposed method. These sequences differ broadly from one another in terms of frame rate, bit depth, motion/texture, color, and spatial resolution, and were not used in the training of the decision trees.

Compression efficiency was computed in terms of BD-rate and BD-PSNR, and the encoding times were obtained with the *Intel VTune Amplifier XE* software profiler [31]. The coded video bit rate and PSNR, as well as the encoding times, were compared with the corresponding values obtained when the unmodified HM encoder was used to encode the same sequences. All video sequences were encoded with QPs 22, 27, 32, and 37, using the *Main* profile and the standard RA temporal configuration defined in CTC [29].

It should be pointed out that the HM encoder is an HEVC software implementation developed during the standardization process for tests and documentation purposes, so that it is not optimized for real-time operation. Still, as the definition of the partitioning structures is a high-level decision that does not directly affect the operation performed at any particular HEVC encoder module (though it influences the number of encoding iterations performed in the RDO process), similar complexity reductions are expected to be achieved in other encoder implementations, such as the x265 software [38]. It is also important to notice that the current version of the HM encoder already includes several complexity reduction techniques, such as the *rough mode decision* for intra-frame prediction, the *coded block flag*-based early termination, the early CU termination algorithm, and the early *SKIP* mode decision algorithm. The schemes proposed in this paper were implemented on top of all these methods, providing additional CCRs, and they were compared with the original HM encoder with all its built-in early terminations enabled. All schemes were first separately evaluated and then jointly implemented.

TABLE X
R-D-C RESULTS FOR THE CODING TREE EARLY TERMINATION

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	-0.007	0.000	23.0	-0.030
BQTerrace	-0.008	0.000	28.4	-0.027
BasketballDrill	+0.425	-0.017	30.1	1.414
BasketballPass	+0.128	-0.006	23.9	0.534
Cactus	+0.207	-0.007	41.0	0.506
ChinaSpeed	+0.131	-0.007	29.2	0.449
Kimono1	+0.552	-0.020	44.0	1.253
PeopleOnStreet	+0.089	-0.004	16.1	0.556
SlideEditing	+0.353	-0.057	71.3	0.495
SteamLocomotiveTrain	+0.971	-0.022	60.3	1.608
Average	**+0.284**	**-0.014**	**36.7**	**0.774**

TABLE XI
R-D-C RESULTS FOR THE PU EARLY TERMINATION

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.299	-0.014	45.7	0.655
BQTerrace	+0.091	-0.002	54.4	0.168
BasketballDrill	+0.491	-0.020	44.6	1.101
BasketballPass	+0.449	-0.020	42.5	1.055
Cactus	+0.401	-0.012	48.6	0.827
ChinaSpeed	+1.001	-0.051	47.1	2.127
Kimono1	+0.689	-0.024	50.9	1.353
PeopleOnStreet	+1.021	-0.048	37.4	2.733
SlideEditing	+0.206	-0.030	68.1	0.302
SteamLocomotiveTrain	+0.969	-0.022	57.2	1.694
Average	**+0.562**	**-0.024**	**49.6**	**1.132**

TABLE XII
R-D-C RESULTS FOR THE RQT EARLY TERMINATION

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.073	-0.003	8.9	0.820
BQTerrace	-0.086	+0.001	7.9	-1.093
BasketballDrill	+0.049	-0.002	6.5	0.753
BasketballPass	+0.033	-0.002	6.8	0.482
Cactus	+0.042	-0.001	7.2	0.575
ChinaSpeed	+0.131	-0.006	6.7	1.959
Kimono1	+0.169	-0.006	5.6	2.988
PeopleOnStreet	+0.249	-0.012	6.2	4.022
SlideEditing	-0.244	+0.036	8.9	-2.739
SteamLocomotiveTrain	+0.132	-0.004	7.3	1.806
Average	**+0.055**	**0.000**	**7.2**	**0.758**

A. Rate-Distortion-Complexity Results for Single Schemes

Tables X–XII present R-D-C results for the three proposed schemes implemented separately. The tables show that the PU early termination (Table XI) yields the largest CCR levels, decreasing the total encoding time in a range from

37.4% to 68.1% (on average, 49.6%). These large reductions are justified due to the fact that the scheme is applied to inter-predicted CUs and as inter-frame prediction is the most time-consuming task in the HEVC encoder, the complexity reductions achieved with early terminated inter-predicted CUs have an important impact in the overall encoding complexity. The *coding tree* early termination scheme (Table X) reduces computational complexity in a range from 16.1% to 71.3% (on average, 36.7%), while the use of RQT early termination (Table XII) results in a CCR varying from 5.6% to 8.9% (on average, 7.2%).

It is possible to notice from the results shown in Table X that the CCR values provided by the *coding tree* early termination are correlated with the texture characteristics of the videos. Sequences with large homogeneous areas (e.g., *SlideEditing* and *SteamLocomotiveTrain*) are those which present the largest complexity reductions, because such areas are usually encoded with large CU sizes, allowing the early termination to halt the decision process in the first *coding tree* depths without significant loss of encoding performance. High-resolution videos usually have larger numbers of homogeneous areas, but this is not always the case. For example, *BQTerrace* and *PeopleOnStreet* are examples of high-resolution videos that do not present very large complexity reductions because they are composed of detailed texture. Videos with small spatial resolution (e.g., *BQSquare*, *BasketballDrill*, and *BasketballPass*) also present smaller complexity reductions, since they are mostly encoded with small-sized CUs.

The largest CCR achieved with the PU early termination (Table XI) is also observed in those videos with large homogeneous areas, since in these cases the decision process is halted more frequently after testing the largest PU size ($2N \times 2N$). However, differently from the *coding tree* early termination, the complexity reductions achieved with the PU early termination also depend on the motion characteristics of the video. For example, the *BQTerrace* video sequence presents a larger complexity reduction with the PU early termination than with the *coding tree* early termination because it shows a scene in slow motion (mostly camera movement), which means that testing MSM or large PUs in reference frames is usually enough to find a good prediction. On the other hand, fast-motion scenes (e.g., *BasketballDrill* and *BasketballPass*) are those that present the smallest complexity reductions, since more modes need to be tested.

Regarding the RQT early termination (Table XII), all videos present similar complexity reduction levels, since this solution is less dependent on video characteristics and more dependent on intermediate encoding results (prediction residue). In terms of compression efficiency, the RQT early termination causes the smallest bit rate increases both in absolute and relative numbers. When not considering the CCRs achieved, the BD-rate increase caused by the scheme is insignificant (on average, +0.06%). Besides, this increase relative to the CCR is the smallest between all schemes (on average, 0.76), as shown in the rightmost column of Table XII.

Note that in some sequences a small BD-rate decrease was noticed. A careful analysis of the encoding results for these cases showed that in some areas of some frames (e.g., frame 10 of *SlideEditing*) the early terminations lead to the use of larger partitioning structures than those chosen by the original HM encoder. This analysis also showed that the motion fields for this and the following frames obtained when using the early termination procedures were more coherent than those obtained when using the original encoder. Even though these early decisions do not yield the best R-D efficiency possible for the current CTU, it appears that the bits saved through the increased use of *MSM* and *SKIP* modes result in lower BD-rates. As these particular video sequences present very homogeneous and moderate-motion characteristics, small or no image quality decreases are noticed due to these decisions, which, associated to the bit rate decrease, leads to the negative BD-rate values shown in the tables.

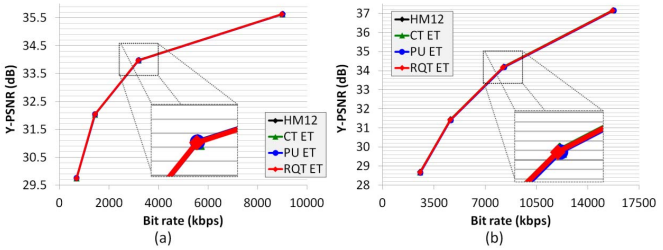

Fig. 8. R-D efficiency for (a) *BQTerrace* and (b) *PeopleOnStreet* video sequences encoded with the original HM and the three early termination schemes implemented separately.

In comparison with the six configurations tested in Section II-B, the three proposed schemes present much better R-D-C efficiency results. While the BD-rate/CCR ratio for those configurations varied between 3.2 and 44.2, the three proposed early termination schemes resulted in ratios equal to 0.77, 1.13, and 0.76.

Fig. 8 shows the R-D efficiency of the three proposed schemes for two video sequences encoded with different bit rates. In both charts of Fig. 8, the curves represent R-D results for the original encoder (*HM 12*), the *coding tree* early termination (*CT ET*), the PU early termination (*PU ET*), and the RQT early termination (*RQT ET*) implementations, respectively. The charts show results for the *BQTerrace* and the *PeopleOnStreet* video sequences [Fig. 8(a) and (b)], which are, respectively, those that presented the best and worst R-D-C efficiency considering the three early termination schemes. We can observe that the R-D efficiency achieved with the proposed schemes is very close to that of the original encoder, since the curves overlap. A closer detailed look (300% zoom boxes) shows that the curves overlap even in the worst-case video sequence (*PeopleOnStreet*). This happens because in the zoomed boxes, the space between horizontal lines is exactly 0.2 dB but, as presented in Tables X–XII, the BD-PSNR decreases caused by the proposed early terminations are much smaller (between 0 and 0.02 dB).

B. Rate-Distortion-Complexity Results for Joint Schemes

The early termination schemes were jointly implemented in HM to evaluate the encoding R-D-C efficiency when further CCRs are required. Tables XIII–XV present results

TABLE XIII
R-D-C RESULTS FOR JOINT CODING TREE AND PU EARLY TERMINATIONS

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.378	-0.017	54.3	0.697
BQTerrace	+0.283	-0.006	72.1	0.393
BasketballDrill	+1.295	-0.052	55.7	2.324
BasketballPass	+0.873	-0.038	50.9	1.717
Cactus	+0.999	-0.031	64.0	1.561
ChinaSpeed	+1.410	-0.072	56.3	2.505
Kimono1	+2.745	-0.096	67.7	4.052
PeopleOnStreet	+1.463	-0.068	43.3	3.382
SlideEditing	+1.293	-0.190	86.6	1.493
SteamLocomotiveTrain	+2.573	-0.059	77.8	3.305
Average	**+1.331**	**-0.063**	**62.9**	**2.118**

TABLE XIV
R-D-C RESULTS FOR JOINT PU AND RQT EARLY TERMINATIONS

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.378	-0.018	51.3	0.737
BQTerrace	+0.188	-0.005	58.9	0.320
BasketballDrill	+0.446	-0.018	48.2	0.925
BasketballPass	+0.735	-0.032	46.6	1.580
Cactus	+0.442	-0.013	52.7	0.839
ChinaSpeed	+1.206	-0.061	50.5	2.388
Kimono1	+0.876	-0.031	53.9	1.627
PeopleOnStreet	+1.269	-0.059	40.8	3.113
SlideEditing	+0.075	-0.011	72.1	0.104
SteamLocomotiveTrain	+1.408	-0.032	60.9	2.311
Average	**+0.702**	**-0.028**	**53.6**	**1.311**

TABLE XV
R-D-C RESULTS FOR JOINT CODING TREE AND RQT EARLY TERMINATIONS

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.091	-0.004	30.3	0.301
BQTerrace	-0.028	0.000	56.4	-0.050
BasketballDrill	+0.453	-0.018	33.6	1.349
BasketballPass	+0.074	-0.003	29.0	0.255
Cactus	+0.284	-0.009	44.5	0.637
ChinaSpeed	+0.242	-0.012	33.1	0.733
Kimono1	+0.839	-0.030	46.7	1.796
PeopleOnStreet	+0.396	-0.018	20.9	1.895
SlideEditing	+0.544	-0.072	73.2	0.744
SteamLocomotiveTrain	+0.553	-0.015	62.6	0.884
Average	**+0.345**	**-0.018**	**43.0**	**0.802**

for three implementations that use two early termination schemes in conjunction: *coding tree* and PU early terminations (Table XIII), PU and RQT early terminations (Table XIV), and *coding tree* and RQT early terminations (Table XV). Finally, results for an implementation that conglomerates the three early termination schemes are presented in Table XVI.

The results show that the largest CCRs are achieved when all the early termination schemes are jointly implemented in HM. In this case, the CCRs vary from 46.4% to 87.6% (65.3%, on average, as shown in Table XVI). Note that the reduction achieved is not equivalent to the sum of the reductions achieved with each scheme separately. This happens due to the nature of the partitioning structures used in HEVC. For example, when constraining the number of R-D evaluations performed to decide the best *coding tree* structure, the overall number of PU partitioning evaluations is also decreased, since PUs are defined within CUs. Similarly, the number of RQT evaluations also decreases because CUs are used as roots for RQTs.

TABLE XVI
R-D-C RESULTS FOR JOINT CODING TREE, PU, AND RQT EARLY TERMINATIONS

Video	BD-rate (%)	BD-PSNR (dB)	CCR (%)	Ratio BD-rate/CCR
BQSquare	+0.406	-0.019	59.0	0.689
BQTerrace	+0.282	-0.007	74.4	0.379
BasketballDrill	+1.182	-0.048	58.3	2.029
BasketballPass	+0.958	-0.042	54.1	1.773
Cactus	+1.006	-0.031	66.3	1.517
ChinaSpeed	+1.547	-0.078	58.8	2.629
Kimono1	+3.013	-0.105	69.2	4.355
PeopleOnStreet	+1.740	-0.081	46.4	3.752
SlideEditing	+0.920	-0.134	87.6	1.050
SteamLocomotiveTrain	+2.493	-0.058	79.1	3.154
Average	**+1.355**	**-0.060**	**65.3**	**2.075**

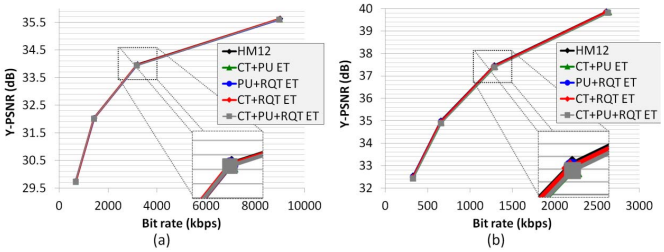

Fig. 9. R-D efficiency for (a) *BQTerrace* and (b) *Kimono1* video sequences encoded with the original HM and the four combinations of early termination schemes.

In terms of R-D efficiency, the implementation with all schemes presented the largest losses in comparison with the remaining versions, as expected. However, the average BD-rate increase of 1.36% is still negligible when the massive CCR of 65.3% is considered. For this reason, this would be the most advisable solution for an implementation that admits a small loss in R-D efficiency.

The smallest CCRs are achieved when the *coding tree* and RQT early terminations are jointly implemented. The average CCR achieved with this implementation is 43% (Table XV), which is close to the reduction achieved when only the *coding tree* early termination is used (36.7%, as shown in Table X). In terms of R-D efficiency, the joint *coding tree* and RQT early terminations yields the smallest losses (+0.35% in BD-rate), so that this would be the most advisable solution for an implementation that admits very small compression efficiency losses.

Fig. 9 shows R-D efficiency results for the four joint implementations. In both charts of Fig. 9, the curve labeled as *HM12* represents R-D results for the original encoder, while

TABLE XVII
COMPARISON WITH RELATED WORKS

Category	Works	BD-rate (%)	CCR (%)	Ratio BD-rate/CCR
CT ET	Seunghyun [9]	+0.6	50	1.20
	J.-Hycok [10]	+1.2	48	2.50
	Goswami [11]	+1.7	38	4.42
	Jian [12]	+1.9	43	4.42
	Shen [27]	+1.4	45	3.11
	Proposed	**+0.28**	**37**	**0.77**
PU ET	Vanne [14]	+1.3	51	2.55
	Zhao [15]	+5.9	50	11.8
	Khan [17]	+1.3	44	2.88
	Proposed	**+0.56**	**50**	**1.13**
RQT ET	Shi [21]	+1.4	22	6.36
	Proposed	**+0.05**	**7.2**	**0.76**
CT+PU ET	Wei-Jhe [18]	+5.1	43	5.11
	Liquan [19]	+1.5	42	3.55
	Shen [20]	+1.9	41	4.54
	Proposed	**+1.33**	**63**	**2.12**
PU+RQT ET	**Proposed**	**+0.7**	**54**	**1.31**
CT+RQT ET	**Proposed**	**+0.35**	**43**	**0.80**
CT+PU+RQT ET	**Proposed**	**+1.36**	**65**	**2.08**

the curves labeled as *CT + PU ET*, *PU + RQT ET*, and *CT + RQT ET* represent R-D results for the joint *coding tree* and PU early terminations, the joint PU and RQT early terminations, and the joint *Coding Tree* and RQT early terminations, respectively. The curve *CT + PU + RQT ET* presents results for the implementation that includes all schemes. The charts show results for the *BQTerrace* and the *Kimono1* video sequences [Fig. 9(a) and (b)], which are, respectively, those that presented the best and worst R-D-C efficiency results in the joint implementations. As in the previous comparisons for the separately implemented schemes, the R-D efficiency achieved in the joint cases is also very close to that of the original encoder. Even in the worst-case video, a detailed look (300% zoom) shows that the curves almost overlap.

VI. COMPARISONS WITH PREVIOUS WORKS

The best related works reviewed in Section III have been analyzed and compared with the schemes proposed in this paper. To allow a fair comparison, only those works that reported BD-rate results and CCR values relative to the overall encoding process using the original HM as reference were selected. Besides, all the compared works were also tested with the *Main* profile, QPs 22, 27, 32, and 37, and the RA temporal configuration (except for [9], which focuses on intra-predicted CUs and thus was only tested with the *All Intra* configuration). All works were tested for at least seven video sequences with at least four different spatial resolutions, except for [21], which was kept in the comparisons because it was the only comparable work in its category.

Table XVII presents results in terms of BD-rate and CCR for all compared works. Since each related work presents different values for BD-rate and complexity reduction, the ratio between these two quantities (BD-rate/CCR) was used to permit comparisons of the competing complexity reduction methods in terms of R-D efficiency loss per computational complexity saved. Using this performance indicator, a method that presents a given BD-rate and CCR is ranked higher than some other method with higher BD-rate but equal (or smaller) CCR (i.e., higher BD-rate/CCR ratio). The works are grouped according to the partitioning structure constrained for complexity reduction and the average results for the proposed schemes are reproduced for comparison at the end of each group. We can conclude from the data in Table XVII that all the schemes proposed in this paper achieved the best BD-rate/CCR ratios in comparison with related works in their corresponding category.

In the *coding tree* early termination category, the best performing related work [9] has a BD-rate/CCR ratio equals to 1.2, which is still larger than the ratio of the scheme proposed in Section IV-B. However, as previously mentioned, the method in [9] is only applicable to intra-predicted CUs, which means that its complexity reductions are only achieved in *All Intra* temporal configurations. All related works focused on PU early termination present a ratio at least twice larger than that obtained with our proposed scheme. Moreover, our scheme achieves a CCR similar to or even larger than the reductions obtained by the related works. Finally, although the only comparable RQT early termination work achieves complexity reduction levels larger than the achieved with our scheme, it produces a BD-rate/CCR ratio equals to 6.36, which is 8.4 times larger than the ratio of our scheme.

The joint early termination schemes proposed in this paper are presented in the last lines of Table XVII and compared with joint *coding tree* and PU early termination methods found in the literature. It is possible to notice that all our joint schemes achieve better results in terms of both R-D efficiency and CCR. Besides, our joint scheme that integrates the three early terminations achieves complexity reductions that go far beyond any other related work, while still maintaining a low BD-rate increase and, consequently, BD-rate/CCR ratios below the achieved by the comparable joint implementations.

VII. CONCLUSION

This paper presented a set of early termination schemes that aim at reducing the computational complexity of the HEVC encoding process. The schemes use DM tools to build decision trees that exploit intermediate encoding results to decrease the high computational complexity inherent to the decision of the best *coding tree*, PU, and RQT structures. Each early termination was separately developed and implemented in the HM software after an extensive analysis on the information gain yield by each possible attribute. The three implementations were then combined in pairs and finally all together in joint schemes, aiming at further reducing the HEVC computational complexity.

The efficiency of the decision trees was validated through extensive experiments using a set of video sequences different from those used in the training phase. Experimental results

have shown that an average complexity reduction of 65% can be achieved when the three early termination schemes are implemented together, with a compression efficiency loss of only 1.36% in BD-rate. Such complexity reductions go beyond those achieved in any other previously published work. Besides, all the proposed schemes in this paper yield better R-D-C efficiency than their comparable related works. The proposed schemes do not add any computationally intensive operations to the HEVC encoding process and the decision trees use only intermediate encoding results, so that they are easily reproducible.

REFERENCES

[1] *High Efficiency Video Coding Text Specification Draft 10*, document JCTVC-L1003, ISO/IEC-JCT1/SC29/WG11, Geneva, Switzerland, 2013.

[2] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[3] J. Vanne, M. Viitanen, T. D. Hamalainen, and A. Hallapuro, "Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1885–1898, Dec. 2012.

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

[5] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Performance and computational complexity assessment of high-efficiency video encoders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1899–1909, Dec. 2012.

[6] *High Efficiency Video Coding (HEVC) Test Model 15 (HM 15) Encoder Description*, document JCTVC-Q1002, ISO/IEC-JCT1/SC29/WG11, Valencia, Spain, 2014.

[7] F. Bossen, B. Bross, K. Suhring, and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1685–1696, Dec. 2012.

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

[9] C. Seunghyun and K. Munchurl, "Fast CU splitting and pruning for suboptimal CU partitioning in HEVC intra coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 9, pp. 1555–1564, Sep. 2013.

[10] L. Jong-Hyeok, P. Chan-Seob, and K. Byung-Gyu, "Fast coding algorithm based on adaptive coding depth range selection for HEVC," in *Proc. IEEE Int. Conf. Cons. Electron.-Berlin*, Sep. 2012, pp. 31–33.

[11] K. Goswami, B. G. Kim, D. Jun, S. H. Jung, and J. S. Choi, "Early coding unit (CU) splitting termination algorithm for high efficiency video coding (HEVC)," *Electron. Telecommun. Res. Inst. J.*, vol. 36, no. 3, pp. 407–417, 2014.

[12] X. Jian, L. Hongliang, W. Qingbo, and M. Fanman, "A fast HEVC inter CU selection method based on pyramid motion divergence," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 559–564, Feb. 2014.

[13] 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. Consum. Electron.*, vol. 57, no. 4, pp. 1866–1874, Nov. 2011.

[14] J. Vanne, M. Viitanen, and T. D. Hamalainen, "Efficient mode decision schemes for HEVC inter prediction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 9, pp. 1579–1593, Sep. 2014.

[15] T. Zhao, Z. Wang, and S. Kwong, "Flexible mode selection and complexity allocation in high efficiency video coding," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1135–1144, Dec. 2013.

[16] T. Guifen and S. Goto, "Content adaptive prediction unit size decision algorithm for HEVC intra coding," in *Proc. Picture Coding Symp.*, May 2012, pp. 405–408.

[17] M. U. K. Khan, M. Shafique, and J. Henkel, "An adaptive complexity reduction scheme with fast prediction unit decision for HEVC intra encoding," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2013, pp. 1578–1582.

[18] H. Wei-Jhe and H. Hsueh-Ming, "Fast coding unit decision algorithm for HEVC," in *Proc. Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, Oct./Nov. 2013, pp. 1–5.

[19] S. Liquan, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective CU size decision method for HEVC encoders," *IEEE Trans. Multimedia*, vol. 15, no. 2, pp. 465–470, Feb. 2013.

[20] X. Shen, L. Yu, and J. Chen, "Fast coding unit size selection for HEVC based on Bayesian decision rule," in *Proc. Picture Coding Symp.*, May 2012, pp. 453–456.

[21] Y. Shi, Z. Gao, and X. Zhang, "Early TU split termination in HEVC based on quasi-zero-block," in *Proc. 3rd Int. Conf. Electr. Electron.*, 2013, pp. 450–454.

[22] C. Kiho and E. S. Jang, "Early TU decision method for fast video encoding in high efficiency video coding," *Electron. Lett.*, vol. 48, no. 12, pp. 689–691, 2012.

[23] G. Fernández, P. Cuenca, L. O. Barbosa, and H. Kalva, "Very low complexity MPEG-2 to H.264 transcoding using machine learning," in *Proc. ACM Int. Conf. Multimedia*, Santa Barbara, CA, USA, 2006, pp. 931–940.

[24] G. Fernandez-Escribano *et al.*, "Low-complexity heterogeneous video transcoding using data mining," *IEEE Trans. Multimedia*, vol. 10, no. 2, pp. 286–299, Feb. 2008.

[25] L. P. Van *et al.*, "Fast transrating for high efficiency video coding based on machine learning," in *Proc. 20th IEEE Int. Conf. Image Process.*, Sep. 2013, pp. 1573–1577.

[26] R. Garcia, D. Ruiz-Coll, H. Kalva, and G. Fernandez-Escribano, "HEVC decision optimization for low bandwidth in video conferencing applications in mobile environments," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops*, Jul. 2013, pp. 1–6.

[27] X. Shen and L. Yu, "CU splitting early termination based on weighted SVM," *EURASIP J. Image Video Process.*, vol. 2013, no. 4, pp. 1–11, 2013.

[28] C. H. Lampert, "Machine learning for video compression: Macroblock mode decision," in *Proc. 18th Int. Conf. Pattern Recognit.*, 2006, pp. 936–940.

[29] *Common Test Conditions and Software Reference Configurations*, document JCTVC-J1100, ISO/IEC-JCT1/SC29/WG11, Stockholm, Sweden, 2012.

[30] G. Bjontegaard, "Calculation of average PSNR differences between RD-curves," in *Proc. 13th VCEG Meeting*, Austin, TX, USA, 2001, pp. 1–5.

[31] *Intel VTune Amplifier XE*. [Online]. Available: http://software.intel.com/, accessed May 6, 2014.

[32] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann, 1993.

[33] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: An update," *ACM SIGKDD Explorations Newslett.*, vol. 11, no. 1, pp. 10–18, 2009.

[34] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley, 1991.

[35] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[36] N. Japkowicz, "The class imbalance problem: Significance and strategies," in *Proc. Int. Conf. Artif. Intell.*, 2000, pp. 111–117.

[37] G. Correa, P. Assuncao, L. Agostini, and L. A. da Silva Cruz, "Complexity scalability for real-time HEVC encoders," *J. Real-Time Image Process.*, to be published.

[38] *x265 HEVC High Efficiency Video Coding H.265 Encoder*. [Online]. Available: http://x265.org/, accessed May 6, 2014.

Guilherme Correa (M'08) received the B.Sc. degree in computer science from Federal University of Pelotas, Pelotas, Brazil, in 2009 and the M.Sc. degree in computer science from Federal University of Rio Grande do Sul, Porto Alegre, Brazil, in 2010. He is currently working toward the Ph.D. degree in electrical and computer engineering with University of Coimbra, Coimbra, Portugal.

He is also a Research Assistant with the Institute for Telecommunications, Lisbon, Portugal, where he is involved in video coding algorithms, power-aware video coding, and digital systems design.

Pedro A. Assuncao (M'98–SM'14) received the Licenciado degree and M.Sc. degrees in electrical engineering from University of Coimbra, Coimbra, Portugal, in 1988 and 1993, respectively, and the Ph.D. degree in electronic systems engineering from University of Essex, Colchester, U.K., in 1998.

He is currently a Professor of Electrical Engineering and Multimedia Communication Systems with Polytechnic Institute of Leiria, Leiria, Portugal, and a Researcher with Institute for Telecommunications, Lisbon, Portugal. He has authored or co-authored over 100 papers in conferences and journals, seven book chapters, and holds four U.S. patents. His research interests include 2-D, 3-D and plenoptic video coding, complexity control and networking, multiple description coding, error concealment, and quality evaluation.

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

He has been a Professor at UFPel since 2002. Since 2013 he has been the UFPel Executive Vice President for Research and Graduation Studies. He is currently the Leader of the Group of Architectures and Integrated Circuits at UFPel. He has authored over 100 published papers in journals and conference proceedings. His research interests include 2-D and 3-D video coding, algorithmic optimization, arithmetic circuits, field-programmable gate array-based design, and microelectronics.

Luis A. da Silva Cruz (M'11) received the Licenciado degree and M.Sc. degrees in electrical engineering from 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 Rensselaer Polytechnic Institute, Troy, NY, USA, 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 an Assistant Professor since 2000. He is currently a Researcher with Institute for Telecommunications, Lisbon, Portugal, where he is involved in video processing and coding, wireless communications, and medical image processing.