CHAPTER 3

QBMO Motion Estimation Search Algorithm Performance Evaluation for Video Compression

Asral Bahari Jambek, Arief Affendi Juri, Rizalafande Che Ismail and Mohd Nazrin Md Isa
School of Microelectronic Engineering, Universiti Malaysia Perlis, Malaysia
asral@unimap.edu.my

3.1. Introduction

The high demand for higher quality and a lower bitrate has driven the video community to produce better video compression standards from time to time. For example, H.264 is able to achieve better video quality results by more than 2 dB and 3 dB as compared to MPEG-4 and H.263, respectively. Furthermore, with the same video quality, H.264 is able to produce a 50% bit-rate reduction as compared to MPEG-4 [1-3].

However, these improvements come at the cost of an increase in computational time. As compared to H.263, H.264 the encoding and decoding times increase by three and two times, respectively. Most of these increments are contributed by the motion estimation (ME) module, which plays a vital part in determining the quality of the video compression output. The module can consume from 70% (one reference frame) to 90% (five reference frames) of the total encoding time [4-6]. This shows that reducing the ME computational load will result in a significant overall computational load reduction.

With the H.264 standard, the reference software implements an UMHexagonS algorithm as one part of its ME. The algorithm combines several ME techniques to create a fast search ME algorithm. It achieves good compression efficiency and faster searches compared to the full search algorithm. Thus, this algorithm will be focus in this work. UMHexagonS consists of five different steps, namely: initial search point prediction, unsymmetrical cross search, small full search, uneven multi-hexagon-grid search, and extended hexagon-based search, the latter of which contains a small diamond search [8-10].

3.2. A Quadrant-Based Multi-Octagon Search Algorithm (QBMO)

To further reduce the computational load in UMHexagonS, the quadrant-based multi-octagon search algorithm (QBMO) is proposed [7]. The algorithm focuses on the fourth step of the UMHexagonS algorithm by implementing the multi-octagon-grid search to replace the multi-hexagon-grid search. Furthermore, a quadrant-based search is implemented on top of multi-octagon-grid search, which results in four quadrant shapes, as shown in Figure 3.1. The quadrant to be used is dependent on the motion vector (MV) of the block in the previous frame. An example of determining the quadrant is given in Figure 3.2. The quadrant where the MV of the

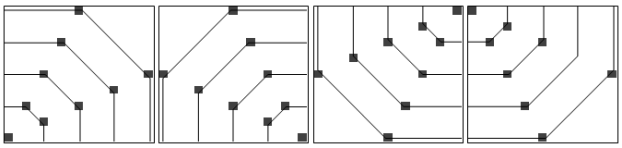
previous frame is located - labelled with a triangle, as shown in Figure 3.2(a) - will be chosen as the quadrant where the search will be performed.

To evaluate the performance of QBMO, the algorithm will be compared against existing algorithms. First, the study will measure the performance against the individual algorithms implemented in each step of UMHexagonS. Next, the performance improvement when QBMO's implementation is combined with another algorithm will be studied.

As discussed in Section 3.1, the second step of UMHexagonS is an unsymmetrical-cross search. This algorithm evaluates the best match surrounding the search centre with 24 total search points. In this work, the irregular-cross template [10] is chosen as the second stage of the UMHexagonS algorithm. While the standard unsymmetrical cross search is emphasized only on the horizontal search, the irregular cross template emphasizes both the horizontal and vertical motion conditions. Furthermore, the algorithm biases the search location towards the best-predicted initial place location, as shown in Figure 3.3. In this example, the dark round shape represents the current block's location and the square shape represents the best initial place. The implemented cross template points are represented by the white round shape.

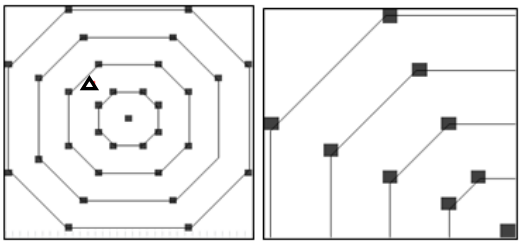
The third step of UMHexagonS is a small full search, where it will evaluate a 5x5 search area around the central location, with a total of 25 search points. To reduce the computational load involved in this step, a small full search of 3x3 is used, as proposed in [11]. It is found that 80% of the MV is distributed within the 5x5 search area, while 70% of the MV is distributed within the 3x3 search area. The 10% reduction in terms of MV distribution coverage during the search is compensated for by the reduction of more than 60% of the search candidate. This greatly reduces the computational complexity of this step.

The fifth step of UMHexagonS implements a conventional extended hexagon-based search. This work replaces this algorithm with horizontal and vertical hexagon searches [11], as shown in Figure 3.4. The selection of the search pattern to be used is decided based upon the block size. A 16x16 or 8x8 block size utilizes a uniform hexagon shape, as in Figure 3.4 (a). For a 16x8 or 8x4 block, the horizontal hexagon shape, as in Figure 3.4 (b), will be used. The vertical hexagon shape, as in Figure 3.4(c), will be used by an 8x16 or 4x8 block size. A summary of the equivalent implementation for UMHexagonS is shown in Table 3.1.



(a) (b) (c) (d)

Figure 3.1. Four quadrant of the quadrant-based multi-octagon search.



(a) (b)

Figure 3.2: Determining the multi-octagonal search quadrant: (a) the triangle represents the MV of the block located in the previous frame, (b) the chosen quadrant for the current block.

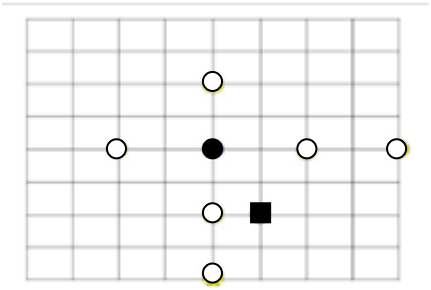
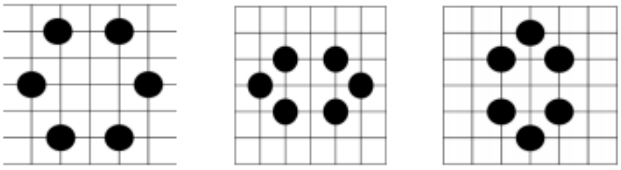


Figure 3.3. The red point is the current block's location, the blue point is the best initial place, and the yellow points are the cross template points [10].



(a) (b) (c)

Figure 3.4. Three hexagonal patterns implemented in [11]: (a) Uniform hexagon (b) Horizontal hexagon (c) Vertical Hexagon.

Table 3.1: Summary of the equivalent algorithm for UMHexagonS

No	UMHexagonS Steps	Equivalent Implementation
1	Initial prediction	-
2	Unsymmetrical cross search	Irregular cross search [10]
3	5x5 full search	3x3 full search [11]
4	Multi hexagon-grid search	QBMO
5	Extended hexagon search	Horizontal and vertical hexagon [11]

3.3. Results and Analysis

In this work, the simulation was performed using an Intel Core i5 processor 430M (2.26GHz, 3MB l3 Cache), 2GB RAM and Windows 7 Home Premium 64-bit. The algorithm was tested on the H.264 reference software (JM17.2) with the simulation setup as shown in Table 3.2 using six test video sequences: Bus, Football, Foreman, Silent, News and Hall. All the video test sequences were in CIF format (352x288 pixels), which targeted portable application. These video samples have been categorized into three groups, namely: aggressive motion (the Bus and Football videos), medium motion (the Foreman and Silent videos) and low motion (the News and Hall videos).

In order to evaluate the performance of each algorithm, ten simulation settings are performed, as shown in Table 3.3. Algorithm 1 represents the conventional UMHexagonS algorithm of the reference software. This algorithm will become the reference algorithm when comparing the performance with other algorithms. Algorithms 2 through to 10 replace the specific step in the UMHexagonS algorithm with one of the algorithms listed in Table 3.3. In the table, the symbol 'tick' denotes that the conventional UMHexagonS step is being used in that specific step, while the name of the algorithm is shown if the step is performed by any equivalent algorithm.

Two types of simulations were performed for this work, Type A and Type B. The Type A simulation replaces one step of the UMHexagonS algorithm at a time. This is represented by Algorithms 2 through to 7. The purpose of the simulation was to measure the effectiveness of the QBMO as compared to other existing methods. The Type B simulation replaces all the steps in UMHexagonS (except for step 1) with other algorithms. This is represented by Algorithms 9 and 10, as shown in Table 3.3. This simulation measures further reductions when QBMO is used with other existing algorithms in order to improve the overall performance of UMHexagonS.

Table 3.4 through to Table 3.6 give the performance results for each video benchmark in terms of the ME simulation time (MET), the PSNR and the bitrate. From the table, Algorithms 2-9 give a higher MET reduction time for aggressive motion videos. This is due to the fact that aggressive video motions tend to have a bigger MV, since their best match location is typically located further away from their original locations. On the other hand, low motion videos tend to have their best match location located at the same or close to their original positions. This causes the algorithms have only a slight impact on these types of videos, because most of the time they will enter into early termination steps. If this happens, the ME step will be skipped and the original position will be taken as the best position.

For the Type A simulation, comparing Algorithms 1 through to 7, Algorithm 7 is able to outperform all the other algorithms in terms of MET, whereby it gives a reduction of up to 18.2%. This is achieved without sacrificing the quality of the video (PSNR) or the bitrate. For Algorithm A, the maximum decrease in PSNR is only about 0.015dB, with an average PSNR drop of 0.002dB. However, this small change in the PSNR will not affect the visual quality significantly, since it will not be visible to the human eye [13]. The worst bitrate increment for Algorithm 8 is only 1.21% (or 18.89 kb/s). This shows that the QBMOs are effective in reducing the computational load for the ME engine in video compression.

For the Type B simulation, both Algorithms 8 and 9 utilize the same algorithm for steps 2, 3 and 5. However, for step 4, Algorithm 9 uses a multi-octagon-grid search, whereas

Algorithm 10 uses a QBMO search algorithm. Based on the results, Algorithm 10 gives the highest reduction in terms of MET for all video sample categories, with a maximum reduction of 28.66% in MET; moreover, this reduction is achieved without degrading the PSNR or bitrate significantly. From the table, the worst PSNR drop is only 0.015. The highest bitrate increment for Algorithm 10 is only 1.43% (or 5.64kb/s).

Table 3.2: H.264 Parameters for the Algorithms' Simulation

Parameter	Value
Profile	Baseline
Level	4.0
Codec	JM17.2
Image format	CIF (352x288 pixels)
MV search range	32
Frame rate	30 fps
RD optimization	On
Total number of reference	5
Sequence type	IPPP
Entropy coding	CAVLC
Encoded frames	100
Motion Estimation for component	Y (luma)

Table 3.3: Simulation Results for the Bus Video Sample

Algo-rithm. No.	1st Step: initial prediction	2nd Step: Unsymmetrical cross search	3rd Step: 5x5 full search	4th Step: Multi hexagon-grid search	5th Step: Extended hexagon search
1	✓	✓	✓	✓	✓
2	✓	Irregular cross search	✓	✓	✓
3	✓	✓	New 3x3 square search	✓	✓
4	✓	✓	✓	Full octagon search	Full octagon search
5	✓	✓	✓	Multi-octagon-grid search	✓
6	✓	✓	✓	✓	Horizontal and vertical hexagon
7	✓	✓	✓	QBMO	✓
8	✓	Irregular cross search	New 3x3 square search	Multi-octagon-grid search	Horizontal and vertical hexagon
9	✓	Irregular cross search	New 3x3 square search	QBMO	Horizontal and vertical hexagon

TABLE 3.4: Motion Estimation Simulation Time (Met) Percentage Difference As Compared To the Conventional UMHexagonS Algorithm.

Algo-rithm. No.	Features	Bus	Football	Foreman	Silent	News	Hall	Average
1	UMHexagonS	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	Irregular cross search	-0.28	-0.05	0.23	6.37	-0.64	6.39	2.01
3	New 3x3 square search	-1.40	-4.21	-1.64	5.05	-1.07	5.72	0.41
4	Full octagon search	-6.98	-10.16	-1.80	4.43	-1.98	6.23	-1.71
5	Multi-octagon-grid search	-8.06	-11.77	-3.03	3.24	-2.69	5.41	-2.82
6	New extended hexagon grid search	-1.39	-2.27	-1.17	4.75	-1.66	6.27	0.76
7	QBMO	-12.29	-18.21	-5.79	-9.16	-4.39	-5.99	-9.31
8	Combined Algorithm 2,3,5 & 7	-11.82	-17.31	-6.05	0.89	-3.95	3.64	-5.77
9	Combined Algorithm 2,3,6 & 7	-20.45	-28.66	-12.32	-15.16	-9.60	-11.73	-16.32

Table 3.5: PSNR difference as compared to the conventional UMHexagonS algorithm.

Algo-rithm. No.	Features	Bus	Football	Foreman	Silent	News	Hall	Average
1	UMHexagonS	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	Irregular cross search	-0.003	-0.009	-0.003	-0.003	0.005	-0.012	-0.004
3	New 3x3 square search	0.010	-0.004	-0.010	0.005	-0.006	-0.005	-0.002
4	Full octagon search	0.003	-0.014	-0.014	0.001	0.019	-0.011	-0.003
5	Multi-octagon-grid search	0.013	-0.015	-0.022	-0.005	0.008	-0.004	-0.004
6	New extended hexagon grid search	-0.003	-0.010	-0.029	-0.006	0.010	-0.002	-0.007
7	QBMO	0.019	-0.015	-0.012	-0.007	0.002	-0.001	-0.002
8	Combined Algorithm 2,3,5 & 7	0.010	-0.019	-0.010	0.001	0.008	-0.005	-0.003
9	Combined Algorithm 2,3,6 & 7	0.012	-0.015	-0.014	0.002	0.016	-0.008	-0.001

Table 3.6: Bitrate difference as compared to the conventional UMHexagonS algorithm.

Algo-rithm. No.	Features	Bus	Football	Foreman	Silent	News	Hall	Average
1	UMHexagonS	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	Irregular cross search	-0.04	-0.01	-0.01	0.05	-0.06	-0.10	-0.03
3	New 3x3 square search	0.02	0.12	0.34	0.04	-0.17	-0.06	0.05
4	Full octagon search	0.40	0.61	0.21	-0.19	-0.57	0.16	0.10
5	Multi-octagon-grid search	0.78	0.60	0.43	0.21	0.22	0.10	0.39
6	New extended hexagon grid search	0.19	0.16	0.37	0.01	-0.28	0.11	0.09
7	QBMO	1.07	1.21	0.63	0.08	-0.09	0.07	0.49
8	Combined Algorithm 2,3,5 & 7	0.67	0.19	0.64	0.24	-0.21	-0.51	0.17
9	Combined Algorithm 2,3,6 & 7	1.26	0.92	1.43	0.28	-0.48	-0.10	0.55

3.4. Conclusion

This paper discusses the performance comparison of the UMHexagonS motion estimation search algorithm utilizing a quadrant-based multi-octagon search. Based on the experimental results, it is seen that the proposed algorithm is able to reduce the motion estimation time by up to 18.21% it is implemented in step 4 of the UMHexagonS. A further reduction of 28.66% is achieved when combining the algorithm with other algorithms for steps 2, 3, and 5. In the future, this work will be further extended to evaluate the effectiveness of this technique for H.265 motion estimation implementation.

References

[1] A. Jimenez-Moreno, E. Martinez-Enriquez, F. Diaz-de-Maria, "Mode Decision-Based Algorithm for Complexity Control in H.264/AVC", *IEEE Transactions on Multimedia*, vol 15, issue 5, pp. 1094-1109, Aug. 2013

[2] Weiyao Lin, K. Panusopone, D. M. Baylon, Sun Ming-Ting, Chen Zhenzhong, Li Hongxiang, "A Fast Sub-Pixel Motion Estimation Algorithm for H.264/AVC Video Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, vol 21, issue 2, pp. 237-242, 2011.

[3] Jianning Zhang, Yuwen He, Shiqiang Yang and Yuzhuo Zhong, "Performance and Complexity Joint Optimization for H.264 Video Coding", *Proceeding of International Symposium on Circuit an System*, 2003, Vol. 2, pp. 888-891.

[4] Zhou Wei and Zhou Xin, "A fast hierarchical 1/4-pel fractional pixel motion estimation algorithm of H.264/AVC video coding,", *8th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2013, pp.891-895.

[5] H.F. Ates, Y.Altunbasak, "Rate-Distortion and Complexity Optimized Motion Estimation for H.264 Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.18, no.2, pp.159,171, Feb. 2008.

[6] Zhibo Chen and Yun He, "Fast Integer and Fractional Pel Motion estimation", *Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG*, Geneva, 2002.

[7] Arief Effandi Juri and Asral Bahari Jambek, "Improved UMHexagonS Algrithm and Architecture for Low Power H.264 Video Compression", M.Sc Thesis, Universiti Malaysia Perlis, Malaysia, 2013.

[8] Yufeng Li; Jufei Xiao; Wei Wu, "Motion estimation based on H.264 video coding," *5th International Congress on Image and Signal Processing (CISP)*, 2012, pp.104-108.

[9] Xie Lifen, Huang Chunqing and Chen Bihui, "UMHexagonS search algorithm for fast motion estimation," *3rd International Conference on Computer Research and Development (ICCRD)*, 2012, vol.1, no. pp.483 - 487.

[10] Peng Huang and Cui-Hua Li, "Irregularity-cross multi-hexagon-grid search algorithm for fastmotion estimation on H.264", *2nd International Conference on Computer Engineering and Technology*, 2010, vol. 3, pp. 587-592.

[11] Li, H. Y., Liu, M. J., & Zhang, Z. Q. "A New Fast Motion Estimation Algorithm Based on H.264. *International Conference on Multimedia Information Networking and Security*, 2009, vol 1, pp. 287-290,.

[12] Xingyu, W., & Guiju, L.. Optimization on Motion Estimation Algoriithm Based on H264. *3rd International Conference on Advance Computer Theory and Engineering*, 2010, vol. 5, pp. 590-593.

[13] Thomos, N., Boulgouris, N., & Strintzis, M, "Optimized transition of JPEG2000 streams over wireless channels. IEEE Trans. *Image Processing*, vol 1, no 15, pp54 – 67, 2006.