BRIEF PAPER

High-quality Hardware Integer Motion Estimation for HEVC/H.265 Encoder

Chuang ZHU[†a)], **Jie LIU**[††], **Xiao Feng HUANG**[†††], ***Nonmembers*, *and*** **Guo Qing XIANG**[††], ***Student Member***

SUMMARY This paper reports a high-quality hardware-friendly integer motion estimation (IME) scheme. According to different characteristics of CTU content, the proposed method adopts different adaptive multi-resolution strategies coupled with accurate full-PU modes IME at the finest level. Besides, by using motion vector derivation, IME for the second reference frame is simplified and hardware resource is saved greatly through processing element (PE) sharing. It is shown that the proposed architecture can support the real-time processing of 4K-UHD @60fps, while the BD-rate is just increased by 0.53%.

key words: *high-quality, integer motion estimation, multi-resolution, PE*

1. Introduction

Video compression plays a vital role in multimedia application areas under limited bandwidth and storage. The High Efficiency Video Coding standard (HEVC), which brings about 50% coding efficiency improvement compared to the previous generation encoder H.264/AVC, is the most recently released video compression technology by the Joint Collaborative Team on Video Coding (JCT-VC) [1]. In HEVC, as shown in Fig. 1, each picture is partitioned into a series of coding tree units (CTUs), and each CTU can be further divided into coding units (CUs). Each CU can be inter-predicted by using different prediction units (PUs), and a PU partitioning structure has its root at the CU level. Perform integer motion estimation (IME) for all these PUs can greatly reduce the temporal redundancies.

However, the high computational complexity of fully executing IME poses a big challenge for video encoders. Pan et al. presented a fast motion estimation method to reduce the encoding complexity of the H.265/HEVC encoder based on the best motion vector selection correlation among the different size prediction modes [2]. Although the proposed algorithm can achieve an average of 20% motion estimation time saving, the computing complexity is still very high for many applications. Hardware acceleration was proved to be a promising solution for IME of H.265/HEVC in the past several years [3], [4]. The authors of work [3] designed and realized an efficient hardware integer motion estimator for an HEVC video encoder based on the full search algorithm. The proposed method can achieve 30 fps for 4K video formats. Nguyen et al. proposed an optimized hardware design of IME for 8K HEVC video encoder [4]. Although the work can achieve 8K video processing in an FPGA, the hardware resource consumption is very huge. Besides, the R/D performance is not available. In work [5], the authors designed a high-throughput motion estimation system for HEVC encoder, which allows the processing of 2160p@30fps at 400MHz. However, the proposed scheme just supports one reference frame and the coding loss is very large. In our previous research, we have designed a multi-resolution motion estimation algorithm (MMEA) to lower the computational complexity for H.264/AVC [6], [7]. Although MMEA works well on the previous H.264 standard, the direct application of it to HEVC is still not enough when facing the tremendous computing complexity. In this paper, we further build a 4K-UHD@60fps real-time IME for HEVC encoder by using a variety of novel strategies coupled with MMEA, which remains high coding efficiency. The proposed scheme supports 2 reference frames and achieves higher coding efficiency when compared to the state-of-the-art.

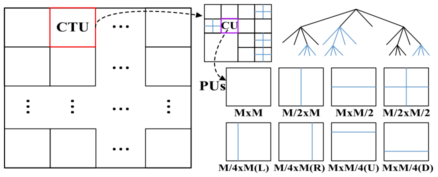


Fig. 1 Subdivision of a CTU into CUs and the PU modes

Outline. The remainder of the paper is structured as follows. In Sect. 2, we present our proposed high-quality hardware-friendly IME, including current CTU content analysis, CU depth decision, IME for two reference frames, and the search window. In Sect. 3, we give our IME architecture and the implementation results are discussed in Sect. 4. Finally, in Sect. 5 we conclude our paper.

2. Proposed High-Quality Hardware-Friendly IME

To decrease the complexity, for the current CTU we first generate the spatial homogeneity (*SH*) and the temporal stationarity (*TS*) characteristics through analyzing the image content. We then conduct CU depth decision (CUDD). After CUDD, the IME for the first reference frame (Ref. 0) is executed and the corresponding integer best motion vectors

Manuscript received August 3, 2018.
Manuscript revised May 25, 2019.
Manuscript publicized August 13, 2019.

[†]The author is with the Beijing University of Posts and Telecommunications, China.
[††]The authors are with the Peking University, China.
[†††]The author is with Hanzhou Dianzi University, China.
a) E-mail: czhu@bupt.edu.cn

DOI: 10.1587/transele.2018ECS6012

(MVs) are generated. To further alleviate the IME processing load, we skip the normal IME for the second reference frame (Ref. 1) and perform a fast IME algorithm based on the best MVs of Ref. 0.

2.1 Current CTU Content Analysis

For the current CTU, this part will extract information used for CUDD. For each CTU, the *SH* and *TS* are calculated. The *SH* is evaluated by the edge information, which is extracted using the Sobel edge operator, as shown in (1).

$$SH = \sum\nolimits_{y=0}^{63} \sum\nolimits_{x=0}^{63} |E| = \sum\nolimits_{y=0}^{63} \sum\nolimits_{x=0}^{63} |E_x| + |E_y| \qquad (1)$$

where

$$E_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \otimes P; E_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \otimes P \qquad (2)$$

The *TS* is computed by using (3). $P(x,y)_t$ and $P(x,y)_{t-1}$ are the pixels in current CTU and the colocated CTU of the previous frame.

$$TS = \sum\nolimits_{y=0}^{63} \sum\nolimits_{x=0}^{63} abs(P(x,y)_t - P(x,y)_{t-1}) \qquad (3)$$

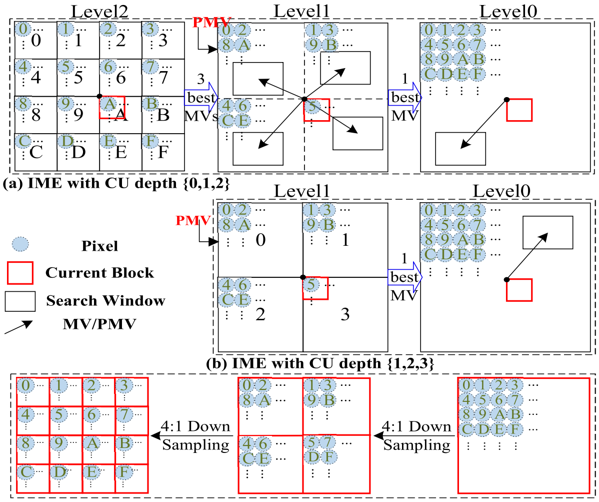
2.2 CU Depth Decision

In HEVC, the splitting depth of the CTU $CU_{Depth} = 0, 1, 2$ and 3 correspond to CU size of 64×64, 32×32, 16×16 and 8×8. We apply the texture information into our CUDD process, which is different with the previous MMEA because there is no CUDD issue in H.264. The higher *SH* value means the texture is very complex and smaller CU size should be used as the candidate, and vice versa. The CU depth type of search range is divided into two categories: {0,1,2} and {1,2,3}. The proposed CU depth range decision strategies are shown in (4). T_{depth} is a threshold, which is calculated as the average SH of selected areas whose final CU depth ranges from 0 to 2 in the previously encoded frames. In the following, we will adjust IME strategies according to the result of CUDD. For example, if {1,2,3} is selected, which indicates CTU with complex texture, we will use two-level MMEA and adopt 4×4 basic block to perform variable block size (VBS) motion estimation in the finest level. The reason is that the downsampling operation will introduce signal aliasing, and the smaller block size will decrease the rate of block mismatching.

$$CU_{depth} \in \begin{cases} \{0,1,2\} & if \quad SH \leqslant T_{depth} \\ \{1,2,3\} & if \quad SH > T_{depth} \end{cases} \qquad (4)$$

2.3 IME for Ref. 0

The previous adaptive MMEA for H.264 [7] selects 1 to 3 levels according to the texture of the current macroblock (MB). However, the CTU (64×64) in HEVC covers a larger area than one MB (16 × 16) in H.264. Both the stationary area and the homogeneous area [7] may exist in the same CTU. Thus for our MMEA in HEVC, we perform adaptive MMEA according to the results of CUDD, and we use at least 2 search levels in each situation.



(a) IME with CU depth {0,1,2}

(b) IME with CU depth {1,2,3}

(c) Current block down sampling flow

Fig. 2 Proposed IME for Ref. 0

Figure 2 depicts our proposed IME for Ref. 0. We adopt three-level adaptive MMEA when CU depth {0,1,2} is selected in the previous step, and we use two-level adaptive MMEA for {1,2,3} case. Like the previous MMEA, the original current block and all reference pixels in the whole search window are directly down-sampled into different resolutions in the same manner.

IME with CU depth {0,1,2}: In this case, IME consists of 3 resolution levels and level 0 is the bottom level. The direct 4:1 down-sample is performed at level 0 to form its upper level 1, and the same down-sample operation is executed upon level 1 to form level 2. As a result, level 0 is sub-divided into four level 1 sub-windows, and level 1 is further sub-divided into sixteen level 2 sub-windows. The three-level IME is performed from the coarsest 16:1 direct down-sampled level 2 to the finest unsampled level 0 in each sub-frame. At level 2, the whole search window (SW) is divided into 16 sub-windows and thus 16-way parallel full search can be adopted to accelerate IME speed. Then one candidate MV with the minimum cost in each sub-window is generated and totally 16 candidates are produced. Finally, 3 best MVs are chosen from 16 candidates as part of the input for level 1. At level 1, the 3 selected best MVs together with one PMV are used as the initial search centers. Then the winning candidate of level 1 will be chosen as the search center for level 0. At level 0, VBS motion estimation is performed. The VBS type includes M×M, (M/2)×M, M×(M/2), (M/2)×(M/2), M×(M/4), (M/4)×M, (M is the CU width), whose costs (such as SAD) all can be derived from the cost of basic block 8×8 [8].

IME with CU depth {1,2,3}: The selected smaller CU sizes (such as 8×8 and 16×16) indicate complex texture in the current CTU, and 16:1 down-sampling operation will introduce inevitable performance degradation. Thus, for this situation, we just use 2 resolution levels: level 1 and level 0. The direct 4:1 down-sampling to level 0 is performed and 4 sub-windows are produced for level 1. Level 1 is the top level and (0, 0) will be selected as the search center for each sub-window. Besides, PMV, as one search center, is also taken into consideration on level 1. Then 5 candidate MVs will be produced and the MV with the minimum cost is chosen for level 0. At level 0, VBS motion estimation is also performed and the search results are then passed to fractional motion estimation (FME) module. Different with CU depth {0,1,2}, the VBS motion estimation is performed based on block 4×4.

2.4 IME for Ref. 1

Our previous MMEA adopts the same searching scheme for different reference frames [7]. To further reduce the coding complexity of HEVC, in this work we conduct fast search scheme for Ref. 1 by using the searching results of Ref. 0.

For Ref. 1, the above multi-resolution IME for Ref. 0 will not be performed. A search center MV_{R1} is derivated based on MV_{R0}, which is the best level 1 MV of Ref. 0, and the computing strategy is similar to the *derivation process for collocated motion vectors* of HEVC standard [9], as shown in (5). Then, a full search of VBS motion estimation will be executed with a small window centered at MV_{R1} on Ref. 1.

$$\begin{aligned} MV_{R1} = {} & Clip3(-32768, 32767, Sign \\ & (distScaleFactor * MV_{R0})* \\ & ((Abs(distScaleFactor * MV_{R0}) + 127) >> 8)) \end{aligned} \tag{5}$$

where $Clip3$ is clip function, $Sign$ is sign function, $distScaleFactor$ is picture distance scale factor and Abs is absolute operator.

2.5 Search Window

To decrease the total resource consumption, Ref. 0 IME and Ref. 1 IME are performed serially. Suppose a search window of level L has the form: $[-W_x^L, W_x^L) \times [-W_y^L, W_y^L)$, then for our system the total time of IME for one CTU (take CU depth {0,1,2} for example and we adopt CTU-level pipeline architecture) is

$$\begin{aligned} T_{total} &= T_{Ref.0} + T_{Ref.1} + T_{ctrl} \\ &= (2W_x^{L2}/4) \times (2W_y^{L2}/4)/(4 \times 4) \\ &+ (2W_x^{L1} \times (2W_y^{L1}) + (2W_x^{L0} \times (2W_y^{L0}) \\ &+ (2W_x^{Ref.1} \times (2W_y^{Ref.1}) + T_{ctrl} \end{aligned} \tag{6}$$

where T_{ctrl} is time consumption of control logic. As we target real time processing of 4K-UHD@60fps under 300M Hz, thus the T_{total} is limited to $300 \times 10^6/(3840 * 2160 * 60/64/64) = 2469 cycles$. In this paper, we use the search window strategies listed in Table 1 (T0 is a threshold) when taking (6) and TS value into consideration. This is different with the previous MMEA [7], of which the searching window is fixed for a specific level.

Table 1 Search windows for the proposed IME

Ref.	Level	Search Window				
		-	CU Depth{0,1,2}		CU Depth{1,2,3}	
0	2	W_x^{L2} W_y^{L2}	128 96		N/A	
	1	-	TS>T0	TS≤T0	TS>T0	TS≤T0
		W_x^{L1} W_y^{L1}	18 18	16 16	20 20	18 18
	0	W_x^{L0} W_y^{L0}	10 10	11 11	10 10	10 10
1	0	W_x^{L0} W_y^{L0}	11 11	13 13	10 10	13 13

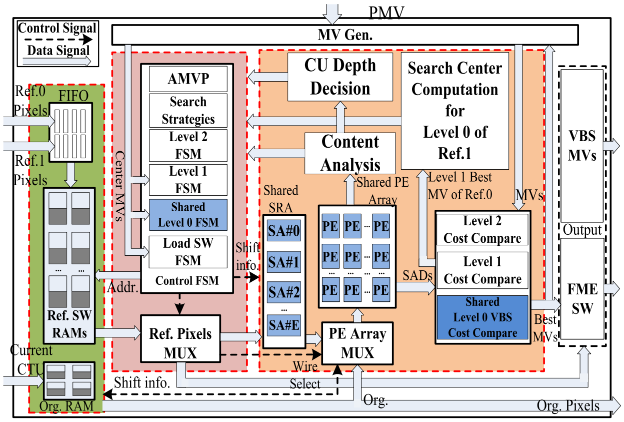


Fig. 3 Proposed IME architecture

3. IME Hardware Architecture

The proposed IME architecture is depicted in Fig. 3. The reference SW and original pixels are first stored in the RAM arrays and then passed to the computing unit with the cooperation of control module. At last, the generated best MVs and the other information (such as SW and original pixels) are transmitted to FME. Ref. 0 IME and Ref. 1 IME share the level 0 FSM, shift register array (SRA), the processing element (PE) and cost computation.

4. Implementation Results

The results contain video coding efficiency comparisons and hardware implementation.

Performance of proposed IME: The proposed algorithm is implemented on HM16. The matching distortion criterion SAD is adopted and 2 reference frames are used. The adopted four QPs are 27, 32, 37 and 42. Low delay B (LDB) is configured and the coding structure (CS) is IBBB [10]. Class A, class B, class C (used to verify the

Table 2 Performance comparisons

Alg.	Ref. Num.	CS	Class	Resolution	BDBR Inc. (%)
Work [5] v.s. HM	1	N/A	A	2560 × 1600	1.45
			B	1920 × 1080	1.56
			C	832 × 480	2.92
			E	1280 × 720	2.98
			Average		**2.23**
Proposed v.s. HM	2	IBBB	A	2560 × 1600	0.5
			B	1920 × 1080	0.9
			C	832 × 480	0.5
			E	1280 × 720	0.2
			Average		**0.53**

Table 3 Comparison with other FPGA architectures

Design	Wrok [3]	Wrok [4]	Wrok [5]	Proposed
Standard	H.265	H.265	H.265	H.265
Ref. Num.	1	1	1	2
Search Range	32 × 32	N/A	129 × 127	256 × 192
CTU Size	32 × 32	64 × 64	64 × 64	64 × 64
Clock (MHz)	318	142	200	300
Res. (LUT)	49K	667K	26K	180K
Throughput	4K@37	8K@43	1080p@60	4K@60
Memory	36KB	N/A	72KB	14KB
Technology	Virtex-7	Virtex-7	Arria II GX	Virtex-7

performance for small-resolution sequence) and class E are included in the evaluation. Each class contains a series of test sequences with the same resolution. The coding performance on HEVC testing datasets is not available for works such as [4], and thus it is not included in our comparisons. In our experiments, the method in work [5] is included for comparison and the test results are tabulated in Table 2. Compared with the original HM standard, our scheme just introduces 0.53% percent bitrate increase, which achieves the best coding efficiency of hardware fast IME as far as we know. Work [5] just supports one reference frame IME, and it brings higher coding efficiency degradation (2.23% bitrate increase).

Hardware implementation Results: Our IME design is first implemented in Verilog HDL, simulated and synthesized into Virtex7 FPGA. Under frequency of 300MHz, our IME can support the real-time processing of 4K-UHD@60fps. The proposed design is also synthesized by the 0.18-μm CMOS process. Only a total of 2160K gates and 14KB SRAM are consumed. The reference data sharing search technique [11] is used to save on-chip memory usage. In Table 3, we compare our result with previous state-of-the-art architectures on FPGA platforms. Our proposed architecture supports search range 256 × 192, which is the biggest search area among four schemes. By using motion vector derivation, our scheme supports 2 reference frames with no additional hardware resource consumption through skipping the normal motion estimation for the second reference picture. Work [4] achieves higher throughput (8K@43fps), but it utilizes more resources when compared with our method. In work [5], the authors present the implementation results both for FPGA and ASIC platforms, and we include the FPGA result for comparison. Our design achieves a good tradeoff between throughput and hardware cost with negligible coding performance loss.

5. Conclusion

A high-quality hardware-friendly IME scheme is reported. According to different characteristics of CTU content, the proposed method adopts different adaptive multi-resolution strategies coupled with accurate full-PU modes IME at the finest level. Besides, by using motion vector derivation, IME for the second reference picture is simplified through processing element (PE) sharing. The designed hardware IME can support the real-time processing of 4K-UHD @60fps, while the BD-rate is just increased by 0.53%. Our design achieves a good tradeoff between throughput and hardware cost, and produces higher coding efficiency compared with the recent hardware IME scheme.

Acknowledgments

This work has been supported in part by the Beijing Natural Science Foundation (4182044), and the Fundamental Research Funds for the Central Universities.

References

[1] G.J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," IEEE Trans. Circuits Syst. Video Technol., vol.22, no.12, pp.1649–1668, 2012.

[2] Z. Pan, J. Lei, Y. Zhang, et al., "Fast motion estimation based on content property for low-complexity H. 265/HEVC encoder," IEEE Trans. Broadcast., vol.62, no.3, pp.675–684, 2016.

[3] E. Alcocer, R. Gutierrez, O. Lopez-Granado, et al., "Design and implementation of an efficient hardware integer motion estimator for an HEVC video encoder," Journal of Real-Time Image Processing, vol.16, no.2, pp.547–557, 2019.

[4] N.V. Thang, V.D. Tung, and N.D. Hoan, "An optimized hardware design of Integer Motion Estimation HEVC for encoding 8K video," 2017 4th NAFOSTED Conference on Information and Computer Science, IEEE, pp.319–324, 2017.

[5] G. Pastuszak and M. Trochimiuk, "Algorithm and architecture design of the motion estimation for the H. 265/HEVC 4K-UHD encoder," Journal of Real-Time Image Processing, vol.12, no.2, pp.517–529, 2016.

[6] X.H. Ji, C. Zhu, H.Z. Jia, X. Xie, and H. Yin, "A Hardware-Efficient Architecture for Multi-Resolution Motion Estimation Using Fully Reconfigurable Processing Element Array," Proc. ICME, pp.1–6, July 2011.

[7] J. Liu, X. Ji, C. Zhu, H. Jia, X. Xie, and W. Gao, "Adaptive multi-resolution motion estimation using texture-based search strategies," 2014 IEEE International Conference on Consumer Electronics (ICCE), pp.363–366, 2014.

[8] C.Y. Chen, C.T. Huang, Y.H. Chen, et al., "Level C+ data reuse scheme for motion estimation with corresponding coding orders," IEEE Trans. Circuits Syst. Video Technol., pp.553–558, 2006.

[9] Coding H E V. Recommendation ITU-T H. 265[J]., "International Standard ISO/IEC," 2013: 23008-2, 2013.

[10] F. Bossen, "Common test conditions and software reference configurations," JCTVC-L1100, 2013.

[11] X. Huang, K. Wei, H. Yin, C. Zhu, H. Jia, and D. Xie, "Three-level pipelined multi-resolution integer motion estimation engine with optimized reference data sharing search for AVS," Journal of Real-Time Image Processing, vol.15, no.1, pp.43–55, 2018.