computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

A NEW OPTIMIZED IMPLEMENTATION OF A FAST INTRA PREDICTION MODE DECISION ALGORITHM FOR HEVC STANDARD

Hajar Touzani [1], Ibtissem Wali [2], Fatima Errahimi [1], Anass Mansouri [3], Nouri Masmoudi [2] and Ali Ahaitouf [1]

[1] Laboratory Intelligent systems, Georesources and renewable Energies, Faculty of Sciences and Technologies, University Sidi Mohamed Ben Abdellah, Fez, Morocco, touzanihajar1@gmail.com
[2] Laboratory of Electronics and Information Technologies, National School of Engineering, University of Sfax, Sfax, Tunisia
[3] National School of Applied Sciences, University Sidi Mohamed Ben Abdellah, Fez, Morocco

Paper history:
Received 22 November 2019
Received in revised form 17 April 2020
Accepted 17 October 2020
Available online 30 December 2020

Keywords:
H.265/HEVC;
Encoding;
Intra prediction;
ARM;
Embedded Linux;
HM16.7;
Bit rate and
PSNR.

Abstract: New and stronger video compression standard was developed during the last years, called H.265/HEVC (High Efficiency Video Coding). This standard has undergone several improvements compared to H.264/AVC (Advanced Video Coding). In intra prediction block, 33 directional intra prediction modes were included in H.265 to have an efficient coding instead of 8 modes that were used in H.264 in addition to planar and DC modes, which has generated computational complexities in the new standard. Therefore one of the most issues for embedded implementation of HEVC is time reduction of the encoding process. In this paper, an embedded implementation of a fast intra prediction algorithm is performed on ARM processors under the embedded Linux Operating System. Experimental results included the comparison between the original HM16.7 and the proposed algorithm show that the encoding time was reduced by an average of 61.5% with an increase of 1.19 in the bit rate and a small degradation in the PSNR of 0.05%.

1. INTRODUCTION

Videos became these days more popular among consumers than any other type of content. Statistics provided in the world video marketing said that by 2020 online video will make up more than 80% of all consumer internet traffic. In addition, the trend toward higher video qualities is developed to UHD (Ultra High Definition). So, the need of powerful video compression standards that can support higher resolutions with the huge use of videos and that can insure the required bandwidth to transmit video content is demanded. For this purpose, HEVC standard was announced in 2013. This standard can support higher video resolutions that can achieve 8K x 4K UHD [1]. It has also the ability to offer up to 50% (70% in the latest versions) of data compression at the same level of quality compared to H.264/AVC (Advanced Video Coding) that was the most used before [2].

In HEVC video coding, each picture is split into slices, slice segments and tiles, containing the CTUs (Coding Tree Units). Each CTU is divided in one or more CUs (Coding Units) which sizes 2N x 2N (N = 4, 8,16 and 32). Then the CU is partitioned into one or more PUs (Prediction Units) and one or more TUs (Transform Units) to form the Recursive QuadTree (RQT) which CU is the root. This method of partitioning aids the coding process to be more flexible [3] compared to H.264/AVC standard which was based on macroblocks with reduced sizes and partitions.

In H.265 standard three types of the pictures are used; the I frame that applies only the intra prediction, the P frames and B frames are used in the inter prediction (motion estimation and compensation). The output of the prediction block (residual signal) is the difference between the predicted picture and the reference picture. This

output is transformed with 2D DCT and ICT transforms (2 Dimensional Discrete Coding Transform and Integer Coding Transform) and then quantized, finally the transformed quantized coefficients are compressed using the CABAC (Context-Adaptive Binary Arithmetic Coding) algorithm that involves the functions of binairization, context modelling and arithmetic coding [4].

The present work focuses on an optimized method of implementing the overall proposed model [5] on ARM processors that can more accelerate the encoding process of the intra prediction for a more recent version of HM which is HM16.7 reference software. Firstly, the modifications of the intra prediction process are made in HM16.7. Secondly, in the context of software performance testing, these modifications are implemented in a standard platform (Intel Core i7, 3.6 GHz, 4GB on RAM memory) under the Linux (Ubuntu 16.04) operating system based on the Common test conditions and software reference configurations provided by the JCT-VC [6]. Finally, an embedded implementation of the proposed models is performed on octa-core ARM processors under an optimized Linux operating system. As a result, the encoding time was accelerated by around 3 times in ARM processor compared to the original intra prediction process.

The rest of the paper is organized as follows, related works are reviewed in section 2, section 3 describes the complexity of the algorithm of the intra prediction process in the HEVC/H.265 video compression standard. Part 4 provides the description of the proposed algorithms for the intra prediction. In Section 5, the hardware implementation of the proposed model is analyzed with a brief presentation of the used platform and finally we provide the experimental results and comparisons with other works.

2. RELATED WORKS

As mentioned in the Introduction, the HEVC standard has introduced many methods that make this video compression standard more powerful. As a result of these newly integrated methods, the HEVC/H.265 video coding can achieve the double of compression ratio with a same quality compared to H.264, but this has generated a big complexity on the compression computations. A lot of analysis in the works has been carried out to evaluate the complexity of the HEVC standard, earlier studies [7] present the results of the profiling of HEVC standard on different platforms for the all intra (when inter prediction is disabled) configuration file. Results show that after the transform/quantization block that consumes the lengthy encoding time through the RDOQ (Rate Distortion Optimization Block) process, the prediction (intra and inter prediction) process takes the great amount of time (almost half of the total of encoding time) compared to other blocks. Concerning the random access configuration when the inter prediction is activated, Bossen et al. [8] showed an analysis of HEVC complexity for both the encoder and the decoder using different experimental tests. Their results show that the prediction block consumes more than 60% of encoding time especially in the Rate Distortion (RD) Cost functions that are presented by the TComRdCost class and that accounts for 40% of encoding time. This high percentage demonstrates the complexity presented in the prediction block and more particularly in the general coder control block that is responsible for the decision and that uses, for the most often, the RD Cost functions to decide the best partitioning and modes to be used in the prediction. To reduce this complexity, an active area of researches was concentrated on how to accelerate the intra prediction process. Venugopal et al. [9] have proposed a fast model of the intra prediction. It was concerning a rapid template matching for the intra prediction of HEVC standard. The best template match is derived from the reconstructed samples by matching three best Template Matches (TM) in the sense of minimizing the SSD (Sum of Square Differences), the averaged superposition of these three best TMs is used as the prediction of the current Prediction Block (PB). As results of this method, there was a gain in the bit rate by a percentage of 1.15% and an increase of run-time of 33%. Another way to reduce the complexity of HEVC intra prediction is to reduce the number of the modes verified in the decision process [8, 9]. For instance, in the Xie et al. paper [10] a set of tests on split and no split of CU with different QP (Quantization Parameter) sizes in addition to an analysis of the candidate list and the modes that are more selected as the best ones are considered. Based on these tests, the authors propose two innovative algorithms, the first one concerns a fast CU division algorithm based on a set of a predefined threshold values. The second one concerns a fast intra mode decision algorithm that excludes some modes from the candidate list to minimize the computation complexity of intra process. As a result, the combination of the two algorithms can save to 49.6% of the encoding time; there is also an optimization of the bit rate of 1.3% but with a loss in the PSNR of 0.31dB. The other work [11] was based on the correlation between the CTU texture and the optimum CU partition. Authors of [11] proposed two algorithms, the CDRP (CTU Depth Range Prediction) that can reduce the splitting of the CU partition and the IPMS (Intra Prediction Mode

Selection) that is proposed in order to optimize the decision process of intra prediction by minimizing the number of candidate modes. The combination between the two algorithms can reduce the running time by 60% with an increase of 1.45% in the bit rate. In another paper, Azgin et al. [12] reported an optimized architecture for the intra prediction concerning the sizes 4 x 4, 8 x 8, 16 x 16 and 32 x 32 for only the angular prediction mode. This block was then implemented on an FPGA. The distinctive characteristic in this paper was the multiplication function that is implemented using the DSP (Digital Signal Processor) blocks instead of using the adders and shifters. As a result, this hardware implementation has up to 36.66% less energy consumption than the original one and can implement up to 55 full HD (High Definition) video frames per second. Kibeya et al. [5] have designed two fast intra prediction algorithms concerning the optimization of the RMD (rough mode decision) process. The goal of their work was the reduction of the necessary encoding time for the intra prediction decision part by minimizing the number of candidate modes evaluated in the decision stage. The modifications were performed on the HM10.0 (HEVC test Model) reference software on a standard platform characterized by an Intel processor Core TM i7-3770 under Windows 7 Operating System. This work has reduced the encoding time by an average of 46.13% compared to the original algorithm of HEVC. More details of this model are presented in section 3.

The embedded implementations of such a powerful video compression standard like HEVC on ARM processors is a huge challenge, because of its high performance and powerful efficiency and a very high level of complexity. In addition that it targets a wide variety of mobile and consumer applications including mobile phones and tablets. Almost all the ARM implementations of HEVC standard that was performed these last few years concern the decoder process that is much easier than the encoder. For instance, Smei et al. [13], exploit the parallel tools used in HEVC standard (Tiles and slices) to perform a pipelining method for the decoder and then they implement this method on a dual ARM platform (Zedboard) [13]. This method was able to minimize the decoding time by 30% compared to the sequential method. Other embedded implementation of HEVC decoder that applies a parallel optimized method was reported by Liu et al. [14] for the multi-view video decoding using the multi-threading. The results show that the proposed method is 5 times faster in the ARM platforms.

3. INTRA PREDICTION IN HEVC

The main goal of the intra prediction block [15] is to achieve a higher coding efficiency by minimizing the spatial redundancies between the adjacent samples using the reconstructed reference samples of each video frame. In the intra prediction, three main steps are executed as shown in Fig. 1.

After the construction of the reference sample array, the intra prediction is performed on the current sample using one of the 35 intra prediction modes shown in Fig. 2. Finally, in the post processing step, a filter is applied on the current block to reduce the discontinuities between the current and the reference blocks.

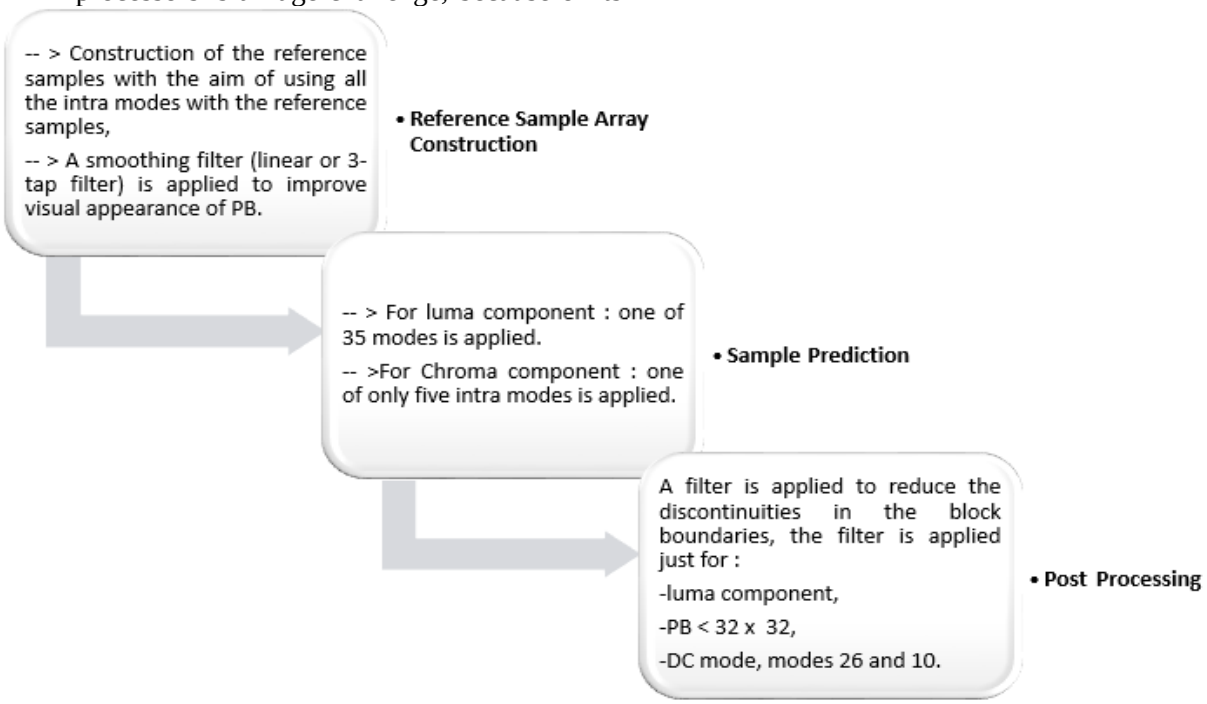

Figure 1–Description of the three steps of the intra prediction block

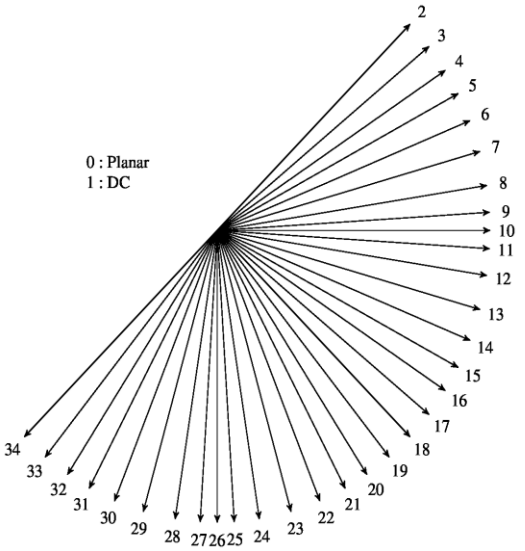

Figure 2 – The 35 intra prediction modes

The 35 intra modes (see Fig. 2) used in the second step (Sample Prediction) consists of:

– 33 angular prediction directions (No. 2-34) that are mostly used in objects with directional structures. This type of intra mode can be obtained by projecting the current sample to the reference sample array applying one of these 33 intra modes.

– DC (No. 1) and planar modes (No. 0) are used in the smooth image areas [16].

Before executing these three steps, it is required to choose the best intra mode with the minimal overhead, this means that we have to choose one of the 35 intra modes prediction that can give the best trade-off between a reduced distortion and a higher bit rate. In that regard three steps are performed:

1. Before calculating the Rate Distortion Optimization (RDO), the Hadamard transform is executed for all possible intra prediction modes by calculating the RD-cost(1) (equation 1) [17] and the SATD (Sum of Absolute Transformed Differences) using the equations below:

$$RD\text{-}cost(1) = SATD + \lambda * R, \quad (1)$$

where, *SATD* – distortion, λ – Lagrangian multiplier parameter, *R* – bit rate needed to encode the prediction mode.

$$SATD = \sum |H*(C_{ij}-R_{ij}) * H, \quad (2)$$

where, $C_{i,j}$ – the current pixels, $R_{i,j}$ – the reference pixels, *H* – the Hadamard transform matrix.

As a result, a Prediction Mode Table (PMT) is constituted which contains a list of candidates listed from smaller to higher costs.

The number of candidates depends on the size of PUs.

– For the 4x4 and 8x8 PUs, eight intra modes are chosen,

– For the rest of PU sizes only three are chosen.

This is called the RMD process.

2. After constructing the PMT using the Hadamard transform, three MPMs (Most Probable Modes) are added to the PMT, which are based on the modes of the left and top neighboring blocks.
3. In a final step, we perform the RDO process by calculating the new RDcost (RD-cost(2)) of each candidate in the PMT using the equation below [17].

$$RD\text{-}cost(2) = SSD + \lambda * R, \quad (3)$$

where *SSD* – the absolute sum of the difference between the current and the original sample, λ – Lagrangian multiplier parameter and *R* – the number of bits needed to encode the prediction mode.

4. DESCRIPTION OF THE PROPOSED FAST INTRA PREDICTION PROCESS

The modifications performed in the intra prediction process consist of three steps executed before choosing the best intra prediction mode. For that, two algorithms are designed:

4.1 FAST INTRA-PREDICTION ALGORITHM BASED ON EARLY DETECTION OF ZERO TRANSFORM AND QUANTIZED COEFFICIENTS

This method uses threshold values to decide the best mode among the 35 modes without the assessments of all these modes. The assumed thresholds are the same as those reported by Kibeya et al. [5]. We then follow the steps described in Fig. 3 to find the best mode among the 35 possible.

The SATD is calculated for a candidate *i*, if the calculated SATD value is smaller than the SATD of the threshold, and then this mode is selected as the best one, and the SATD is calculated for the next candidate.

4.2 FAST INTRA-PREDICTION ALGORITHM BASED ON REFINEMENT OPERATION

As mentioned in [5], a list of statistical experiments was performed, using different Qp values (22, 27, 32 and 37) for all classes (A, B, C, D and E). During these experiments, it has been observed two principal remarks:

– The intra modes that are more used include planar, DC, directly horizontal and directly vertical. Table 1 summarizes the more used modes for each

PU size as reported in [5].

1. Select the N intra-prediction modes as **i = {0, N}.**
   Where: N = 4 for 64 x 64 pixels
   N = 15 for 32 x 32 and 16 x 16
   N = 17 for 8 x 8 and 4 x 4
2. Calculate the SATD for the N candidates.

First_best_mode = min SATD

3. If **SATD First_best_mode > SATD First_best_mode – 1**, go to step 4 else go to step 7.
4. **Mode (i) = First_best_mode – 1.**
5. **If** SATD mode(i) > mode(i-1) **then** i = i – 1 **and repeat step 5, else go to step 10.**
6. **if** SATD First_best_mode > SATD First_best_mode + 1**, go to 8, else go to step 10.**
7. **Mode (i) = First_best_mode + 1**.
8. If **SATD mode(i) > mode(i+1)** then **i = i + 1** and repeat step 9, else go to step 10.
9. R-list = three modes with the min SATD.
10. Calculate RdCost SSD for the three modes.
11. **Best_mode = min RdCost_SSD**.

Figure 3 – Description of the first algorithm

– The PUs with a reduced size are more used compared to ones with a largest size.

The idea emerging from these results is then to reduce the number of intra modes in RMD process and calculate the RDcost based on SATD just for the modes that are more used for each PU size (shown in Table 1). This method is more detailed in the following steps:

1. Prediction stage: Instead of calculating the Hadamard transform for all the 35 modes in the original intra prediction, the proposed model releases these computations on only the candidate modes illustrated in the Table 1. The mode with the minimal SATD is selected as the best one and indicated by a variable called "first-best-mode".

2. Refinement stage: The goal of this step is to refine the first step by calculating the SATD of the two neighboring direction modes, as a result an R-list composed of three candidates is generated. More details concerning this step are performed in Fig. 4.

3. Final decision stage: This last step is similar to the one of the original HEVC method (as described in section 2), the RDO based on the SSD is calculated for the three modes in the R-list obtained from the second step. Finally, the mode with the minimal RD-cost is selected as the best intra mode.

1. Select the N intra-prediction modes as **i = {0, N}.**
   Where: N = 4 for 64 x 64 pixels
   N = 15 for 32 x 32 and 16 x 16
   N = 17 for 8 x 8 and 4 x 4
2. Calculate the SATD for the N candidates.

First_best_mode = min SATD

3. If **SATD First_best_mode > SATD First_best_mode – 1**, go to step 4 else go to step 7.
4. **Mode (i) = First_best_mode – 1.**
5. **If** SATD mode(i) > mode(i-1) **then** i = i – 1 **and repeat step 5, else go to step 10.**
6. **if** SATD First_best_mode > SATD First_best_mode + 1**, go to 8, else go to step 10.**
7. **Mode (i) = First_best_mode + 1**.
8. If **SATD mode(i) > mode(i+1)** then **i = i + 1** and repeat step 9, else go to step 10.
9. R-list = three modes with the min SATD.
10. Calculate RdCost SSD for the three modes.
11. **Best_mode = min RdCost_SSD**.

Figure 4 – Description of the second algorithm

Table 1. The most modes used for each PU

PU sizes	Number of candidates	The modes
64 x 64 pixels	4 + 3 MPMs	0, 1, 10, 26, 3 MPMs
32 x 32 pixels	14 + 3 MPMs	0, 1, 3, 6, 9, 10, 12, 15, 18, 21, 24, 26, 27, 30, 33, 3 MPMs
16 x 16 pixels	14 + 3 MPM	0, 1, 3, 6, 9, 10, 12, 15, 18, 21, 24, 26, 27, 30, 33, 3 MPMs
8 x 8 pixels	19 + 3 MPMs	0, 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 3 MPMs
4 x 4 pixels	19 + 3 MPMs	0, 1, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 3 MPMs

5. EMBEDDED IMPLEMENTATION OF HEVC ON ARM PLATFORM

5.1 HM (HEVC TEST MODEL) SOFTWARE

In order to demonstrate and to study the coding performance of the HEVC coding standard, the (JCT-VC) was developed a reference software HEVC test Model (HM) with various features that can help users to have a rich checking compliance.

HM is provided as a source code developed in C++ that includes both encoder and decoder functionalities, and can be implemented on various platforms. The modifications performed in this work

concerns the HM 16.7 version.

The JCT-VC have introduced a set of common test conditions and software reference configurations that can be used in experiments in order to make easier the comparison of the outcome of experiments. Three configurations of files are commonly used. All intra when all pictures are coded as "I frame" using just the intra prediction process, in Random Access and low delay, both intra and inter prediction are used. In this paper the experiments will be performed using only the All Intra configuration file to test the intra prediction process with different test sequences (classes from A to E) and different quantization parameter values.

To implement the HM16.7 on the ARM platform, a cross compilation of the HM on a host machine should be performed before, in order to generate an executable file supported by the target machine (ARM processor). For that, it is necessary to install the cross toolchain corresponding to the target platform used, then the binary files of the HEVC encoder are copied to the SD card and executed using the different test sequences.

5.2 DESCRIPTION OF THE ARM PLATFORM

The development platform selected for our embedded implementation was the Banana Pi M3 development board [18]. This is a super low cost single board computer that can support a variety of operating systems including Android and Arch Linux.

This hardware platform is characterized by the following key features:

– Processor: ARM Cortex-A7 Octa-Core 1.8 GHz,

– Memory: 2GB LPDDR3 SDRAM.

The operating system used in this work is the embedded Linux, due to its low cost (freely available source code), ease of customization and its stable kernel. It was also integrated in the SD card. We have designed a Linux Operating System from scratch, this system is scaled in such a way that it consumes less power and material resources, by reducing the size of the image kernel, so that the execution of the HM 16.7 will benefit of the eight cores of the used platform and a big size of memory will be used to execute the code. This way has effectively more optimized the implementation of the HM16.7.

After preparing and booting the image using a dumb-terminal emulation program, the Linux operating system is started and the directory /home can be chosen to execute the HEVC software.

6. EXPERIMENTAL RESULTS AND DISCUSSIONS

6.1 OVERVIEW

After testing the algorithms (algorithm 1 and algorithm 2) implemented in HM16.7 on the Intel processor, different implementation approaches were analyzed and tested on ARM processors in order to highlight the performances of the used algorithm on this platform using the optimized embedded Linux system. The test sequences used in both of these experiments are described in Table 2.

Table 2. Test sequences

Classes	Sequences	Resolution	Frame rate [Hz]
A	PeopleOnStreet	2560 x 1600	30
B	BasketballDrive	1920 x 1080	50
C	Vidyo1	1280 x 720	60
D	RaceHorces	832 x 480	30
E	BQSquare	416 x 240	60

For each platform, we have performed 20 different implementations for the original intra-algorithm and 20 for the modified fast intra-algorithm of HM16.7. The 20 implementations were tested using several test sequences with different classes from A to E (Table 2), with different values of quantization parameter (22, 27, 32 and 37) according to the common JCT-VC test conditions, and using the All-Intra-Main configuration file. The parameters FEN (fast encoder decision), FDM (fast decision for merge RD cost), RDOQ (Rate Distortion Optimization Quantization), SAO (Sample Adaptive Offset) and AMP (Asymmetric Motion Partitions) are all enabled. The rest of parameters are described in the All-intra configuration file [19].

In order to compare the performances of the modified model with those of the original one, we have used the Bjontegaard [20] delta bit rate (BDBR) and the Bjontegaard delta peak signal-to-noise ratio (BDPSNR) to verify both the efficiency of the bit rate and the quality of the output video that is encoded using the proposed method. The PSNR of YUV can be calculated as follows:

$$PSNRyuv = ((PSNRy * 6) + PSNRu + PSNRy) / 8, \quad (4)$$

where PSNRy – the PSNR of luma component, PSNRu and PSNRv are the PSNR for the chroma components.

Additionally, the computational complexity of the original algorithm is evaluated in comparison with the proposed intra algorithm in terms of

encoding time that is calculated using the equation bellow:

$$\Delta T=((Torg-Tprop))/Torg, \quad (5)$$

where *Torg* – is the encoding time of the original model, *Tprop* – the encoding time of the proposed model.

6.2 RESULTS AND COMPARISONS

Experimental results, presented in the Tables 3 and 4, are obtained using the above described method in section 5.1.

Table 3 presents the experimental implementation results on the Intel platform (Intel Core i7, 3.6 GHz, 4GB on RAM memory) under the Linux (Ubuntu 16.04) operating system.

The encoding time and the bit rate are optimized respectively by a percentage that can reach 67% and 3% compared to the original intra prediction algorithm, but the quality of the output video is decreased by a very small percentage of 0.05% in average. In comparison with Kibeya et al. [5] results, the proposed intra prediction algorithms are more responsive in the HM16.7 reference software than the version HM10.0. Our respective results show that the averages of the encoding time are 64.898% for the present implementation and 46.13% for Kibeya et al. [5] implementation, which corresponds here to a more optimization of around 20%.

Table 3. Implementation results on Intel platform

		Fast intra prediction algorithm vs the original in Intel		
Test Sequences	Qp	BDBR(%)	BDPSNR(%)	Timing results(%)
Class A	22	0.94	-0.04	65.87
	27	1.3	-0.037	65.93
	32	1.73	-0.044	64.83
	37	2.03	-0.007	65.41
Average of class A		1.5	-0.032	65.51
Class B	22	1.5	-0.01	65.98
	27	2.06	-0.005	66.01
	32	1.93	-0.006	65.95
	37	1.96	-0.036	66.10
Average of Class B		1.86	-0.014	66.01
Class C	22	1.9	-0.03	66.13
	27	2.45	-0.035	65.86
	32	2.61	-0.066	66.21
	37	2.65	-0.09	66.87
Average of Class C		2.4	-0.055	66.3
Class D	22	2.3	-0.015	60.58
	27	2.44	-0.013	61.34
	32	2.15	-0.017	61.61
	37	2.06	-0.015	61.64
Average of Class D		2.23	-0.015	61.3
Class E	22	1.05	-0.095	65.22
	27	0.11	-0.113	65.84
	32	1.28	-0.131	66.05
	37	1.27	-0.166	64.38
Average of Class E		0.93	-0.12	65.37
Average of all classes		1.784	-0.05	64.898

Table 4 gives the implementation results of the encoding time, the BDBR and the BDPSNR of the used models compared with the original model on ARM platform and based on the Bjontegaard method. As can be seen the implementation results indicate that a gain of 61.5% in average of encoding time by the proposed model compared to the original HM16.7 is achieved. This highlights that the embedded implementation on ARM platform of these algorithms can reduce the computations complexity by a factor of 63% in maximum and a minimum of 59.31%. Additionally, the bit rate was optimized by a maximum of 2.1% in BDBR and a minimum of 0.21%.

Table 4. Implementation results on ARM platform

		Fast intra prediction algorithm vs the original on ARM		
Test Sequences	Qp	BDBR(%)	BDPSNR(%)	Timing results(%)
Class A	22	0.85	-0.05	60.25
	27	1.16	-0.03	62.5
	32	0.5	-0.05	61.56
	37	1.68	-0.05	62.30
Average of class A		1.05	-0.045	61.65
Class B	22	0.65	-0.02	62.26
	27	1.47	-0.005	61.22
	32	1.89	-0.005	60.73
	37	0.21	-0.01	62.85
Average of Class B		1.05	-0.01	61.76
Class C	22	1.62	-0.02	61.66
	27	1.97	-0.03	60.97
	32	2.08	-0.05	60.94
	37	2.10	-0.06	60.29
Average of Class C		1.94	-0.04	60.96
Class D	22	0.46	-0.03	61.11
	27	0.87	-0.03	60.73
	32	1.09	-0.04	60.50
	37	1.36	-0.05	59.31
Average of Class D		0.94	-0.03	60.41
Class E	22	0.69	-0.09	62.86
	27	1.02	-0.15	62.89
	32	1.14	-0.09	62.21
	37	1.14	-0.13	61.96
Average of Class E		0.99	-0.115	62.48
Average of all classes		1.19	-0.049	61.5

On the other hand, there was a loss in the PSNR with a too negligible value of 0.049%. Looking at these results, we can deduce the great performance of the embedded implemented algorithms for the intra prediction process on ARM processors that have speed up the encoding process by around 3 times.

For classes C, D and E, we can see in the Table 4 that the Quantization parameter (Qp) value has an influence on the bit rate and the encoding time, that is when we increase the value of the Qp the bit rate increases but the encoding time decreases.

Table 5 gives the results of the embedded implementations of the original HM16.7 and the proposed HM16.7 intra prediction algorithms on ARM platform in comparison with previously reported results [5]. Concerning Kibeya et al. work [5] the fast intra prediction model is executed in an Intel®Core TM i7-3770 @ 3.4 GHz CPU and 12 GB RAM platform for the HM10.0 version. The bit rate, PSNR and encoding time saving comparisons are illustrated in Table 5. By applying the optimized Linux operating system to implement the fast intra prediction algorithms, the results show that the encoding time is more performant in the ARM processor with a gain of about 15.37% in addition to the significant gain in the bit rate by a factor of 1.19%, also the loss in the PSNR is lower here by a factor of 0.2%.

Table 5. Results of the comparison between the platforms used to implement the proposed algorithm

	Proposed fast intra prediction algorithm on ARM (HM16.7)			**Proposed fast intra prediction algorithm on Intel (HM10.0)**		
Test Sequences	BDBR(%)	BDPSNR(%)	ΔT (%)	BDBR(%)	BDPSNR(%)	ΔT (%)
Class A	1.05	-0.18	61.65	4.1	-0.26	46.24
Class B	1.05	-0.01	61.76	3.13	-0.12	44.69
Class C	1.94	-0.04	60.96	3.17	-0.19	36.65
Class D	0.94	-0.03	60.41	3.41	-0.23	36.39
Class E	0.99	-0.115	62.48	6.57	-0.45	66.68
Average of all sequences	1.19	-0.049	61.5	4.07	-0.25	46.13

Another comparison of the encoding time, bit rate and the PSNR of the proposed model implemented with those reported in [10],[11] is performed. The results are shown in Table 6, which demonstrates the performance in terms of acceleration and video quality of our embedded algorithm compared to the state-of-art fast methods. As can be seen, the average of the time saving of the implemented method is overcome of 12% compared to the case of the algorithms proposed by Xie et al. in [10] and 1.86% in the case of Zhu et al. in [11] results. Additionally, also compared to these methods, we obtained here a lower degradation of the PSNR for the output video (Table 6). On the other hand, their bit rate is more optimized by still negligible values (0.11% and 0.26) respectively for Xie et al. [10] and Zhu et al. [11].

Table 6. Fast Intra Prediction algorithms comparison

	Proposed overall algorithm			**Xie et al. algorithm**			**Zhu et al. algorithm**		
Sequences	BDBR(%)	BDPSNR(%)	ΔT(%)	BDBR(%)	BDPSNR(%)	ΔT(%)	BDBR(%)	BDPSNR(%)	ΔT(%)
Class A	1.05	-0.18	61.65	0.85	-0.25	48.0	1.27	-0.06	61.69
Class B	1.05	-0.01	61.76	2.59	-0.29	49.6	1.38	-0.05	61.76
Class C	1.94	-0.04	60.96	0.9	-0.26	49.9	1.36	-0.07	56.25
Class D	0.94	-0.03	60.41	0.83	-0.29	48.6	1.2	-0.07	51.19
Class E	0.99	-0.115	62.48	1.58	-0.35	51.15	2.12	-0.09	70.49
Average of all classes	1.19	-0.049	61.5	1.3	-0.31	49.6	1.45	-0.07	59.64

7. CONCLUSION

This paper presents an implementation on ARM platform of algorithms that concerns the selection of the best mode between the 35 intra modes defined in the intra prediction block of HEVC/H.265 video encoding standard. The focus has been on the RMD process when the number of verified modes is reduced in order to minimize the computation complexity of HEVC standard. These methods have been applied on the HM10.0 under an Intel Core and have achieved an average of 46.13% of reduction in encoding time in comparison with the original algorithm.

In this paper, this method is applied on the HM16.7 instead of HM10.0 and is implemented on ARM architecture under an optimized embedded Linux operating system. As a result, the run-time was reduced by an average of 61.5% and a maximum of 2.1% of bit rate was added in HM16.7 reference software.

Our short-term perspective is to optimize more in terms of both the algorithm and the embedded implementations within the objective of reaching a real-time implementation of the HEVC/H.265 standard on different platforms by optimizing more the complex functionalities of this standard.

8. REFERENCES

[1] J. Kufa, L. Polak and T. Kratochvil, "HEVC/H.265 vs. VP9 for Full HD and UHD video: Is there any difference in QoE?," *Proceedings of the International Symposium ELMAR*, Zadar, Croatia, 2016, pp. 51-56.

[2] J-R. Ohm, G J. 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 Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1669-1684, 2012.

[3] I. Kim, J. Min, T. Lee, W. Han and J. Park, "Block partitioning structure in the HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1697-1706, 2012.

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

[5] H. Kibeya, F. Belghith, M. Ali Ben Ayed, and N. Masmoudi, "Fast intra-prediction algorithms for high efficiency video coding standard," *Journal of Electronic Imaging*, Sfax, Tunisia, vol. 25, no. 1, article 013028, 2016.

[6] G. J. Sullivan, "Common test conditions and software reference configurations," *Joint Collaborative Team on Video Coding (JCT-VC)*, Geneva, 2013.

[7] H. Touzani, F. Errahimi, A. Mansouri and A. Ahaitouf, "Implementation analysis of HEVC encoding on Zynq platform under embedded Linux," *Proceedings of the International Conference on Wireless Technologies, Embedded and Intelligent Systems (WITS)*, 2017, pp. 1-5.

[8] F. Bossen, B. Bross, K. Suhring and D. Flynn, "HEVC complexity and implementation analysis," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1685–1696, 2012.

[9] G. Venugopal, P. Merkle, D. Marpe and T. Wiegand "Fast template matching for intra prediction," *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Beijing, China, 2018, pp. 1692-1696.

[10] X. Xie, X. Xin and H. Wang, "A fast coding unit division and mode selection method for HEVC intra prediction," *Proceedings of the 4th International Conference on Systems and Informatics (ICSAI)*, Hangzhou, China, 2017, pp. 1302–1307.

[11] W. Zhu, Y. Yi, Hanyu Zhang, P. Chen and Hua Zhang, "Fast mode decision algorithm for HEVC intra coding based on texture partition and direction," *Journal of Real-Time Image Processing*, vol. 22, no. 1, pp. 1-18, 2018.

[12] H. Azgin, A. Can Mert, E. Kalali and I. Hamzaoglu, "An efficient FPGA implementation of HEVC intra prediction," *Proceedings of the IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, 2018, pp. 1302–1307.

[13] H. Smei, K. Smiri and A.Jemai, "A new profiling and pipelining approach for HEVC decoder on ZedBoard platform," *Advances in Science, Technology and Engineering Systems Journal*, vol. 2, no. 6, pp. 40-48, 2017.

[14] W. Liu, J. Li, Y. B. Cho, "A novel architecture for parallel multi-view HEVC decoder on mobile device," *EURASIP Journal on Image and Video Processing*, vol. 2017, no. 1, pp. 102-120, 2017.

[15] D. Patel, T. Lad and D. Shah, "Review on intra-prediction in high efficiency video coding (HEVC) standard," *International Journal of Computer Applications*, vol. 132, no. 13, pp. 27-30, 2015.

[16] J. Lainema, F Bossen, W. Han, J. Min and K. Ugur, "Intra coding of the HEVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1792-1801, 2012.

[17] M. Jamali and S. Coulombe, "RDO cost modeling for low-complexity HEVC intra coding," *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, Vancouver, BC, Canada, 2016, pp. 1-5.

[18] Sinovoip and Foxconn team. What is Banana Pi M3. [Online]. Available at: www.banana-pi.org/m3.html.

[19] High Efficiency Video Coding (HEVC). [Online]. Available at: https://hevc.hhi.fraunhofer.de.

[20] G. Bjontegaard, "Calculation of average PSNR differences between RDcurves," Document VCEG-M33, 2001.

[21] K. Reuze, P. Philippe, O. Deforges and W. Hamidouche, "Intra prediction modes signalling in HEVC," Proceedings of the 2016 IEEE Picture Coding Symposium (PCS), Nuremberg, 2016, pp. 1-5.

Hajar Touzani *received her Embedded System engineering degree from the National School of Applied Sciences, Fez, Morocco in 2015. She is currently pursuing her PhD degree in Laboratory Intelligent systems, Georesources and renewable Energies, Faculty of Sciences and Technologies, University Sidi Mohamed Ben Abdellah of Fez. Her PhD work focuses on the real time embedded implementations, video coding and image processing, Software/Hardware implementations.*

Ibtissem Wali *completed her PhD in Electronics and Information Technologies Laboratory from the National School of Engineering of Sfax, Tunisia, in 2018. Her main research activities are focused on signal, image and video processing and algorithm optimizations.*

Fatima Errahimi, *teacher and researcher at the Faculty of Sciences and Technologies in Sidi Mohammed Ben Abdellah University of Fez. She got PhD degree on Automatic Control Systems and robotics in 2004. Her special fields of interest include the integration of renewable energy sources in building energy management, optimization control for demand side units in households and smart grid, signal and image processing. She participated in many research projects among which the IRESEN one on low cost CPV in Morocco. She supervised many PhD thesis and several post graduate students.*

Anass Mansouri *received Ph.D degrees in Microelectronics and Embedded System from Faculty of Sciences and Technologies of Fez, MOROCCO, in 2009. He is an Associate Professor in National School of Applied Sciences of Fez. His major research interests include VLSI and embedded architectures design, video and image processing, as well as Software/Hardware design and optimization. He is member of the Intelligent systems, Georesources and renewable Energies and head of the team Embedeed systems, electronics and telecommunication.*

Nouri Masmoudi *received electrical engineering degree from the Faculty of Sciences and Technics Sfax, Tunisia, in 1982, the DEA degree from the*

National Institute of Applied Sciences Lyon and University Claude BernardLyon, France in 1984. He received his Ph.D. degree from the National School Engineering of Tunis (ENIT), Tunisia in 1990. Since 2000, he has been a group leader Circuits and Systems in the Laboratory of Electronics and Information Technology. His research activities have been devoted to several topics: Design, Telecommunication, Embedded Systems, Information Technology, Video Coding and Image Processing.

Ali Ahaitouf *teacher and researcher in the University of Sidi Mohammed Ben Abdellah of Fez. He obtained his PhD in electronics in 1998. He is director of the Intelligent systems, Georesources and renewable Energies. His research field covers microelectronics and solar compounds, digital and analog design of the integrated circuits, Image and data compression. He managed many research multilateral projects, related to analog design optimization, electronics component characterization and optimization as well as solar energy under concentration (CPV). He supervised a dozen of PhD thesis and published more than fifteen papers in renowned journals.*