Fast Motion Estimation Algorithm for HEVC Video Encoder

Purnachand Nalluri[1,2], Luis Nero Alves[1,2], Antonio Navarro[1,2]
nalluri@av.it.pt, nero@av.it.pt, navarro@av.it.pt

[1]Institudo de Telecomunicações,
Pólo-Aveiro, Campus Universitário de Santiago,
3810-193 Aveiro, Portugal

[2]Departamento de Electrónica, Telecomunicações e
Informática, Universidade de Aveiro
Campus Universitário de Santiago
3810-193 Aveiro, Portugal

***Abstract*—Video compression is required in applications like video network communications, video conference, broadcasting, live streaming and video storage. H.265/HEVC is the latest video compression standard, jointly developed by JCT-VC that provides the highest compression efficiency without significant loss in original video source quality. Among all the tools in HEVC encoder, Motion Estimation (ME) is one of the most complex tasks. The present paper analyses the ME algorithm present in HEVC standard reference software and proposes two improvements to the algorithm. Our results show a decrease on the computational complexity by almost 30% with negligible loss in the video quality.**

***Keywords*—Video Compression; HEVC; Motion Estimation**

I. Introduction

HEVC is the latest video coding standard currently under joint development by ISO/IEC MPEG and ITU-T VCEG. The MPEG and VCEG have established JCT-VC (Joint Collaborative Team on Video Coding) [1]. The main goal of HEVC is to increase compression performance compared to existing standard H.264/AVC in a range of 50% reduction in bitrate without affecting output video quality [2]. For doing this, many new coding tools were implemented and upgraded from H.264/AVC standard. One of them is the quad-tree based coding structure. The HEVC coding structure is more generalized into quad-tree based coding units (CUs), as shown in Fig.1 (a). Each CU is recursively sub-divided into quad-tree based prediction units (PUs), of either intra-type or inter-type or skip type or merge type. Each PU is further sub-divided into quad-tree based transform units (TUs).

For compressing a video, the encoder typically exploits spatial redundancy (inside a frame) and temporal redundancy (between frames). To exploit temporal redundancy, the video encoder uses predictive encoding. In this process, each block in a video frame searches for best matched block in past/future frame ROI (Region of Interest, technically termed as search window), and only the motion vector (which denotes the shift of entire block in past/future frame) is encoded instead of encoding the entire block. This process is called Motion Estimation (ME) and reduces the bitrate of entire video to a huge extent, at the cost of huge increase in computational complexity. So, reducing the computational complexity of ME

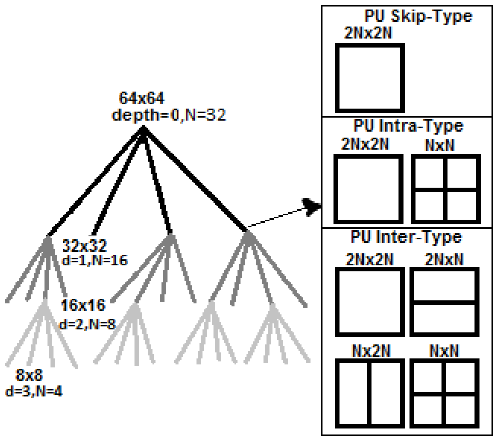


Fig.1(a) HEVC Coding Structure

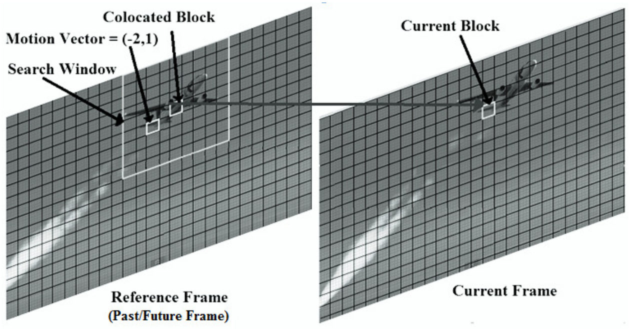


Fig.1(b) Illustration of ME Process

process, without affecting rate-distortion performance is a challenging task. The process of ME is shown in Fig.1 (b).

As shown in Fig.1(b), the task of ME is to search best matched block of current frame in the search window. The best matched block is the block which has minimum cost value, defined in Eq.1.

$$J_{MV} = D + \lambda_{Motion} . R(MV - PMV) \tag{1}$$

where J is the lagrangian cost, D is the distortion between current and reference block, λ denotes lagrangian multiplier, PMV denotes the predicted motion vector, MV denotes the estimated motion vector and R represents bitrate or bits required to encode the motion vector difference "PMV-MV".

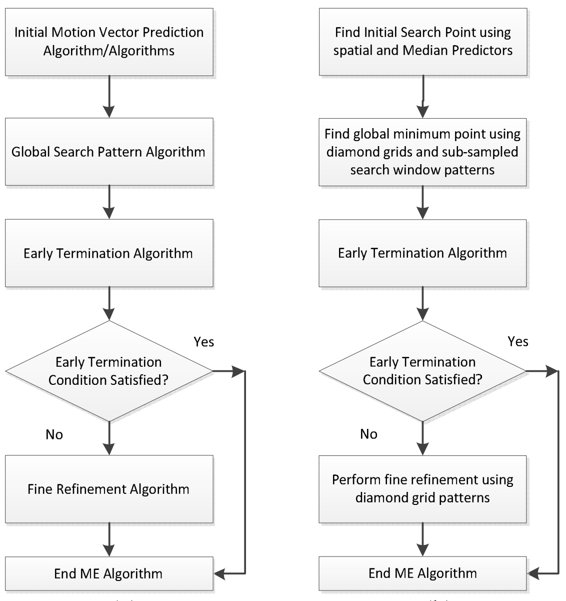

Initial Motion Vector Prediction Algorithm/Algorithms
Global Search Pattern Algorithm
Early Termination Algorithm
Early Termination Condition Satisfied?
Yes
No
Fine Refinement Algorithm
End ME Algorithm
Find Initial Search Point using spatial and Median Predictors
Find global minimum point using diamond grids and sub-sampled search window patterns
Early Termination Algorithm
Early Termination Condition Satisfied?
Yes
No
Perform fine refinement using diamond grid patterns
End ME Algorithm
(a)
(b)

Fig.2.Illustration of ME Tools in (a) Typical Fast ME Algorithm (b) TZS ME Algorithm

The distortion is measured by using the matching criteria like SAD (Sum of Absolute Difference) or SSD (Sum of Squared Difference) or MSE (Mean Square Error). The most commonly used and simplest matching function is SAD defined in Eq.2.

$$SAD(x,y)=\sum_{i=1}^{M}\sum_{j=1}^{N}\left|C(x,y)-R(x+i,y+j)\right| \qquad (2)$$

where C represents the current block and R denotes reference block and MxN is the size of the current block.

If the ME algorithm searches every block in search window, then it is called full search ME algorithm. If the algorithm, skips some of the blocks, that are less likely to be the best matched block, then it is classified as fast search ME algorithm. The present paper provides an improvement to existing fast ME algorithm present in HEVC encoder of reference software HM [3]. Section II provides detailed explanation of ME algorithm and its tasks. Section III explains the proposed improvements. Section IV shows the simulation results and finally section V summarizes with concluding remarks.

II. Motion Estimation Tools and Algorithms

There are numerously many fast ME algorithms, that were developed for fast video encoding. The MPEG based standards including MPEG-2, H.264/AVC and HEVC use hybrid fast ME algorithms to estimate the best Motion Vector (MV). All these fast ME algorithms have four common tools to find the minimum cost block, as explained in the following section.

A. Fast ME Coding Tools

The most common tools for fast ME algorithms are 1) Initial Search point Prediction (ISP) Algorithm 2) Global Search Pattern (GSP) 3) Early Termination (ET) Algorithm 4) Fine Refinement Algorithm. The ISP algorithm aims at predicting the starting point of the ME process, by using the MVs of previously coded neighboring blocks. The GSP algorithm searches a global estimate of the motion vector by using grid patterns and reduces the possibility of the MV in getting trapped to local minima. The ET algorithm terminates the ME algorithm to save the computation time, if the current cost of MV falls below a predefined threshold (fixed or adaptive threshold). Finally, if the ET condition is not satisfied, the fine refinement algorithm converges to local minima, using fixed search patterns. The coding tools for a typical fast ME algorithm is shown in Fig.2 (a).

B. Fast ME Algorithms

The H.264/AVC introduces 3 fast ME algorithms - EPZS (Enhanced Predictive Zonal Search), UMHexS (Unsymmetrical-cross Multi-Hexagonal grid Search), and SUMHexS (Simple UMHexS) algorithms [4-5]. The EPZS algorithm uses four prediction algorithms for ISP stage and finds the best starting point. It does not use any global search pattern. The UMHexS and SUMHexS algorithms use basic prediction algorithms for ISP stage but focus more on global search patterns. They use two patterns – unsymmetrical cross pattern and multi-hexagonal grid patterns for finding the global minima. The HEVC uses TZS (Test Zone Search) algorithm for ME [6-7]. It has two global search patterns – exponential diamond grid patterns and search window sub-sampled pattern. The detailed flow of TZS ME algorithm is shown in Fig.2 (b).

C. The TZS ME Algorithm

As shown in flowchart of Fig.2 (b), in the initial step, the TZS ME algorithm uses spatial up, left, up-right, median predictors to predict initial search point. After ISP stage, the TZS algorithm uses diamond grid pattern and sub-sampled search window pattern to find the global minimal cost search point, as shown in Fig. 3(a), and Fig.4(a). The grid pattern, shown in Fig.3 (a) is an exponentially increasing grid pattern. The initial diamond close to starting search point (center point) has a stride-length (distance between center point and vertices of pattern) of 1. The stride-length continues to increase from starting point by a factor of 2 until the search range, i.e. 2,4,8,16,32 and so on. The sub-sampled search window samples the entire search window uniformly, and finds the minimum cost MV. Then the minimum cost point in both the search patterns is taken as the global minimal point. In the fine refinement stage, the TZS algorithm performs diamond grid pattern search with the new minimal point as search center. The refinement process continues, until the new search is the center point of the diamond grid pattern.

III. Proposed Improvements to HEVC Fast ME Algorithm

The present paper proposes two improvements to the TZS fast ME algorithm present in the HEVC reference software. The first improvement is at global search pattern stage, and the second improvement is at fine refinement stage. Each of these improvements is explained in the following sub-sections.

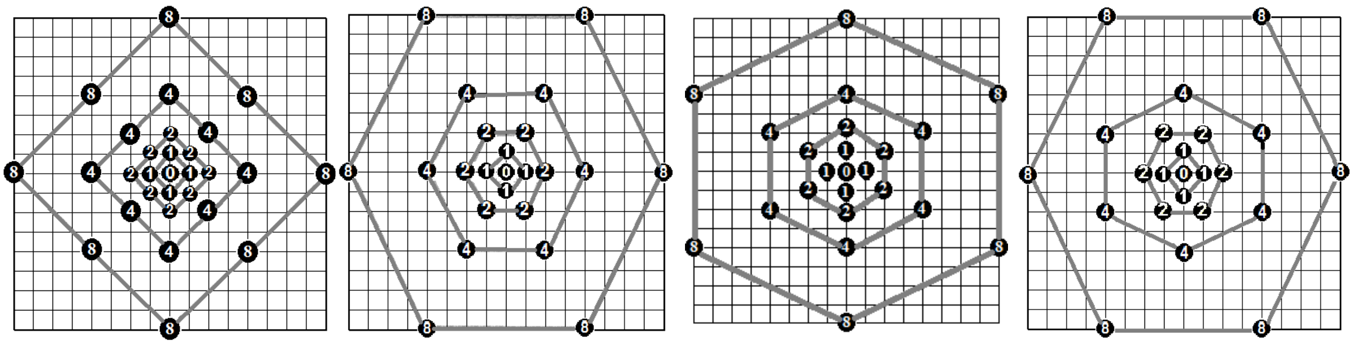
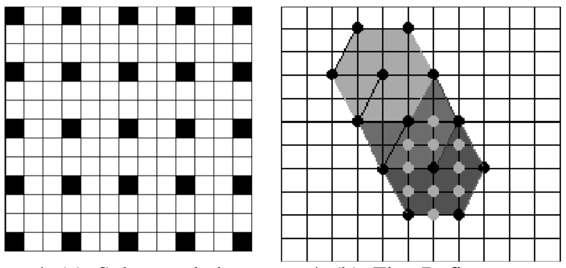

(a) Diamond Pattern (b) Horizontal Hexagonal Pattern (c) Vertical Hexagonal Pattern (d) Rotating Hexagon Pattern

Fig.3. Grid Search Patterns for Motion Estimation with Stride Length 8.


4. (a). Sub-sampled Search Window 4. (b). Fine Refinement Algorithm

A. Rotating Hexagon Pattern For Global Search

As seen in Fig.3 (a), the diamond grid pattern has 8 search points per each grid. If we replace the diamond pattern with hexagonal pattern as shown in Fig 3 (b), then each grid has only 6 points. For a search range of 64, there will be 6 grids in diamond or hexagonal patterns, and hence there will be 53 points (8x6+4+1) for diamond and 41 points (6x6+4+1) for hexagonal pattern. The number of search points for diamond (N_D) and hexagonal (N_H) pattern, for a given search range 'R' are shown in Eq.3 and Eq.4. Hence, there will be

$$N_D = 5 + 8\log_2 R \quad (3)$$
$$N_H = 5 + 6\log_2 R \quad (4)$$

computational complexity reduction of around 20% for each current block. However, the horizontal hexagon shown in Fig.3(b) is the most appropriate for estimating horizontal motion and less appropriate for estimating horizontal motion since it has more horizontal search points than vertical points, and for estimating vertical motion the horizontal hexagons performs more loops in fine refinement stage. On the other hand, the vertical hexagons shown in Fig.3(c) are good for vertical motion and lose performance for estimating horizontal motion. Hence the present paper proposes a rotating hexagonal pattern as shown in Fig.3 (d) for balancing the motion estimation in both horizontal and vertical directions.

B. Hexagonal Based Fine Refinement

The TZS ME algorithm uses the same diamond grid pattern for performing the fine refinement of MV. Once the global minimum point is estimated, it is most likely that the optimal point lies in the vicinity of the global MV and the distortion function in the local refinement stage is a monotonically decreasing function. Hence a gradient descent based algorithm using hexagonal pattern is implemented for refining the MV. The algorithmic steps are shown in Fig.4 (b). In the first step, the refinement algorithm forms a small hexagon with 6 search points, around the global minimum point. The minimal point in the hexagon is taken as the new center point to form a new hexagon. The search continues until the minimum point in the hexagon is the center point. Then the algorithm checks for the remaining unchecked 10 points around the center point, as shown in Fig.4 (b).

IV. SIMULATION RESULTS

The simulations have been carried out on HEVC reference software HM 9.0 [3]. Table 1 shows the comparison results for ME time and number of ME search points. TZS_D and TZS_{RH} indicate the results for TZS algorithm with diamond and rotating hexagonal pattern. TZS_{RHFR} denotes the results for rotating hexagonal pattern with hexagonal fine refinement. Fig.4 shows the RD (Rate Distortion - bitrate vs. PSNR) curves for all the three patterns TZS_D, TZS_{RH}, TZS_{RHFR} with QPs (Quantization Parameters) 22, 27, 32 and 37. The PSNR represents the video quality and is measured using the Eq.5.

$$PSNR(dB) = 10.\log_{10}\left(\frac{255^2}{MSE}\right) \quad (5)$$

where MSE represents mean square error and 255 is the maximum value of a 8-bit pixel (luminance component).

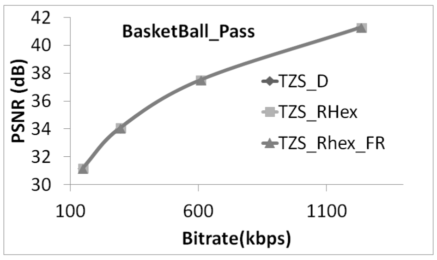
The simulation results show that the rotating hexagonal pattern gains 15% of ME speed or 16% reduction of total number of search points, compared to diamond pattern. Further, modification of fine refinement stage gains 30% of ME complexity which means that the total number of search points is reduced by 30%. The RD curves in Fig.4 and the Bjontegaard-delta [8] results in Table 2 also show that there is negligible loss in rate-distortion performance.

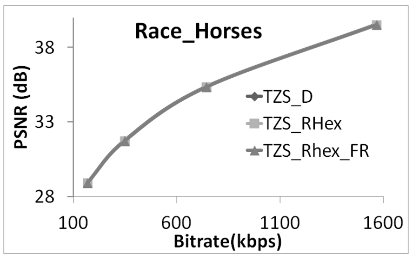
V. CONCLUSION

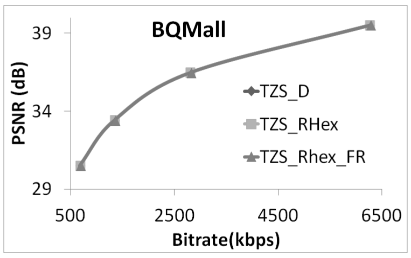
The fast ME algorithm present in HEVC reference software is analyzed and improved. Simulation results show that computational complexity was reduced by 30% without affecting rate-distortion performance. The algorithm can be further improved by adding more initial search point prediction algorithms.

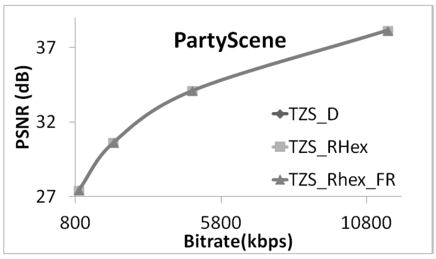
Table 1. ME Time Comparison Results between Diamond and Rotating Hexagon Patterns

Sequence	QP	ME Time (sec)			Δ ME Time (%)		No. of Search Points 'N' (x 10^9)			ΔN (%)	
		TZS_D	TZS_{RH}	TZS_{RHFR}	TZS_D vs. TZS_{RH}	TZS_D vs. TZS_{RHFR}	TZS_D	TZS_{RH}	TZS_{RHFR}	TZS_D vs. TZS_{RH}	TZS_D vs. TZS_{RHFR}
BasketballPass (416x240 @ 50fps)	22	71	58	44	18.31	38.03	0.676	0.549	0.437	18.76	35.31
	27	60	50	41	16.67	31.67	0.558	0.453	0.364	18.76	34.79
	32	50	41	34	18.00	32.00	0.433	0.353	0.289	18.41	33.29
	37	42	34	27	19.05	35.71	0.333	0.272	0.225	18.49	32.40
RaceHorses (416x240 @ 30fps)	22	130	118	92	9.23	29.23	1.297	1.141	0.918	12.02	29.18
	27	119	105	85	11.76	28.57	1.133	0.990	0.795	12.59	29.83
	32	101	91	72	9.90	28.71	0.915	0.784	0.633	14.31	30.82
	37	84	73	59	13.10	29.76	0.688	0.583	0.474	15.28	31.14
BQMall (832x480 @ 60fps)	22	284	246	198	13.38	30.28	2.734	2.370	1.929	13.31	29.44
	27	252	218	176	13.49	30.16	2.315	1.985	1.621	14.25	29.96
	32	217	185	148	14.75	31.80	1.856	1.566	1.295	15.64	30.23
	37	180	154	129	14.44	28.33	1.458	1.227	1.021	15.81	29.95
PartyScene (832x480 @ 50fps)	22	245	210	174	14.29	28.98	2.401	2.058	1.703	14.30	29.09
	27	224	187	157	16.52	29.91	2.093	1.776	1.471	15.14	29.74
	32	195	160	137	17.95	29.74	1.717	1.433	1.197	16.52	30.29
	37	163	133	115	18.40	29.45	1.353	1.114	0.939	17.70	30.62
FourPeople (1280x720 @ 60fps)	22	176	145	120	17.61	31.82	1.642	1.367	1.184	16.75	27.90
	27	161	130	110	19.25	31.68	1.469	1.223	1.079	16.77	26.60
	32	147	120	102	18.37	30.61	1.328	1.095	0.987	17.48	25.67
	37	134	112	97	16.42	27.61	1.209	0.990	0.911	18.14	24.68
Average					15.54	30.70				16.02	30.05









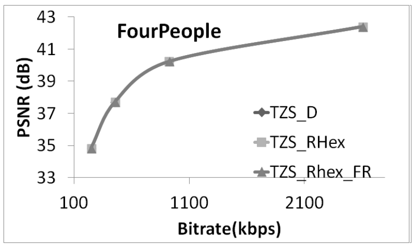


Fig.4. RD Curves for Video Sequences with QP=37,32,27,22.

Table 2. Bjontegaard Delta Results.

Sequence	BD-PSNR-Y (dB)		BD-BitRate (%)	
	TZS_D vs. TZS_{RH}	TZS_D vs. TZS_{RHFR}	TZS_D vs. TZS_{RH}	TZS_D vs. TZS_{RHFR}
BasketballPass	-0.011	-0.014	0.208	0.296
RaceHorses	-0.027	-0.045	0.561	0.933
BQMall	-0.004	-0.045	0.104	0.564
PartyScene	-0.007	-0.011	0.153	0.257
FourPeople	-0.006	-0.011	0.234	0.382
Average	**-0.011**	**-0.025**	**0.252**	**0.487**

ACKNOWLEDGEMENTS

This work is supported by "Fundação para a Ciência e a Tecnologia (FCT)" Portugal grant Ref. SFRH/BD/73266/2010.

REFERENCES

[1] B. Bross, W.-J. Han, G. J. Sullivan, J.-R. Ohm, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 8,"ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC) document JCTVC-J1003, July 2012.

[2] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, Thomas Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard" IEEE Transactions on Circuits and Systems for Video Technology, vol. pre-pub, Issue-99, Dec 2012.

[3] HM Reference Software 9.0 [Online]. Available: https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware.

[4] A.M. Tourapis, H.-Y. Yeon, P. Topiwala, "Fast ME in the JM reference software" ITU-T/ISO/IEC Joint Video Team (JVT) document JVT-P026, July 2005.

[5] X. Xu, Y. He, "Comments on Motion Estimation Algorithms in Current JM Software" ITU-T/ISO/IEC Joint Video Team (JVT) document JVT-Q089, Oct 2005.

[6] N. Purnachand, L. N. Alves, A. Navarro, "Improvements to TZ search motion estimation algorithm for multiview video coding." IEEE - International Conference on Systems, Signals and Image Processing (IWSSIP-2012), pp. 388-391, April 2012.

[7] N. Purnachand, L. N. Alves, A. Navarro, "Fast Motion Estimation Algorithm for HEVC." IEEE International Conference on Consumer Electronics-Berlin (ICCE-Berlin 2012), pp. 34-37, September 2012

[8] Bjontegaard, G, "Calculation of average PSNR difference between RD curves",VCEG-M33, 2001.