A Novel Two-Step Integer-pixel Motion Estimation Algorithm for HEVC Encoding on a GPU

Keji Chen[1], Jun Sun[1,2], Zongming Guo[1,2(✉)], and Dachuan Zhao[3]

[1] Institute of Computer Science and Technology of Peking University, Beijing, China
{chenkeji,jsun,guozongming}@pku.edu.cn
[2] Cooperative Medianet Innovation Center, Shanghai, China
[3] Advanced Micro Devices Co., Ltd., Beijing, China
Dachuan.Zhao@amd.com

Abstract. Integer-pixel Motion Estimation (IME) is one of the fundamental and time-consuming modules in encoding. In this paper, a novel two-step IME algorithm is proposed for High Efficiency Video Coding (HEVC) on a Graphic Processing Unit (GPU). First, the whole search region is roughly investigated with a predefined search pattern, which is analyzed in detail to effectively reduce the complexity. Then, the search result is further refined in the zones only around the best candidates of the first step. By dividing IME into two steps, the proposed algorithm combines the advantage of one-step algorithms in synchronization and the advantage of multiple-step algorithms in complexity. According to the experimental results, the proposed algorithm achieves up to 3.64 times speedup compared with previous representative algorithms, and the search accuracy is maintained at the same time. Since IME algorithm is independent from other modules, it is a good choice for different GPU-based encoding applications.

Keywords: High Efficiency Video Coding (HEVC) · Graphics Processing Unit (GPU) · Integer-pixel Motion Estimation (IME) · Two-step algorithm

1 Introduction

High Efficiency Video Coding (HEVC) [1] is the latest generation video coding standard. With enhanced coding tools, HEVC achieves about 50% bit-rate reductions at similar Mean Opinion Score (MOS) compared with H.264/AVC [2]. Among the enhanced coding tools, HEVC introduces a quadtree-like coding structure. The frames are first partitioned into Coding Tree Units (CTUs) with a maximum size of 64×64, then the CTUs are further partitioned into Coding Units (CUs) and Prediction Units (PUs). For each CU, there are up to 8 different kinds of PU partitions.

L. Amsaleg et al. (Eds.): MMM 2017, Part II, LNCS 10133, pp. 28–36, 2017.
DOI: 10.1007/978-3-319-51814-5_3

Within the encoding process, Motion Estimation (ME) is applied to find proper Motion Vectors (MVs) for different PUs. ME investigates the MV Candidates (MVCs) within a specific region determined by the search range and the start MV, and tries to find the MVC with the least cost. Generally, ME is partitioned into Integer-pixel ME (IME) and Fractional-pixel ME (FME). IME investigates the integer-pixel MVCs among the whole search region and find out the best integer-pixel MVC for FME. Since IME is the topic of this paper, the MVCs in the following part means the integer-pixel MVCs only.

The cost of MVCs determines which MVC is better. In typical IME methods, Rate Distortion Optimization (RDO) [3] framework is applied with the following formula:

$$cost = D + \lambda * R, \tag{1}$$

where D can be measured with Sum of Absolute Difference (SAD), R can be derived from the distance between the MVC and the start MV, and λ is the Lagrange multiplier.

It is complex to retrieve the best MVC for different PUs in a quad-tree. One possible way to increase the speed is to use Graphic Processing Units (GPUs). GPUs are essential for modern personal computers. With many-core architecture, numerous tasks can be gathered as a kernel and be processed simultaneously on a GPU. Synchronizations of GPU are quite expensive. Besides, sufficient independent tasks should be provided to make full use of the cores. Compared with Central Processing Units (CPUs), GPUs are more energy saving and powerful when processing numerous independent tasks.

GPU-based IME algorithms are extensively applied in previous works. Since synchronizations are quite expensive, CPU-based multiple-step IME algorithms are not adopted. Early GPU-based IME methods [4,5] are usually based on Full Search algorithm, which introduces only one synchronization and achieves the best accuracy. However, since Full Search is complex, these methods are not satisfactory. S. Radicke *et al.* propose an algorithm based on Frayed Diamond Search Pattern (FDSP) [6]. Only the MVCs of a predefined pattern are searched in this work. And Recursive Sum of Absolute Differences (RSAD), which accumulates the SADs of basic blocks for the large blocks, is introduced to avoid redundant SAD calculations. Although the complexity is significantly reduced with FDSP, the synchronization cost is overrated for high motion videos. C. Jiang *et al.* propose an algorithm called DZfast [7]. By judging the possible zone of the best MVC, DZfast reduces the complexity for static area with lower search ranges. However, DZfast is designed for parallelizing within a macroblock with the multiview extension of H.264/AVC. The feature of independent views is utilized, which makes it unsuitable to be applied to HEVC directly. Since IME algorithm is fundamental in GPU-based encoding, it should be carefully studied and further optimized.

The main contribution of this paper is to propose a novel two-step IME algorithm for GPU-based encoding, which combines the advantage of one-step algorithms in synchronization and the advantage of multiple-step algorithms in complexity. According to experimental results over the widely used encoder x265

[8], the proposed algorithm achieves up to 9.75 times and 3.64 times speedup compared with Full Search and FDSP respectively, while the search accuracy is still maintained.

The rest of this paper is organized as follows. Section 2 introduces the proposed two-step IME algorithm. Section 3 presents the analysis of different IME algorithms. The detailed performance evaluation of the proposed algorithm is given in Sect. 4. And finally, the conclusions are made in Sect. 5.

2 The Proposed Two-Step IME Algorithm

The greedy algorithm is applied in the traditional CPU-based IME algorithms. The IME process is first divided into multiple steps, and only the MVCs around the start MV are investigated in each step. After a step, the cost of the MVCs in this step is summarized, and the best MVC among them is found as the start MV of the next step. Thus, regions far from the best MVC of each step will be neglected. To apply similar algorithms on a GPU, synchronizations will be introduced after each step. In summary, the greedy algorithm reduces the complexity. But the many synchronizations become a heavy burden for GPU-based encoding.

To avoid the synchronizations, previous GPU-based methods [4–6] apply one-step IME algorithms. The costs of the MVCs are only summarized once when all the MVCs are investigated. And the best MVC is found directly. Since there's only one step, the MVC distribution of this step should be quite dense, so that the search region can be well covered and good search accuracy can be achieved. As a result, although there's only one synchronization in one-step algorithms, the MVC count is large, and the complexity is significant.

By combining the advantage of multiple-step and one-step algorithms, we propose a two-step IME algorithm for GPU-based encoding. In the proposed algorithm, the IME process is divided into two steps. Only the MVCs around the best MVC of the first step are investigated in the second step. As a result, although an extra synchronization is introduced, the extra cost is limited and the complexity reduction is significant. Thus, the overall speed can be increased.

The two steps of the proposed IME algorithm are based on corresponding Search Patterns (SPs). The SP of the first step changes with the search range, while the SP of the second step is a fixed 8×8 pattern. The SPs of the two steps are carefully designed, so that every MVC within the search region has chance to be selected as the best MVC. Within a specific region, the denser the MVC of the first step, the higher the possibility it can be investigated in the second step. Considering that the MVCs closer to the start MV are more likely to be the best MVC, the SP of the first step is designed denser around the start MV. To reduce the complexity, the SP of the first step does not need to be too dense since the second step is always applied. Figure 1 shows the SP of the first step when search range is 16, and Fig. 2 shows the SP of the second step. In these two figures, the black point represents the start MV and other points represent the MVCs investigated in each step.

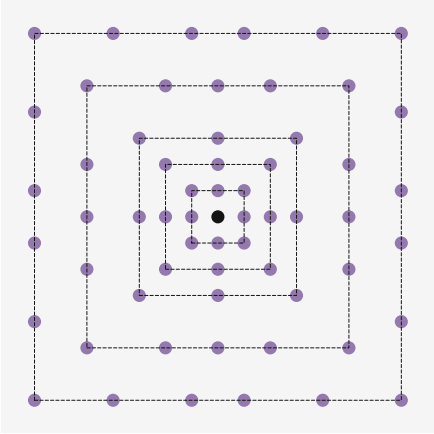


Fig. 1. The SP of the first step in the proposed two-step IME algorithm when search range is 16.

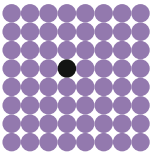


Fig. 2. The SP of the second step in the proposed two-step IME algorithm.

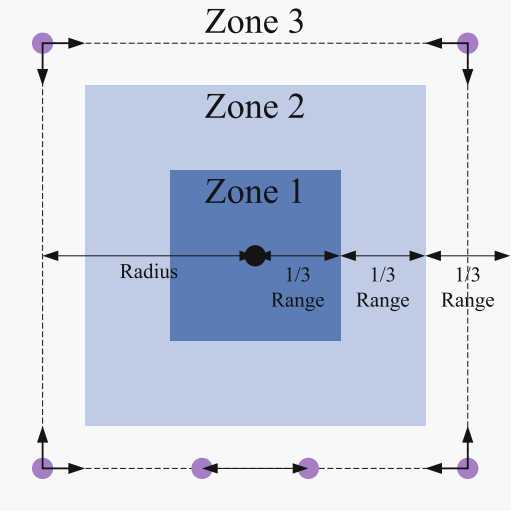


Fig. 3. The algorithm to derive the SP of the first step for different search range in the proposed two-step IME algorithm.

The algorithm to derive the SP of the first step for different search range is shown in Fig. 3. According to the search range, the search region is divided into 3 zones. The density of MVCs decreases from Zone 1 to Zone 3. The MVCs lie on several circles, which are depicted as dashed squares. The distance between the start MV and the circle is defined as the circle's radius. In the proposed algorithm, the radius increases by 2 in Zone 1, and increases by 4 and 6 separately in Zone 2 and Zone 3. For each circle, the MVC assigning distance is set as the minor between the radius and 6. The MVCs are assigned starting from the four angles. When the distance between two MVCs in the middle of an edge is greater than the assigning distance, an extra MVC will be assigned between them.

The proposed algorithm is applied on basic blocks first. For large blocks, RSAD method is applied in the first step. However, since the second step start MV of large blocks might be different from that of basic blocks, separate SAD calculations are applied. The basic block is set as the minimum PU, which is 8×8 in the experiment of this paper.

To implement the proposed algorithm on a GPU, the device memory consumption should be considered. Before applying IME algorithms, the frames and their referenced frames should be stored on device memory. The generated IME results should also be stored. Besides, if RSAD is applied, the SADs of the basic blocks should be stored to derive the SADs of large blocks. For the proposed two-step algorithm, RSAD is only applied in the first step. Thus, only the SADs of the first step should be stored. Meanwhile, the best MVCs of the first step are utilized as the start MV of the second step, and they should also be stored.

To make full use of the GPU platform, the IME of a whole frame is applied together. For the proposed algorithm, IME is divided into 3 kernels: (1) First step IME for the basic blocks, (2) First step IME for the large blocks, (3) Second step of every block size. 8 threads are assigned for each basic block. Thus, for a frame with 416×240 resolution, there can be 12480 threads working independently in a kernel. For higher resolution, there will be more independent threads.

3 Analysis of Different IME Algorithms

To present the advantage of the proposed algorithm, three different IME algorithms are analyzed in this section. The comparison between them is shown in Table 1, where *MVC Count* means the count of overall investigated MVCs, *Sync. Count* means the synchronization count, and *Full Covered* means whether all the MVCs in the search region have chance to be investigated. It can be observed that the proposed algorithm significantly reduces the MVC count of basic blocks with only one extra synchronization. Meanwhile, only a part of MVCs in the search region have chance to be the best MVC in FDSP, and all the MVCs have chance in Full Search and the proposed algorithm.

The effect of RSAD method on complexity should be considered. For large blocks, RSAD is applied in Full Search and FDSP, but it can only be applied in the first step of the proposed algorithm. However, there are always 64 MVCs in the second step, thus the complexity increment is constant for a specific PU

Table 1. Analysis of different IME algorithms

Algorithm	Search Range	MVC Count	Sync Count	Full Covered
Full Search	16	1089	1	Yes
	32	4225	1	Yes
	48	9409	1	Yes
	64	16641	1	Yes
FDSP	16	241	1	No
	32	865	1	No
	48	2001	1	No
	64	3521	1	No
Proposed	16	125	2	Yes
	32	233	2	Yes
	48	433	2	Yes
	64	709	2	Yes

partitioning strategy. It will be shown in Sect. 4 that for high motion videos, this increment is not significant compared with the complexity reduction from the MVC count.

The device memory consumption is an advantage of the proposed algorithm. Since RSAD is applied for all MVCs with Full Search and FDSP, the SADs of basic blocks should be all stored. Meanwhile, only the SADs of the first step with the proposed algorithm should be stored. When search range is 64, Full Search will store 16641 SADs for each basic unit. Suppose each SAD is 2 bytes, for a video with 2560×1600 resolution, about 2 G bytes will be utilized. For FDSP, which stores 3521 SADs for each basic unit, about 450 M bytes are necessary. For the proposed algorithm, only the 645 SADs of the first step need to be stored for each basic unit. Then, only about 83 M bytes are utilized. Extra device memory is necessary to store the best MVCs of the first step with the proposed algorithm. But it is only about 500 K bytes, and can be ignored.

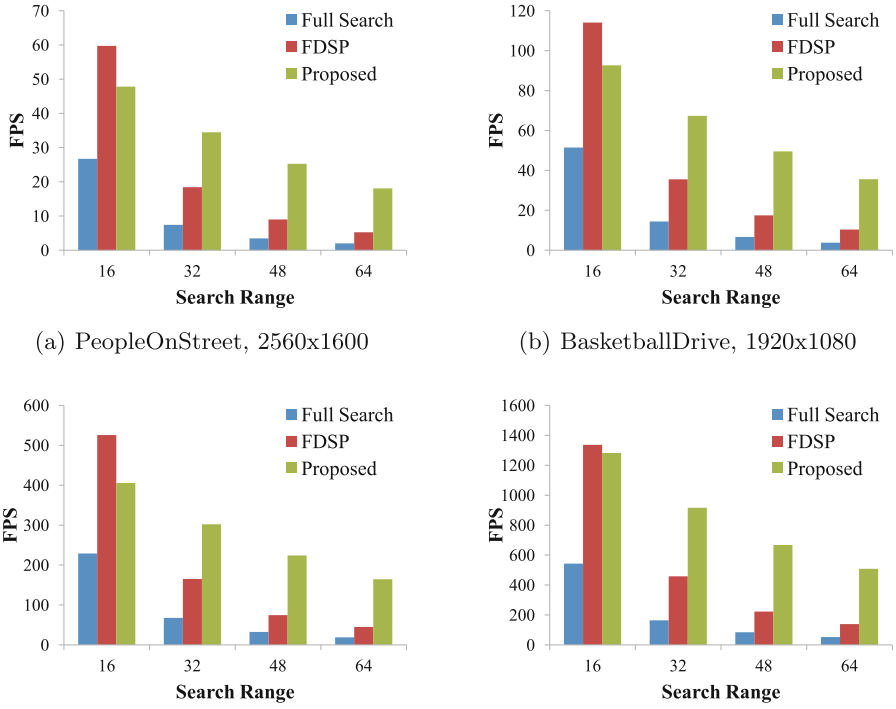
4 Experimental Results

Detailed experimental results are given in this section. According to our experiments, λ in (1) does not affect the result much, and it is set to 10 (corresponding Quantization Parameter is 32) in IME process. The experimental platform is described in Table 2. And the results are collected under HEVC common test conditions [9] by applying the default parameter preset of x265.

The speed comparison between different IME algorithms using different sequences is shown in Fig. 4. The speed is measured in Frames Per Second (FPS), and is derived from the IME time of each algorithm on the GPU. It can be observed that the FPS reduction speed of the proposed algorithm is much

Table 2. Experimental platform

GPU	AMD Radeon R9-290X
Maximum power	250 w
Sreaming processors	2816
Core freq.	1 GHz
Device memory	4 GB
Device memory freq.	5 GHz
Computing framework	OpenCL 1.0



(a) PeopleOnStreet, 2560x1600

(b) BasketballDrive, 1920x1080

(c) BasketballDrill, 832x480

(d) BasketballPass, 416x240

Fig. 4. The speed comparison between different IME algorithms using different sequence with search range of 16, 32, 48 and 64.

slower than that of Full Search and FDSP with the increment of the search range, which is an obvious advantage when search range is large.

The detailed results are illustrated in Table 3, where SR represents the search range, and ΔS represents the speedup compared with Full Search algorithm. Each result is summarized by averaging the results of all 8-bit HEVC standard sequences in that class listed in [9]. By replacing the original IME algorithm

Table 3. Experimental results of different IME algorithms

SR	Class	Algorithm	FPS	BD-Rate	ΔS	SR	Class	Algorithm	FPS	BD-Rate	ΔS
16	A	Full Search	26.8	0.00%	1.00	32	A	Full Search	7.4	0.00%	1.00
		FDSP	59.6	1.24%	2.22			FDSP	18.4	1.63%	2.48
		Proposed	47.5	0.75%	1.77			Proposed	34.6	1.07%	4.65
	B	Full Search	51.5	0.00%	1.00		B	Full Search	14.4	0.00%	1.00
		FDSP	114.0	0.90%	2.22			FDSP	35.6	0.91%	2.46
		Proposed	92.4	0.54%	1.79			Proposed	67.5	0.64%	4.68
	C	Full Search	229.7	0.00%	1.00		C	Full Search	67.7	0.00%	1.00
		FDSP	522.6	1.38%	2.28			FDSP	165.3	1.49%	2.44
		Proposed	408.3	1.01%	1.78			Proposed	298.9	1.21%	4.41
	D	Full Search	583.9	0.00%	1.00		D	Full Search	164.7	0.00%	1.00
		FDSP	1369.7	1.02%	2.54			FDSP	461.1	1.20%	2.80
		Proposed	1259.6	0.74%	2.34			Proposed	904.9	0.85%	5.49
	E	Full Search	114.8	0.00%	1.00		E	Full Search	32.7	0.00%	1.00
		FDSP	255.3	0.48%	2.22			FDSP	81.4	0.51%	2.48
		Proposed	202.8	0.24%	1.77			Proposed	151.0	0.38%	4.61
48	A	Full Search	3.5	0.00%	1.00	64	A	Full Search	2.0	0.00%	1.00
		FDSP	9.0	1.84%	2.59			FDSP	5.2	2.09%	2.59
		Proposed	25.3	1.23%	7.28			Proposed	18.1	1.53%	9.01
	B	Full Search	6.7	0.00%	1.00		B	Full Search	3.8	0.00%	1.00
		FDSP	17.4	0.97%	2.61			FDSP	10.4	1.03%	2.70
		Proposed	49.6	0.75%	7.42			Proposed	35.6	0.80%	9.28
	C	Full Search	32.3	0.00%	1.00		C	Full Search	18.8	0.00%	1.00
		FDSP	74.5	1.67%	2.31			FDSP	44.9	1.76%	2.39
		Proposed	222.5	1.33%	6.88			Proposed	163.6	1.42%	8.70
	D	Full Search	84.6	0.00%	1.00		D	Full Search	52.3	0.00%	1.00
		FDSP	222.1	1.39%	2.63			FDSP	138.5	1.41%	2.65
		Proposed	662.6	1.07%	7.84			Proposed	503.8	1.09%	9.63
	E	Full Search	15.2	0.00%	1.00		E	Full Search	8.2	0.00%	1.00
		FDSP	39.2	0.64%	2.57			FDSP	23.3	0.64%	2.83
		Proposed	112.1	0.49%	7.36			Proposed	80.1	0.49%	9.75

of x265 and keeping other modules the same, PSNR and bitrate of different algorithms are derived and summarized as BD-Rate. BD-Rate is also derived by comparing with Full Search. The effect that does not use RSAD in the second step can be observed from the results. When the search range is 16, the complexity increment caused by RSAD is significant and the proposed algorithm is slower than FDSP. However, when the search range is larger, the proposed algorithm outperforms the other two algorithms. Up to 9.75 times speedup of

the proposed algorithm is observed compared with Full Search, and up to 3.64 times speedup is observed compared with FDSP. Meanwhile, the results show that the BD-Rate of the proposed algorithm is better than FDSP and is close to Full Search. Considering that Full Search always investigates every MVC and precisely finds out the best one, the results indicate that the search accuracy with the proposed algorithm is well maintained.

5 Conclusion

In this paper, a two-step IME algorithm is proposed for HEVC encoding on a GPU. By dividing IME into two steps with the proposed algorithm, the MVC count as well as complexity is significant reduced. According to the experimental results, up to 9.75 times and 3.64 times speedup is achieved compared with Full Search and FDSP respectively, and the search accuracy is well maintained at the same time. Since IME algorithm is fundamental and independent from other modules in GPU-based encoding, the proposed two-step IME algorithm is a good choice for different GPU-based encoding applications.

Acknowledgment. This work was supported by National Natural Science Foundation of China under contract No. 61671025 and National Key Technology R&D Program of China under Grant 2015AA011605.

References

1. Sullivan, G.J., Ohm, J.-R., Han, W.-J., Wiegand, T.: Overview of the High Efficiency Video Coding (HEVC) standard. IEEE Trans. Circuits Syst. Video Technol. **22**(12), 1648–1667 (2012)
2. Wiegand, T., Sullivan, G.J., Bjntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Trans. Circ. Syst. Video Technol. **13**(7), 560–576 (2003)
3. Sullivan, G.J., Wiegand, T.: Rate-distortion optimization for video compression. IEEE Signal Process. Mag. **15**(6), 74–90 (1998)
4. Chen, W.-N., Hang, H.-M.: H.264/AVC motion estimation implmentation on Compute Unified Device Architecture (CUDA). IEEE International Conference on Multimedia and Expo, pp. 697–700, June 2008
5. Rodríguez-Sánchez, R., Martínez, J.L., Fernández-Escribano, G., Claver, J.M., Sánchez, J.L.: Reducing complexity in H.264/AVC motion estimation by using a GPU. IEEE International Workshop on Multimedia Signal Processing, pp. 1–6, October 2011
6. Radicke, S., Hahn, J.-U., Wang, Q., Grecos, C.: Bi-predictive motion estimation for HEVC on a Graphics Processing Unit (GPU). IEEE Trans. Consumer Electron. **60**(4), 728–736 (2014)
7. Jiang, C., Nooshabadi, S.: A scalable massively parallel motion and disparity estimation scheme for multiview video coding. IEEE Trans. Circ. Syst. Video Technol. **26**, 346–359 (2016)
8. x265 Developers: x265 HEVC Encoder/H.265 Video Codec (2015). http://www.x265.org/
9. Bossen, F.: Common test conditions and software reference configurations. Document JCTVC-L1100, January 2013