Efficient Search Algorithm for Fast Motion Estimation

Dong-Min Park*, Tong-Ill Kwak, Bo-Hyun Hwang, Jong-Ho Yun and Myung-Ryul Choi
Dept. of EECI, Hanyang Univ., 1271, Sa 1-dong, Ansan, Gyunggi-do, 425-791, Korea
Phone: +82-31-400-4036, E-mail: dia0620@asic.hanyang.ac.kr

Keywords : motion estimation, fast search, frame difference

Abstract

Block-matching motion estimation plays an important role in video coding. In this paper, we propose an Efficient Search Algorithm for Fast Motion Estimation. The proposed algorithm detects motion variation for reducing computational complexity before determining motion vector. Experimental results show that the proposed algorithm has good performance than conventional algorithms through quantitative evaluation.

1. Introduction

Motion Estimation (ME) techniques are widely used in video compression because it is important technique to remove temporal redundancy between successive frames of an image sequence. Motion estimation performs up to 50% of the computations encountered in the entire coding system and is commonly remarked as the computationally most intensive part of the video coding system [1].

The most popular technique for motion estimation is Block Matching Algorithm (BMA) [2]. For each block of luminance samples in the current frame, the motion estimation algorithm searches a neighboring area of the reference frame for a 'matching' $N \times N$ area. The best match is the one that minimizes the energy of difference between the current $N \times N$ block and the matching $N \times N$ area.

The optimal solution for BMA is the Full Search (FS) algorithm that is necessary to carry out a comparison of the current block with every possible region of the search window. The Mean Absolute Difference (MAD) or Sum of Absolute Difference (SAD) matching criteria are considered to be statistically optimal solutions to the matching process.

The FS motion estimation process of an image frame divided into $N \times N$ blocks, is defined as

$$SAD^{t}(u,v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|MB^{t}(i,j) - MB^{t-1}(i+u, j+v)\right| \quad (1)$$
$$-s \le u, v \le s-1$$

Where SAD (u, v) is computed criteria between current block and reference block within search range. $MB^{t}(i, j)$ and $MB^{t-1}(i, j)$ are luminance value of the pixel in the current and reference frames respectively. s determines search range.

The Motion Vector (MV) is defined as

$$MV = \arg \min_{-s \le u, v \le s-1} SAD(u,v) \quad (2)$$

Where MV is selected minimum SAD (u, v) within search range.

Many fast search algorithms have proposed to reduce the computational complexity of the FS algorithm. The two-dimensional logarithmic search algorithm [3], diamond search (DS) [4], three-step search (TSS), new three-step search (NTSS) [5] algorithms are among the class of fast search algorithms by reducing the number of search points in the process of block motion estimation.

In this paper, we propose an Efficient Search Algorithm for Fast Motion Estimation for low computational complexity. This paper is organized as follows. In Sec.2, the proposed algorithm is represented. Experimental results and conclusions are given in Sec.3, and 4, respectively.

2. The proposed algorithm

The major objective of the proposed algorithm is for reducing computational complexity. The proposed algorithm divides into two parts through detecting of motion variation. One part of no motion variation extracts *Motion Vector* (*MV*) without an operation of determining *MV*. The other part of having motion

variation has four processing steps for determining *MV*. Detecting motion variation uses frame difference. The flow chart of the proposed algorithm is shown by Figure 1.

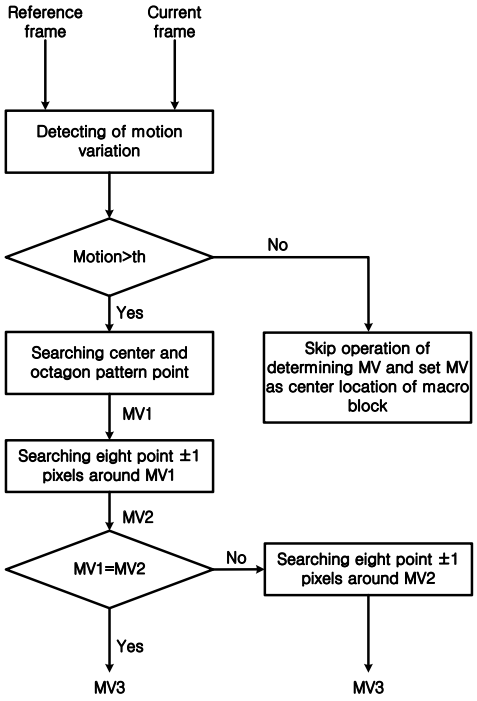


Fig. 1. The flow chart of the proposed algorithm

A. Detecting motion variation

The proposed algorithm is carried out detecting motion variation value using Sum of Difference Value (*SDV*). *SDV* is calculated sum of difference between current macro block and reference macro block. The reason of using *SDV* is simple for detecting motion variation. *SDV* is defined as

$$SDV^{t} = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|MB^{t}(i,j) - MB^{t-1}(i,j)\right| \quad (3)$$

$$SDV_{mean} = \frac{\left(SDV_{current} + SDV_{left} + SDV_{upperleft} + SDV_{upper} + SDV_{upperright}\right)}{5} \quad (4)$$

SDV decides whether current block is motion variation block or not. SDV_{mean} will be calculated mean of SDV^{t} with neighboring blocks which are left block, upper block, upper left block and upper right block because motion variation is highly correlated neighboring blocks. SDV_{mean} is compared with threshold (*th*). If SDV_{mean} is smaller than *th* this block is block of no motion variation. If SDV_{mean} is bigger than *th* this block is block of motion variation.

B. Extracting Motion Vector

The proposed algorithm uses *SAD* (*u*, *v*) for extracting motion vector. *MV* is selected minimum *SAD* (*u*, *v*) within search range. *SAD* (*u*, *v*) and MV are defined as the equation (1) and the equation (2) respectively.

This algorithm divides into two parts using SDV_{mean} for reducing computational complexity. If SDV_{mean} is smaller than threshold, *MV* is determined center location without an operation which extracts *MV*. If SDV_{mean} is bigger than threshold, *MV* is extracted from four processing steps. The proposed algorithm is proposed reduced search range for reducing computational complexity. This search range is shown by Figure 2.

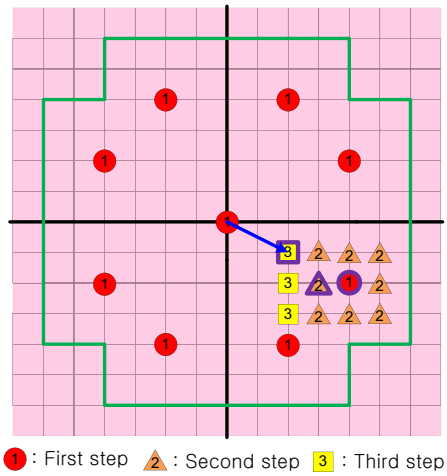


Fig. 2. Search pattern of the proposed algorithm

Boundary of line of plus shape is represented reduced search range. We can verify that this search range includes almost *MV* [6].

Four processing steps are shown by Figure 1 and Figure 2. Four processing steps in the proposed algorithm calculate SAD among center and octagon point in first step. MV1 is extracted from minimum SAD of the first step. MV2 is predicted by minimum SAD of the second step among eight point ±1 around MV1. If MV1 and MV2 are same location in third step, MV1 of first step is determined as final vector. If the position of MV1 and MV2 is not same, second step is performed around MV2. MV3 of fourth step is determined as final motion vector.

3. Experimental result

In this section, we present the experimental results of the proposed algorithm compared with various conventional algorithms.

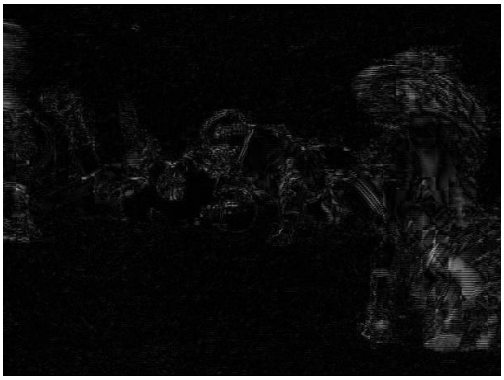
TABLE 1. Average PSNR and average complexity of search point each block

	Football (640×480)		*Salesman* (640×480)		*Flower Garden* (640×480)		*Ice* (720×480)	
	PSNR(dB)	Complexity of Search Point(%)	PSNR(dB)	Complexity of Search Point(%)	PSNR(dB)	Complexity of Search Point(%)	PSNR(dB)	Complexity of Search Point(%)
FS	22.51	100	36.62	100	24.21	100	27.61	100
TSS	19.67	11.11	34.25	11.11	16.71	11.11	23.28	11.11
DS	22.48	10.54	35.7	6.05	16.77	9.0	23.1	11.68
Proposed	22.55	4.72	35.8	0.9	21.3	6.45	26.7	2.4

TABLE 1 shows simulation results in test sequences. The experimental set up is as follow. The distortion measurement of Sum of Absolute Difference (SAD) is used for a block size of 16 × 16, and search window size of ±7. Four standard sequences "*Football*" (640×480, 30 frames), "*Salesman*" (640×480, 30 frames), "*Flower Garden*" (640×480, 30 frames) and "*Ice*" (720×480, 30 frames) are evaluated, which vary in motion content.

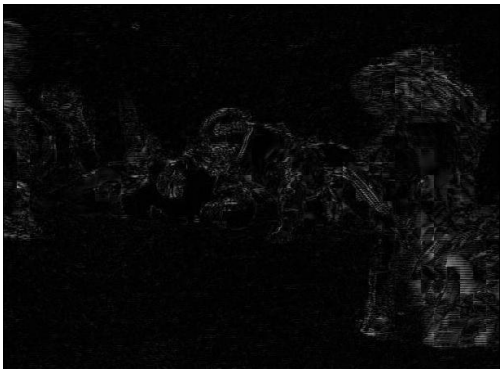
Experimental results are shown that the proposed algorithm reduces average computational complexity than Diamond Search (DS) about 5.82% when the proposed algorithm and DS are compared with Full Search (FS) in *Football* sequence. The proposed algorithm achieves similar PSNR with conventional algorithms.

Fig. 3 represents residual energy of 5th frame in football sequence. We verify that the proposed algorithm is good performance than Three Step Search (TSS) in visual evaluation. And the proposed algorithm has similar result with Diamond Search (DS) and Full Search (FS) in visual evaluation.

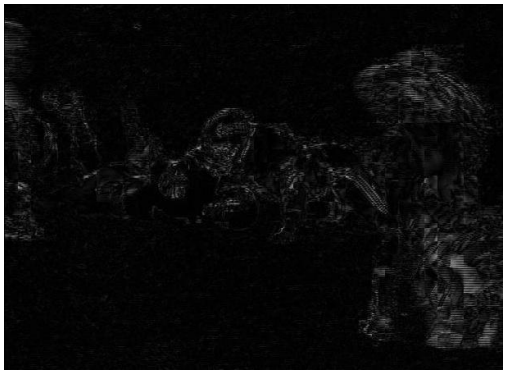


(a) Full Search

(b) Three Step Search



(c) Diamond Search



(d) The proposed algorithm

Fig. 3. Residual frame of football

4. Conclusion

In this paper, we propose an Efficient Search Algorithm for Fast Motion Estimation for reducing computational complexity. We verify that the proposed algorithm has good performance in quantitative and visual evaluation. This algorithm is also simple and efficient in motion estimation. Therefore, it is possible to achieve efficiently implementation of hardware.

5. References

1. Iain E.G. Richardson “VIDEO CODEC DESIGN", JOHN WILEY & SONS, LTD, pp. 93-101, 2002.
2. L.K. Liu and E. Feig, “A block-based gradient descent search algorithm for block motion estimation in video coding,” IEEE Trans. Circ. Syst. and Video Technol., vol. 6, pp. 419-422,Aug. 1996.
3. J. R. Jain and A. K. Jain, “Displacement measurement and its application in interframe image coding,” IEEE Trans. Commun., vol. COM-29, pp. 1799–1808, Dec. 1981.
4. S. Zhu and K. K. Ma, “A new diamond search algorithm for fast blockmatching motion estimation,” IEEE Trans. Image Processing, vol. 9, pp. 287–290, Feb. 2000.
5. R. Li, B. Zeng, and M. L. Liou, “A new three-step search algorithm for block motion estimation,” IEEE Trans. Circuits Syst. Video Technol., vol. 4, pp. 438–442, Aug. 1994.
6. C. H. Cheung and L. M. Po, “Novel cross-diamond-hexagonal search algorithms for fast block motion estimation,” IEEE Trans. Multimedia, vol. 7, no. 1, pp. 16–2, Mar. 2005.