A NEW ALGORITHM FOR FAST BLOCK-MATCHING MOTION ESTIMATION BASED ON FOUR-NEIGHBORHOOD BLOCKS

Ibrahim Nahhas

Doctoral Degree Programme (2), FIT BUT

E-mail: xnahha00@stud.fit.vutbr.cz

Supervised by: Martin Drahansky

E-mail: drahan@fit.vutbr.cz

Abstract: Block matching for motion estimation has been widely used in video compression for efficient transmission and storage of video data. The motion estimation is a process which determines the motion between two frames of a video signal. This paper presents a new algorithm for fast block matching algorithm based on Four-neighborhood search (FNS), this algorithm can significantly speed up the computation of the block matching by reducing the number of checked points. Theoretically has been shown that 88% to 92% of operations can be saved while maintaining the quality of video.

Keywords: Motion Estimation, Block Matching, Four-neighborhood Search.

1. INTRODUCTION

Image and video compression was and still is a very active field of research and development for over 20 years and many different systems and algorithms for compression and decompression have been proposed and developed. Video compression algorithms operate by removing redundancy in the temporal, spatial and/or frequency domains and the goal of the motion estimation and compensation is to reduce temporal redundancy between transmitted frames [1]. Changes between video frames may be caused by object motion, camera motion and lighting changes. It is possible to estimate the trajectory of each pixel between successive video frames, producing a field of pixel trajectories known as the optical flow. However, this is not a practical method of motion compensation for several reasons. An accurate calculation of optical flow is very computationally intensive and it would be necessary to send the optical flow vector for every pixel to the decoder. To have a practical method of motion estimation we divide the frame into non-overlapped, equally paced, fixed size small rectangular sections, called „blocks“, and determine all pixels inside the block to have the same vector motion to compensate the movement of „blocks“ of the current frame. In contrast, the block matching technique is simple, straightforward, and very efficient yet. It has been by far the most popularly utilized motion estimation technique in video coding. In fact, it has been adopted by all the international video coding standards: ISO MPEG-1 and MPEG-2, and ITU H.261, H.263 and H.264.

2. BLOCK MATCHING

The block matching motion estimation technique based on segmentation the current frame into blocks and the determination of all pixels inside the block have the same displacement vector, which was estimated by finding its best-matched counterpart in the previous frame. The block size needs to be chosen properly. In general, the smaller the block size, the more accurate, but leading to more motion vectors to be estimated and encoded, which means an increase in computation and side information. As a compromise, a size of 16×16 pixels is considered to be a good choice – this has been specified in the international video coding standards such as H.261, H.263 and MPEG-1, MPEG-2 [2][3]. The Figure 1 illustrates the principle idea of block matching technique, where segment an image frame at the moment t_n into non-overlapped $p\times q$ rectangular blocks. Consider one of the blocks centered at (x, y). It is assumed that the block is translated as a whole. Consequently, only one displacement vector needs to be estimated for this block. In order to estimate the displacement vector, a rectangular search window is opened in the frame t_n-1 and centered at the pixel (x, y) as in Figure 2, than a rectangular correlation window of the same size $p\times q$ is opened with the pixel located in its center. A certain type of similarity measure (correlation) is calculated. After this matching process has been completed for all candidate pixels in the search window, the correlation window corresponding to the largest similarity becomes the best match of the block under consideration in frame t_n and there are various cost functions to calculate best matching, computationally expensive is Mean Absolute Difference (MAD) [9] given by equation (1), Mean Squared Error (MSE) [2] given by equation (2) and Sum of Absolute Difference (SAD) [4] given by equation (3).

$$MAD = \frac{1}{N^2}\sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\left|C_{ij} - R_{ij}\right| \qquad (1)$$

$$MSE = \frac{1}{N^2}\sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\left(C_{ij} - R_{ij}\right)^2 \qquad (2)$$

$$SAD = \sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\left|C_{ij} - R_{ij}\right| \qquad (3)$$

Where N is the side of the macro bock, C_{ij} and R_{ij} are the pixels being compared in current macro block and reference macro block, respectively. The relative position between these two blocks (the block and its best match) gives the displacement vector. In this paper, we will focus on Four-neighborhood search and compare it with 5 different types of block matching algorithms varying in number of positions searched and computational cost of the matching error.

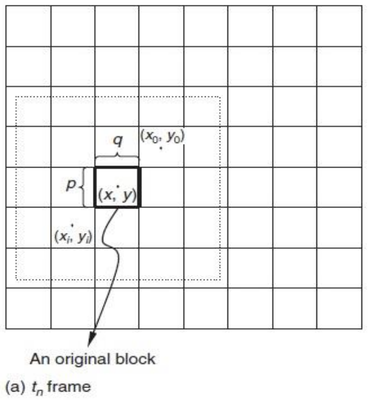

Figure 1: Principle idea of block matching.

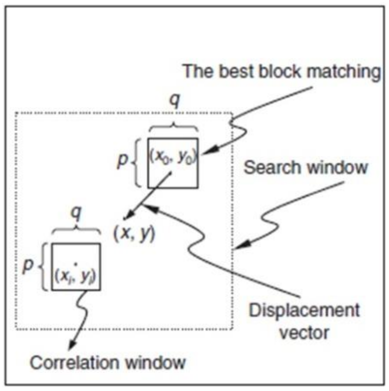

Figure 2: Matching process.

2.1. Block matching Search Algorithms

Full Search [10] algorithm is the first and simplest algorithm for block matching. In an exhaust searching for the best matching, the correlation window is moved to each candidate position within the search and the minimum dissimilarity gives the best matching. Full search algorithm provides the highest PSNR but at the same time suffers from long computational time so needs an improvement with maintaining the same PSNR and there are number of block matching algorithms which have been developed to accelerate the block matching process to reduce the search time which poses great challenge for real-time codec implementation. We classify these techniques into three categories:

- Partial Search Set techniques reduce the number of the searched points.
- Partial-Matching-Error techniques reduce the computational cost of the matching error for each search points.
- Hybrid techniques are combination of first and second categories to further improve the efficiency of search techniques.

3. FOUR-NIEGHBORHOOD SEARCH (FNS)

The basic idea behind the proposed (FNS) relies on reducing the number of checked points within macro- block using simple search strategy depends on calculating the weight between the values of centered point with the four points surrounding it and compare the result with two values the first one is the ideal value and when cost function gives this value the searching procedure stops while the second one is the updated value which has initial value from dissimilarity from the same centered point at previous and actual frame and change during the process depending on the better matching every step. NFS also depends on center biased searching and sets a variable pattern size of steps starting from one in first step and increase by one only and only if the centered point achieves the better matching relative to surrounding points. The number of steps varies depending on the type of motion activity of video and ranges from 1 to no limits steps and the number of checked points depends also on type of motion activity of video and ranges from 5 to no limit. The cost function used in FNS is SAD criteria [4] given by equation (3). We summarize the algorithm first step calculate dissimilarity between central point and its four surrounding points with distance equal one now we have three possibilities: If the central point or one the surrounding points achieve the ideal value of weight we stop the searching directly. If the central point has the better value of weight we save the same central point with increase the size of distance by one and repeat calculating the weight with new distance while if one of surrounding points has the better value of weight we change the central point to it and repeat the procedure from beginning. We repeat this operation eight times or even have a best matching. Figures 3 till 6 illustrate the algorithms with different scenarios. FNS algorithm can significantly speed up the computation of the block matching, theoretically has been shown that 88% to 92% of operations can be saved while maintaining the quality of video which should be close to that of Full Search and also FNS is very effective in real time applications because it is very simple and saves much of computational time.

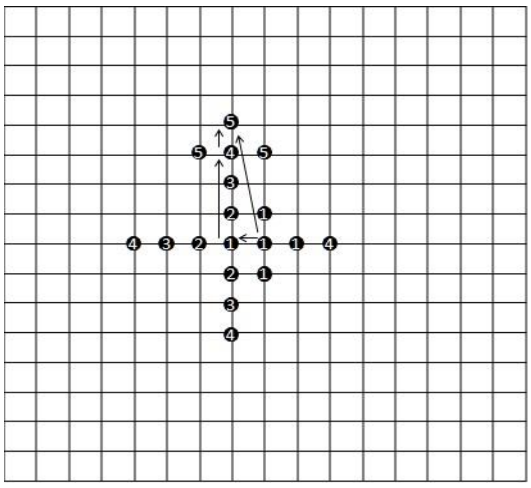
Figure 3: FNS – Scenario 1

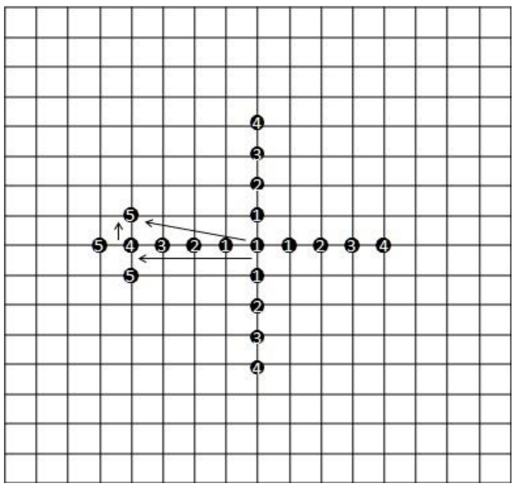
Figure 4: FNS – Scenario 2

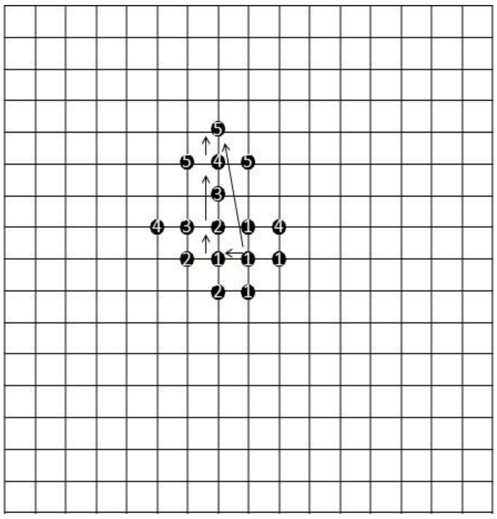
Figure 5: FNS – Scenario

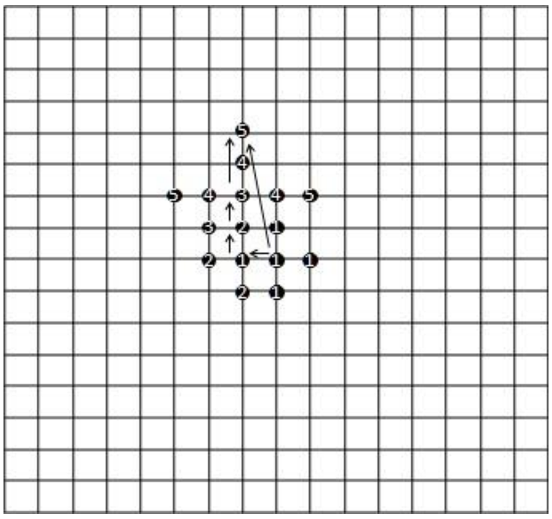
Figure 6: FNS – Scenario

One problem that occurs with the FNS is that it uses a small size of step in the first round, which becomes inefficient for large motion estimation and needs more time to get a best matching so it is excellent performance for low motion activity video and good performance for high motion activity video.

4. RESULTS

To assess the performance of the proposed FNS algorithm, we compared these algorithms with other fast search algorithms (Full Search [10], Three Step Search [5], New Three Step Search [6], Four Step Search [7], Diamond Search [8]). The comparisons of fast search algorithms was based on applying the different algorithms on the scenario with typical size of block 16×16 pixels and maximal movement 8 to find the best matching and compare between performances of algorithms. The Table 1 indicates statistically the PSNR difference (ΔPSNR) between that of Full Search and that of a fast search algorithm and the average number of searches required per macro block (SP) using the six block matching algorithms and also speed up of all 6 fast full algorithms relative to full search algorithm and all results.

Table 1: Results of BMAs in terms of ΔPSNR (DB) and number of SPs per block.

	FS	TSS	NTSS	FSS	DS	FNS
ΔPSNR	0.00	-0.24	-0.26	-0.29	-0.25	-0.23
SP	225.00	25.00	24.00	22.00	20.00	17.50
Speedup	1.00	9.00	9.37	10.22	11.25	12.85

During the course of this project all of the above 6 algorithms have been implemented, using luminance popular video sequences of 50 frames using CIF formats (352×288) with large motion activity ”Stefan” video sequence. Table 2 indicates maximum value of PSNRs of different search algorithms for “Stefan” video for the first 50 frames, average number of searched positions using the 6 block matching algorithms and speed up of all 6 algorithms.

Table 2: Results using the “Stefan” video sequence.

	FS	TSS	NTSS	FSS	DS	FNS
PSNR	20.989	20.710	20.660	20.614	20.619	20.630
SP	225.000	23.029	21.106	20.642	19.035	17.50
Speedup	1.000	9.770	10.660	10.900	11.820	12.85

5. CONCLUSION

A Four-neighborhood search algorithm (FNS) was proposed. This algorithm significantly speeds up the motion estimation procedure and substantially decreases the checked points and computational time, when compared with fast search algorithms, still providing similar quality performances. As a consequence, it was proven as to be specially suited to be implemented in most embedded systems with restricted computational requirements, i.e. it is often adopted by portable devices and for real time applications.

AKNOWLEDGEMENTS

This research has been realized under the support of the following grants: “Security-Oriented Research in Information Technology” – MSM0021630528 (CZ), “Tools and Methods for Video and Image Processing for the Fight against Terrorism” – MV VG20102015006 (CZ) and “The IT4Innovations Centre of Excellence” – IT4I-CZ 1.05/1.1.00/02.0070 (CZ).

REFERENCE

[1] Iain E.G. Richardson, Video Coding Concepts, in H.264 and MPEG-4 Video Compression, Essex, 2003, England, ISBN 0-470-84837-5.

[2] A. Barjatya, Block Matching Algorithms For Motion Estimation, Dept. of Electrical Engineering, Utah State University, Utah, DIP, 2004.

[3] D. V. Manjunatha et al, Comparison And Implementation Of Fast Block Matching Motion Estimation Algorithms For Video Compression, International Journal of Engineering Science and Technology, Vol. 3, No. 10, p. 5, 2011.

[4] Z. Ahmed et al., Fast Computations of Full Search Block Matching Motion Estimation (FCFS), PGNeT Conference, 2011.

[5] T. Koga et al., Motion compensated interframe image coding for video conference, Proceedings of NTC8, 1981.

[6] R. Li, B. Zeng and M.L. Liou, A new three-step search algorithm for block motion estimation, IEEE Transactions of Circuits and Systems for Video Technology, Vol. 4, No. 4, pp. 438–442, 1994.

[7] L.M. Po and W.C. Ma, A Novel Four-Step Search Algorithm for Fast Block Motion Estimation, IEEE Transactions of Circuits And Systems For Video Technology, Vol. 6, No. 3, pp. 313-317, 1996.

[8] S. Zhu and K.K. Ma, A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation, IEEE Transactions of Image Processing, Vol. 9, No. 2, pp. 287-290, 2000.

[9] C.M. Lin and S.C. Kwatra, Motion compensated interframe color image coding, International Conference on Communications, Amsterdam, Part 1, pp. 516 –520,1988.

[10] M. Ahmadi and M. Azadfar, Implementation of fast motion estimation algorithms & comparison with full search method in H.264, IJCSNS International Journal of Computer Science & Network Security, Vol. 8, No. 3, pp. 139-143, 2008.