Inter-Prediction Optimizations for Video Coding Using Adaptive Coding Unit Visiting Order

Ivan Zupancic, *Student Member, IEEE,* Saverio G. Blasi, *Member, IEEE,* Eduardo Peixoto, *Member, IEEE,* and Ebroul Izquierdo, *Senior Member, IEEE*

***Abstract*—The flexible partitioning scheme and increased number of prediction modes in the High Efficiency Video Coding (HEVC) standard are largely responsible for both its high compression efficiency and computational complexity. In typical HEVC encoder implementations, Coding Units (CUs) in a Coding Tree Unit (CTU) are visited from top to bottom at each level of recursion to select the optimal coding configuration. In this paper, a novel approach is presented in which CUs in a CTU can be adaptively visited also in a reverse bottom to top visiting order. This Reverse CU (RCU) visiting order allows for different algorithmic optimizations for further complexity reduction of many HEVC encoding steps, especially under challenging conditions, such as highly textured or fast moving content. In particular, algorithms to reduce complexity of HEVC depth selection, mode decision and inter-prediction are presented here based on the coding information obtained from higher depths when using the RCU visiting order. Experimental results show that enabling different stages of the proposed algorithm can achieve average speed-ups from 16.3% to 36.6% compared to fast reference HEVC implementation with pre-built speed-ups enabled (up to 51.2% in some cases), for 0.3% to 2.2% BD-rate penalty.**

***Index Terms*—video coding, HEVC, mode decision**

I. Introduction

IMPROVING video coding standards is necessary to allow for more efficient exchange of video signals, especially when considering high spatial or temporal resolutions. For this reason, the H.265/High Efficiency Video Coding (HEVC) standard [1] was developed by the Joint Collaborative Team on Video Coding (JCT-VC) as a successor to the successful H.264/Advanced Video Coding (AVC) [2]. HEVC reportedly achieves 35 - 40% of bitrate reduction for a given level of objective visual quality compared to AVC [3], [4].

It has been shown that one of the key factors contributing to the higher efficiency of HEVC with respect to AVC is its improved flexible partitioning scheme [5]. HEVC allows frames to be flexibly partitioned to adapt to local characteristics of the content. In particular, a frame is first divided in a number of Coding Tree Units (CTUs) of fixed size. Each CTU is then partitioned in Coding Units (CUs) following a recursive quadtree structure. CUs are assigned a depth, depending on their size and the corresponding level of recursion. Each CU at depth d can then be partitioned into four CUs at depth $d+1$ with half the width and height of their parent CU. This process can be recursively repeated up to a minimum CU size of 8×8 luma samples. The encoder selects the best configuration for each CTU, following a process denoted as *depth selection*.

The content of each CU can then be predicted using a variety of different modes, each identifying a different sub-partitioning of the CU in Prediction Units (PUs). In the case of inter-prediction, up to 8 modes are considered, as shown in Figure 1. Motion Estimation (ME) and compensation is then computed independently on each PU. Similarly to previous standards, the SKIP mode is also considered in inter-predicted CUs. When using intra-prediction, a single PU is always considered spanning the entire CU, except for CUs at the highest depth which may be split into 4 equally sized PUs. The process of testing different PU modes and calculating the corresponding Rate-Distortion (RD) cost to select a mode with the minimum cost is referred to as *mode decision*.

Finally, when testing inter-prediction modes, the encoder considers a set of candidate Motion Vectors (MVs) based on the ME algorithm used. An RD cost is estimated for each MV in the set, to select the MV at minimum cost for the current PU. This process is repeated for all PUs within a CU, and it is referred to in this paper as *prediction*.

Most HEVC encoder implementations test a variety of possible options, and select the optimal coding configuration and parameters specific for the current portion of the signal being encoded based on an RD cost. In theory, in order to obtain the best possible coding efficiency, the encoder should test all possible combinations of these options to find the optimal solution for the current block. Clearly, such approach is extremely computationally expensive and can drastically limit the usage of HEVC, especially when handling high resolution content. For this reason, many authors have proposed efficient HEVC encoder schemes under different constraints. In this paper, a novel scheme is presented to target HEVC encoding at low computational costs, based on adaptive block testing order. When analyzing small blocks first, information collected during the encoding can be used to limit the number of testing options in larger blocks and hence reduce the complexity.

Ivan Zupancic and Ebroul Izquierdo are with the School of Electronic Engineering and Computer Science at Queen Mary University of London, Mile End Road, E1 4NS, London, UK (email: {i.zupancic, ebroul.izquierdo}@qmul.ac.uk)

Saverio G. Blasi was with the School of Electronic Engineering and Computer Science at Queen Mary University of London at the time when this work was done, and is now with the British Broadcasting Corporation, Research and Development (BBC R&D), London, UK (email: saverio.blasi@bbc.co.uk).

Eduardo Peixoto is with the Department of Electrical Engineering, Universidade de Brasilia, Brasilia, Brazil (e-mail: eduardopeixoto@ieee.org)

This research utilized Queen Mary's MidPlus computational facilities, supported by QMUL Research-IT and funded by EPSRC grant EP/K000128/1. This work was partially supported by CNPq under grant 444864/2014-8. Some of the content belongs to 4EVER consortium, rights reserved to the 4EVER partners and their licensors. The 4EVER research Project is coordinated by Orange Labs and has received funding from the French State (FUI/Oseo) and French local Authorities (Région Bretagne) associated to the European funds FEDER.

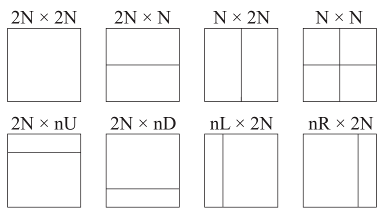


Fig. 1. HEVC inter-prediction partitions.

The rest of this paper is organized as follows. An overview of previously proposed fast algorithms for HEVC is presented in Section II. An analysis of typical HEVC encoder implementations using available fast algorithms is presented in Section III. The proposed adaptive CU visiting order and Reverse CU (RCU) optimizations are then illustrated in Section IV. A comprehensive experimental evaluation of the proposed scheme is presented in Section V, and finally Section VI concludes the paper.

II. Related Work

The combination of depth selection, mode decision and inter-prediction is extremely time consuming for the encoder, which has to test a huge number of different options before selecting the optimal configuration for each CTU. For this reason, many methods have been proposed to reduce the complexity of each of these three processes.

A. Fast Methods for Depth Selection

After introducing a new CTU partitioning scheme in HEVC, many methods have been proposed for fast depth selection. A speed-up is already included in the HM reference software [6], referred to as Early CU termination (ECU) [7]. The ECU examines the optimal mode selected after mode decision on a given CU. If this is SKIP, then the CU is not split and no sub-CUs are tested. A method was proposed by Zhang *et al.* [8] in which similarity among neighboring CTUs is studied to reduce the depth search range. CTUs are divided in three classes based on a defined similarity measure and only a restricted range of depths is tested for each class. Xiong *et al.* [9] proposed an approach in which the optical flow is estimated from the down-sampled frames and then used to compute a parameter which decides whether CU should be split or not. An approach was proposed by Shen *et al.* [10] which limits the CTU depth levels. When using this algorithm, a weighted sum of the depths from spatially and temporary neighboring CTUs is used to predict the depth range for a current CTU. Finally, Correa *et al.* [11] recently proposed schemes to early terminate the CU partitioning scheme and predict the optimal PU and residual quadtree structures. The schemes are based on decision trees obtained through data mining techniques.

B. Fast Methods for Mode Decision

Mode decision has been extensively studied in the past and numerous fast methods were proposed in the context of HEVC. There are already two pre-built speed-ups included in the HM reference software, referred to as Early Skip Detection (ESD) [12] and Coding Flag Mode (CFM) [13]. When using ESD, the encoder tests the 2N×2N inter mode first. SKIP mode is then tested and its cost is compared with the cost obtained for the inter 2N×2N mode. In case SKIP is selected, no further modes are tested on the current CU. When using CFM, the root *coded block flag* (CBF) of the current PU is analyzed after testing each inter-prediction mode. Such flag specifies whether all quantized and transformed residuals in all components are zero or not. If CBF is zero, the current mode is good in an RD sense, and no other modes are tested for the current CU.

In the second part of the method proposed by Shen *et al.* [10], testing of inter and intra modes is skipped for homogeneous regions. Three early termination strategies to avoid testing inter and intra modes on sub-CUs are proposed based on detection of homogeneous regions. Another method was also proposed by Shen *et al.* [14] in which inter-prediction modes of spatially and temporally neighboring CUs are used to predict the motion activity of the current CU. Based on this prediction, CUs are divided in categories where only selected modes are tested for each category. Xiong *et al.* [15] proposed an algorithm based on Markov random fields, where a maximum *a posteriori* approach based on the RD cost is conducted to evaluate whether CUs should be split or not.

A method was proposed by Vanne *et al.* [16] to speed up the inter mode decision in HEVC. Their approach is focused on the selection of Symmetric (SMP) and Asymmetric (AMP) Motion Partition modes. Testing of either SMP and/or AMP modes is limited based on the current CU depth or the Quantization Parameter (QP) value. Recently, Ahn *et al.* [17] proposed a method which utilizes the sample adaptive offset parameters, MVs, Transform Unit (TU) size and CBF information to estimate the temporal complexity. Spatial and temporal complexity are then combined to develop fast decision methods. Finally, in our previous work on fast mode decision [18], statistics on the modes and costs found on previously encoded PUs are collected and used to build online cost distributions. Modes are then sorted based on the expected probability, and only the most probable modes are tested.

C. Fast Methods for Inter-Prediction

The inter-prediction step has also been investigated to find fast methods to compute the MVs for a PU. The HM reference software already makes use of a fast ME algorithm based on Enhanced Predictive Zonal Search (EPZS) [19]. When using EPZS, a certain number of MV candidates are considered. These are computed using information from spatially or temporally neighboring blocks, and also testing some predefined MVs. Then, pattern searches are performed in the surrounding of such candidates to find the optimal MV solution for the current block. Recently, we proposed a fast integer precision ME method [20] to early terminate the ME process. The method is applied immediately before performing EPZS, on each of the possible EPZS starting points to possibly skip the integer ME search. Finally, in our previous work on fractional precision ME, a fast algorithm for reducing complexity of HEVC fractional precision ME was proposed [21]. In this approach, local features of each PU (behavior of the residual error samples) and global features of each frame (amount of edges) are used to adaptively skip the fractional precision ME.

Most of the above-mentioned algorithms are tailored to work particularly well for sequences which contain many homogeneous regions or slow motion activity. Typically, these conditions also correspond to sequences which can be easily compressed using HEVC, resulting in very high compression efficiency. Very few methods have been developed to target fast encoder decisions under more challenging conditions, such as highly textured regions or fast motion activity. In these cases, most of the reported methods provide very little complexity reductions or typically suffer from high efficiency losses. Conversely, the methods proposed in this paper make use of a different encoding scheme based on a reverse CU visiting order within a CTU to optimize all three encoding stages. As a result, the proposed encoders significantly reduce the computational complexity of HEVC encoding at low efficiency losses, and work particularly well when encoding highly textured or fast motion content.

III. Analysis of the HEVC Encoder

While historically in previous video coding technology the ME module typically accounted for the majority of the coding complexity [22], the complexity of modern HEVC encoder implementations is more equally distributed among the mode decision, depth selection and prediction steps. In this section, we analyze the algorithms for fast encoding used in the HM reference codec and present an encoder complexity profile.

To measure compression efficiency, the Bjøntegaard Distortion rate (BD-rate) [23] is used. This is a performance measure of an encoder with respect to an anchor at the same levels of quality, in percentage. Positive values of the BD-rate mean a decrease in compression efficiency with respect to the anchor. Total encoding speed-up was also computed for each of the 4 tested QPs for each sequence using the following formula:

$$\Delta T_i = \frac{T_A - T_M}{T_A} \times 100\%, \tag{1}$$

where T_A denotes the total encoding time for the anchor encoder, and T_M denotes the total encoding time for the tested encoder. Finally, arithmetic mean of ΔT_i for all 4 points was computed to obtain the average encoding speed-up ΔT.

A. Performance of Fast HEVC Implementations

The JCT-VC established a set of coding configurations, referred to as Common Test Conditions (CTC) [24], which regulate usage of the HM software to provide a uniform benchmark for testing and evaluation of video coding algorithms. When using CTC, the encoder computes an RD cost for every possible option. As expected, this gives the best RD performance at a very high computational cost. In the rest of this paper, for brevity, we will refer to the usage of HM under these conditions as the *CTC* encoder.

A simple way to reduce the computational cost of the *CTC* encoder could be to restrict the range of depths that can be visited within the CTU. Here we show three example encoders, namely *Depth*$_A$, *Depth*$_B$, and *Depth*$_C$, each corresponding to testing a restricted range of depths inside the CTU. In particular, *Depth*$_A$ tests only CUs of sizes 64×64, 32×32 and 16×16; *Depth*$_B$ tests only CUs of sizes 64×64 and 32×32; and *Depth*$_C$ tests only CUs of sizes 32×32, 16×16 and 8×8. These encoders were tested on a variety of sequences at different resolutions, using 4 QP values (22, 27, 32, and 37). Results for these three simple encoders are shown in Table I, with *CTC* as an anchor. On average, *Depth*$_A$ achieves 3.7% BD-rate increase for 32.1% time savings, *Depth*$_B$ obtains 13.0% BD-rate increase for 61.1% time savings, while *Depth*$_C$ achieves 3.7% BD-rate increase for 15.7% time savings. It can be seen that these losses are heavily dependent on the sequences being encoded. In some cases, obtained losses are unacceptable for most applications.

There are more sophisticated speed-up tools available in the literature, such as the ECU, ESD and CFM speed-ups pre-built in the HM encoder implementation. Each of these speed-ups was enabled on top of a *CTC* encoder, to obtain the *ECU*, *ESD* and *CFM* encoders, respectively. Results of such encoders are also presented in Table I. From the table, it can be seen that the *ECU* encoder obtains on average a 0.4% BD-rate increase for 39.6% time savings. Similarly, the results for *ESD* show a 0.2% BD-rate increase for 32.1% time savings. Finally, enabling the CFM tool in HM achieves on average a 0.7% BD-rate increase for 36.3% time savings. The ESD, ECU and CFM speed-ups can be combined on top of *CTC* to provide an even faster implementation. The corresponding encoder is referred in the rest of this paper as *FAST-HM*. The results for *FAST-HM* are also shown in Table I, where it can be seen that it achieves on average a 1.5% BD-rate increase for 49.4% time savings.

More detailed examination of the results in Table I reveals some cases where *FAST-HM* has a very limited impact on reducing the encoding time. This happens typically in case of highly textured and/or fast moving sequences. In these cases, the SKIP mode is selected rarely (meaning ESD and ECU are not often used), and similarly many non-zero residuals can be expected due to difficulty in producing good predictions of each PU (meaning CFM is also not used often). In the next subsection, the encoding time needed for particular HEVC tasks when using *CTC* and *FAST-HM* was analyzed for selected sequences.

B. Profiling Analysis of the HM Encoder

The HM encoder complexity was further analyzed for two specific sequences: *Kimono* and *Riverbed*. Figure 2 shows the profiling results for these sequences when using both the *CTC* and *FAST-HM* encoders. Three parts of the HEVC encoding loop are shown in the figure: the time spent by the encoder at each depth, the time spent testing each of the prediction modes while testing depth 1, and the time spent doing different ME tasks while testing depth 1. These sequences were selected as they represent the cases in which the built-in HM speed-ups have a high and low impact (as can be seen from Table I).

For the *Riverbed* sequence, the available speed-ups do not work very well. It can also be seen in Table I that the three simple depth constrained encoders outperform the speed-ups found in *FAST-HM*. For instance, *Depth*$_B$ provides very

TABLE I
RESULTS FOR VARIOUS MODIFICATIONS OF HEVC ENCODER VERSUS *CTC-HM* ANCHOR UNDER RA-MAIN CONFIGURATION.

		Depth$_A$		*Depth$_B$*		*Depth$_C$*		*ESD*		*ECU*		*CFM*		*FAST-HM*	
	Sequence	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]
3840 × 2160	Manege	8.5	30.2	33.1	59.6	1.8	16.6	0.7	30.4	0.8	34.8	2.1	35.2	3.3	47.6
	Marathon	2.3	30.7	10.1	60.6	2.4	15.2	0.4	24.1	0.4	27.4	0.9	29.2	1.7	36.9
	ParkJoy	0.7	32.0	4.1	61.8	2.0	14.6	0.2	21.1	0.4	24.6	0.3	24.4	1.1	32.0
	Sedof	7.2	31.5	25.8	60.9	2.4	14.4	0.4	32.7	0.7	38.8	1.4	38.5	2.8	50.4
BBC-UHD	Book	1.5	30.5	6.0	59.2	12.5	16.9	0.2	37.8	−0.1	46.7	0.6	40.3	0.6	56.3
	CalendarAndPlants	3.4	32.2	14.1	60.6	6.9	15.9	0.4	45.0	0.8	58.0	1.0	48.5	2.5	68.8
	MenAndPlants	2.1	32.4	10.1	61.0	5.6	14.3	0.3	39.3	0.3	49.2	0.9	43.7	1.4	59.1
	ParkAndBuildings	3.3	33.2	8.7	61.9	6.7	13.6	0.2	41.7	0.3	52.8	0.5	46.2	1.2	63.9
	Vehicles	1.9	32.9	6.5	61.8	3.3	15.3	0.2	37.6	0.4	48.2	0.4	40.7	1.0	58.0
EBU-UHD	LupoCandlelight	3.2	32.3	9.6	60.6	10.8	16.5	0.2	38.5	0.3	56.6	0.5	47.2	1.1	66.9
	RainFruits	1.7	32.7	9.0	61.2	4.9	15.7	0.2	43.0	0.4	55.7	0.4	45.7	1.4	65.9
2560 × 1600	ParkJoy	1.0	31.5	5.5	61.3	2.6	16.1	0.2	22.6	0.5	26.6	0.5	25.8	1.4	34.1
Class A	Traffic	5.3	31.9	16.3	60.7	3.2	17.2	0.2	40.7	0.6	50.5	0.7	46.0	2.4	63.0
	PeopleOnStreet	7.3	29.0	29.0	59.2	1.5	17.4	0.4	20.2	0.6	18.7	1.5	28.1	2.4	31.5
1920 × 1080	CrowdRun	7.1	30.5	25.5	60.8	0.8	14.6	0.3	19.2	0.6	21.4	1.1	26.6	2.3	30.9
	DucksTakeOff	−0.1	33.0	1.3	63.3	1.4	14.0	0.1	14.2	0.3	16.7	0.2	16.7	0.9	22.1
	Riverbed	0.1	31.4	0.6	62.4	0.5	14.5	0.1	5.8	0.0	3.2	0.1	7.5	0.2	7.0
Class B	Kimono	1.3	32.9	5.0	61.0	3.9	16.2	0.3	33.9	0.5	40.9	0.6	38.6	1.5	51.1
	Parkscene	3.8	32.8	12.6	60.6	2.4	16.9	0.2	38.8	0.5	47.5	0.7	44.0	1.9	59.4
	Cactus	6.3	31.8	21.2	60.4	2.5	16.1	0.3	33.0	0.7	39.4	0.8	37.2	2.5	50.4
	BasketballDrive	4.3	31.3	15.0	60.5	4.0	15.8	0.4	30.3	0.4	35.0	1.0	35.2	1.5	45.6
	BQTerrace	4.3	32.1	14.4	61.1	4.2	15.9	0.4	35.9	0.7	42.8	0.8	40.2	2.5	55.1
1280 × 720	ParkJoy	3.0	31.1	14.0	59.9	0.8	15.9	0.2	21.5	0.5	24.2	0.5	25.9	1.6	33.0
	ParkRun	0.9	32.5	5.5	60.8	0.8	15.0	0.1	23.6	0.3	28.8	0.2	25.7	0.7	35.6
	DucksTakeOff	0.2	33.7	2.1	63.4	1.2	14.8	0.1	17.2	0.4	20.3	0.1	19.3	1.0	25.3
Class E	FourPeople	7.4	33.6	19.6	61.4	2.7	16.2	0.0	48.6	0.0	63.8	0.3	51.4	0.7	75.0
	Johnny	7.5	34.6	19.5	61.9	6.5	17.1	0.0	53.3	−0.1	69.9	0.6	55.9	0.6	81.2
	KristenAndSara	7.8	33.9	19.8	61.6	5.2	17.0	0.2	49.7	0.0	65.0	0.7	53.1	0.8	76.3
Average		3.7	32.1	13.0	61.1	3.7	15.7	0.2	32.1	0.4	39.6	0.7	36.3	1.5	49.4

small efficiency losses for significant time savings. The main reason is that the ESD, ECU and CFM speed-ups are rarely triggered in this sequence, hence *FAST-HM* does not lead to any significant computational complexity savings. This can be seen in the charts in Figure 2, where the distribution of encoding time per task is largely unchanged between *CTC* and *FAST-HM*. To further validate these findings, we also computed the number of times each of the speed-ups is used when using the *ECU*, *ESD*, and *CFM* encoders for each CU in the first 8 frames of the sequences. The analysis showed that ECU, ESD, and CFM were used on 14.6%, 22.5%, and 28.9% of the CUs, respectively.

An opposite behavior can be observed for the *Kimono* sequence, where existing speed-ups work very well. The distribution of encoding time in this sequence is quite different between *CTC* and *FAST-HM*, as shown in Figure 2. It can be seen, for instance, that the relative complexity of depth 0 is significantly higher for *FAST-HM* than for *CTC*. This happens because when ECU is triggered the encoder avoids testing lower depths, resulting in increased time distribution towards low depths when using the speed-ups. Also, the relative complexity of the SKIP mode is higher in *FAST-HM* than in *CTC*, as this mode is always tested while both ESD and CFM may be triggered to avoid testing other modes. The number of times each speed-up is triggered was also computed for the first 8 frames of *Kimono* and shows that ECU, ESD, and CFM were used on 32.0%, 70.4%, and 75.2% of the CUs, respectively.

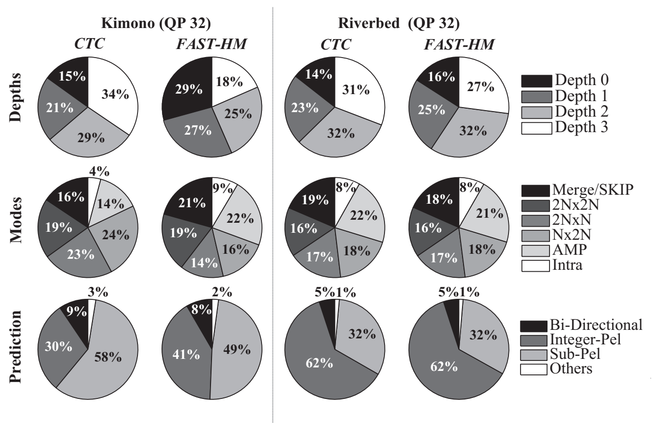


Fig. 2. HEVC encoding time distribution for different depths, modes and inter-prediction tasks are shown for *Kimono* and *Riverbed* sequence using the *CTC* and *FAST-HM* encoders.

These results highlight that existing speed-ups can provide high encoding complexity reductions, and cannot be ignored in a practical fast HEVC implementation. Hence, we enable these speed-ups in the proposed algorithm. However, they may not be as effective under challenging conditions, such as highly textured or fast moving sequences. Therefore, we also address these cases in this paper.

IV. FAST HEVC CODING USING REVERSE CU

This section presents the proposed Reverse CU (RCU) framework and the novel algorithms and tools developed with the goal of reducing HEVC encoding complexity. The algorithms are mainly targeted at coding under challenging conditions, such as high resolution of the input sequences. For this reason, only the content at resolutions equal or higher than 1280×720 luma samples was considered here.

In particular, content at 3840×2160 (2160p), 2560×1600 (1600p), 1920×1080 (1080p) and 1280×720 (720p) luma samples was considered. For simplicity, during the tests these resolutions were grouped into two resolution groups: the *UHD-group* includes the 2160p and 1600p formats, while the *HD-group* includes the 1080p and 720p formats.

The algorithmic development presented in the rest of this section was performed using a 10 sequence *Training set* comprising 5 sequences from *UHD-group* and 5 from *HD-group*. In particular, the 2160p sequences *ConstructionField*, *RushHour*, *TallBuildings*, *Library*, and *TrafficFlow* [25] were part of *UHD-group*, while the 1080p sequences *ChristmasTree*, *Station*, *Wisley*, and *ParkJoy*, along with the *Mobcal* 720p sequence were part of *HD-group*. These sequences were selected to represent a variety of different content. Unless otherwise specified, all experiments reported in this section were obtained after encoding the first 9 frames (including the first intra frame) from the *Training set* using JCT-VC CTC [24] under RA-Main configuration. Note that sequences in the *Training set* were excluded from the test set used in the Results section to evaluate performance of the proposed algorithms.

A. The Reverse CU Framework

The typical testing order of CUs within a CTU (assuming a maximum depth of 2) is illustrated in Figure 3 (a), referred to as Normal CU (NCU) visiting order. In this paper, another approach is also used, referred to as RCU visiting order, in which CUs in the CTU are visited in a different order. When RCU is used on a CTU, each time the encoder needs to select whether a CU should be encoded in its full configuration or it should be split, the latter is tested first. Due to the recursive nature of the quadtree, applying RCU on a CTU implies that the first CUs to be tested are those at the maximum allowed depth. The RCU visiting order is presented in Figure 3 (b).

CU visiting order can also be considered in the context of graph theory, as a tree search problem. The RCU visiting order corresponds to Post-order tree traversal [26], in which all the child nodes are always visited before their parent nodes. That is different from Depth-First Search (DFS) strategy [27] used in conventional HEVC, in which child nodes are visited depending on their parent nodes. Notice that a bottom-to-top approach has already been used before in video coding, but in the context of intra-prediction, to visit the transform blocks in the recursive structure used for transform and quantization [28], or to calculate gradient intensity of the CU [29]. Recently, it has also been used in the context of transcoding from AVC to the HEVC [30].

Assuming that all CUs in the CTU are tested, simply using RCU has no effects on HEVC coding. The RCU framework can be used though to define fast algorithms, as illustrated in the rest of this paper. Compared to our previous work [31], the method proposed in this paper introduces numerous enhancements and optimizations. In our previous work, RCU visiting order was selected in cases when at least two neighboring CUs had maximum depth level greater or equal to 2, and there was no depth selection. In this paper, a thorough analysis has been performed to define an enhanced depth prediction and RCU selection algorithm. Mode decision in our previous work was based on MV Variance Distance (MVVD), while in the proposed approach we use probabilistic model based on Naïve Bayes classifier. Finally, in this paper we are introducing a prediction stage based on MVVD, which does not have a correspondent in our previous work.

B. RCU Framework and Depth Selection

Among the new coding tools introduced in HEVC, usage of larger CU sizes is particularly suited for encoding homogeneous and uniform regions in a video sequence. Many approaches for fast depth selection, such as ECU, have been proposed with the aim of detecting those cases to stop the recursion in the CTU and avoid testing unnecessary depths. On the other hand, the analysis presented in Section III showed that there are sequences where the ECU speed-up is rarely used or provides relatively high efficiency losses. This mainly happens in those regions which are difficult to predict. Such regions are best encoded using very small CUs at high depths, which allow the prediction to be computed at a finer level of granularity. Therefore, if a method to predict the expected maximum depth in the CTU is defined, and if such predicted depth is relatively high, then RCU can be used to start examining CUs at high depths, and stop the search before reaching lowest depth of the CTU, therefore possibly reducing complexity without affecting the coding efficiency.

Hence, in order to utilize the RCU framework and correspondingly reduce the complexity of HEVC depth selection, RCU should be selectively used along with NCU, with the following rationale: when the predicted maximum depth for a given CTU is low, NCU should be used, while in contrast, when the predicted maximum depth is high, RCU should be selected. Since CUs at depth 0 cannot coexist in a CTU with any other CUs at other depths, to further reduce computational complexity when RCU is used in a CTU, CUs at depth 0 are never tested, and conversely, if NCU is used, CUs at depth 3 are never tested. Notice that when using the conventional NCU visiting order, the pre-implemented ECU speed-up can also be used, as in typical HEVC implementations. The problem of selecting whether to use the RCU or NCU visiting order becomes then that of predicting the maximum depth for a given CTU, which has already been investigated in the past [8] - [10]. In this paper, a similar approach to predict the

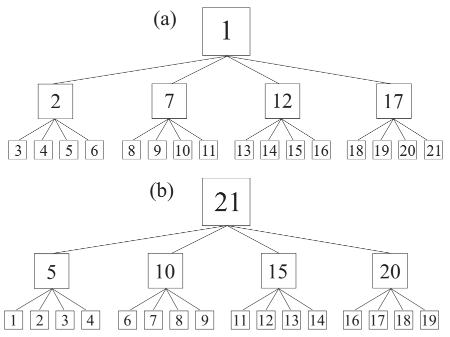


Fig. 3. Visiting order of CUs using conventional HEVC implementations (a) or the proposed RCU approach (b), assuming a maximum depth of 2.

TABLE II
MAXIMUM DEPTH CORRELATION BETWEEN THE CURRENT CTU AND NEIGHBORING REGIONS $r_0, ..., r_4$.

Region	r_0	r_1	r_2	r_3	r_4
Correlation	0.76	0.73	0.66	0.70	0.71

maximum CTU depth is used based on the depths found in the spatially neighboring CTUs.

In particular, assume that the encoder needs to encode a certain CTU with a size of L×L luma samples. All previously encoded sub-CUs contained within five square regions of L/2×L/2 samples immediately on top, top-left, and left of the current CTU are considered. These regions are illustrated in Figure 4, denoted as $r_0, ..., r_4$. For each $i = 0, ..., 4$, all sub-CUs contained within the region r_i are considered, and finally the maximum depth of such sub-CUs is extracted, denoted here as D_i. Notice that in case one of the neighboring CTUs is encoded with a depth 0, the maximum depths in the regions contained within such CTU are considered to be 0 (e.g., D_0 and D_1 are 0 in case the CTU on the left is encoded with depth 0).

The method proposed in this paper is based on the assumption that such maximum depths are highly correlated with the maximum depth found in the current CTU. To validate this assumption, the first 9 frames from *Training set* were encoded using the *CTC* encoder. For each CTU, the maximum neighboring depths D_i, $i = 0, ..., 4$ were computed. Finally, the maximum depth found in the current CTU was extracted, referred to here as D_{max}. The correlations between D_{max} and D_i, $i = 0, ..., 4$ were then calculated, as shown in Table II. The table shows that high correlation can be expected between the maximum depth found in each region r_i, and the maximum depth of the current CTU. It also highlights the fact that all the correlations are very similar. Following from these findings, it is clear that the depths D_i can be used to predict the expected maximum depth in the current CTU. Such prediction is performed in this paper by means of a simple weighted sum. Moreover, due to the similarity among all correlations in Table II, all weights w_i can be set to 1. Finally, the following parameter can be defined:

$$D_p = \sum_{i=0}^{N-1} D_i w_i, \tag{2}$$

where N is the number of the available neighboring regions of L/2×L/2 samples, $N \leq 5$. D_p ranges from 0 to 15 and is referred to in this paper as the Depth Sum.

A further analysis was then performed to study the relationship between the Depth Sum and the actual maximum depth found in a CTU. Table III shows the percentage of CTUs coded with a certain maximum depth for each value of the Depth Sum. Clearly, higher values of D_p correspond to higher maximum depths in the current CTU, which correspond to those cases where RCU should be used. Conversely, lower values of D_p correspond to generally low depths in the current CTU which means NCU should be used. By appropriately defining a threshold T_p, NCU is used on a CTU in case $D_p < T_p$, and RCU is used otherwise.

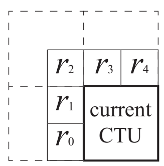


Fig. 4. Neighboring CUs considered while deciding about using RCU.

The problem is then that of appropriately defining the threshold T_p to perform such decision. To derive this threshold, the sum of two F-scores was computed using the depth information after encoding first 9 frames from *Training set*. F-score has been widely used in statistical analysis of binary classification as an accuracy measure [32]. It is defined as the harmonic mean of *precision* and *recall*, where *precision* is defined as the number of True Positive (TP) results divided by the number of all positive results, where the latter is obtained as the sum of TP and False Positive (FP) results. Similarly, *recall* is defined as the number of TP results divided by the sum of TP and False Negative (FN) results. The first F-score was computed considering the accuracy of selecting RCU. In particular, a CTU was considered as a TP if RCU is selected and the maximum depth in that CTU is different than 0, whereas a CTU was considered as an FP if RCU is selected but the maximum depth of the CTU is 0. Finally, FN CTUs are those where NCU is selected, but the maximum CTU depth is 3. Analogously, the second F-score was computed to measure the accuracy of NCU following a similar process. The two F-scores were summed for all possible values of T_p, and the threshold corresponding to the highest sum was selected. The highest F-score sum of 1.6653 was obtained for $T_p = 6$ and hence it is used in this paper.

Formally, assume that CUs can span from a maximum depth D_{max} to a minimum depth D_{min}. Then, in case a CTU is classified to use RCU, depth 0 is not tested, namely $D_{min} = 1$. Conversely, if NCU is used, depth 3 is not tested, namely $D_{max} = 2$. In addition to this, even more speed-ups can be obtained by avoiding testing of low depths in case the Depth Sum is very high. For instance, if D_p is greater than 13, it means that most of the neighboring CUs in the surrounding of the current CTU were encoded at depth 3. In this case, following from the correlations in Table II, it is unlikely that the current CU should be encoded with low depths. Formally, a second threshold $T_{p2} > T_p$ can be defined, such that when $D_p \geq T_{p2}$, also testing of CUs at depth 1 is avoided in the CTU. Following from experimental analysis on *Training set*, $T_{p2} = 14$ is selected in this paper as a trade-off between coding efficiency and encoder complexity. It is worth mentioning that limiting the depth selection to test only one depth could potentially lead to error propagation, since it significantly affects the value of D_p. This may mislead the encoder to always test only one depth for the rest of the frame, and hence was not considered in the proposed algorithm.

The depth prediction accuracy of the proposed method was further validated on first 8 frames (excluding the first intra frame) from *Training set*, and compared with other depth prediction scheme [10]. Experiment showed that depth prediction accuracy for the method proposed in [10] is 94.52%, while the

TABLE III
PROBABILITIES OF MAXIMUM CTU DEPTH OCCURRENCE FOR DIFFERENT VALUES OF DEPTH SUM.

	Depth Sum															
Maximum CTU depth	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0.9536	0.7976	0.6215	0.4144	0.3230	0.2488	0.2176	0.1718	0.1350	0.1029	0.0650	0.0435	0.0258	0.0126	0.0053	0.0022
1	0.0309	0.1252	0.2358	0.1742	0.2065	0.3887	0.2085	0.1634	0.1266	0.0904	0.0636	0.0510	0.0280	0.0138	0.0094	0.0022
2	0.0103	0.0489	0.0956	0.3485	0.2102	0.2069	0.3019	0.2680	0.2647	0.1997	0.2643	0.1143	0.0982	0.0410	0.0309	0.0052
3	0.0052	0.0283	0.0471	0.0629	0.2603	0.1555	0.2720	0.3968	0.4738	0.6070	0.6070	0.7911	0.8480	0.9326	0.9544	0.9904

depth prediction accuracy for our method was 96.01%. Such high accuracy can be explained from the fact that the depth prediction is performed using information from regions highly correlated with the current CTU. Note that in case when not all the neighboring CU information is available, selection between using a NCU or RCU is obtained using a different threshold value ($Tp = 4$). When encoding the first CTU in a frame, no neighboring CUs are available. Therefore, NCU visiting order is used without any depth limitation. The final depth selection algorithm with the optimizations proposed here (illustrated in Figure 5), is referred to as *Stage 1* in the rest of this paper.

C. Mode Decision with RCU Framework

In case RCU is used on a CTU, when testing a CU at depth $d < D_{max}$, information regarding the encoder decisions on the four sub-CUs at depth $d + 1$ is already available. This information could possibly be used to reduce the number of modes to test for the current CU. Many of the parameters selected for the sub-CUs could be used at this purpose. In particular, it can be assumed that the prediction modes found on the sub-CUs are highly related with the mode selected on the CU.

An analysis has been carried out to investigate such assumption. Prediction modes were divided in 3 classes: SKIP, Inter, and Intra. The first 8 frames (excluding the first intra frame) from *Training set* were encoded and used to generate Table IV. The table shows the sample probability that the mode of a CU is in one of these three classes, depending on the number of modes in each class found in the sub-CUs. All possible combinations of classes in the sub-CUs are shown in the table. It can be observed from the table that there is a high dependency between the optimal modes in sub-CUs and the optimal mode for current CU. In all cases when all 4 sub-CUs are encoded using modes in the same class, that same class is used with the highest probability in the upper CU. The same happens in most cases when 3 sub-CUs are encoded using modes in the same class. More importantly, if a certain class

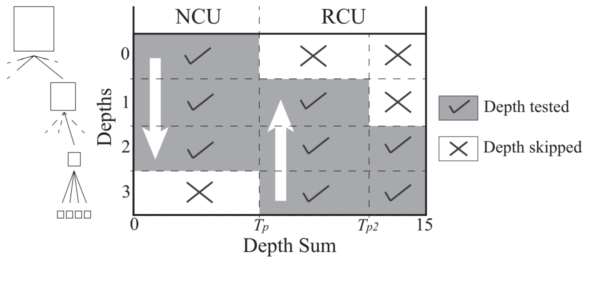


Fig. 5. RCU depth selection depending on the Depth Sum (*Stage 1*).

TABLE IV
PROBABILITY OF A SPECIFIC MODE TO BE SELECTED AS AN OPTIMAL FOR A GIVEN CU DEPENDING ON THE OPTIMAL MODES FOUND IN 4 SUB-CUS.

Number of modes selected in 4 sub-CUs			Optimal mode selected in upper CU		
SKIP	Inter	Intra	SKIP	Inter	Intra
4	0	0	0.9447	0.0065	0.0488
0	4	0	0.0077	0.9816	0.0107
0	0	4	0.0463	0.0078	0.9459
3	1	0	0.5258	0.2126	0.2616
3	0	1	0.3614	0.0103	0.6284
1	3	0	0.1611	0.7266	0.1123
1	0	3	0.1006	0.0033	0.8961
0	3	1	0.0703	0.6628	0.2668
0	1	3	0.0555	0.1058	0.8387
2	1	1	0.2766	0.0997	0.6237
1	2	1	0.1980	0.3007	0.5013
1	1	2	0.1396	0.0694	0.7910
2	2	0	0.3199	0.4649	0.2152
2	0	2	0.1817	0.0060	0.8123
0	2	2	0.0737	0.3286	0.5976

was not used on any of the sub-CUs, the probability of that mode class being used on the current CU was the lowest. The behavior shown in Table IV proved to be sequence, resolution, QP, and temporal layer independent.

From these findings, it is clear that the modes selected on the sub-CUs can be used to help mode decision on the current CU. Using only this information, instead of more complex features, has the advantage that the decision rule can be performed in a relatively fast way, as only four integer numbers are involved. In this paper, a classifier is defined for this purpose based on the well-known Naïve Bayes algorithm [33].

Naïve Bayes algorithms are classification algorithms where the posterior probability of making a decision conditioned to a set of features is expressed in terms of the prior probability of such decision and the likelihood of these features for each class. Naïve Bayes classifiers are one of the simplest among the probabilistic classifiers, but despite that, they are successfully used in various fields. They assume the value of a particular feature is independent of the value of any other feature, given the class variable. Formally, assume a decision has to be made regarding the outcome of a random variable Y, which can assume values in $y_1, ..., y_D$, based on the observation of a set of n attributes $X_1 ... X_n$. The class with maximum posterior probability is obtained as:

$$Y = \arg\max_{y_k} P(Y = y_k) \prod_{i=1}^{n} P(X_i | Y = y_k). \qquad (3)$$

Where the distributions $P(Y)$ and $P(X_i|Y)$ are estimated from the training data. A complete derivation of Naïve Bayes classifier can be found elsewhere [33].

Similarly, a Bayes classifier can be used to estimate the posterior probability of Y based on the observation of the attributes, and then sort the classes based on such probabilities. In the case of HEVC mode decision, such a classifier can be used to determine the M most probable modes for a given CU based on the optimal modes found in its 4 sub-CUs, and then only test those modes. The optimal modes in each of the 4 sub-CUs can be considered as random variables $X_1 ... X_4$. Each X_i can assume values from 0 to 8, where $X_i = 0$ corresponds to using the SKIP mode on sub-CU i, values $X_i = 1, ..., 7$ correspond to using one of the 7 inter-prediction modes (considering N×N along with 2N×2N), and $X_i = 8$ means the sub-CU i was intra-predicted. Similarly, the mode to select on the current CU can be considered as a random variable Y, which again can assume 9 values. The prior probabilities $P(Y = y_k)$, $k = 0, ..., 8$ can be computed from training sequences as the number of times a certain mode is selected. Similarly, the likelihoods $P(X_i = x_n | Y = y_k)$ can be computed for each sub-CU i, as the number of times mode x_n was selected on i, given y_k was selected on the upper CU.

To obtain these probabilities, the first 8 frames (excluding the first intra frame) from *Training set* were used. An example of conditional probabilities for selecting a mode in sub-CU 0, given the mode in the current CU, is shown for *HD-group* in Table V. For instance, notice that there is 98.51% probability that the optimal mode for the first sub-CU is SKIP, given the optimal mode found for the current CU is SKIP. Finally, while encoding a CU at depth $d < D_{max}$, the four optimal modes found in sub-CUs at depth $d + 1$ are considered. Correspondingly, the posterior probabilities $P(Y = y_k | X_1 ... X_n)$ are computed for $k = 0, ..., 8$. These are sorted in descending order, and only the M most probable modes are tested, where M is a tuning parameter. Smaller values of M correspond to testing fewer modes on each CU, with correspondingly higher speed-ups, while lower losses and speed-ups can be expected when using high values of M.

It is worth pointing out that existing mode decision methods, such as ESD and CFM, are in theory not fully compatible with the proposed mode decision scheme. Both these schemes apply some kind of early termination to the mode decision process, based on the outcomes of previously tested modes. For instance, the outcomes of SKIP and 2N×2N are considered while applying ESD. When using the mode decision scheme proposed here, testing of any mode can be avoided on a CU, including SKIP or inter 2N×2N. If this is the case ESD does not have the information needed to possibly perform the early termination. For this reason, ESD and CFM were adapted to coexist with the proposed scheme. The algorithms are initially disabled on a CU if the algorithm presented in this subsection is used. Then, the speed-ups are possibly re-enabled only if enough information becomes available to the encoder. The optimizations proposed in this subsection are referred to as *Stage 2* in the rest of this paper.

TABLE V
CONDITIONAL PROBABILITIES OF SELECTING A MODE IN SUB-CU 0 GIVEN THE MODE ON CURRENT CU FOR HD-GROUP.

		Optimal CU mode								
	Mode	SKIP	2N×2N	2N×N	2N×uN	2N×nD	N×2N	nL×2N	nR×2N	Intra
Mode at Sub-CU 0	SKIP	0.9851	0.0247	0.6975	0.6006	0.5597	0.637	0.6135	0.5223	0.6746
	2N×2N	0.0033	0.9685	0.0593	0.0354	0.0393	0.0382	0.0424	0.0602	0.0448
	2N×N	0.0043	0.0029	0.1550	0.1389	0.1164	0.1448	0.1257	0.0976	0.1263
	2N×uN	0.0025	0.0010	0.0313	0.0802	0.1476	0.0639	0.0547	0.0454	0.0434
	2N×nD	0.0004	0.0002	0.0049	0.0278	0.0345	0.0202	0.0160	0.0097	0.0122
	N×2N	0.0003	0.0002	0.0038	0.0244	0.0267	0.0157	0.0139	0.0086	0.0099
	nL×2N	0.0032	0.0018	0.0366	0.0608	0.0519	0.0548	0.0830	0.1703	0.0579
	nR×2N	0.0004	0.0003	0.0066	0.0192	0.0127	0.0155	0.0245	0.0598	0.0171
	Intra	0.0003	0.0003	0.0050	0.0128	0.0112	0.0098	0.0263	0.0262	0.0137

D. Prediction with RCU Framework

The final stage of the proposed optimizations consists in reducing the complexity of the prediction step. In Section III it was shown that the large majority of encoding time is used by inter-prediction. For this reason, only inter-prediction is targeted here. Similar to the algorithms illustrated in the previous stages, the idea of this stage is that of exploiting the RCU framework to use information extracted from sub-CUs to make decisions on the current CU. In particular, the algorithm proposed in this section is based on the assumption that the MVs found after ME in the sub-CUs are strongly correlated with the optimal MVs for the currently encoded CU. This assumption was experimentally validated by encoding the first 8 frames (excluding the first intra frame) from the *Training set* and analyzing the optimal MVs for a given CU and its sub-CUs. The experiment showed that in 79% of the cases when all MVs from the same list in the sub-CUs are identical, the same MV is selected for that list in the current CU. Extending this concept, the number of MV candidates to test in the motion search can be limited to reduce the computational complexity.

The problem of measuring the similarity among MVs is not new in video coding. For instance, it has been extensively studied in video transcoding techniques, where MVs extracted from the source codec might need to be manipulated and reused in the target codec. Among the techniques proposed for this purpose, one is particularly well suited for the algorithm proposed in this paper, referred to as MVVD [34]. The MVVD measures similarity in terms of the Euclidean distance between the variance of the horizontal and vertical components of a given set of MVs. As such, it offers several advantages with respect to other techniques: it can be expressed as a single fractional number; it can be applied to a group of an arbitrary large number of MVs; more importantly, it can also be used to compare groups of MVs containing different number of elements; it is relatively fast to compute, especially if it is computed using appropriate approximations.

The first step to compute the MVVD consists in scaling all MVs in the given set to the same temporal distance from the current frame. The idea is that, while MVs that point to different reference frames are theoretically not comparable, they can be projected to the same reference frame assuming the linearity of the motion flow. While this assumption is not always satisfied, it is good enough for the purpose of measuring similarity, as will be shown in the rest of this subsection.

Formally, assume the MVVD needs to be computed on an area spanning a number of PUs, for a given reference list k. Define as J the number of MVs available in list k, namely the number of inter-predicted PUs in the area which are either bi-predicted, or uni-predicted with list k. Each MV can be defined in terms of three integers $MV_j = \{x_j, y_j, t_j\}$, where x_j and y_j

are the MV vertical and horizontal components, respectively, and t_j is the temporal index of the reference frame the MV points to, $j = 0, ..., J-1$. Consider then the reference frame at minimum temporal index, $t_{min} = \min_{j=0,...,J-1}\{t_j\}$. Define as t_{cur} the temporal index of the current frame being encoded. Each MV can then be scaled to $\tilde{MV}_j = \{\tilde{x}_j, \tilde{y}_j, t_{min}\}$, where $\tilde{x}_j = \alpha_j x_j$, $\tilde{y}_j = \alpha_j y_j$, and:

$$\alpha_j = \frac{t_{cur} - t_{min}}{t_{cur} - t_j}. \quad (4)$$

As a result of this process, all the J scaled MVs extracted from list k point to the same reference frame. MVVD then measures the sample variance of the two MV components and finally returns the Euclidean distance between the two variances. Before applying this process, the available MVs need to be further processed, to account for the fact that each CU may contain a different number of available MVs. Consider for instance that MVVD needs to be applied to a region comprised of two CUs, where the first CU is predicted using the 2N×2N mode with list k (therefore only contains a single MV), while the second is predicted using N×N with list k (therefore contains 4 MVs), $J = 5$ in this example.

To overcome the bias problem, MVs can be replicated in such a way that always 4 MVs are considered in each CU in which there is at least one MV available in list k, regardless of the inter-prediction mode being used. MVs from CUs predicted with 2N×2N mode are replicated 4 times, MVs from CUs predicted with N×N mode are not replicated, and all other MVs are replicated 2 times. The output of this process is a set of G replicated and scaled MVs. Due to the fact that all these MVs point to the same reference frame, they are completely characterized by their components, and for simplicity are referred to as $MV_g = \{x_g, y_g\}$, $g = 0, ..., G-1$ in the rest of this subsection. Finally, the variance of each component is computed. This is obtained for the x component as:

$$\sigma_x^2 = \frac{\sum_{g=0}^{G-1}(x_g^2 - \bar{x}^2)}{G-1}, \quad (5)$$

and analogously for the variance σ_y of the y component. The MVVD of a given area S for a reference list k can be computed as:

$$\delta_\sigma\{S, k\} = \sqrt{(\sigma_x^2)^2 + (\sigma_y^2)^2}. \quad (6)$$

The MVVD as defined in Eq. (6) can be used as a measure of the "dissimilarity" of MVs extracted from region S, list k. Values of $\delta_\sigma\{S, k\}$ close to 0 indicate that all MVs are fairly similar within the area. MVVD can be used before performing inter-prediction using a given mode, to possibly limit the number of MV candidates to test. The idea is that if all MVs from all sub-CUs of the current CU are similar, then they are likely to be good candidates for the current CU, which means ME can be performed testing only the J original MVs $\{x_j, y_j, t_j\}$. The process of only testing this limited set of J MVs is referred to here as *constrained-ME*. Rather than using an adaptive threshold for MVVD to determine a similarity of MVs, a different sequence invariant approach is used.

The idea is that of computing the MVVD on the entire region spanned by all four sub-CUs, and then computing it on selected sub-regions spanning smaller areas of the current CU. In case the MVVD found on the entire region is smaller than that found on the sub-regions, it can be assumed that MVs are all similar. In the following, refer to the areas spanned by four sub-CUs as A, B, C and D respectively. The MVVD is computed independently on $A \cup B$ and $C \cup D$, where the maximum of these two values is taken as an indication of the maximum dissimilarity in horizontal direction H:

$$\delta_\sigma\{H, k\} = \max\{\delta_\sigma\{A \cup B, k\}, \delta_\sigma\{C \cup D, k\}\},$$

and similarly for vertical direction V:

$$\delta_\sigma\{V, k\} = \max\{\delta_\sigma\{A \cup C, k\}, \delta_\sigma\{B \cup D, k\}\}.$$

Finally, the MVVD is computed on the entire region:

$$\delta_\sigma\{FULL, k\} = \delta_\sigma\{A \cup B \cup C \cup D, k\}.$$

Eventually, if:

$$\delta_\sigma\{FULL, k\} \leq \delta_\sigma\{V, k\} \text{ and } \delta_\sigma\{FULL, k\} \leq \delta_\sigma\{H, k\}, \quad (7)$$

then the current CU is classified to use *constrained-ME*, and only J MVs are tested when performing ME on list k, $k = 0, 1$. In such case, ME precision is limited based on the MV precision in sub-CUs, but it is never lower than half-pel precision. The prediction algorithm proposed in this subsection is referred to as *Stage 3* in the rest of this paper.

E. Proposed Scheme and Encoders

Figure 6 summarizes the entire encoding framework and optimizations presented in this paper. The areas shaded in grey in the figure correspond to the three stages of optimization for depth selection, mode decision and prediction. Apart from *Stage 1* which is a prerequisite for the other two stages, stages are independent to each other, which means they can be used either separately or in any possible combination. By appropriately enabling or disabling the stages and tuning the corresponding parameters of each stage, a variety of different encoders can be defined to target different encoding speeds at correspondingly different efficiency costs. In this subsection three possible pre-set encoders are defined, referred to here as *Encoder 1*, *Encoder 2* and *Encoder 3*. Each of the encoders is based on the *FAST-HM* encoder defined in Section III.

Encoder 1: the first proposed encoder targets relatively small speed-ups, for very low compression efficiency losses. Hence, such encoder only considers optimizations in *Stage 1*. The two thresholds for the Depth Sum are set to $T_p = 6$ and $T_{p2} = 14$, as illustrated in Subsection IV-B.

Encoder 2: the second proposed encoder targets consistent speed-ups at acceptable performance losses. Only *Stage 1* and *Stage 2* are used in this case. The depth selection parameters are the same as those used in *Encoder 1*. *Stage 2* is used with $M = 4$, namely maximum 4 modes are tested on each CU.

Encoder 3: the final proposed encoder targets very high levels of speed, while allowing slightly higher compression efficiency losses. All three stages of optimizations are enabled here. Differently than *Encoder 2*, the mode decision is performed in a more strict fashion with $M = 3$. Finally, if inter-prediction modes are enabled to be tested on a CU, *Stage 3* is used to possibly reduce the number of MV candidates to test.

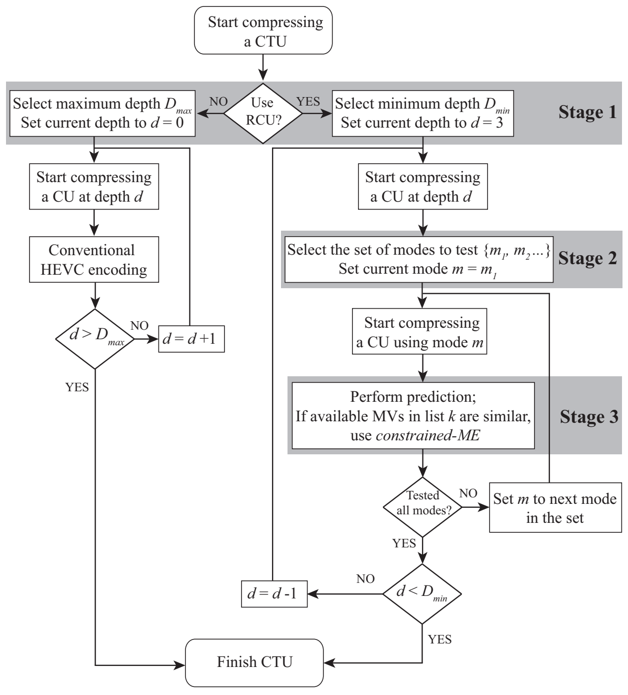
V. Results

The three encoders proposed in this paper were validated and compared with state-of-the-art techniques through extensive experimental evaluation using the following setup. The HM code version 12.0 [6] was used for the implementations. All tests were run using Dual 6-core 2.4 GHz 12 M Cache Intel Westmere (E5645) machines with 24 GB of RAM. The encoders were compared with the approaches proposed by Shen *et al.* [10], [14], with the fast encoder proposed by Vanne *et al.* [16], and with our previously proposed RCU method [31]. All experiments were performed according to JCT-VC CTC [24] under the RA-Main (RA) and LB-Main (LB) configurations using the ECU, ESD and CFM speed-ups whenever possible. Since the approach proposed by Shen *et al.* [10] is not compatible with ECU and ESD, only CFM is enabled when testing such encoder; all three speed-ups are used in any other test. Notice that in their paper, Vanne *et al.* [16] propose a variety of different algorithms achieving different levels of encoding speed. In this paper, the QP dependent algorithm achieving fastest encoding was used for comparison with the proposed methods. The encoders were validated on sequences at various resolutions (720p, 1080p, 1600p, and 2160p) from several well-known test sets [24], [25], [35], [36]. RD plots for different resolutions for all the sequences used for evaluation purposes were computed using *CTC* and are shown in Figure 7. Results are presented here in terms of BD-rates where *FAST-HM* was used as anchor in all tests. Also, encoding times for the tested encoders were measured and averaged on all QPs. These were used to compute speed-ups with respect to *FAST-HM* using Eq. (1).

Full results for RA and LB configurations are shown in Table VI and Table VII, respectively. As can be seen from the tables, all the tested methods perform rather similar under both configurations. *Encoder 1* introduces negligible BD-rate losses while still achieving on average 16.3% additional speed-up compared to *FAST-HM*. *Encoder 2* achieves on average 33.3% speed-up, for 1.6% BD-rate losses. Finally, even higher average speed-up of 36.6% compared with fast HEVC implementations is obtained for *Encoder 3*, with still acceptable average BD-rate losses of 2.1%. The proposed encoders outperform all other tested methods. The approach proposed by Shen *et al.* [10] achieves on average 1.9% BD-rate losses, for 25.5% speed-ups. The QP dependent method proposed by Vanne *et al.* achieves on average 1.5% BD-rate losses for 30.3% speed-ups. Our previously proposed RCU method achieves on average 0.8% BD-rate losses for 18.7% speed-ups. Finally, another approach from Shen *et al.* [14] achieves on average 0.7% BD-rate losses for 9.6% speed-ups. As can be seen from the results above, *Encoder 2* achieves considerably higher speed-ups for similar levels of compression losses compared with approaches from Shen [10] and Vanne [16]. With respect to our previous version of RCU, the encoders proposed in this paper allow for much higher speed-ups with still reasonable compression losses.

It is important to notice that the proposed methods do not degrade the visual quality. This was verified with the PSNR and Structural Similarity (SSIM) [37] measure analysis. It was observed that the proposed methods obtain equal or higher both PSNR and SSIM values compared to the *FAST-HM* anchor. Hence, higher BD-rate losses for *Encoder 2* and *Encoder 3* are caused by an increased number of bits, rather than degradation of quality.

As already mentioned in the paper, RCU is particularly effective in case of highly textured and complex content, where small CUs at high depths are often used. This corresponds to sequences which are typically more difficult to compress using HEVC, namely those residing in the bottom part of the RD-plots in Figure 7. These sequences are also the ones where pre-built speed-ups do not work particularly well (see Table I). In order to validate this statement, a sub-set of the tested sequences was defined, corresponding to the 2 most



Start compressing a CTU
Use RCU?
NO
YES
Select maximum depth D_{max} Set current depth to $d = 0$
Select minimum depth D_{min} Set current depth to $d = 3$
Stage 1
Start compressing a CU at depth d
Conventional HEVC encoding
$d > D_{max}$
NO
YES
$d = d + 1$
Start compressing a CU at depth d
Select the set of modes to test $\{m_1, m_2 \ldots\}$ Set current mode $m = m_1$
Stage 2
Start compressing a CU using mode m
Perform prediction; If available MVs in list k are similar, use *constrained-ME*
Stage 3
Tested all modes?
NO
YES
Set m to next mode in the set
$d < D_{min}$
NO
YES
$d = d - 1$
Finish CTU

Fig. 6. The proposed fast HEVC encoder scheme based on RCU. The shaded areas illustrate the three stages of optimization proposed in the paper.

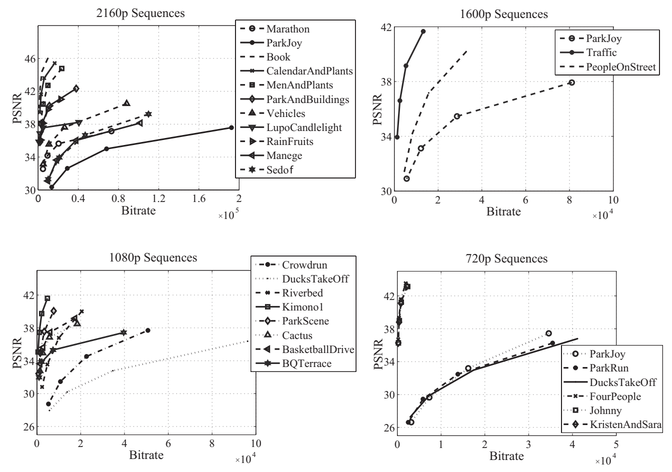


Fig. 7. RD plots for the test sequences used to validate the algorithms.

TABLE VIII
RESULTS FOR THE THREE PROPOSED PRE-SET ENCODERS VERSUS *CTC* AS AN ANCHOR UNDER RA-MAIN AND LB-MAIN CONFIGURATIONS FOR DIFFERENT SEQUENCE GROUPS.

	RA-Main						LB-Main					
	Encoder 1		*Encoder 2*		*Encoder 3*		*Encoder 1*		*Encoder 2*		*Encoder 3*	
Group	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]
2160p	2.0	62.5	3.6	70.1	4.1	71.7	1.6	57.1	3.3	65.5	3.9	67.6
1600p	2.3	53.8	3.6	65.0	4.3	67.3	1.6	48.6	3.0	61.4	3.8	65.0
1080p	1.9	51.5	3.1	61.6	3.6	63.5	1.4	45.2	2.6	56.3	3.1	58.5
720p	1.4	62.4	2.2	71.4	2.5	72.8	1.4	56.0	2.2	65.6	2.5	67.6
Average	1.9	57.5	3.1	67.0	3.6	68.8	1.5	51.7	2.8	62.2	3.3	64.7

challenging sequences at each resolution. Such sequences are highlighted in Italic in Tables VI and VII, where average results on this sub-set is also presented in the table. As can be seen, when applied on "complex sequences", *Encoder 1* does not bring any losses compared to *FAST-HM*, but adds 20.5% more speed-up. More importantly, *Encoder 2*, with only 1.1% BD-rate losses for 42.0% speed-ups, outperforms all other tested approaches by a large margin. As expected *Encoder 3* achieves even higher speed-ups with still acceptable losses.

Performance of the proposed methods was further investigated. It is expected that the percentage of CTUs that use certain CU visiting order will vary significantly depending on the sequence content. That means that sequences with mostly uniform or static areas will tend to select NCU for vast majority of their CTUs. This implies that the proposed fast mode decision and prediction algorithms are used only on a fraction of overall CTUs, since Stage 2 and 3 optimizations are not used in CTUs where NCU is selected. As a result, the proposed methods will provide limited speed-ups for sequences residing in the top left corner of RD-plots in Figure 7, compared to other methods that use fast mode decision for all CTUs. However, it should be noted that any fast mode decision method could be used in CTUs where NCU is selected to increase the speed-ups.

Finally, the proposed algorithm was compared with CTC HM anchor without any speed-ups enabled. All the sequences from Tables VI and VII were grouped based on their resolution and the results for each group are shown in Table VIII. As can be seen, *Encoder 1* achieves on average 57.5% of time savings for 1.9% BD-rate losses. *Encoder 2* achieves on average 67.0% of time savings for 3.1% BD-rate losses. Finally, *Encoder 3* achieves 68.8% of time savings for 3.6% BD-rate losses.

VI. CONCLUSIONS

The new flexible CTU partitioning scheme introduced in HEVC is responsible for a significant portion of the overall compression gains with respect to its predecessor AVC. However, testing all possible partitions inside the CTU comes at very high computational costs. In this paper, we propose the adaptive CU visiting framework where CUs in a CTU can be visited in reverse or conventional visiting order which are selectively used in each CTU, and only selected depths are tested. Moreover, by testing the CUs at higher depths first, information on optimal modes and outcomes of ME found in the four sub-CUs can be used to limit the number of modes to test for current CU and to speed up the ME process. Three different pre-set encoders were proposed in the paper to target increasing levels of encoding speed, at correspondingly different levels of compression efficiency. Experimental results show that the proposed methods greatly outperform previous state-of-the-art algorithms reducing the encoding time by an average 16.3% to 36.6%, for 0.3% to 2.2% luma BD-rate losses, respectively. It was observed that proposed methods are performing particularly well when encoding highly complex sequences, which are difficult to compress using conventional HEVC implementations.

REFERENCES

[1] ITU-T, "High efficiency video coding," ITU-T Rec. H.265, Tech. Rep., version 3: Apr 2015.

[2] D. Marpe, T. Wiegand, and G. Sullivan, "The H.264/MPEG4 advanced video coding standard and its applications," *IEEE Commun. Mag.*, vol. 44, no. 8, pp. 134–143, Aug 2006.

[3] J. Ohm, G. Sullivan, H. Schwarz, T. K. Tan, and T. Wiegand, "Comparison of the coding efficiency of video coding standards - including high efficiency video coding (HEVC)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1669–1684, Dec 2012.

[4] J. Vanne, M. Viitanen, T. 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.

[5] R. Weerakkody and M. Mrak, "High efficiency video coding for ultra high definition television," in *Proc. 2013 NEM Summit*, Oct 2013.

[6] ITU. HM Reference Software. May 2016. [Online]. Available: https://hevc.hhi.fraunhofer.de/HM-doc/

[7] K. Choi, S. H. Park, and S. E. Jang, "Coding tree pruning based CU early termination," JCTVC-F092, Tech. Rep., Jul 2011.

[8] Y. Zhang, H. Wang, and Z. Li, "Fast coding unit depth decision algorithm for interframe coding in HEVC," in *Data Compression Conference (DCC), 2013*, Mar 2013, pp. 53–62.

[9] J. Xiong, H. Li, Q. Wu, and F. Meng, "A fast HEVC inter CU selection method based on pyramid motion divergence," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 559–564, Feb 2014.

[10] L. Shen, 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.

[11] G. Correa, P. Assuncao, L. Volcan Agostini, and L. da Silva Cruz, "Fast HEVC encoding decisions using data mining," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 4, pp. 660–673, Apr 2015.

[12] J. Yang, J. Kim, K. Won, H. Lee, and B. Jeon, "Early SKIP detection for HEVC," JCTVC-G543, Tech. Rep., Nov 2011.

[13] R. H. Gweon, Y.-L. Lee, and J. Lim, "Early termination of CU encoding to reduce HEVC complexity," JCTVC-F045, Tech. Rep., Jul 2011.

[14] L. Shen, Z. Zhang, and Z. Liu, "Adaptive inter-mode decision for HEVC jointly utilizing inter-level and spatiotemporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1709–1722, Oct 2014.

[15] J. Xiong, H. Li, F. Meng, S. Zhu, Q. Wu, and B. Zeng, "MRF-based fast HEVC inter CU decision with the variance of absolute differences," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2141–2153, Dec 2014.

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

[17] S. Ahn, B. Lee, and M. Kim, "A novel fast CU encoding scheme based on spatiotemporal encoding parameters for HEVC inter coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 422–435, Mar 2015.

[18] S. G. Blasi, E. Peixoto, B. Macchiavello, E. M. Hung, I. Zupancic, and E. Izquierdo, "Context Adaptive Mode Sorting for Fast HEVC Mode Decision," in *Image Processing (ICIP), 2015 IEEE Int. Conf. on*, Sep 2015.

[19] A. Tourapis, O. Au, and M.-L. Liou, "Highly efficient predictive zonal algorithms for fast block-matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 934–947, Oct 2002.

[20] I. Zupancic, S. Blasi, and E. Izquierdo, "Multiple early termination for fast HEVC coding of UHD content," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE Int. Conf. on*, Apr 2015, pp. 1419–1423.

[21] S. Blasi, I. Zupancic, E. Izquierdo, and E. Peixoto, "Adaptive precision motion estimation for HEVC coding," in *Picture Coding Symp. (PCS), 2015*, May 2015, pp. 144–148.

TABLE VI

RESULTS FOR THE 3 PROPOSED ENCODERS COMPARED WITH STATE-OF-THE-ART FAST ALGORITHMS FROM SHEN ET AL. [10] AND [14], VANNE ET AL. [16], AND OUR PREVIOUS WORK [31] VERSUS *FAST-HM* AS AN ANCHOR (WITH ESD, ECU, AND CFM ENABLED) UNDER RA-MAIN CONFIGURATION.

		Encoder 1		*Encoder 2*		*Encoder 3*		*Shen* [10]		*Vanne* [16]		*RCU* [31]		*Shen2* [14]	
	Sequence	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]
3840 × 2160	*Manege*	0.2	19.9	1.7	39.0	2.4	43.3	3.8	38.5	2.3	31.1	1.6	25.2	1.7	10.0
	Marathon	0.2	16.4	2.0	39.0	3.0	44.3	2.0	34.4	2.2	31.6	1.3	23.6	0.8	8.6
	ParkJoy	0.2	18.5	1.5	40.3	2.0	43.4	1.7	31.7	1.2	29.5	1.1	25.2	0.4	7.1
	Sedof	0.1	17.4	1.2	37.2	1.6	40.2	3.2	32.0	1.6	30.6	1.0	21.4	1.2	10.5
BBC-UHD	Book	0.7	16.8	2.4	22.1	2.9	23.4	2.1	29.6	1.2	33.3	0.4	2.4	0.6	12.7
	CalendarAndPlants	0.6	11.7	3.2	26.7	4.1	31.4	2.7	24.6	2.9	32.3	1.1	13.1	1.6	13.0
	MenAndPlants	0.5	13.4	2.7	30.0	3.7	33.7	3.0	28.7	2.3	33.6	1.0	12.9	1.0	12.4
	ParkAndBuildings	0.6	12.5	1.9	28.2	2.3	31.8	1.1	23.1	1.8	33.2	1.0	13.6	0.6	12.6
	Vehicles	0.3	16.1	1.3	34.5	1.6	38.5	1.2	29.2	0.9	29.6	0.9	21.4	0.4	10.7
EBU-UHD	LupoCandlelight	0.4	14.7	1.4	26.5	1.7	29.3	0.9	20.6	1.4	31.9	0.3	13.5	0.7	11.4
	RainFruits	0.1	12.1	1.6	24.9	2.1	27.6	0.5	15.5	1.2	32.0	0.9	11.8	0.6	11.7
2560 × 1600	*ParkJoy*	0.2	19.7	1.5	39.5	2.1	42.2	1.7	32.5	1.5	30.6	1.4	24.8	0.5	8.8
Class A	Traffic	0.7	12.2	1.6	28.2	2.1	32.1	2.0	27.1	1.7	31.7	0.8	16.6	1.1	14.3
	PeopleOnStreet	−0.2	22.2	1.4	43.6	2.5	49.1	4.3	46.7	1.6	32.2	0.7	30.9	1.4	11.5
1920 × 1080	*CrowdRun*	−0.4	25.8	0.6	47.1	1.1	51.2	2.2	40.7	1.3	28.2	1.4	35.2	0.9	8.0
	DucksTakeOff	−0.1	19.9	0.6	41.8	0.9	45.5	0.7	27.8	1.0	26.7	0.4	26.6	0.2	3.8
	Riverbed	0.1	23.1	0.9	32.6	1.1	33.9	0.3	24.6	0.3	29.9	0.1	3.3	0.1	2.8
Class B	Kimono1	0.5	16.0	1.5	26.2	2.0	28.5	0.5	18.2	2.0	33.3	0.5	7.1	0.7	11.7
	ParkScene	0.4	12.5	1.2	30.2	1.7	33.7	1.1	20.4	1.4	30.6	0.8	18.1	0.8	12.2
	Cactus	0.2	15.4	2.0	33.5	2.6	37.6	2.7	28.9	1.9	30.3	0.7	19.0	1.1	9.8
	BasketballDrive	0.7	14.6	2.6	33.3	3.4	36.8	3.5	26.2	2.1	31.9	0.7	16.6	0.8	7.9
	BQTerrace	0.6	15.8	1.8	36.2	2.2	39.7	1.4	23.3	1.6	29.3	1.0	28.1	0.9	8.4
1280 × 720	ParkJoy	−0.4	23.1	0.2	42.8	0.5	47.4	0.3	30.8	0.7	26.6	0.7	35.3	0.5	7.2
	ParkRun	0.1	17.6	0.8	42.6	1.0	46.1	0.2	24.9	0.5	26.1	0.8	33.2	0.2	5.6
	DucksTakeOff	−0.1	20.5	0.5	42.3	0.9	43.8	0.3	26.9	0.9	26.6	0.4	28.5	−0.8	−1.0
Class E	FourPeople	0.9	10.6	2.0	23.9	2.4	27.1	2.7	7.5	1.2	28.4	0.5	9.5	0.6	12.3
	Johnny	1.2	8.0	2.0	19.0	2.5	21.6	2.5	−4.5	1.6	27.4	0.4	5.0	0.7	11.9
	KristenAndSara	1.2	8.6	2.3	20.1	2.6	22.6	4.2	3.1	1.6	29.8	0.2	3.0	0.8	12.5
Overall average		0.3	16.3	1.6	33.3	2.1	36.6	1.9	25.5	1.5	30.3	0.8	18.7	0.7	9.6
Complex sequences average		0.0	20.5	1.1	42.0	1.6	45.6	1.9	33.7	1.3	28.9	1.0	28.7	0.6	6.7

TABLE VII

RESULTS FOR THE 3 PROPOSED ENCODERS COMPARED WITH STATE-OF-THE-ART FAST ALGORITHMS FROM SHEN ET AL. [10] AND [14], VANNE ET AL. [16], AND OUR PREVIOUS WORK [31] VERSUS *FAST-HM* AS AN ANCHOR (WITH ESD, ECU, AND CFM ENABLED) UNDER LB-MAIN CONFIGURATION.

		Encoder 1		*Encoder 2*		*Encoder 3*		*Shen* [10]		*Vanne* [16]		*RCU* [31]		*Shen2* [14]	
	Sequence	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]	BDR [%]	TS [%]
3840 × 2160	*Manege*	0.2	20.1	1.9	40.4	2.6	45.8	4.8	42.9	2.2	34.2	1.6	26.9	2.2	15.5
	Marathon	0.3	16.8	2.8	38.5	4.0	43.6	2.0	34.7	1.7	34.4	1.4	23.0	0.6	11.2
	ParkJoy	0.1	19.1	1.6	40.3	2.0	44.4	1.4	36.5	1.0	33.4	1.1	26.9	0.4	11.9
	Sedof	0.0	19.1	1.2	38.1	1.8	43.3	3.8	37.8	1.7	34.9	1.3	23.8	1.6	15.8
BBC-UHD	Book	0.6	17.5	2.2	24.2	2.7	25.1	1.7	35.5	1.6	35.8	0.3	5.6	0.8	17.9
	CalendarAndPlants	0.5	12.9	3.0	24.3	3.9	29.4	2.9	32.2	2.4	39.0	1.0	11.9	1.6	21.5
	MenAndPlants	0.4	13.6	2.8	27.4	3.8	31.8	2.6	31.7	2.2	37.9	0.9	10.1	1.1	17.5
	ParkAndBuildings	0.8	13.6	1.9	28.7	2.5	32.4	1.2	29.1	1.9	38.3	0.9	15.4	0.7	19.5
	Vehicles	0.1	15.4	1.3	33.6	1.6	38.0	1.1	32.3	1.2	32.4	0.9	22.9	0.6	14.6
EBU-UHD	LupoCandlelight	0.1	11.1	1.1	23.5	1.5	26.7	0.6	21.9	1.7	37.2	0.3	12.2	1.1	19.0
	RainFruits	0.0	12.3	1.3	23.2	1.6	26.0	0.1	24.0	1.2	37.5	0.7	10.4	0.8	18.8
2560 × 1600	*ParkJoy*	0.1	17.2	1.5	37.2	2.1	43.0	1.5	34.0	1.2	31.6	1.1	23.3	0.6	10.7
Class A	Traffic	0.7	13.0	1.8	29.4	2.4	34.2	2.9	33.1	1.9	35.2	1.0	16.9	1.5	21.6
	PeopleOnStreet	0.2	20.5	1.9	43.9	3.1	50.3	4.9	48.6	1.2	34.2	1.0	32.4	1.5	15.3
1920 × 1080	*CrowdRun*	0.1	26.6	1.1	47.4	1.8	52.4	3.3	42.9	0.8	31.2	1.1	35.9	1.0	11.3
	DucksTakeOff	0.1	20.4	0.9	38.2	1.2	41.1	0.7	28.1	1.1	31.0	0.7	17.8	0.1	4.4
	Riverbed	0.1	25.9	0.7	34.9	0.8	34.8	0.2	28.4	0.3	30.2	0.0	6.2	0.0	4.0
Class B	Kimono	0.3	14.1	1.4	25.1	1.8	28.5	0.6	22.7	2.0	36.8	0.5	5.0	0.8	14.4
	ParkScene	0.3	14.4	1.3	32.5	1.9	37.4	1.3	30.4	1.6	36.4	0.8	20.4	1.2	21.0
	Cactus	0.1	16.3	2.1	34.8	2.7	39.6	3.1	34.5	2.0	34.8	1.0	21.3	1.3	15.1
	BasketballDrive	0.7	14.1	3.0	30.3	3.8	34.4	3.3	27.6	1.8	36.2	0.8	13.3	0.9	13.2
	BQTerrace	0.8	14.6	1.9	36.0	2.3	39.9	1.4	26.3	1.4	33.0	1.0	29.2	1.0	12.6
1280 × 720	ParkJoy	−0.4	25.0	0.3	45.7	0.4	49.6	0.6	35.2	0.6	28.2	0.5	37.3	0.7	11.0
	ParkRun	−0.1	16.5	0.8	41.7	1.0	46.0	0.2	28.3	0.6	28.6	0.9	31.5	0.3	6.6
	DucksTakeOff	0.1	19.2	0.8	37.3	1.1	40.8	0.5	25.7	1.0	29.0	0.7	18.0	0.1	3.3
Class E	FourPeople	1.0	9.5	2.1	21.3	2.6	25.1	4.1	17.5	2.0	31.9	0.5	5.6	1.5	18.8
	Johnny	0.9	8.0	1.5	16.2	1.9	18.4	2.8	6.3	2.4	33.2	0.2	3.8	1.6	19.4
	KristenAndSara	0.8	8.5	1.8	16.4	2.1	19.9	5.3	10.1	2.6	35.5	0.2	1.6	1.7	19.6
Overall average		0.3	16.3	1.6	32.5	2.2	36.5	2.1	29.9	1.5	34.0	0.8	18.2	1.0	14.5
Complex sequences average		0.1	19.9	1.3	40.8	1.9	45.5	2.1	35.9	1.1	31.6	1.0	26.6	0.8	9.9

[22] W. Choi, B. Jeon, and J. Jeong, "Fast motion estimation with modified diamond search for variable motion block sizes," in *Image Processing (ICIP), 2003 IEEE Int. Conf. on*, vol. 2, Sep 2003, pp. II–371–4 vol.3.

[23] G. Bjontegaard, "Improvements of the BD-PSNR model," ITU-T SG16/Q6, 35th VCEG Meeting, Doc.VCEG-AI11, 2008.

[24] F. Bossen, "Common test conditions and software reference configurations," JCTVC-L1100, Tech. Rep., Oct 2012.

[25] L. Song, X. Tang, W. Zhang, X. Yang, and P. Xia, "The SJTU 4K Video Sequence Dataset," in *Int. Workshop on Quality of Multimedia Experience*, Jul 2013, pp. 34–35.

[26] D. Knuth, *The Art of Computer Programming, Volume 1 (3rd Ed.): Fundamental Algorithms*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 1997.

[27] S. Even, *Graph Algorithms (2nd Ed.)*. Cambridge Univ. Press, 2011.

[28] D. Palomino, E. Cavichioli, A. Susin, L. Agostini, M. Shafique, and J. Henkel, "Fast HEVC intra mode decision algorithm based on new evaluation order in the coding tree block," in *Picture Coding Symp. (PCS), 2013*, Dec 2013, pp. 209–212.

[29] Y. Zhang, Z. Li, and B. Li, "Gradient-based fast decision for intra prediction in HEVC," in *Visual Communications and Image Processing (VCIP), 2012 IEEE*, Nov 2012, pp. 1–6.

[30] J.-F. Franche and S. Coulombe, "Fast H.264 to HEVC transcoder based on post-order traversal of quadtree structure," in *Image Processing (ICIP), 2015 IEEE Int. Conf. on*, Sep 2015, pp. 477–481.

[31] S. Blasi, I. Zupancic, E. Izquierdo, and E. Peixoto, "Fast HEVC coding using reverse CU visiting," in *Picture Coding Symp. (PCS), 2015*, May 2015, pp. 50–54.

[32] D. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, Dec 2011.

[33] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[34] E. Peixoto and E. Izquierdo, "A complexity-scalable transcoder from H.264/AVC to the new HEVC codec," in *Image Processing (ICIP), 2012 IEEE Int. Conf. on*, Sep 2012, pp. 737–740.

[35] R. Weerakkody, M. Naccari, and M. Mrak, "UHD test sequences," JCTVC-O0332, Tech. Rep., Nov 2013.

[36] L. Haglund, "The SVT high definition multi format test set ," Sveriges Television, Tech. Rep., Feb 2006.

[37] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600–612, Apr 2004.