J Real-Time Image Proc (2016) 11:675–691
DOI 10.1007/s11554-014-0407-0

SPECIAL ISSUE PAPER

A new fast motion estimation algorithm using fast mode decision for high-efficiency video coding standard

Fatma Belghith · Hassan Kibeya · Hassen Loukil · Mohamed Ali Ben Ayed · Nouri Masmoudi

Received: 22 October 2013 / Accepted: 4 February 2014 / Published online: 26 February 2014

Abstract High-efficiency video coding is the latest standardization effort of the International Organization for Standardization and the International Telecommunication Union. This new standard adopts an exhaustive algorithm of decision based on a recursive quad-tree structured coding unit, prediction unit, and transform unit. Consequently, an important coding efficiency may be achieved. However, a significant computational complexity is resulted. To speed up the encoding process, efficient algorithms based on fast mode decision and optimized motion estimation were adopted in this paper. The aim was to reduce the complexity of the motion estimation algorithm by modifying its search pattern. Then, it was combined with a new fast mode decision algorithm to further improve the coding efficiency. Experimental results show a significant speedup in terms of encoding time and bit-rate saving with tolerable quality degradation. In fact, the proposed algorithm permits a main reduction that can reach up to 75 % in encoding time. This improvement is accompanied with an average PSNR loss of 0.12 dB and a decrease by 0.5 % in terms of bit-rate.

Keywords Fast encoding · Motion estimation · HDS · SDSP · HEVC

F. Belghith (✉) · H. Kibeya · H. Loukil · M. A. Ben Ayed · N. Masmoudi
Electronics and Information Technology Laboratory, University of Sfax, National Engineering School of Sfax, Sfax, Tunisia
e-mail: fatmabelghithenis@gmail.com

N. Masmoudi
e-mail: Nouri.Masmoudi@enis.rnu.tn

1 Introduction

After the success of H.264/AVC, video compression standards target higher resolutions reaching up to 4K by 2K, known as ultra high definition (UHD) levels. As the demand increased, the International Telecommunication Union and International Organization for Standardization develop a new video coding standard with improved compressing tools focusing on UHD videos. The high-efficiency video coding (HEVC) standard is emerging, aiming to double the compression rates when compared to H.264/AVC for the same video quality [1]. The computational complexity of the encoding and decoding process increases from 2 to 10 times as compared to the H.264/AVC. Motion estimation and motion compensation are two essential processes in block-based video coding, because they reduce the temporal redundancy in consecutive frames. Motion estimation is based on searching the best motion vector in previous or future frames or combining both of them. Reducing the number of searching points decreases the complexity of motion estimation. In addition to that, some algorithms have been proposed to decrease the mode decision complexity for the HEVC encoder achieving a significant time saving with a negligible quality loss. This paper proposes a new motion estimation algorithm (MEA) for HEVC encoder based on the modification of the pattern search. It also explains and adopts new options for fast mode decision used for sub-partitioning large coding unit (LCU). The rest of the paper is organized as follows. In Sect. 2, an overview of the HEVC encoder is presented. Section 3 details some related works on MEA and fast mode decision used in HEVC to save the encoding time. Methods based on fast coding are developed in Sect. 4. In Sect. 5, the motion estimation process is detailed and followed by our proposed algorithms. Then, the

experimental results based on combining our two approaches are given in Sect. 6. Finally, a conclusion and some perspectives are presented in Sect. 7.

2 Overview of the HEVC encoder

2.1 Encoder structure

Video coding layer (VCL) uses the same hybrid approach which is based on intra/inter-prediction modules. This approach was used in the previous standards. Inter-prediction mode is a temporal prediction using motion estimation between different frames and intra prediction mode is a spatial prediction in the same frame. These stages are followed by the integer transformation and scalar quantization. As shown in Fig. 1, quantized coefficients will be then entropy coded using CABAC. As in previous standards, two loop filters that consist of a deblocking and sample adaptive offset (SAO) filters are applied to the reconstructed frame. The deblocking filter is proposed to reduce the blocking artifacts due to the block-based coding. It is applied only on the samples located next to transform unit (TU) or prediction unit (PU) boundaries. SAO is a process that modifies the samples after the deblocking filter. It is a non-linear filter which is obtained through a look-up table. This filter is able to reduce the distortion of the reconstructed frames by adding an offset of the reconstructed pixels [1]. The main feature which makes HEVC different from AVC replaces the macroblock structure by a quad-tree structure.

The principle of the new structure is based on coding tree unit (CTU) instead of macroblock structure. This structure is based on blocks and units. The three basic units used in the quad-tree structure for HEVC are coding unit (CU), prediction unit (PU), and transform unit (TU) [2]. As the HEVC is basically allocated to larger videos, larger macroblocks are needed to encode more efficiently. On the other side, details are provided only through small blocks. This is the challenge of the HEVC standard. While using, the quad-tree provides a compromise between a good quality and less bit-rate. This new structure is more detailed in Sect. 3.

For intra prediction, in HEVC, the size of the smallest block did not change from the H.264/AVC one which is 4×4. In addition, to perform larger bloc's size, the PU can reach up to 64×64 [3]. The number of prediction modes increased to augment the coding efficiency. There are at most 34 intra prediction modes. The angular modes in HEVC are more complex than the directional ones in H.264/AVC since more multiplications are required. The number of intra modes varies with the PU size as shown in Table 1 [4].

To get the best intra prediction mode, the choice in the HEVC reference software is based on computing for each

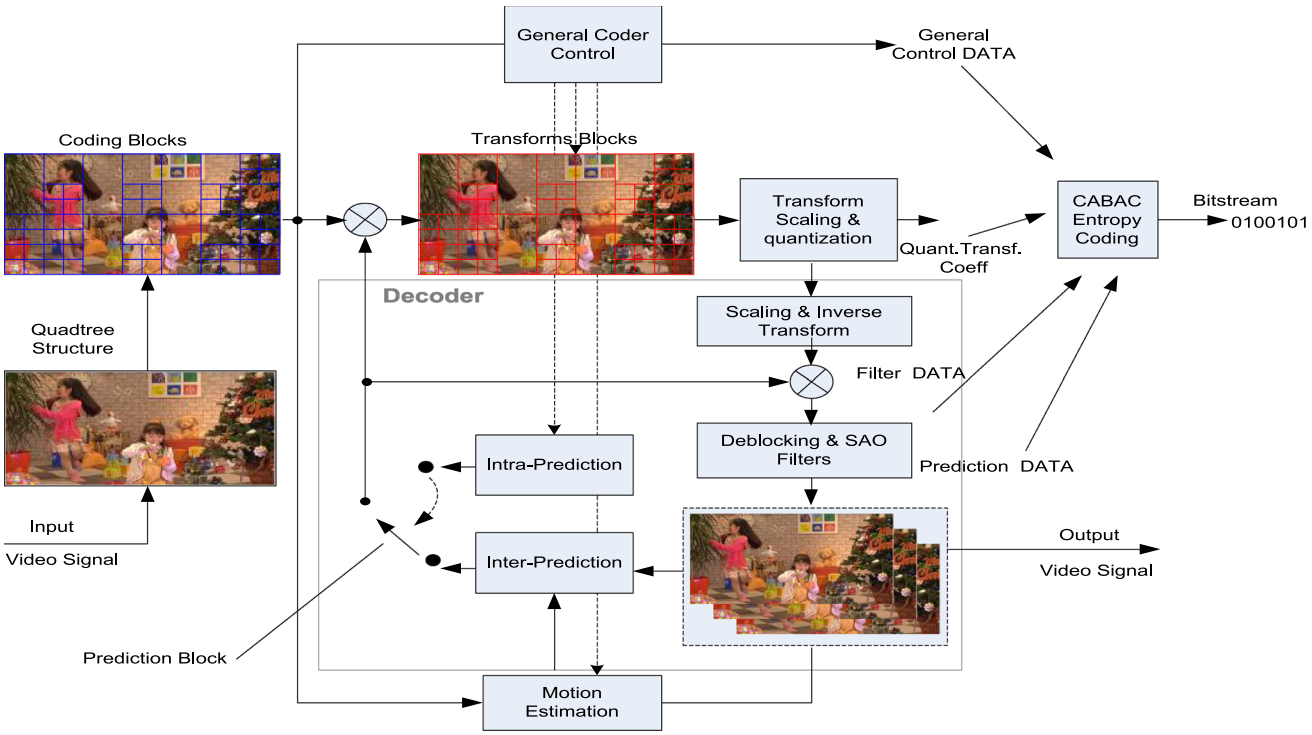


Fig. 1 HEVC encoder structure

Table 1 Number of supported intra modes for each PU size

Prediction unit size	Number of intra angular modes
4 × 4	19
8 × 8	35
16 × 16	35
32 × 32	35
64 × 64	35

mode its rate-distortion cost known as J_{RDO} given by the following formula:

$$J_{\mathrm{RDO}} = \mathrm{SSD} + \lambda \times R \quad (1)$$

where SSD is the sum of squared differences, R is the total bits needed to encode the relative mode, and λ is a value calculated using different parameters such as the quantization parameter (QP), the slice type (I, P or B slice), and the configuration (random access, low delay or intra-only). The best mode is the one with minimum rate-distortion J_{RDO} [5]. In case of processing the chrominance, the five existent modes are: intra planar, intra angular (dir = 26, vertical), intra angular (dir = 10, horizontal), intra DC, and intra derived which mean that the chrominance uses the same angular direction of its corresponding luminance [4].

The process used to encode an inter-picture is to choose a motion vector applied for predicting the current block by exploiting the temporal redundancy present in the video signal. The encoding process for inter-picture prediction consists of forming motion data for each PU that includes an index for a reference picture and motion vector (MV) that will be applied for predicting the samples of each block. The basic theory of motion compensation prediction (MCP) is to reduce the amount of the transmitted information to the decoder. For each block in the frame, the encoder searches in the reference frames to find its best matching block. The MV is coded by transmitting only the index and the difference vector into the candidate list [5]. The differences between vectors are much smaller in amplitude and thus can be coded more efficiently. Motion estimation (ME) is one of the key elements in video coding standard. It is dedicated to achieve high coding performance by reducing temporal redundancy. Developing fast algorithms for ME has been an essential and challenging issue. The most simple and basic way to find the optimal position is the full search (FS) algorithm. It checks every possible point in the search range to select the best one. The best point is chosen after computing the rate-distortion (RD) for each candidate. Although the full search algorithm reaches the global minimum, the computational complexity is usually very excessive. This part will be more detailed in Sect. 5.

Either intra or inter-frame prediction, the residual signal which is the difference between the source frame and the predicted one, is the subject of a conversion to concentrate the signal energy. The purpose of the transformation is to convert the unit into the spatial frequency space. However, as the HEVC is devoted mainly for large resolutions, different sizes from 4 × 4 to 32 × 32 for the TU were used [6]. All transformed coefficients in the TU are equally quantized depending on the QP value. HEVC uses the same uniform-reconstruction quantization (URQ) as H.264/AVC. The range of the QP values is defined from 0 to 51. Quantization scaling matrices are also supported [7].

2.2 HM test reference analysis

The HM software implementation is developed by JCT-VC group as a common reference of the HEVC encoder. Due to the complexity of the HEVC encoder, the HM is supposed to be relatively slower than the AVC encoder as it has extra modules like the quad-tree structure, SAO.etc. Among the contributing factors to the slowness of the HM encoder is a heavy reliance on the rate-distortion optimization. JCT-VC common test conditions defines different configurations for the encoder that can be used [8]. These configurations consist of:

- All intra (AI), where all pictures are encoded using only I slices.
- Random access (RA), where all the pictures are reordered in a pyramidal structure with a random access picture about every 1 s. This configuration may be used in a broadcasting environment.
- Low delay (LD), where only the first frame is encoded using I slices and no picture reordering is used. This configuration is generally used in a videoconferencing environment. This paper focused on motion estimation algorithms and some fast encoding methods which influence only the inter-mode decision. Thus, only RA and LD configurations were considered.

The HM (HEVC test model) encoder has been profiled to determine which of the components is the most time-consuming. Figure 2 shows the time spent in various C++ modules in the HEVC encoder.

These results were obtained with *Vtune* tools when processing "Vidyo1" sequence coded in RA mode with QP = 37. In this configuration, inter-prediction module takes the lion share of the encoding time with up to 40 %. The second important time consumer module is the RdCost that includes a heavy computation of sum of absolute differences (SAD) and other distortion metrics which consumes about 33 % of the total encoding time. Transformation and quantization are equal to 9 %. On the entropy coding front, the amount of time spent in core CABAC operations is relatively small and equal to 3 %. It is also interesting to note that a small amount of time is

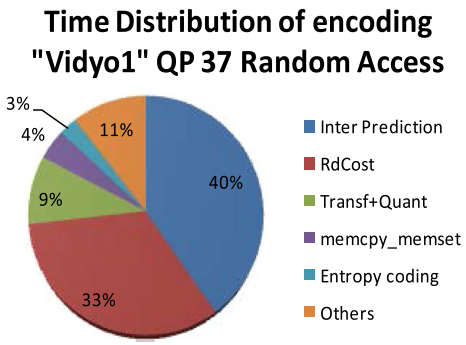


Fig. 2 Encoding time distribution

spent on setting and copying memory areas (memcpy and memset system functions). In this case, around 4 % of the time is spent in these system functions. The Fig. 2 shows that the two most critical modules in term of time consuming are inter prediction module based on motion estimation and RD cost module based on mode decision. That is why in this work, we started exploring these two parts.

3 Related works

The complexity of HEVC is a critical problem, especially when looking forward to a real-time implementation. It is reported [9] that mode decision and motion estimation modules take the lion's share while profiling the encoder. Therefore, in the literature, a number of efforts have been made to explore algorithms based on fast decision and motion estimation. In [10], an algorithm called "Fast Coding unit Size algorithm" was based on Bayesian decision rule saving an average of 41 % from the encoding time with a negligible loss of 1.88 % on RD performance. In [11], they present the "Effective CU size decision" algorithm based on skipping some specific depths which are rarely used in the previous frame. This algorithm provides a time saving reaching 26 % with a negligible coding efficiency loss and a slight bit-rate increase. In [12], the recursive CU splitting process is early terminated according to an adaptive threshold value of the mean square error (MSE). This work investigated the correlation between CU splitting and MSE in the current CU level. When the MSE of the current CU is small, most CUs do not need to be split. If the MSE of the current CU reaches MSE_{th}, the partition process can be terminated. This work reduced the encoding time by 24 % while having a little decrease in quality with 0.3 % in worst cases. At the time of writing, we evaluated a recent work based on the contextual mode information of neighboring CUs to detect the merge skip mode. This proposed achieves the average time-saving factor of 43.67 % in the random access configuration with the HEVC test model (HM) 10.0 reference software. Compared to HM 10.0 encoder, a small bit-rate loss of 0.55 % is observed without significant loss of image quality [13]. In addition to the algorithms previously quoted, other algorithms were implemented in the reference software HM. These algorithms will be detailed in the next section.

In the other hand, a lot of researches are concentrated on improving the motion estimation algorithm to optimize the test zonal search (TZ search). In [14], an algorithm based on a hexagonal pattern was proposed. The proposal contains different algorithms based on this pattern replaced at the first step (initial grid step) and at the refinement stage. The simulation results reveal that the proposed algorithms reduce a major amount (around 40–80 %) of the motion estimation complexity compared to the TZ search algorithm implemented in HEVC reference software. In [15], authors modified the pattern search from diamond to hexagonal as [14]. Further, the algorithm was improved by modifying the searching threshold in each grid in the search area. Results show that the overall encoding time can be reduced by almost 50 % compared to the TZ search algorithm, while maintaining almost the same PSNR and bit-rate.

When analyzing these previous works, we note that different mode decision algorithms were proposed for the HEVC to speed up encoding. Many algorithms were proposed based on fast mode decision, especially on the quadtree structure. Furthermore, other proposed algorithms are based on decreasing the number of search to optimize the motion estimation process. This paper is based on combining the two concepts for an efficient coding time saving.

4 Effective fast decision methods

The coding structure consists of four level structures to have a partitioning in multiple sizes of blocks. The CU is defined as the basic unit chosen always as a square form. It is basically a replacement of (16 × 16) macroblock of H.264/AVC. The principal difference between them is that the CU can have different sizes. The whole processing, including the intra/inter-prediction, the transformation, and the entropy coding, is based on CU [16]. A CU is identified by a "p" depth and a $2N \times 2N$ size. It can be split into several PUs when the split flag is set to zero. Otherwise, it is divided into four smaller CUs with a depth equal to $p + 1$ and sized $N \times N$. Consequently, each CU of depth equal to p is processed in a recursive way.

If the depth of the divided CU reaches the smaller unit sized 8 × 8, the partitioning is stopped [17]. Once the process of the CU partitioning is launched, the prediction methods will be specified. The basic unit for the prediction process is the PU. It should be noted that the PU is defined for all the CUs in each depth and its maximum size is

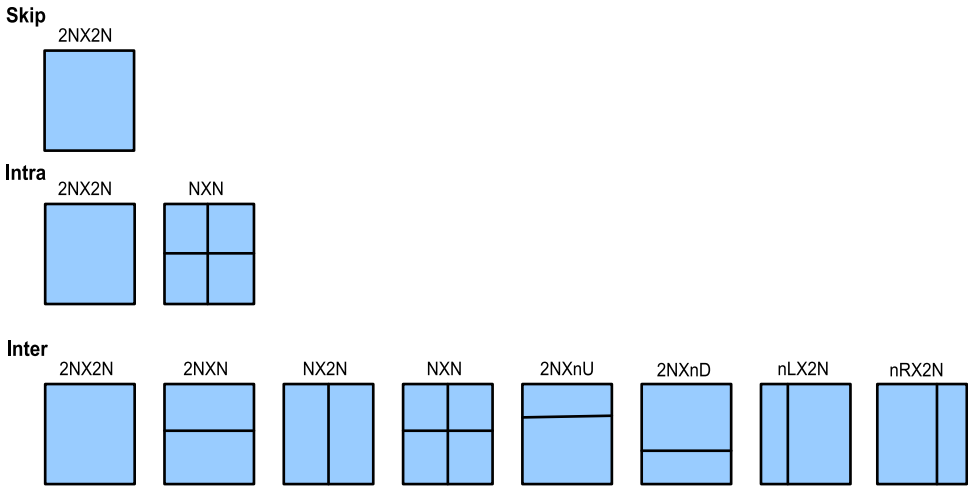


Fig. 3 Partitioning modes of prediction blocks

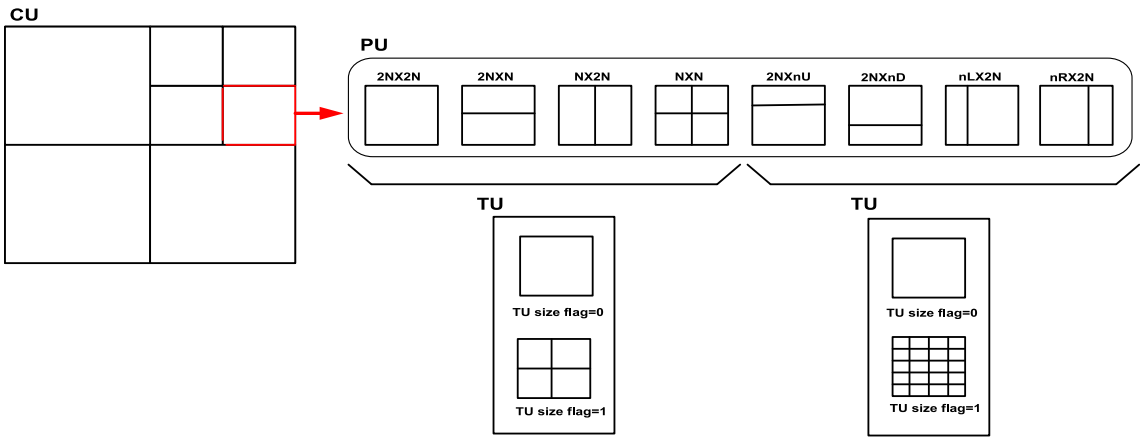


Fig. 4 Relation between different units

limited to its corresponding CU. Two terms are specified in the prediction: the prediction mode and the size of the PU. Similar to the previous standards, the prediction mode can be skip, intra or inter. The possible PU sizes are defined according to the prediction mode [18]. Figure 3 shows the various possible blocks of PU according to the prediction modes.

In addition to the CU and the PU, a TU is used for the transformation and the quantification. This unit is defined separately. It should be noted that the size of the TU can be larger than the PU one, but it should not exceed the size of CU. The size of the TU is fixed by the split flag. When it is set to 1, the TU is partitioned. Figure 4 illustrates the relation between the three different units. For each CU, a PU is defined by specifying the way in which the prediction can be generated. The TU size is specified by the size of the CU and the type of the PU partitioning. This kind of flexible and recursive processing provides several benefits. The first comes from supporting CU sizes greater than the conventional 16×16 size. When the region is homogeneous, larger CUs will be coded leading to a relatively smaller number of symbols compared to the case using several small blocks for the same area. In addition, by eliminating the distinction between macroblock and sub-macroblock and using only the coding unit, the multilevel quad-tree structure can be specified in a very simple way, especially for parallel processing.

Based on this recursive structure, the encoder needs to exhaust all the combinations of CU, PU and TU to find the optimal solution which is a compromise between an optimized rate distortion and a best video quality.

Figure 5 shows the conventional mode decision process for partitioning CUs. Each CU is encoded by determining its best mode. For the current CU, all the modes are tested one by one from the skip to intra $N \times N$. This is done by calculating continuously the RD cost of each CU. The computing process is a stage which takes enormous time and induces a great complexity.

It would be preferable if the encoder can perceive the best mode without carrying out the computing of the RD

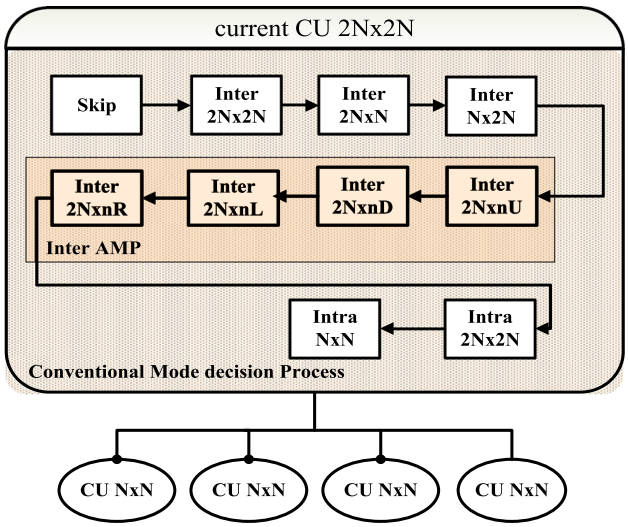


Fig. 5 The conventional mode decision process

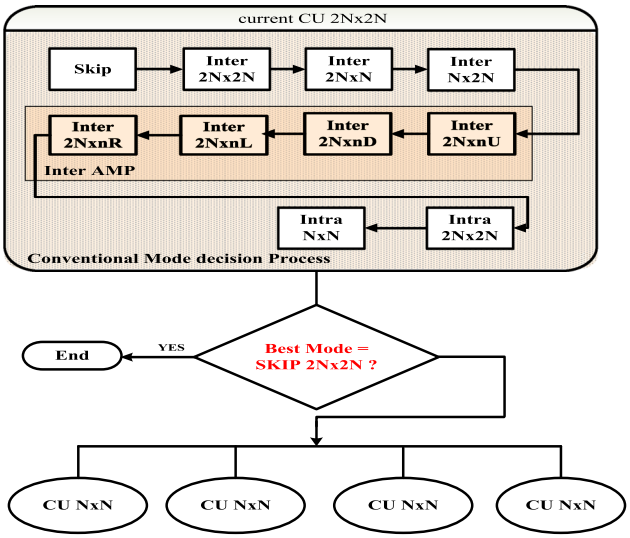


Fig. 6 Early CU algorithm

cost for all the modes. To reduce the computational complexity of the encoder, many fast mode decision algorithms for partitioning were adopted in HEVC test model, such as ECU, ESD and CBF that are detailed below.

4.1 Early CU termination

The early CU termination (ECU) [19] is an algorithm that operates in the passage from depth p to depth $p + 1$. The sub-tree computations can be skipped if the best prediction mode of the current CU is Skip mode as Fig. 6 shows. The best mode is chosen by computing the RD cost. If the minimal RD cost corresponds to the skip mode, there is no need to continue the partitioning.

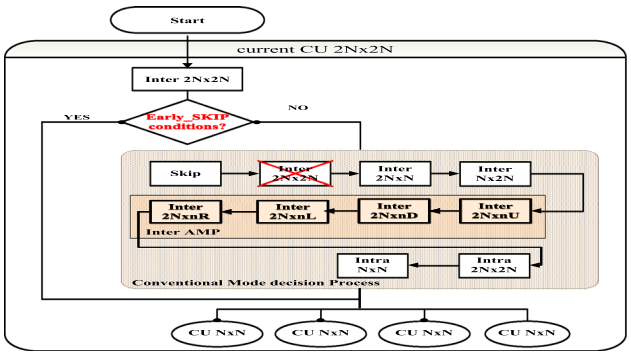


Fig. 7 Early skip detection algorithm

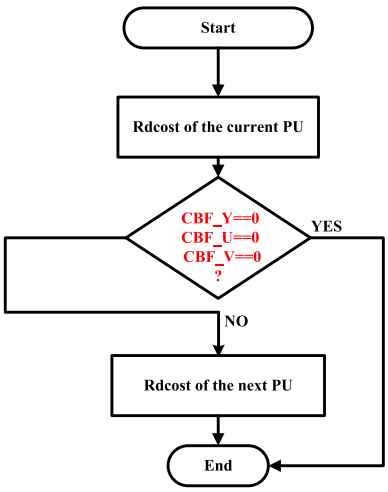


Fig. 8 Coded block flag algorithm

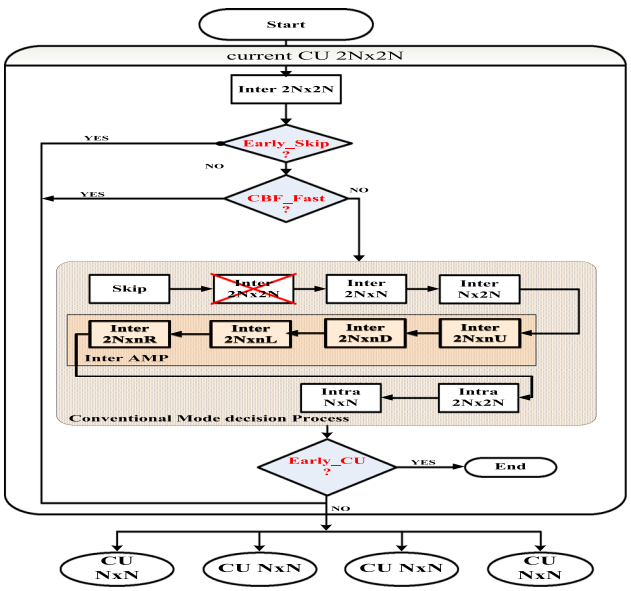


Fig. 9 Our proposed fast algorithm for partitioning

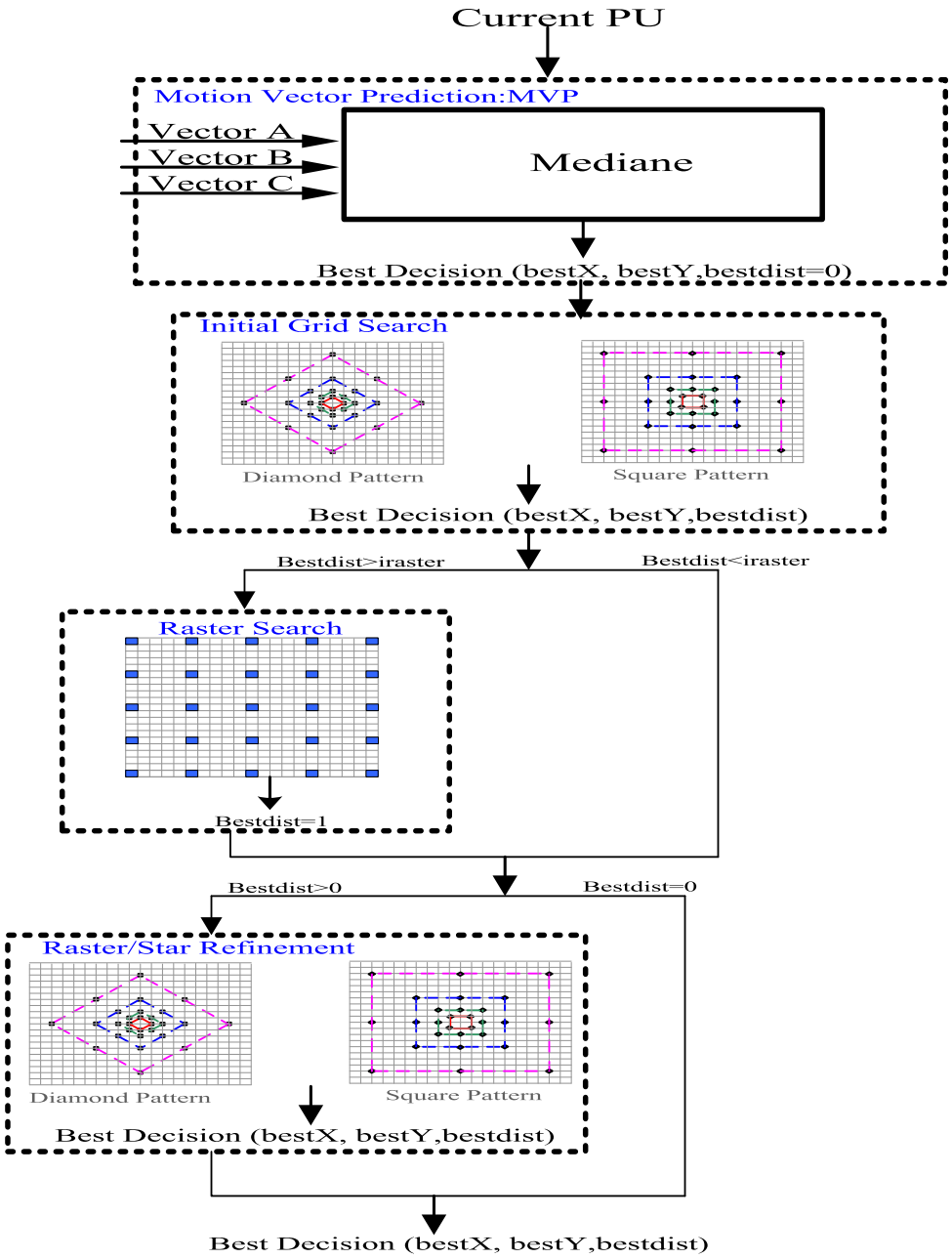


Fig. 10 TZ search algorithm diagram

4.2 Early skip detection

Some statistics show that the skip mode was the most chosen mode [20]. This explains the fact that a great improvement is obtained when the detection of the skip mode is anticipated. Choosing the skip mode is a very effective tool for coding as it represents a coded block without residual information.

The early skip detection algorithm (ESD) represents a simple checking of the differential motion vector (DMV) and the coded block flag (CBF) which are the two conditions called as "early Skip conditions" after searching the best inter $2N \times 2N$ mode as shown in Fig. 7. The current CU searches two modes inter $2N \times 2N$ (AMVP and merge) before checking the skip mode. After selecting the mode having the minimum RD cost, the DMV and CBF are checked. If DMV of the best inter $2N \times 2N$ mode is equal to (0, 0) and CBF is equal to zero, the best mode of current CU will be set to skip mode. Thus, the remaining PU modes are not investigated anymore [21].

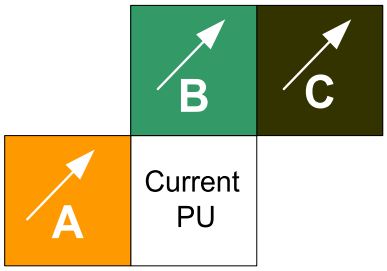


Fig. 11 Motion vector prediction

4.3 Coded block flag algorithm

The coded block flag (CBF) is an algorithm that allows an early detection of the best prediction mode [22]. RD cost is computed for each mode of the PU belonging to a CU. Then, the coded block flag is carried out. If CBF is zero (that means all transform coefficients are zeros) after encoding a PU for luminance (Y) and two chrominance (U, V) components, the next PU encoding process of the CU is terminated. The algorithm is described in Fig. 8.

4.4 Proposed algorithm for fast partitioning

All the methods detailed below are already implemented in the HM encoder. But they are disabled in the HEVC common coding conditions.

Our proposed algorithm takes advantage of the three algorithms (ESD, ECU and CBF). Our new algorithm is illustrated in Fig. 9 as follows:

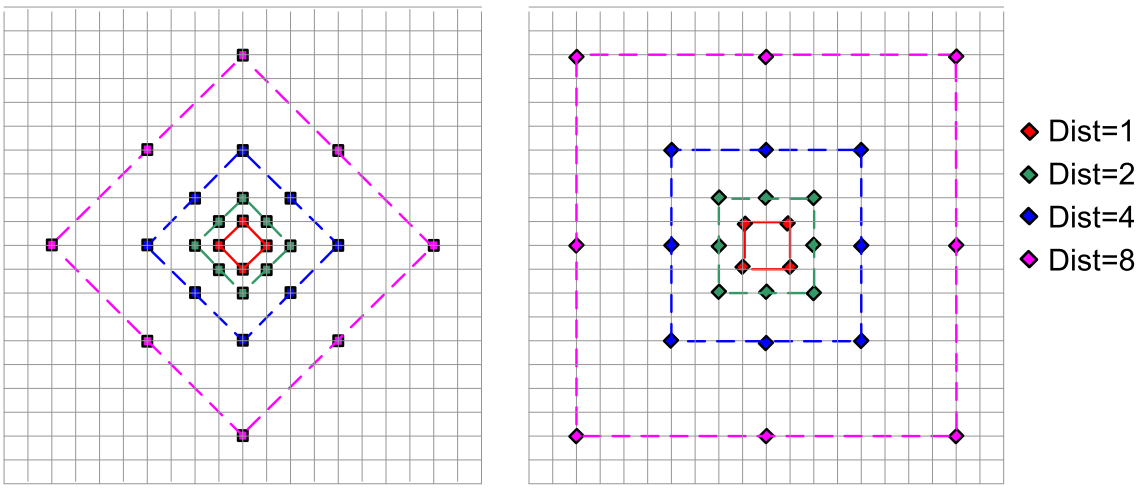
The inter $2N \times 2N$ is first tested to take advantage from the ESD strategy. Then, ESD and CBF conditions are tested. If the conditions are verified, the evaluation of the other modes stops. Otherwise, we evaluate the conventional mode decision process. Once for a CU all modes were checked, the ECU can be tested by verifying if the best mode is skip or not to decide the partitioning or not.

5 Motion estimation process

As the profiling results show, the inter-prediction takes a considerable part on time consuming as well as the mode decision part. The inter-prediction is conceptually simple in HEVC, but comes with some overhead compared to the previous standard H.264/AVC. Motion estimation is an important stage in video coding because it exploits temporal redundancies present within a sequence. The main purpose of motion estimation is to obtain the MV. This requires an important computing power. Implementing fast algorithms for motion estimation can reduce the complexity of computing, thus reducing the encoding time, while preserving the same quality and bit-rate. The block matching methods have been more specifically developed in the framework of video coding [23]. All standards to date, including HEVC, are based on this paradigm. HEVC adopts two algorithms.

5.1 Full search algorithm

This algorithm presents a very intuitive choice for the search area in the whole reference image. It consists of processing all the pixels in the search window to find the best motion vector having the minimum SAD. While using the FS algorithm as the main tool for the inter-prediction, motion estimation spends more than 96 % of the required encoding time [24]. Other techniques exist which are able



(a) Diamond Pattern **(b)** Square Pattern

Fig. 12 Search patterns for motion estimation

to find the motion vector with less significant complexity, but they are suboptimal. In fact, the chosen vector is not always the best one. TZ search is an algorithm implemented in the HM software test model and reduces the computational complexity compared to the full search algorithm. In fact, when applying TZ search, it increases the speed by a ratio of 23 compared to FS with small quality losses.

5.2 TZ search algorithm

The TZS algorithm is a combination of four stages as shown in Fig. 10: motion vector prediction, zonal search, raster search and refinement stage.

The flow of the complete algorithm can be divided into four steps in the following sub-sections:

5.2.1 MVP: motion vector prediction

The TZS algorithm employs the median predictor obtained using left predictor, up predictor, upper right predictor as Fig. 11 shows. The minimum of these predictors is selected as a starting point for the next search steps.

$$\text{Median}(A, B, C) = A + B + C - \text{Min}(A, \text{Min}(B, C)) - \text{Max}(A, \text{Max}(B, C))$$

5.2.2 Initial grid search

At this step, a diamond or a square pattern is used to find the search window while varying the distance (dist) by a power of 2, from 1 to search range which defines the maximum length of the search window. The patterns used are either diamond or square depending on the configuration.

A sample grid with a search range 8 (maximum distance = 8) for the diamond (a) and the square pattern (b) is shown in Fig. 12. All the points of the pattern are tested. The best point having the minimum SAD is chosen as the center search point for further steps. The distance obtained for the best point is stored in the "*Bestdist*" variable which will be evaluated in further steps.

5.2.3 Raster search

The raster search is a simple full search on a down-sampled version of the search window. As Fig. 13 shows, raster search is done for "*iRaster*" = 5. A predefined value "*iRaster*" for raster scan can be set in the configuration file [25]. The condition to perform the raster search is that "*Bestdist*" (obtained from previous step) is superior to "*iRaster*". If this condition is satisfied, "*Bestdist*" is changed to "*iRaster*" value. Else, no raster search is conducted.

5.2.4 Raster/star refinement

This final step is a refinement of the motion vectors obtained from previous steps. The refinement can be raster or star. Only one of the refinement methods is enabled.

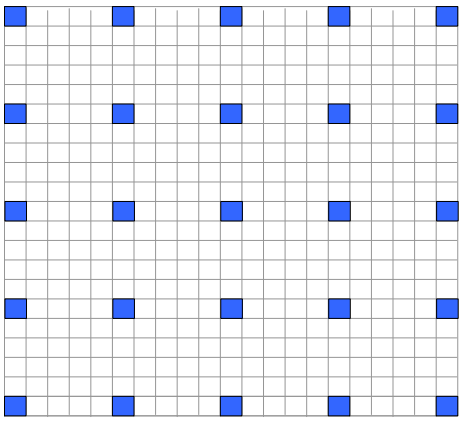


Fig. 13 Raster search for iRaster = 5

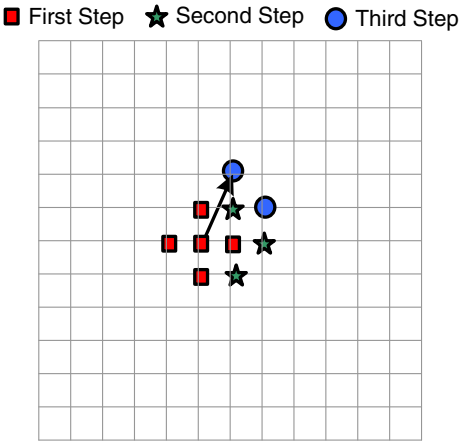


Fig. 14 Small diamond search pattern

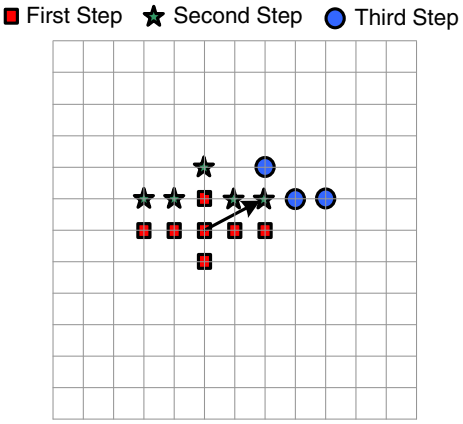


Fig. 15 Horizontal diamond search pattern

Either square or diamond pattern is used. These details are fixed in the TZ configuration. The difference between the two refinement types is that the raster refinement is obtained by down-scaling the *Bestdist* obtained in the previous step and the star refinement is done by up-scaling it. The down-scaling is done until *BestDist* will be equal to 1. The up-scaling stops when the *BestDist* arrives at the search range value.

Table 2 Evaluation criteria

Criteria	Description	Formula
ΔPSNR (dB)	PSNR loss	PSNRp-PSNRo
ΔBR (%)	Bit-rate increase	(BPp/BPo − 1) × 100 %
BDPSNR (dB)	Bjontegaard average ΔPSNR	Defined in [31]
BDBR (%)	Bjontegaard average ΔBR	Defined in [31]
ΔT (%)	Encoding time speedup	(Tp/To − 1) × 100 %

5.3 Proposed fast algorithms for motion estimation

Our proposal for fast algorithms for motion estimation consists of improving the TZ search algorithm by modifying the pattern. In these algorithms, the first steps MVP and the initial grid search were kept. The change starts from the raster search step. As from the initial grid search, a best point was chosen; there is no need to have an additional search while the refinement stage exists. Consequently, the stage of refinement is conducted directly.

This step will be based on two different algorithms namely small diamond search pattern (SDSP) [26] or horizontal diamond search (HDS) [27] instead of using the simple diamond or square pattern search.

5.3.1 Small diamond search pattern

The small diamond search pattern algorithm [26] consists of operating on the central point and its four neighbors. The search consists of a loop of small diamond search pattern until the best matched point corresponds to the center of the diamond. Figure 14 shows the SDSP search pattern

Table 3 Results of the proposed SDSP and HDS versus the original algorithm

	Sequences	SDSP			HDS		
		ΔBR (%)	ΔPSNR (dB)	ΔT (%)	ΔBR (%)	ΔPSNR (dB)	ΔT (%)
Class A 2,560 × 1,600	Traffic	1.17	−0.01	−2	1.11	−0.01	−1.9
	PeopleOnStreet	0.73	−0.01	−9.9	0.61	−0.01	−10
Average Class A		0.95	−0.01	−5.95	0.86	−0.01	−5.95
Class B 1,920 × 1,080	Kimono	0.89	0	−11.1	0.75	0	−11.1
	ParkScene	0.37	0	−3.8	0.34	0	−3.7
	Cactus	0.25	0	−6.9	0.24	0	−6.8
	BasketballDrive	1.01	0	−13	0.75	0	−13
	BQTerrace	0.03	0	−4	0.05	0	−3.9
Average Class B		0.51	0	−7.76	0.43	0	−7.7
Class C 832 × 480	BasketballDrill	1.17	−0.01	−6.7	1.14	0	−6.5
	BQMall	0.65	−0.01	−5.2	0.67	−0.01	−5.3
	PartyScene	0.16	0	−4	0.14	0	−3.8
	RaceHorses	1.48	−0.01	−11.8	1.4	−0.01	−11.9
Average Class C		0.87	−0.0075	−6.9	0.84	−0.005	−6.88
Class D 416 × 240	BasketballPass	0.21	0	−5.1	0.14	0	−5.3
	BQSquare	0	0	−0.1	0.04	0	−0.2
	BlowingBubbles	0.07	−0.01	−4.9	0.03	0	−3.4
	RaceHorses	1.52	−0.01	−8.8	1.37	−0.02	−8.9
Average Class D		0.45	−0.005	−6.27	0.4	−0.005	−5.87
Class E 1,280 × 720	Vidyo1	0.01	0	−1	0.07	0	−1.2
	Vidyo3	0.1	0	−1.3	0.14	0	−1.6
	Vidyo4	0.39	0	−3.1	0.32	0	−2.7
Average Class E		0.16	0	−1.8	0.18	0	−1.83
Average		0.51	−0.01	−4.8	0.52	0	−5.65

presenting the final motion vector chosen. The main improvement of this algorithm is the speed performances as SDSP reduces the number of searching point.

5.3.2 Horizontal diamond search

For most videos, we observe that objects move in a translational manner and motion is more likely to be in the horizontal direction than the vertical one [28].

Figure 15 illustrates the HDS algorithm that consists of a "small diamond" with two extra points on the horizontal direction added. To find the best point, the algorithm is repeated until the best search corresponds to the center point or the limits of the search window are crossed.

In fact, by adapting one of the two proposed search patterns, we can overcome the effect of omitting the step of global minimum search and hence save computational time that can be reduced by a factor up to three.

6 Experimental results

6.1 Experimental conditions

To evaluate the performance of the proposed approaches, different algorithms were implemented on the recent available HEVC reference software (HM 10.0) [29]. The evaluation was done for the fast mode decision algorithm, the two proposed motion estimation algorithms and then when combining these two approaches. All the implementations were compared to the original algorithm in terms of PSNR, bit-rate and encoding time speedup. The experimental conditions are similar to the common test conditions [8] in HEVC. The largest coding unit (LCU) is fixed to 64 × 64 with a maximum depth level equal to 4, resulting in a minimum coding unit size of 8 × 8. The maximum search range used is set to 64 and the entropy coder used is CABAC. The number of frames taken in each sequence is 100.

All the simulations were carried on a Windows 7 OS platform with Intel ®core TM i7-3770 @ 3,4 GHz CPU and 12 GB RAM. The proposed algorithms were evaluated with QPs 22, 27, 32 and 37 using all the sequences recommended by JCT-VC for five resolutions [30] (416 × 240/832 × 480/1,280 × 720/1,920 × 1,080/2,560 × 1,600 formats) for the configuration random access (RA).

6.2 Evaluation criteria

The evaluation was done through the following criteria given in Table 2.

where:

- PSNRp, BPp and Tp represent, respectively, the PSNR index, the bit-rate and the encoding time of the proposed algorithm.
- PSNRo, BPo and To represent, respectively, the PSNR index, the bit-rate and the encoding time of the original algorithm.

6.3 Results

Table 3 evaluates the performance of the two proposed algorithms for ME, namely HDS and SDSP. The two approaches are not very different concerning the performances. For SDSP, the time saving is in average of 5 % while maintaining the same quality for most videos with a little degradation in bit-rate by 0.5 %. In fact, in the worst case, the time saving can be only 0.1 % for the video BQSquare that is characterized by its slow motion as the camera moves from the left to up right slowly. In this video, the background has low motion. Some people are moving in predicable directions with low speed.

As this algorithm is for reducing the motion estimation complexity, the improvement for videos with low motion

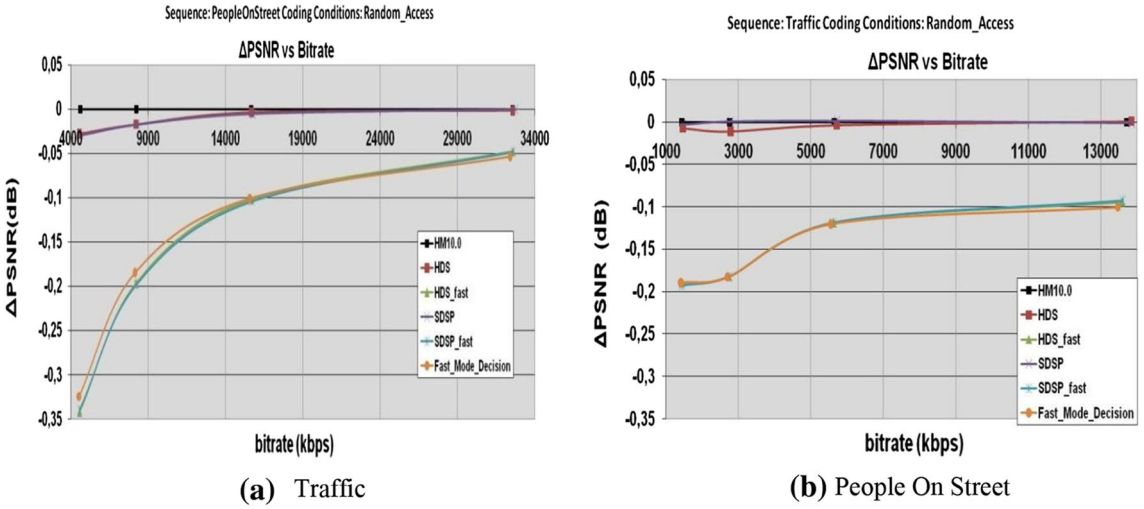
Table 4 Results of the fast decision algorithm (ECU, ESD, CBF) versus the original algorithm

	Sequences	Fast algorithm (ECU, ESD, CBF)		
		ΔBR (%)	ΔPSNR (dB)	Δ*T* (%)
Class A 2,560 × 1,600	Traffic	−1.82	−0.15	−60.30
	PeopleOnStreet	−0.74	−0.17	−28.00
Average Class A		−1.28	−0.16	−44.15
Class B 1,920 × 1,080	Kimono	−0.53	−0.08	−36.98
	ParkScene	−1.29	−0.12	−56.08
	Cactus	−0.98	−0.08	−49.07
	BasketballDrive	−0.56	−0.05	−40.27
	BQTerrace	−1.46	−0.07	−54.67
Average Class B		−0.96	−0.08	−10.19
Class C 832 × 480	BasketballDrill	−0.94	−0.11	−40.28
	BQMall	−0.82	−0.18	−41.40
	PartyScene	−0.78	−0.13	−36.22
	RaceHorses	−0.5	−0.12	−24.02
Average Class C		−0.76	−0.14	−35.48
Class D 416 × 240	BasketballPass	−0.86	−0.16	−44.43
	BQSquare	−1.46	−0.15	−52.84
	BlowingBubbles	−0.96	−0.12	−37.26
	RaceHorses	−0.58	−0.16	−24.28
Average Class D		−0.97	−0.15	−39.70
Class E 1,280 × 720	Vidyo1	−1.75	−0.09	−77.62
	Vidyo3	−1.51	−0.1	−74.02
	Vidyo4	−1.29	−0.07	−70.02
Average Class E		−1.52	−0.08	−73.89
Average		−1.10	−0.12	−40.68

activities is not important contrarily to high-activity sequences, such as PeopleOnStreet and BasketballDrive. The motion is imperceptible and the effectiveness of the algorithm is visible with 11 and 13 % reduced at encoding time. BasketballDrive is a video sequence taken during a basketball game. This sequence contains pictures of high motion activities and high contrast. The background (floor and wall) has rather similar texture. The basketball players

Table 5 Results of the combined algorithms SDSP fast and HDS fast versus the original algorithm

	Sequences	SDSP fast			HDS fast		
		ΔBR (%)	ΔPSNR (dB)	Δ*T* (%)	ΔBR (%)	ΔPSNR Q (dB)	Δ*T* (%)
Class A 2,560 × 1,600	Traffic	−0.64	−0.15	−63.4	−0.77	−0.15	−63.4
	PeopleOnStreet	0	−0.17	−38.2	−0.09	−0.17	−37.6
Average Class A		−0.32	−0.16	−50.8	−0.43	−0.16	−50.5
Class B 1,920 × 1,080	Kimono	−0.41	−0.08	−47.7	−0.29	−0.08	−47.9
	ParkScene	−0.88	−0.11	−58.3	−0.98	−0.11	−58.4
	Cactus	−0.7	−0.08	−54.8	−0.71	−0.08	−55
	BasketballDrive	0.3	−0.05	−52.4	0.43	−0.05	−52.6
	BQTerrace	−1.45	−0.07	−58.9	−1.47	−0.07	−59.1
Average Class B		−0.628	−0.078	−54.42	−0.6	−0.08	−54.6
Class C 832 × 480	BasketballDrill	−0.29	−0.11	−46.3	−0.11	−0.11	−46.6
	BQMall	−0.11	−0.17	−46.8	−0.17	−0.17	−46.9
	PartyScene	−0.66	−0.13	−41	−0.6	−0.13	−41
	RaceHorses	1.03	−0.13	−34.6	0.91	−0.13	−34.6
Average Class C		−0.0075	−0.135	−42.175	0.0075	−0.14	−42.28
Class D 416 × 240	BasketballPass	−0.67	−0.17	−49.8	−0.59	−0.15	−49.7
	BQSquare	−1.37	−0.14	−53.2	−1.38	−0.14	−53.1
	BlowingBubbles	−0.75	−0.12	−42.1	−0.72	−0.11	−41.1
	RaceHorses	0.96	−0.18	−31.2	0.8	−0.17	−31
Average Class D		−0.4575	−0.1525	−44.075	−0.47	−0.14	−43.72
Class E 1,280 × 720	Vidyo1	−1.79	−0.08	−78.4	−1.66	−0.08	−78.5
	Vidyo3	−1.37	−0.1	−74.9	−1.47	−0.1	−75.1
	Vidyo4	−0.92	−0.06	−72.3	−1.06	−0.06	−72.4
Average Class E		−1.36	−0.08	−75.2	−1.4	−0.08	−75.33
Average		−0.55	−0.12	−53.33	−0.58	−0.12	−53.29



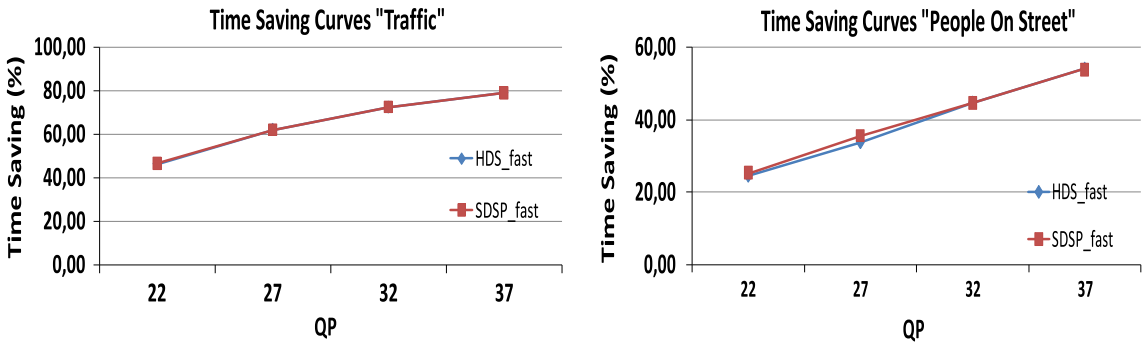
(a) Traffic **(b)** People On Street

Fig. 16 RD curves for sequences (2,560 × 1,600) coded in random access with QP = (22, 27, 32, 37)

are moving in random directions. The proposed algorithm takes advantage of these types of video and shows important values of time saving. Concerning evaluating quality and bit-rate, they are proportional, i.e., a degradation of quality leads to an increase in bit-rate.

Concerning HDS, this algorithm reduces the time execution by an average of 5.7 %. Results show that the majority of videos have maintained the same quality. For bit-rate, the worst cases are for RaceHorses video. In fact, RaceHorses is a sequence that records horse racing. It is a dynamic and motion-filled video. In this video, there are a lot of high-frequency details. Horse tail is usually costly to encode.

Table 4 evaluates the performance of implementing the algorithm for fast partitioning that combines the three approaches of fast coding ECU, ESD, and CBF. As it is



(a) Traffic **(b)** People On Street

Fig. 17 Time saving curves for sequences (2,560 × 1,600) coded in random access with QP = (22, 27, 32, 37)

Table 6 Bjontegaard results of the proposed SDSP fast and HDS fast versus the original HM10.0

	Sequences	SDSP_Fast vs original		HDS_fast vs original	
		BDPSNR (dB)	BDBR (%)	BDPSNR (dB)	BDBR (%)
Class A 2,560 × 1,600	Traffic	−0.13	3.77	−0.12	3.63
	PeopleOnStreet	−0.15	3.36	−0.14	3.28
Average Class A		−0.14	3.57	−0.13	3.45
Class B 1,920 × 1,080	Kimono	−0.09	2.69	−0.08	2.53
	ParkScene	−0.08	2.45	−0.07	2.30
	Cactus	−0.06	2.61	−0.06	2.60
	BasketballDrive	−0.05	−2.26	−0.05	−2.18
	BQTerrace	−0.04	2.02	−0.04	2.06
Average Class B		−0.06	1.50	−0.06	1.46
Class C 832 × 480	BasketballDrill	−0.12	2.85	−0.11	2.68
	BQMall	−0.15	3.63	−0.14	3.51
	PartyScene	−0.09	1.89	−0.09	1.87
	RaceHorses	−0.17	−4.30	−0.16	−4.14
Average Class C		−0.13	1.01	−0.12	0.98
Class D 416 × 240	BasketballPass	−0.12	2.49	−0.11	2.33
	BQSquare	−0.07	1.54	−0.07	1.55
	BlowingBubbles	−0.09	2.20	−0.09	2.17
	RaceHorses	−0.23	−4.85	−0.21	−4.53
Average Class D		−0.13	0.34	−0.12	0.38
Class E	Vidyo1	−0.04	1.18	−0.04	1.19
	Vidyo3	−0.05	1.52	−0.05	1.54
	Vidyo4	−0.04	1.34	−0.03	1.17
Average Class E		−0.04	1.35	−0.04	1.30
Average		−0.10	2.46	−0.10	2.37

shown, results concerning time saving are important and reaching up to 70 % for some videos. In average, not only the encoding time was reduced by 40 %, but also the bit-rate that had decreased by 1 %. Concerning the quality, the degradation varies from 0.07 to 0.16 dB. The results given by the two tables summarize the performance of each algorithm alone.

Table 5 shows the results of combining the two proposed approaches: fast motion estimation algorithm with fast mode decision algorithm. Two algorithms evaluated are HDS fast and SDSP fast. Difference is observed only for motion estimation part. In fact, results are as expected with an important time saving for Vidyo1, Traffic, and BQSquare sequences varying from 53 to 78 %. This saving is justified by the slowness of the motion in these videos. As for resulted sequences quality, we anticipate a slight degradation when implementing these algorithms since some modes will be omitted. However, bit-rate was reduced by 1 % for some videos except for some sequences where we noticed a slight increase in terms of bit-rate for example, BasketBallDrive and RaceHorses videos which are as indicated previously motion-filled videos and containing a lot of texture. We can remark also from analyzing different results that the obtained gain of our final algorithm is equal to the sum of each of the two algorithms that constitute our proposal (fast mode decision algorithm and fast motion estimation). In fact, for People On Street sequence, we have $\Delta T_{\text{Fast\_SDSP}}$ (%) $\approx \Delta T_{\text{Fast}}$ (%) + ΔT_{SDSP} (%).

Figure 16 evaluates the degradation of quality. The curves were done for videos of class A having the highest resolution. The four points represented are calculated for 4 QPs: 22, 27, 22, and 37. When evaluating HDS and SDPS algorithms, the degradation is not visible. For QP equal to 32 and 37, ΔPSNR is null. This means that there is no considerable degradation in quality. Concerning the combined algorithms, the PSNR degradation varies from −0.35 to −0.05. Figure 16 shows that for lower values of QPs, the degradation is important. When QP is important (32 or 37), the curves will be close to each other.

Figure 17 evaluates the time saving while varying QP. The time saving increases proportionally while we increase the QP value. For QP = 37, the encoding time is reduced by 50 % for traffic video and 60 % for PeopleOnStreet video. The speedup is more important for higher QPs due to the fact that skip mode is more likely to be chosen for important values of QP [20].

The comparison was done also using the Bjontegaard criteria which evaluate numerical averages between different RD curves [31]. This criterion is used for evaluating

Table 7 Time distribution for the different algorithms

	Original algorithm (%)	HDS_fast (%)	SDPS_fast (%)
Inter-prediction	40.20	37.80	38.00
RdCost	33.10	30.60	30.70
Transf + Quant	9.30	14.80	14.30
memcpy_memset	4.20	3.50	3.70
Entropy coding	2.60	2.90	2.80
Others	10.60	10.40	10.50

Table 8 Summary of performances of comparing proposed algorithms to the original algorithm

	Liquan [11]			Qin [12]			Park [13]		
	ΔBR (%)	ΔPSNR (dB)	Speedup (%)	ΔBR (%)	ΔPSNR (dB)	Speedup (%)	ΔBR (%)	ΔPSNR (dB)	Speedup (%)
Class A	–	–	–	–	0.1 %	−22.4	0.36	−0.06	−47.74
Class B	0.834	−0.02	−34.8	–	0.3 %	−28.4	0.31	−0.06	−45.60
Class C	1.225	−0.045	−24.5	–	0.2 %	−23.0	0.61	−0.06	−41.74
Class D	1.06	−0.04	−16.25	–	0.2 %	−17.0	0.95	−0.05	−39.61
Class E	1.06	−0.03	−41.33	–	–	–	0.55	−0.05	−43.67
Average	1.05	−0.03	−29.22	–	0.2	−24	0.36	−0.06	−47.74
	Proposed fast algorithm			Proposed algo1 (based on SDSP)			Proposed algo2 (based on HDS)		
	ΔBR (%)	ΔPSNR (dB)	Speedup (%)	ΔBR (%)	ΔPSNR (dB)	Speedup (%)	ΔBR (%)	ΔPSNR (dB)	Speedup (%)
Class A	−1.28	−0.16	−44.15	−0.32	−0.16	−50.80	−0.43	−0.16	−50.5
Class B	−0.96	−0.08	−10.19	−0.63	−0.08	−54.42	−0.6	−0.08	−54.6
Class C	−0.76	−0.135	−35.48	−0.007	−0.14	−42.18	0.007	−0.14	−42.28
Class D	−0.97	−0.148	−39.7	−0.46	−0.15	−44.08	−0.47	−0.14	−43.72
Class E	−1.52	−0.08	−73.89	−1.36	−0.08	−75.20	−1.4	−0.08	−75.33
Average	−1.10	−0.12	−40.68	−0.55	−0.12	−53.33	−0.58	−0.12	−53.29

the proposed algorithm with precision. Table 6 summarizes the results for Bjontegaard average PSNR (BDPSNR) and Bjontegaard average bit-rate (BDBR). The proposed algorithms show a bit-rate saving of 2 % in BDBR and a loss of 0, 1 dB in quality. The best case is for videos with low motion estimation like Vidyo1 and Traffic. While implementing algorithms of fast decision and early termination, it leads to a little decrease in BDBR reaching 3 %. However, for high-activity videos like BasketBallDrive and Racehorses, the loss is relatively important with 2 and 4 % for BDBR.

After implementing these algorithms, the new software is profiled to evaluate the impact of our work. Results are shown in Table 7. The effect of our contribution was clear, especially in the Inter-prediction module and the RD cost module. This improvement was accompanied with a small increase in the transformation module.

Table 8 summarizes the performances of different algorithms. The proposal of Liquan [11] was based on skipping some specific depths used in the analysis of previous frames. This algorithm provided a time saving of 29 % in average, while having a negligible loss in quality and an increase of 1 % in bit-rate. The algorithm implemented by Qin [12] was based on the early termination according to an adaptive threshold value of MSE. This algorithm assures a time saving while maintaining the quality. Another interesting proposal algorithm was presented by Park [13]. His idea was based on detecting the merge skip mode when analyzing the mode information of neighboring CUs. This idea was able to save 47 % of the encoding time while having a little decrease in quality and a small increase in bit-rate. The proposed algorithms were more efficient when having more important time saving, in addition to the decrease in bit-rate.

7 Conclusion and perspectives

HEVC is a new standard that provides a significant amount of increased coding efficiency compared to previous standards. To satisfy a higher coding efficiency, a robust algorithm for the mode decision process is required. Reducing the number of search points in motion estimation helps in further time saving while reducing the RD cost computation. This paper presented two fast algorithms based on optimizing the motion estimation process and using also a fast mode decision algorithm for CU partitioning. Combining both propositions will enable us to achieve an important improvement in time saving reaching up to 78 % and averaging 52 % with a negligible degradation in video quality and decreasing the bit-rate.

As perspectives, we intend to reduce more and more the mode decision partitioning time consuming by setting empirical threshold values for the stop criteria based on QP values to implement the optimized reference on a digital platform adequate to video processing.

References

1. Richardson, I.: HEVC an introduction to high efficiency video coding. In: VCodexVideo Compression (2013)
2. Tai, S., Chang, C., Chen, B., Hu, J.: Speeding up the decisions of quad-tree structures and coding modes for HEVC coding units. In: Advances in Intelligent Systems and Applications (SIST 21), pp. 393–401 (2013)
3. Tiancai, Y., Dongming, Z., Feng, D., Yongdong, Z.: Fast mode decision algorithm for intra prediction in HEVC. In: Internet Multimedia Computing and Service (ICIMCS), China, pp. 300–304 (2013)
4. Bross, B., JCTVC-L1003_v9 :High efficiency video coding (HEVC) text specification draft 10. In: Proceedings of the 12th JCT-VC Meeting, Geneva (2013)
5. Jarno, V., Marko, V., Timo, D.H., Antti, H.: Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs. IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1885–1898 (2012)
6. Jens-Rainer, O., Gary, J.S.: High efficiency video coding: the next frontier in video compression. In: IEEE Signal Processing Magazine, pp. 152–158 (2013)
7. RyeongHee, G., Yung-Lyul, L.: N-Level quantization in HEVC: broadband multimedia systems and broadcasting (BMSB). In: IEEE International Symposium (2012)
8. Bossen, F., JCT-VC-L1100: Common test conditions and software reference configurations. In: Proceedings of the 12th JCT-VC Meeting, Geneva (2013)
9. Gary, J.S., Woo-Jin, H., Thomas, W.: Overview of the high efficiency video coding (HEVC) standard. In: Circuits and Systems for Video Technology (2012)
10. Xiaolin, S., Lu, Y., Jie C.:Fast coding unit size selection for HEVC based on Bayesian decision rule. In: 2012 Picture Coding Symposium, Poland (2012)
11. Liquan, S., Zhi, L., Xinpeng, Z., Wenqiang, Z., Zhaoyang, Z.: Correspondence:an effective CU size decision method for HEVC encoders. IEEE Trans. Multimed. **15**(2), 465–470 (2013)
12. Qin, Y., Xinfeng, Z., Siwei, M.: Early termination of coding unit splitting for HEVC. In: IEEE Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Asia-Pacific (2012)
13. Park, C., Kim, B., Hong, G., Kim, S., Park, C.: Fast coding unit (CU) depth decision algorithm for high efficiency video coding (HEVC). In: Advanced in Computer Science and its Applications, vol. 279, pp. 293–299 (2014)
14. Purnachand, N., Alves L.N., Navarro, A.: Fast motion estimation algorithm for HEVC. In: IEEE Second International Conference on Consumer Electronics, Berlin (2012)
15. Purnachand, N., Alves L.N., Navarro, A.: Improvements to Tz search motion estimation algorithm For multiview video coding. In: International Conference on Systems, Signals and Image Processing (IWSSIP), Austria, pp. 388–301 (2012)
16. Hyang-Mi, Y., Jae-Won, S.: Fast coding unit decision algorithm based on inter and intra prediction unit termination for HEVC. In: IEEE International Conference on Consumer Electronics (ICCE), pp. 300–301 (2013)
17. Il-Koo, K., Junghye, M., Tammy, L., Woo-Jin, H., and Jeong Hoon P.: Block partitioning structure in the HEVC standard. In: Circuits and Systems for Video Technology, vol. 22, No. 12, pp. 1697–1706 (2012)

18. Detlev, M., Senior, M., Heiko, S., Sebastian, B., Benjamin, B., Philipp, H., Tobias, H., Heiner, K., Haricharan, L., Tung, N., Simon, O., Mischa, S., Karsten, S., Martin, W., Thomas W.: Video compression using nested quad-tree structures, leaf merging, and improved techniques for motion representation and entropy coding. In: IEEE Transactions on Circuits and Systems for Video Technology, vol. 20, No. 12, pp. 1676–1687 (2010)
19. Kiho, C., Sang-Hyo, P., Euee S.J.: Coding tree pruning based CU early termination. In: Joint Collaborative Team on Video Coding (JCT-VC). 6th Meeting, Torino (2011)
20. Frank, B., Benjamin, B., Karsten, S., David, F.: HEVC complexity and implementation analysis. In: Circuits and Systems for Video Technology (2012)
21. Yang, J., JCTVC-G543: Early skip detection for HEVC. In: Proceedings of JCT-VC 7th Meeting, Geneva (2011)
22. Bin, L., Jizheng, X., JCTVC-C277: Redundancy reduction in Cbf and merging coding. In: Proceedings of JCT-VC 3rd Meeting, Guangzhou (2010)
23. Jian-Liang, L., Yi-Wen, C., Yu-Wen, H., Shaw-Min, L.: Motion vector coding in the HEVC standard. In: Journal of Selected Topics in Signal Processing IEEE (2013)
24. Felipe, S., Sergio, B., Mateus, G., Luciano, A., Julio, M.: Motion vectors merging: low complexity prediction unit decision heuristic for the inter-prediction of HEVC encoders. In: IEEE International Conference on Multimedia and Expo, pp. 657–662 (2012)
25. Bosse, F., Flynn, D., Sühring, K.: JCTVC-software manual. HM 10.0 Software Manual (2013)
26. Woong, ll C., Byeungwoo, J., Jechang, J.: Fast motion estimation with modified diamond search for variable motion block sizes. In: International Conference on Image Processing, pp. 371–374 (2003)
27. Samet, A., Souissi, N., Zouch, W., Ben Ayed, M.A., Masmoudi, N.: New horizontal diamond search motion estimation algorithm for H.264/AVC. In: Second Symposium on Communication, Control and Signal Processing (ISCCSP), Morocco (2006)
28. Chan, E., Arturo A., Rodriguez, Ghandi, G., Panchanathan, S.: Experiments on block-matching techniques for video coding multimedia systems, vol. 2, No. 5, pp. 228–241 (1994)
29. I1-Koo, K.: High efficiency video coding (HEVC) test model 10 (HM10) encoder description. In: Proceedings of the 12th JCT-VC Meeting, Geneva (2013)
30. Díaz-Honrubia, A., Martínez, J., Cuenca, P.: HEVC a review, trends and challenges. In: 2nd Workshop on multimedia data coding and transmission (2011)
31. Bjontegaard, G.: Calculation of average PSNR differences between RD-curves. In: Doc. VCEG-M33, Austin, TX (2001)

Fatma Belghith was born in Sfax, Tunisia, in 1988. She received her degree in Electrical Engineering from the National School of Engineering (ENIS), Sfax, Tunisia, in 2012. She is working toward her Ph.D. in Electronic Engineering at the Laboratory of Electronics and Information Technology (LETI)-ENIS, University of Sfax. Her current research interests include video coding with emphasis on HEVC standard, hardware implementation using FPGA and embedded systems technology.

Hassan Kibeya was born in 1989. He received his degree in Electrical Engineering from the National School of Engineering (ENIS), Sfax, Tunisia, in 2013. He is currently a PhD student in the Laboratory of Electronics and Information Technology. His main research activities are focused on video signal processing and HEVC standard.

Hassen Loukil received electrical engineering degree from the National School of Engineering-Sfax (ENIS) in 2004. He received his MS and PhD degrees in electronics engineering from Sfax National School of Engineering in 2005 and 2011, respectively. He is currently an assistant professor at Higher Institute of Electronic and Communication of Sfax (Tunisia). He is teaching Embedded System conception and System on Chip. He is currently a researcher in the Laboratory of Electronics and Information Technology and an assistant at the University of Sfax, Tunisia. His main research activities are focused on image and video signal processing, hardware implementation and embedded systems.

Mohamed Ali Ben Ayed was born in Sfax, Tunisia, in 1966. He received his BS degree in Computer Engineering from Oregon State University and MS degree in Electrical Engineering from Georgia Institute of Technology in 1988, his DEA, Ph.D., and HDR degrees in Electronics Engineering from Sfax National School of Engineering in 1998, 2004, and 2008, respectively. He is currently a Maître de Conferences in the Department of Communication at Sfax High Institute of Electronics and Communication. He was a co-founder of ''Ubvideo Tunisia'' in the techno-pole El- GHAZLA Tunis, an international leader company in the domain of video coding technology. He is a member of a research team since 1994 at (LETI, Sfax) in the domain of Electronics and Information Technology and a reviewer in many international and national journals and conferences. He is currently a technical advisor of ''EBREASK video'', a Research and Development company specialized on the next high-efficient video coding generation H265. His current research interests include DSP and VHDL implementation of digital algorithms for multimedia services, and development of digital video compression algorithms.

Nouri Masmoudi received his electrical engineering degree from the Faculty of Sciences and Techniques—Sfax, Tunisia, in 1982, the DEA degree from the National Institute of Applied Sciences—Lyon and University Claude Bernard—Lyon, France in 1984. From 1986 to 1990. He received his Ph.D. degree from the National School Engineering of Tunis (ENIT), Tunisia in 1990. He is currently a professor at the electrical engineering department, ENIS. Since 2000, he has been a group leader 'Circuits and Systems' in the Laboratory of Electronics and Information Technology. Since 2003, he is responsible for the Electronic Master Program at ENIS. His research activities have been devoted to several topics: Design, Telecommunication, Embedded Systems, Information Technology, Video Coding and Image Processing.