HEVC ENCODER OPTIMISATIONS USING ADAPTIVE CODING UNIT VISITING ORDER

I. Zupancic[⋆]*, S. G. Blasi*[⋆]*, E. Peixoto*[†]*, and E. Izquierdo*[⋆]

[⋆]School of Electronic Engineering and Computer Science, Queen Mary University of London
[†]Department of Electrical Engineering, Universidade de Brasilia

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. Each frame in HEVC is partitioned in Coding Tree Units (CTUs) of fixed size, which are then recursively partitioned in Coding Units (CUs). In typical implementations, CUs in a CTU are visited from top to bottom at each level of recursion. In this paper, a different approach is used in which CUs in a CTU can be adaptively visited also in reverse order from bottom to top. Three novel algorithms to reduce complexity of HEVC depth selection, mode decision and inter-prediction are presented, based on this adaptive visiting order. Experimental results show that the proposed encoder achieves on average 38.2% speed-ups compared to fast reference HEVC implementations with pre-built speed-ups enabled, for very limited efficiency losses.

***Index Terms*—** HEVC, inter prediction, depth selection, mode decision

1. INTRODUCTION

Improving video coding standards is necessary to allow for more efficient exchange of video signals, especially when considering high spatial or temporal resolutions. The current state-of-the-art H.265/High Efficiency Video Coding (HEVC) standard [1] reportedly achieves 50% bitrate reduction [2] for the same perceived quality compared to its predecessor H.264/Advanced Video Coding (AVC) [3]. However, this comes at the cost of significantly increased computational complexity.

One of the key factors contributing to the efficiency of HEVC is its flexible partitioning scheme [4]. A frame is first divided in Coding Tree Units (CTUs) of fixed size, which are then partitioned into Coding Units (CUs) following a recursive quadtree structure. Based on the size and level of recursion, each CU is assigned a depth. Reference HEVC implementations test all depths up to a maximum level of recursion, and select the optimal CU size for the content being encoded. CUs are then split into Prediction Units (PUs) in a variety of modes: up to 8 inter-prediction modes (2N×2N, 2N×N, N×2N, N×N, nL×2N, nR×2N, 2N×nU, and 2N×nD), 2 intra-prediction modes (2N×2N and N×N) and a SKIP mode are considered for each CU. Finally, integer and sub-pel uni- and bi-directional Motion Estimation (ME) are performed for each PU. In theory, to obtain the best encoding efficiency, the encoder should test all combinations of such options to find the optimal solution for each CTU. Clearly, such approach is extremely computationally expensive and it can limit the usage of HEVC, especially when handling high resolution content.

In this paper, a novel scheme is presented to target HEVC encoding at low computational costs, based on adaptive visiting order of blocks during the encoding. Information collected while testing small blocks is used to limit the number of options tested in larger blocks and reduce the coding complexity.

2. RELATED WORK

Many methods have been proposed to reduce the complexity of HEVC depth selection, mode decision and inter-prediction.

A speed-up is included in the HM reference software [5] targeting fast depth selection, referred to as Early CU termination (ECU) [6]. ECU examines the optimal mode selected on a given CU: if SKIP mode is selected, the CU is not further split and no sub-CUs are tested. Xiong *et al.* [7] proposed an approach in which the optical flow is estimated from the down-sampled frames and then utilised to decide whether CUs should be split or not. Another approach was proposed by Shen *et al.* [8] to limit the CTU depth levels based on the depths from spatially and temporally neighbouring CUs.

There are already two pre-built speed-ups included in the HM software which target fast mode decision, referred to as Early Skip Detection (ESD) and Coding Flag Mode (CFM). When using ESD [9], the encoder tests the 2N×2N inter mode followed by SKIP mode and compares their costs: if SKIP is selected no further modes are tested on the CU. When using CFM [10], the root *coded_block_flag* (CBF) of the current PU is analysed after testing each mode: if zero, no other modes are tested for the CU. In the second part of the method proposed by Shen *et al.* [8], testing of inter and intra modes is skipped for homogeneous regions which are detected by means of Motion Vector (MV) similarity, neighbouring rate-distortion (RD) cost and SKIP mode selection. A method was proposed by Vanne *et al.* [11] to speed up the process of testing symmetrical and asymmetrical inter modes based on the current CU depth and Quantisation Parameter (QP) value. Recently, Ahn *et al.* [12] proposed a method which utilises the Sample Adaptive Offset (SAO) parameters, MVs, Transform Unit (TU) size and CBF information to estimate the temporal complexity, then used for fast mode decision. Finally, a method was proposed [13] where statistics on the modes and costs found on previously encoded PUs are collected and statistically modeled to test only the most probable PU modes.

A fast ME algorithm based on Enhanced Predictive Zonal Search (EPZS) [14] is already used by default in the HM reference software. When using EPZS, MVs from neighbouring blocks are considered as starting points for Motion Estimation (ME). Then,

Email: {i.zupancic, s.blasi, ebroul.izquierdo}@qmul.ac.uk, eduardopeixoto@ieee.org

This research utilised Queen Mary's MidPlus computational facilities, supported by QMUL Research-IT and funded by EPSRC grant EP/K000128/1.
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
This work was partially supported by CNPq under grant 444864/2014-8.

pattern searches are performed to find the optimal MV solution. Recently, a method was proposed [15] to early terminate the ME process by applying a pattern search around the starting point candidate to possibly skip the ME search.

3. FAST HEVC CODING USING ADAPTIVE CU VISITING

Opposite to the typical CU visiting order within a CTU, known as Depth-First Search (DFS) strategy [16], another approach is also considered here, referred to as Reverse CU (RCU) visiting order. When using RCU, child CUs in a quadtree are always visited before their parent CU, as shown in Fig. 1. Such visiting order is known in the literature as Post-order tree traversal strategy [17]. The RCU framework can be used to derive fast HEVC schemes, as illustrated in the rest of this paper. This kind of bottom to top approach has not been extensively studied in the context of video coding. Palomino *et al.* [18] have used this approach in a fast intra-prediction algorithm, to visit transform blocks in the recursive structure used for transform and quantisation. Franche *et al.* [19] have used this approach when transcoding AVC bitstreams to the HEVC.

In our previous work [19], we proposed an RCU framework which focused only on mode decision. In this paper, we extend and improve this approach to speed up other encoder parts. We first propose a new method to adaptively select the CU visiting order and depth range to be tested (referred to as *Stage 1*). We then propose a fast mode decision algorithm based on probabilistic model (referred to as *Stage 2*) and a fast ME algorithm (referred to as *Stage 3*) developed specifically to work with information available only when the RCU visiting order is used.

3.1. Adaptive CU Framework and Depth Selection

When using HEVC, homogeneous and uniform regions tend to be encoded with CUs of larger sizes, while textured and high motion regions are usually encoded with smaller-sized CUs. Based on this, the depth of neighbouring CUs can be used to predict the optimal depth for the current CU. In case high CU depths are predicted in a given area, the RCU visiting order may be more efficient because testing of lower depths can be skipped. Else, Normal CU (NCU) visiting order should be used, and testing of higher depths can be skipped. The problem becomes then that of predicting the maximum depth for a given CTU using information from neighbouring blocks. The maximum depths found in spatially neighbouring regions from left, above, and above-left of the current CTU have been shown to have high correlation with the maximum depth of the current CTU [8]. Denote the maximum depths found in regions $r_0, ..., r_4$, as illustrated in Fig. 2, as $D_0, ..., D_4$. The following parameter can then be defined:

$$D_p = \sum_{i=0}^{N-1} D_i w_i, \quad (1)$$

where N is the number of the available neighbouring regions, $N \leq 5$, and the weighting factor w_i is set to 1 in order to give the same importance to all neighbours. D_p ranges from 0 to 15 and is referred to as the Depth Sum.

An analysis was performed on a 10 sequence training set to study the relationship between Depth Sum and maximum depth selected in a CTU. Table 1 shows, for each value of the depth sum, the percentage of CTUs coded with a certain maximum depth. Clearly, CTUs with high values of D_p tend to be encoded with high maximum depths: in these cases RCU can be used and testing of CUs at depth 0 can be skipped. Conversely, low maximum depth is more likely selected in CTUs with low D_p: in these cases NCU can be used and testing of CUs at depth 0 can be skipped. The selection between NCU and RCU visiting order is performed based on a threshold T_p. The training set was used to compute two F-scores using precision and recall data when selecting RCU and NCU. $T_p = 6$ was finally selected, corresponding to the highest sum of the F-scores for each possible threshold value. NCU is used on a CTU if $D_p < T_p$, and RCU is used otherwise. In order to further reduce complexity of depth selection when using RCU, a second threshold $T_{p2} > T_p$ was defined, so that when $D_p \geq T_{p2}$, testing of CUs at depth 1 is also avoided. Following from empirical analysis, $T_{p2} = 14$ was selected here. Note that in case when less than 5 neighbouring CUs are available, selection between using NCU or RCU is obtained using a different threshold value, $T_p = 4$. In case when encoding the first CTU in a frame, no neighbouring CUs are available. Therefore, NCU visiting order is used without any depth limitation. Experimental verification proved that the accuracy of the proposed depth selection method is 96.01%. This step is referred to as *Stage 1* in the rest of the paper.

3.2. Mode Decision with Adaptive CU Framework

In case RCU is used on a CTU, information from the four sub-CUs at depth $d + 1$ is already available when testing a CU at depth $d \leq 2$. This can be used to help mode decision on the current CU and possibly reduce the number of modes to test. A classifier is used here for this purpose based on the well-known Naïve Bayes algorithm [20], to determine the M most probable modes for a given CU based on the optimal modes found in its 4 sub-CUs. Formally, the optimal modes selected in each of the 4 sub-CUs can be considered as random variables $X_1...X_4$, where $X_i = 0$ corresponds to SKIP mode, 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 sub-CU i was intra-predicted. Similarly, the optimal

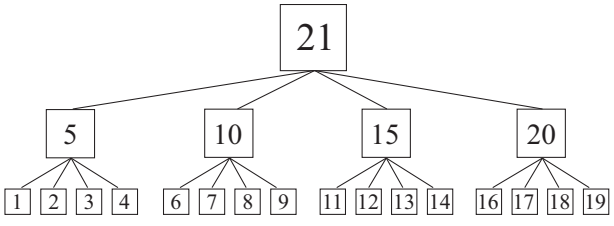

Fig. 1. CU visiting order when using RCU order, assuming a maximum depth of 2

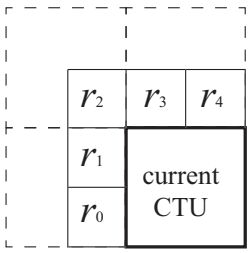

Fig. 2. Neighbouring CUs used to predict depth.

Table 1. Probabilities of maximum CTU depth occurrence for different values of Depth Sum.

Maximum CTU depth	Depth Sum															
	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

mode for the current CU can be considered as a random variable Y, which again can assume the same 9 values. Naïve Bayes algorithms are based on the assumption that $X_1...X_n$ are independent given Y. If this is the case, then the following holds:

$$P(X_1...X_n|Y) = \prod_{i=1}^{n} P(X_i|Y). \tag{2}$$

This expression can be used to derive the posterior probability that $Y = y_k$ given $X_1...X_n$:

$$P(Y = y_k|X_1...X_n) = \frac{P(Y = y_k)P(X_1...X_n|Y = y_k)}{\sum_{j=1}^{D} P(Y = y_j)P(X_1...X_n|Y = y_j)}. \tag{3}$$

Applying Eq. (2) to Eq. (3), the posterior probability that Y will take any given value can be estimated. Finally, the class with maximum posterior probability can be calculated as:

$$Y = \arg\max_{y_k} P(Y = y_k) \prod_{i=1}^{n} P(X_i|Y = y_k). \tag{4}$$

The prior probabilities $P(Y = y_k)$, $k = 0, ..., 8$ can be computed during the classifier training, as the number of times a certain mode was 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 that y_k was selected on the larger CU. While encoding a CU at depth $d \leq 2$, the four modes found in sub-CUs at depth $d + 1$ are considered. The posterior probabilities $P(Y = y_k|X_1...X_n)$ are computed for $k = 0, ..., 8$ and sorted in descending order. Finally, only the M most probable modes are tested in the CU, where M is a parameter which can be used to define the strength of the algorithm. Optimisations proposed in this subsection are referred to as *Stage 2* in the rest of the paper.

3.3. Prediction with Adaptive CU Framework

The final stage of the proposed optimisations consists in reducing the complexity of the inter-prediction step. Similarly to *Stage 2* optimisations, the algorithm proposed here makes use of information from sub-CUs. In particular, the similarity of MVs is used to limit the number of MV candidates to test during ME. MV similarity is computed using a measure known as MV Variance Distance (MVVD) [21]. 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. 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. Formally, assume the MVVD needs to be computed on an area spanning a number of PUs, for a given reference list k, where J is the number of available MVs in 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, for $j = 0, ..., J - 1$. Denote with t_{min} and t_{cur} reference frame at minimum temporal index and temporal index of the current frame being encoded, respectively. 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:

$$\alpha_j = \frac{t_{cur} - t_{min}}{t_{cur} - t_j}. \tag{5}$$

After scaling, the variance of each component can be computed as:

$$\sigma_x^2 = \frac{\sum_{j=0}^{J-1}(\tilde{x}_j^2 - \bar{x}^2)}{J - 1}, \tag{6}$$

and analogously for the variance σ_y of the y component. The MVVD of a given area S for a given reference list k is then defined as:

$$\delta_\sigma\{S, k\} = \sqrt{(\sigma_x^2)^2 + (\sigma_y^2)^2}. \tag{7}$$

Values of $\delta_\sigma\{S, k\}$ close to 0 indicate that the MVs are similar. Eq. (7) can be used on MVs from the sub-CUs of a given CU. The MVVD is first computed on all MVs from the whole region spanned by the entire CU area, and then on sub-regions covering half the CU area, independently. In case the first value (using all MVs on the whole region) is lower than the values found in the sub-regions, the area is considered to be smooth in terms of temporal activity. In this case only the J MVs selected in the sub-CUs are tested in the larger CU, and the remaining ME is skipped. Otherwise, ME is performed as in conventional HEVC encoding. Optimisations proposed in this subsection are referred to as *Stage 3* in the rest of the paper.

3.4. Proposed Scheme and Encoders

By appropriately enabling or disabling the stages proposed in this paper and tuning the corresponding parameters, different encoders can be defined to target different encoding speeds. Three encoders are defined here for experimental evaluation of the proposed methods:

Encoder 1 considers only optimisations in *Stage 1* as illustrated in Subsection 3.1.

Encoder 2 considers optimisations in *Stage 1* and *Stage 2*. $M = 4$ is used for *Stage 2* optimisations.

Encoder 3 considers all three stages. $M = 3$ is used for *Stage 2* optimisations (to target higher encoder speeds).

4. EXPERIMENTAL RESULTS

The three encoders proposed in this paper were validated and compared with state-of-the-art techniques through extensive experimental evaluation. The HM reference software version 12.0 [5] was used as basis for the implementation. 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.* [8], with the fast encoder proposed by Vanne *et al.* [11], and with our previously proposed RCU method [22]. All experiments were performed according to JCT-VC CTC

Table 2. Results for the three proposed preset encoders compared with state-of-the-art fast algorithms from Shen et al. [8], Vanne et al. [11], and our previous version of RCU [22] versus *HM-FAST* (with ESD, ECU, and CFM speed-ups enabled) as an anchor under RA-Main configuration.

		Encoder 1		*Encoder 2*		*Encoder 3*		*Shen* [8]		*Vanne* [11]		*RCU* [22]	
	Sequence	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
	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
	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
	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
	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
	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
	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
	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
	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
	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
	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
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
	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
	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
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
	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
	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
	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
	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
	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
	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
	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
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
	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
	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
Overall average		0.2	17.1	1.5	34.7	2.1	38.2	1.7	28.3	1.5	30.5	0.8	20.3

[23] under the RA-Main configuration. As the methods target fast HEVC encoding, in order to achieve the highest encoding speeds, the pre-available ECU, ESD and CFM speed-ups presented in Section 2 were enabled in all the experiments, both when testing our algorithms and the competing techniques and anchors. The anchors where ECU, ESD, and CFM speed-ups were enabled are denoted as *HM-FAST* in the rest of this section. Sequences at various resolutions (720p, 1080p, 1600p, and 2160p) from well-known test sets [23], [24], [25], [26] were used. Results are presented here in terms of BD-rates where *HM-FAST* was used as anchor in all tests. Also, encoding times required by the tested encoders were measured to calculate the time saving:

$$\Delta T_i = \frac{T_A - T_M}{T_A} \times 100\%, \tag{8}$$

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.

Full results are shown in Table 2. *Encoder 1* introduces negligible BD-rate losses (0.2%) while still achieving on average 17.1% additional speed-up compared to *HM-FAST*. In one case (the *CrowdRun* sequence), up to 25.8% speed-ups are achieved while obtaining negative BD-rate of -0.4%. *Encoder 2* achieves on average 34.7% speed-up, for 1.5% BD-rate losses, with the highest speed-up of 47.1% for *CrowdRun* sequence for 0.6% BD-rate losses. Finally, even higher average speed-ups of 38.2% compared with fast HEVC implementations are obtained for *Encoder 3*, with still acceptable average BD-rate losses of 2.1%. Again, the highest speed-up of 51.2% was obtained for *CrowdRun* sequence, for 1.1% BD-rate losses. The proposed encoders outperform all other tested methods. The approach proposed by Shen *et al.* [8] achieves on average 1.7% BD-rate losses, for 28.3% speed-ups. The QP dependent method proposed by Vanne *et al.* achieves on average 1.5% BD-rate losses for 30.5% speed-ups. Finally, our previously proposed RCU method achieves on average 0.8% BD-rate losses for 20.3% speed-ups. As can be seen from the results above, *Encoder 2* achieves considerably higher speed-ups for the similar levels of compression losses compared with approaches from Shen [8] and Vanne [11].

When compared against the anchors encoded with CTC under the RA configuration without any pre-built speed-ups enabled, *Encoder 1* achieves on average 55.9% speed-up, for 1.9% BD-rate losses. *Encoder 2* achieves even higher speed-up of 65.6% for 3.2% of BD-rate losses, while *Encoder 3* achieves 67.5% of speed-up for 3.7% of BD-rate losses.

5. 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 preset 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 method outperforms previous state-of-the-art algorithms.

6. REFERENCES

[1] G. Sullivan, J. Ohm, W. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 22, no. 12, pp. 1649–1668, 2012.

[2] 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. on Circuits and Systems for Video Tech.*, vol. 22, no. 12, pp. 1669–1684, 2012.

[3] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 13, no. 7, pp. 560–576, 2003.

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

[5] ITU. HM Reference Software. December 2015. [Online]. Available: https://hevc.hhi.fraunhofer.de/HM-doc/

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

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

[8] L. Shen, Z. Liu, X. Zhang, W. Zhao, and Z. Zhang, "An effective CU size decision method for HEVC encoders," *Multimedia, IEEE Transactions on*, vol. 15, no. 2, pp. 465–470, Feb 2013.

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

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

[11] J. Vanne, M. Viitanen, and T. Hamalainen, "Efficient mode decision schemes for HEVC inter prediction," *IEEE Trans. on Circuits and Systems for Video Tech.*, vol. 24, no. 9, pp. 1579–1593, Sept 2014.

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

[13] 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 *IEEE Int. Conference on Image Processing*, Sept 2015.

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

[15] 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 International Conference on*, April 2015, pp. 1419–1423.

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

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

[18] 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 Symposium*, Dec 2013, pp. 209–212.

[19] 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 International Conference on*, Sept 2015, pp. 477–481.

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

[21] E. Peixoto and E. Izquierdo, "A complexity-scalable transcoder from H.264/AVC to the new HEVC codec," in *IEEE International Conference on Image Processing*, Sept 2012, pp. 737–740.

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

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

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

[25] L. Song, X. Tang, W. Zhang, X. Yang, and P. Xia, "The SJTU 4K video sequence dataset," in *International Workshop on Quality of Multimedia Experience*, July 2013, pp. 34–35.

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