AN ANALYSIS ON THE PERFORMANCE OF FAST BLOCK MATCHING MOTION ESTIMATION ALGORITHMS

Redzuan Abdul Manap[1] **Mazran Esro**[2] **Amat Amir Basari**[3], **Badrul Hisham Ahmad**[4]

[1],[2],[3],[4] Faculty of Electronic and Computer Engineering, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia.

[1]redzuan@utem.edu.my [2]mazran@utem.edu.my [3]amat@utem.edu.my [4]badrulhisham@utem.edu.my

Abstract

To achieve a high compression ratio in coding video data, a method known as Motion Estimation (ME) is often applied to reduce the temporal redundancy between successive frames of a video sequence. One of ME techniques, known as Block Matching Algorithm (BMA), has been widely used in various video coding standards. In recent years, many of these BMAs have been developed with similar intention of reducing the computational costs while at the same time maintaining the video signal quality. In this paper, several fast BMAs, namely Three Step Search (TSS), New Three Step Search (NTSS), Four Step Search (4SS) as well as Diamond Search (DS), are first selected to be implemented onto various type of standard test video sequence using MATLAB before their performances are compared and analyzed in terms of peak signal-to-noise ratio (PSNR), number of search points needed as well as their computational complexity. Simulation results demonstrate that DS algorithm has speed up other algorithm's computational work up to 47.8% on average though at the same time gives poorer performance in terms of average PSNR to others. For PSNR analysis, NTSS algorithm performs the best with up to 3.203 dB improvement on video quality.

Keywords*: Block Match Algorithm; Motion Estimation; Video Coding; Video Compression*

I. INTRODUCTION

Due to limited channel bandwidth and strict requirements of real time video playback, efficient coding of video data in video file formats becomes vital process for many visual communication and multimedia applications and requires a very high compression ratio. To achieve this objective, the large amount of temporal correlation, better known as temporal redundancy, need to be properly identified and eliminated. One effective and popular method to reduce the temporal redundancy between successive frames of a video sequence is called Motion Estimation (ME). One of ME techniques is known as Block Matching Algorithm (BMA). BMA has been widely used in various video coding standards such as ITU-T H.261/263 and MPEG-1/2/4 [1] – [4] and in any motion compensated video coding techniques owing to its high compression efficiency and its simplicity for implementation. Thus, to ensure the processing delay is reduced while at the same time maintaining good image quality, fast and accurate block matching search algorithm is highly desirable.

The full search (FS) algorithm is considered as the simplest BMA. All possible displacements in the search window are exhaustively evaluated to find the best matching block. The fact that the global optimal solution can be achieved using this kind of method proves the advantage of FS algorithm. However, due to its high computational complexity, it is not considered as a good choice for real-time video coding implementation. Thus, many fast BMAs have been proposed and developed in the last two decades, for example, 2-D logarithmic search (TDL) [5], three step search (TSS) [6], cross search (CS) [7], new three step search (NTSS) [8], four step search (4SS) [9], diamond search (DS) [10], cross diamond search (CDS) [11], new cross diamond search (NCDS) [12], cross diamond hexagonal search (CDHS) [13] and directional gradient descent search (DGDS) [14], to reduce the computational cost as much as possible while at the same time making ME accuracy degradation as least as possible. Different search patterns and search strategies are employed in these fast BMAs to find the global optimal solution with reduced number of search points as compared with the FS algorithm.

In this paper, several of these fast BMAs, namely TSS, NTSS, 4SS and DS are proposed to be implemented onto standard test video sequences using MATLAB. Their performances are then compared and analyzed in terms of peak signal-to-noise ratio (PSNR), number of search points needed as well as their computational complexity in order to determine their suitability to different motion content represented in those video sequences.

II. Block Matching Algorithm

In general, BMA is concerned with estimating the amount of motion on a block by block basis. In a typical BMA, each frame is divided into non-overlapping blocks of M×N pixels. The current block of pels is compared with a corresponding block in the previous frame within a search area of size (M+2*p*×N+2*p*)as shown in Fig. 1 where p is the maximum displacement allowed, A is the block in the current frame, B is the block in the previous frame, and *C* is the search window in the previous frame [15]. The displacement between these two corresponding blocks is referred to as the motion vector (MV). It is found by the best match based on a certain matching cost function.

Various measures can be used as the matching cost function such as cross-correlation function (CCF), mean squared difference (MSD) and mean absolute difference (MAD).

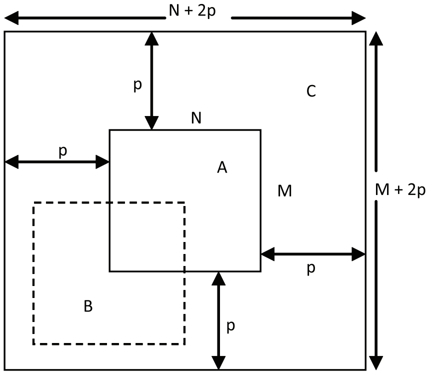

Fig. 1. The current and previous frames in a search window.

In the CCF measurement, the correlation function has to be maximized in order to obtain the best match. However, a good motion tracking would not be obtained using this measurement especially when the displacement was not large [16]. Meanwhile, in MSD and MAD cases, the distortion or matching error between the blocks must be minimized to obtain the best match. The matching function for MSD and MAD respectively are defined as [5]:

$$MSD(i,j) = \frac{1}{MN}\sum_{k}\sum_{l}\left[S_f(k,l) - S_{f-1}(k+i,l+j)\right]^2 \quad (1)$$

$$MAD(i,j) = \frac{1}{MN}\sum_{k}\sum_{l}\left[S_f(k,l) - S_{f-1}(k+i,l+j)\right] \quad (2)$$

Where S_f (k,l) is the location of the pel at the upper most left in the block of the current frame,*f*, and S_{f-1}(k+i,l+j) is the location of the pel on the previous frame, f-1, shifted by the i pels and *j* lines. The best match is represented by the smallest MSD(*i,j*) or MAD(*i,j*) within the search window respectively.

III. Fast Block Matching Algorithms

A. Three Step Search Algorithm

Koga et al.[6] use a three step search procedure to determine MV estimation.

The algorithm is known as TSS algorithm and it is used for displacement computation up to 7 pels/frame. The search procedure for this algorithm is described as follows:

Step 1: 9 checking points centered at the origin of the search window are first searched with a step size of p/2. The check point with the minimum cost function value is selected as the centre point for the second step.

Step 2: Another 8 checking points surrounding the new centre with half the previous step size are checked. Again, the check point with the minimum cost function value is selected as the centre point for the third step.

Step 3: In this last step of the procedure, the step size is again halved. 8 more checking points surrounding the new centre is checked. The MV is given by the position of the point that gives the minimum cost function value in this stage.

B. New Three Step Search Algorithm

Rather than using the uniform distribution as being used in TSS algorithm, which becomes inefficient for small motion estimation, NTSS algorithm emphasis on the use of centre-biased MV distribution, which is one of real world image sequence's characteristics. The search procedure for NTSS differs from TSS by firstly, employing a centre-biased checking point pattern in its first step and secondly, incorporating a halfway-stop technique for stationary or quasi-stationary blocks. The details of the algorithm are given below [8]:

Step 1: In addition to the original checking points used in TSS algorithm, 8 extra points are added, which are the eight neighbours of the search window centre.

Step 2: A halfway-stop technique is used for stationary and quasi-stationary block in order to fast identify and then estimation for these blocks:

If the minimum cost function value in the first step occurs at the search window centre, the search is stopped. This is called the first-step-stop.

If the minimum cost function point in the first step is one of the eight neighbours of the search window centre, the search in the second step will be performed only for another eight neighbouring points surrounding the new minimum cost function point. The search is then stopped. This is called the second-step-stop.

If a) and b) are not the case, the second and the third step of TSS algorithm are implemented.

C. Four Step Search Algorithm

For the maximum displacement of 7 pels/frame, this 4SS algorithm utilizes a centre-biased search pattern with nine checking points on a 5×5 window in the first step instead of a 9×9 window used in TSS and NTSS algorithms. The algorithm can be summarized as follows [9]:

Step 1: A minimum cost function point is found from a nine-checking-points pattern on a 5×5 window located at the centre of the search window. If the point is located at the centre of the search window, go to Step 4; otherwise go to Step 2.

Step 2: The search windowed is maintained in 5×5. However, the search pattern will depend on the position of the previous minimum cost function point.

a. If the previous minimum

cost function point is located at the corner of the previous search window, five additional checking points are used.

b. If the previous minimum cost function point is located at the middle of horizontal or vertical axis of the previous search window, three additional checking points are used.

If the minimum cost function point is found at the centre of the search window, go to Step 4; otherwise go to Step 3.

Step 3: The searching pattern strategy is the same as Step 2, but finally it will go to Step 4.

Step 4: The search window is reduced to 3×3 and the direction of the overall MV is considered as the minimum cost function point among these nine searching points.

D. Diamond Search Algorithm

Instead of typical rectangular shape used in TSS, NTSS and 4SS algorithms, the DS algorithm uses a diamond shape which tries to behave as an ideal circle-shaped coverage for considering all possible directions of an investigating MV. The algorithm is summarized as follows [10]:

Step 1: The large diamond search pattern (LDSP) is centered at the origin of the search window, and the nine checking points of LDSP are tested. If the minimum cost function point calculated is located at the centre, go to Step 3. Otherwise, go to Step 2.

Step 2: The minimum cost function point found in the previous search step is re-positioned as the centre point to form a new LDSP. If the new minimum cost function point obtained is located at the centre position, go to Step 3. Otherwise, recursively repeat this step.

Step 3: Switch the search pattern from LDSP to small diamond search pattern (SDSP). The minimum cost function point found in this step is the final solution for the MV which points to the best matching block.

IV. Experimental Setup

For all simulation work, the block size is fixed at16×16. To make a consistent comparison to previously developed research works in ME, block matching is conducted within a 15×15 search window. In other words, the maximum displacement allowed is set at ±7 in horizontal and vertical directions. Besides that, the full-pixel grid is set for all BMAs concerned and frame distance between the predicted frame and the original frame is set to be 1. MAD, as opposed to MSD, is used as the matching cost function in the search procedure due to the fact it does not require multiplication operation, hence further reducing the block matching computational requirements.

To be in agreement with previously developed research works in ME, video sequences selected for the analysis are the standard sequences used in previous works. Well-known and commonly used test sequences are selected as listed in Table I.

V. Simulation Results

Although the number of frame to be simulated in the source code is specified by the user, only the first 25 frames of the available test video sequences is considered to be simulated and analyzed in this paper. This is done to present much easier-to-understand analysis for the purpose of the novice readers.

Table II compares the performance of the selected fast BMAs in terms of average number of search points required to obtain MV per block per frame while the average PSNR per block per frame for these fast BMAs is given in Table III.

Table II.
Average Number Of Search Points Per Block Per Frame

Video	Algorithm				
	FS	4SS	DS	NTSS	TSS
Akiyo	225	17	13	17.005	25
News	225	17	13.091	17.315	25
Salesman	225	17.051	13.127	17.533	25
Coastguard	225	17	13	23.806	25
Tennis	225	19.114	13.538	19.067	25
Foreman	225	19.926	17.211	22.796	25

Table III.
Average Psnr Per Block Per Frame

Video	Algorithm				
	FS	4SS	DS	NTSS	TSS
Akiyo	45.012	45.012	44.968	45.012	45.012
News	37.932	37.924	37.294	37.932	37.899
Salesman	37.558	37.540	36.664	37.546	37.541
Coastguard	32.404	32.404	29.903	32.404	32.404
Tennis	31.929	31.073	26.716	31.567	31.434
Foreman	28.881	28.794	27.314	28.868	28.775

Based on Table II, DS algorithm greatly outperformed FS, TSS, NTSS and 4SS algorithms in terms of search points required in determining the MV of the block. For small-motion sequences like "Akiyo" and "News", an average of 23.3%, 24.0% and 47.8% speed improvement is achieved by DS algorithm over 4SS, NTSS and TSS algorithm respectively. For moderate-motion sequences like "Salesman" and "Coastguard", an average of 23.3% 35.3% and 47.8% speed improvement is achieved by DS algorithm over 4SS, NTSS and TSS algorithm respectively. For relatively higher degree of motion sequences such as "Foreman" and "Tennis", DS algorithm achieves an average of 21.4%, 26.8% and 38.5% speed improvement compared to 4SS, NTSS and TSS algorithm. Thus, it can be said that the average searched points per block with the observation DS < 4SS < NTSS < TSS < FS is manifested for the video sequences with maximum displacements of $p=\pm 7$.

From the result tabulated in Table III, it can be shown that NTSS algorithm gives the closest performance to FS algorithm in terms of video quality when compared to other fast BMAs, regardless the motion content. It is followed by either TSS or 4SS algorithms while slight degradation in quality is produced by DS algorithm. On average, up to 0.341 dB video quality improvement is achieved by NTSS algorithm for small-motion sequences while for moderate-motion sequences, the achieved improvement is up to 1.692 dB. For large-motion sequences, NTSS algorithm achieves a quality improvement of up to 3.203 dB.

To gain further insights, the average number of search points per block and the average PSNR value per block are plotted on frame-by-frame basis for "Salesman" and "Tennis" video sequences as shown in Fig. 2, Fig.3, Fig.4 and Fig.5. Both sequences are chosen to represent the performance of the algorithms when dealing with small-motion and large-motion sequences respectively.

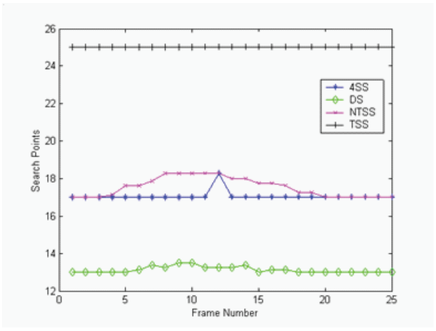

Fig. 2. Average search points per block per frame for "Salesman" sequence.

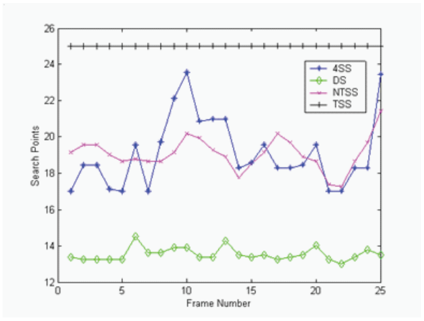

Fig. 3. Average search points per block per frame for "Tennis" sequence

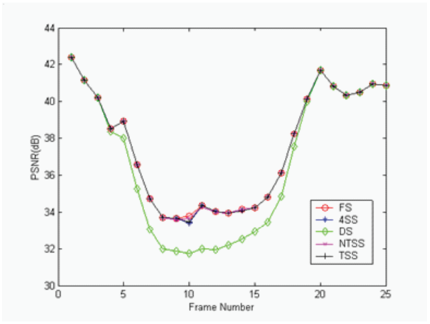

Fig. 4. Average PSNR per block per frame for "Salesman" sequence.

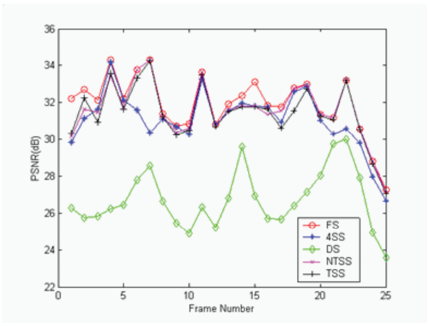

Fig. 5. Average PSNR per block per frame for "Tennis" sequence.

From observations made on Fig.2 and Fig.3, the superiority of DS algorithm to other fast BMAs in terms of number of search points needed is clearly illustrated. Meanwhile, for PSNR analysis, all the selected algorithms perform considerably well with NTSS algorithm gives the closest PSNR value to FS algorithm.

VI. CONCLUSION

In this paper, several well-known fast BMAs are selected to be implemented onto several types of standard test video sequences in order to determine their suitability to different motion content represented in those video sequences. Simulation results show that DS algorithm is the best among the selected algorithms in terms of the number of required search points, where it sped up other algorithms computational work up to 47.8% regardless the motion content of the sequences. At the same time, the results also show that NTSS algorithm has constantly outperform other algorithms in terms of PSNR analysis with up to 3.203 dB improvement on video quality.

VII. ACKNOWLEDGMENT

The authors would like to thank Universiti Teknikal Malaysia Melaka (UTeM) for their sponsorship through Short Term Grant (STG) scheme that makes this research work possible.

VIII. REFERENCES

Video Codec for Audiovisual Services at p ×46 kbit/s, ITU-T Recommendation H.261, March 1993.

Video Coding for Low Bit Rate Communication, ITU-T Recommendation H.263, February 1998.

Information Technology – Coding of Audio Visual Objects – Part 2: Visual, ISO/IEC 14 469-2 (MPEG-4 Visual), 1999.

K. R. Rao and J. J. Hwang. 1996.Techniques and Standards for Image, Video and Audio Coding. Englewood Cliffs, NJ: Prentice Hall,

H. Gharavi and M. Mills, 1990. Blockmatching motion estimation algorithms – New results, IEEE Trans. Circuits and Systems, vol. 37, pp. 649-651.

T. Koga, K. Iimuna, A. Hirano, Y. Iijima, and T. Ishiguro, 1981. Motion compensated interframe coding for video conferencing, in Proc. National Telecommunications Conf., vol. 4, New York, pp. G5.3.1-G5.3.5.

M. Ghanbari, 1990. The cross-search algorithm for motion estimation, IEEE Trans. Commun., vol. 38, pp. 950-953.

R. Li, B. Zheng, and M. L. Liou, 1994. A new three-step search algorithm for block motion estimation," IEEE Trans. Circuits Syst. Video Technol., vol. 4, pp. 438-443.

L. M. Po and W. C. Ma, 1996. A novel four-step search algorithm for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol., vol. 6, pp 313-317.

S. Zhu and K. K. Ma. 2000. A new diamond search algorithm for fast block-matching motion estimation, IEEE Trans. Image Processing, vol. 9, pp. 287-290.

C. H. Cheung and L. M. Po, 2002. A novel cross-diamond search algorithm for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol., vol. 12, pp. 1168-1177.

H. Jia and L. Zhang. 2004. A new cross diamond search algorithm for block motion estimation," in Proc. IEEE International Conf. on Acoustics, Speech and Signal Processing, vol. 3, pp. 357-360.

C. H. Cheung and L. M. Po, 2005. Novel cross diamond hexagonal search algorithm for fast block motion estimation, IEEE Trans. Multimedia, vol. 7, pp. 16-22.

L. M. Po, K. H. Ng, K. W. Cheung, K. M. Wong, and C. W. Ting, 2009. Novel directional gradient descent searches for fast block motion estimation, IEEE Trans. Circuits Syst. Video Technol., vol. 19, pp. 1189-1195.

R. Srinivasan and K. R. Rao, 1985. Predictive coding based on efficient motion estimation," IEEE Trans. Commun., vol. COM-33, pp. 888-896.

J. R. Jain and A. K. Jain. 1981 Displacement measurement and its application in interframe image coding," IEEE Trans. Commun., vol. COM-29, pp. 1799-1808.