COMPARATIVE ANALYSIS ON THE PERFORMANCE OF BLOCK MATCHING MOTION ESTIMATION ALGORITHM

[1] MS. A. P. CHAUHAN, [2] PROF. R. R. PARMAR, [3] PROF. S. K. PARMAR, [4] PROF. S. G. CHAUHAN

[1]M.E [Computer Eng] Student, Department Of Computer Engineering, Atmiya Institute of Technology and Science, Rajkot, Gujarat
[2, 3] Asst.Professor, Department Of E & C Engineering, G H Patel College of Engineering & Technology, V.V.Nagar, Gujarat
[4] Asst.Professor, Department Of Computer Engineering, Atmiya Institute of Technology and Science, Rajkot, Gujarat

ankitachauhan2008@gmail.com, rohitrparmar@gmail.com, shankarparmar@gcet.ac.in, sgchauhan@aits.edu.in

***ABSTRACT*:** *The essential requirement of multimedia communication is to achieve high processing speed and a low Computing time simultaneously without loses in video quality. So, the video compression has become an Important for efficient storage and transmission of digital signal in multimedia. It involves video coding exploits the high temporal correlation between successive frames to improve coding efficiency, which is usually achieved by using motion estimation (ME) and motion compensation techniques. Block Matching Motion Estimation is one of the ME technique that has been widely used in various video coding standard. Here this paper presents comparative study of four block matching motion estimation algorithms, namely Exhaustive Search (ES), New Three Step Search (NTSS), Four Step Search (4SS) and Diamond Search (DS) algorithms. All four algorithms are compared based on average number of search points per macro block and their PSNR performance.*

Keywords—Motion Estimation, Motion Compensation, Block matching, ES, NTSS, 4SS, DS

I: INTRODUCTION

Video Compression has become a most popular area of research based on the recent advances of technology and its demand.Video Compression uses in many applications such as DVD-Video, mobile TV digital television, videoconferencing and internet video streaming. Digital Video has large amount of temporal correlation, known as temporal redundancy which must be exploited to be stored and transmitted efficiently. One most efficient and popular technique to reduce the temporal redundancy between adjacent frames of a video sequence is called motion estimation (ME). In ME technique, the current frame is predicted from a previous frame known as reference frame by using motion vectors (MV). Motion estimation has proven to be the most computationally intensive technique for exploiting the temporal redundancy in video sequences. So, it is an essential element of MPEG based compression standards. There are a variety of technique have been proposed by many researchers for motion estimation. Block matching motion estimation (BMME) is mostly selected as ME technique in video coding and implemented in most existing video coding standards because of its simplicity and good performance. Here, we want to make a review and comparative analysis for BMME algorithm namely ES, NTSS, 4SS and DS.

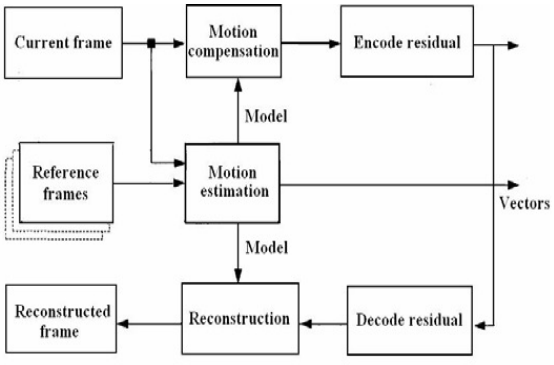

Fig.1 Motion estimation and compansastion block diagram [1].

II: MOTION ESTIMATION AND MOTION COMPENSATION

Motion estimation creates a model of the current frame based on one or more previously encoded

frames which are known as reference frames. These reference frames may be past frames or future frames. The goal for creating a motion estimation algorithm are to model the current frame as accurately as possible that gives better compression performance with acceptable computational complexity. The block diagram of motion estimation and compensation is shown in Figure 1.

The motion estimation creates a model by modifying reference frames to match the current frame as closely as possible (by using a matching criterion). The current frame is motion compensated by subtracting the model from the frame to produce a motion-compensated residual frame. This is coded and transmitted, along with the information required like a set of motion vector for the decoder to recreate the model. At the same time, the encoded residual is decoded and send to the model to reconstruct a decoded copy of the current frame (which may not be same to the original frame because of some coding losses). This reconstructed frame is stored to be used as a reference frame for further frame predictions.

III: BLOCK MATCHING MOTION ESTIMATION

In block matching motion estimation technique[8], frames are divided into non overlapped, equally sized rectangular blocks of N x N pixels. The current block in the current frame is matched against reference blocks within a search area of size (N+2P N+2P) on the reference frame as shown in Figure 2. Where P is maximum motion displacement allowed. These reference blocks are just the displaced versions of current block. The displacement between the current block and reference block are recorded as motion vector (MV). It is found by best matching block based on certain cost function.

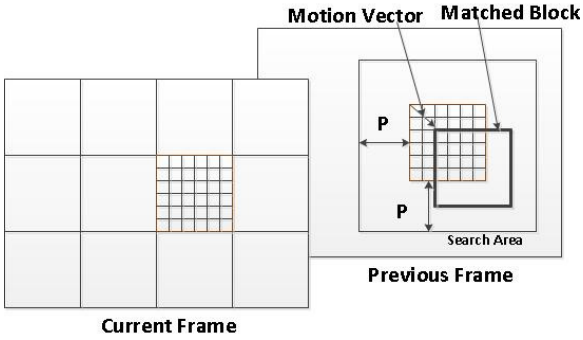

Fig.2 Block Based motion estimation

Various measures can be used as matching cost function but the most popular and less computationally expensive is Mean Absolute Difference (MAD) and Mean Squared Error (MSE) given by equation (i) and (ii) respectively.

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

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

Where size of the macroblock is N x N, C_{ij} and R_{ij} are the pixels being compared in current macro block and reference macro block, respectively. The PSNR of the motion compensated images that are created by using motion vectors and macro clocks from the reference frame given by equation (iii).

$$PSNR = 10\ Log_{10}\left(\frac{(\text{Peak to Peak value of original data})^2}{MSE}\right) \qquad (iii)$$

A. Exhaustive Search

Exhaustive Search [3] is also known as Full Search (FS) The algorithm calculates the cost function at each possible location in the search window. Consider a block of N × N pixels and a search window having a range ±w in both directions in the references frame. There are total $(2w + 1)^2$ search position that need to be examined and finding the best matching block, along with the motion vector is determined by minimum dissimilarity of comparing blocks. As a result of which it finds the best possible match with good accuracy in searching and gives the highest PSNR compare with other BMA but it is the most computationally expensive BMA of all. The drawback of this algorithm is that larger the search Window requires more numbers of computations.

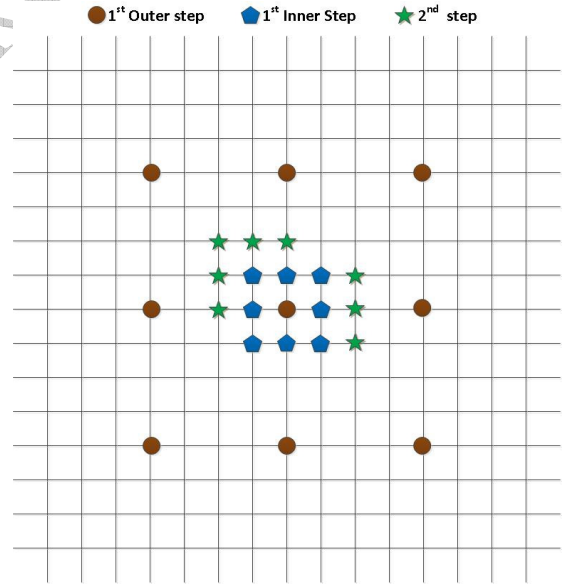

Fig.3 NTSS Search Procedure.

B. New Three Step Search Algorithm:

A new three step search (NTSS) algorithm [5] differs from Three step search (TSS [3]), by employing a center Biased checking point scheme in its first step and secondly incorporating a halfway-stop technique for stationary or quasi-stationary blocks to reduce the computational complexity. This algorithm has the best case of 17 checking points and worst case of 33 checking points. The NTSS searching process are given below and shown in Figure 3.

Step 1: A minimum cost function is found from 8 checking points at 'step size' S = 4 and the other 8 are at S = 1 away from the search origin.
Step 2: A Second step employing a halfway-stop technique for stationary or quasi-stationary blocks.

a) If the minimum cost function value is found at the center of the search window, the search is stop.
b) If the minimum cost function point is at any one of the 8 locations at S = 1, then we change the origin of the search to that point and check for minimum cost function point adjacent to it. The location that gives minimum cost function is the closest match and motion vector is set to that location.
c) If minimum cost function is not find in case (a) and (b), then checking 8 locations at S = 4 with the normal TSS procedure.

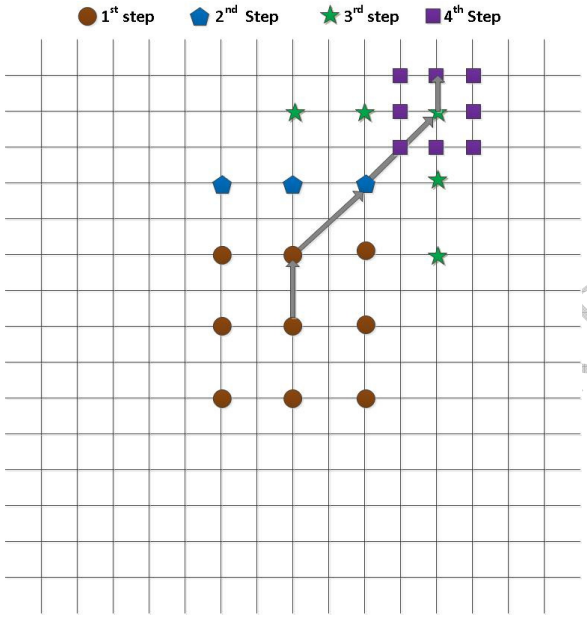

Fig.4 4SS Search Procedure.

C. Four Step Search Algorithm:

Four Step Search Algorithm (4SS) [6] employs center biased searching and has a halfway stop provision as similar to NTSS. The 4SS algorithm uses a search pattern with nine checking points on a 5 x 5 window instead of a 9 x 9 window in the 3SS. The 4SS algorithm reduces the number of computation when compared to the TSS and hence is more robust. It has the best case of 17 checking points and worst case of 27 checking points. The searching procedure of 4SS is shown in Figure 4. The 4SS algorithm can be summarized as follows:
Step 1: If the minimum cost function point is found at the center of search window by finding 9 checking points on 5x 5 windows, go to Step 4; otherwise go to Step 2.
Step 2: the search pattern will depend on the position of the previous minimum cost function point on the search window of size 5x5.

a) If the previous minimum cost function point is found at the corner of the previous search window, five additional checking points are used.
b) If the previous minimum cost function point is found 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 located at the center position of the search window, go to Step 4; otherwise go to Step 3.

Step 3: This step is exactly the same as the step 2, but finally it will go to Step 4.
Step 4: The search window is reduced to 3 x 3 and recorded the direction of the overall motion vector as the minimum cost function point among these nine searching points.

D. Diamond Search Algorithm:

The DS algorithm [7] use diamond shape instead of rectangular shape (used in TSS, NTSS), which tries to investigate the MV for all possible direction in circle shaped coverage. It employs two different types of fixed search patterns, one is Large Diamond Search Pattern (LDSP) and the other is Small Diamond Search Pattern (SDSP).The DS searching process shown in Figure 5 and is summarized as follows:

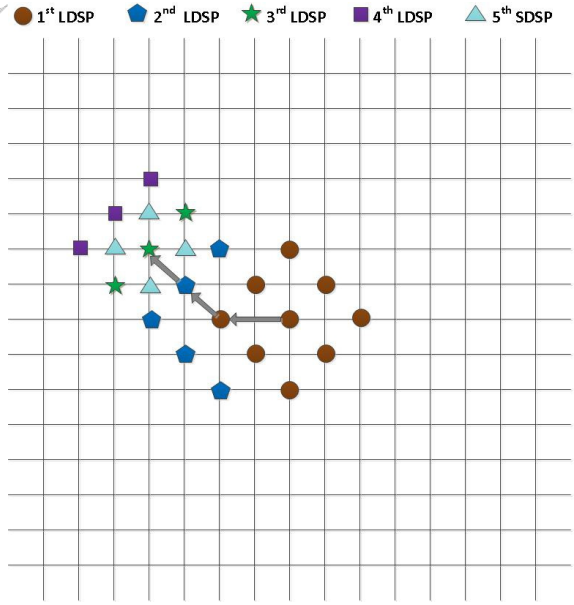

Fig.5 DS Search Procedure.

Step1: The initial step tests 9 checking points of LDSP which are centered at the origin of the search window. If the minimum cost function point calculated is located at the center position, go to Step 3; otherwise, go to Step 2.
Step2: The minimum cost function point found in the previous step is located as the center point to form a new LDSP. If the new minimum cost function point

obtained is found at the center position, go to Step 3; otherwise, recursively repeat this step.

Step3: Apply the search pattern SDSP around the new search origin. The location with the minimum cost function point found in this step is the final solution of the motion vector which indicates to the best matching block.

IV: EXPERIMENTAL RESULT AND ANALYSIS

In order to evaluate four above algorithms, they have been implemented in MATLAB and apply to two different Carphone and Foreman video sequences. Here we take the frame difference between current frame and reference frame is set to 2. In the search procedure we used MAD as matching cost function because it does not requires multiplication. A plot of the average number of searches required per macro block for both video sequence using the four BMA as shown in figure 6 and 8. The PSNR comparison of the compensated images generated using the above four algorithms is shown in Figure 7 and 9.

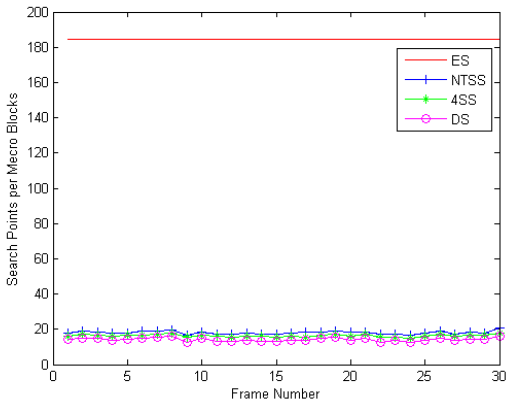

Fig.6 Average Search points per macro block per frame for 'carphone' sequence.

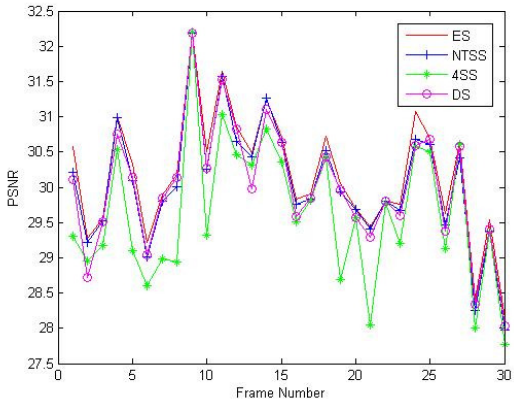

Fig.7 PSNR performance of BMA for 'carphone' sequence.

As are shown by figures, the ES takes on an average around ~184 searches per macro block, while other algorithm reduces number of computation. The figure shows that other algorithm comes pretty close to the PSNR results of ES.

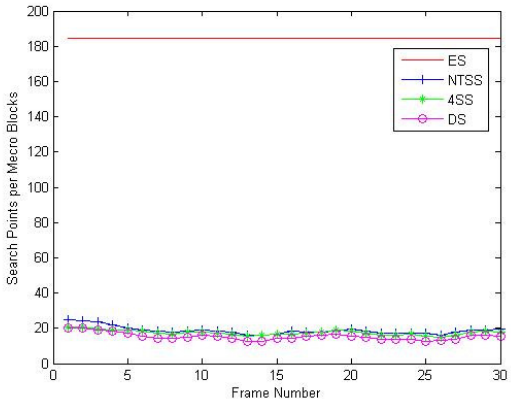

Fig.8 Average Search points per macro block per frame for 'Foreman' sequence.

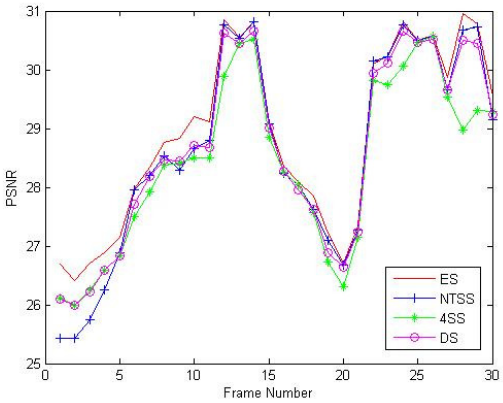

Fig.9 PSNR performance of BMA for 'Foreman' sequence.

VI: CONCLUSION

This paper presents the review of four different block matching (ES, NTSS, 4SS, DS) algorithms in order to compare them together. FS gives the best PSNR and guarantees the minimum errors but the limitation is a large number of computations requires. In DS algorithm, the search window size is not restricted as compared with TSS, NTSS and 4SS. Compared with other algorithm DS reduces the number of computation and its PSNR close to FS. So, from four considerable algorithms DS is the best one to implement and save time processing and get the better PSNR.

REFRENCES

[1] Z. Wei et al "Efficient fast motion estimation algorithm for H.264 based on polynomial model," IMACS Multiconference on computational engineering in System Applications, pp. 1654-1657, Oct.,4-6, 2006.

[2]Shi, Y.Q. and H. Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards2000: CRC Pr I Llc.

[3]P.C.Shenolikar, S.P.Narote,"Different Approaches for Motion Estimation", International conference on "Control, Automation, Communication and Energy

Conservation, International Conference on, 4th -6th June 2009
[4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," Proc. NTC81, pp. C9.6.1-9.6.5, New Orleans, LA. Nov. 1981.
[5] Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 4, pp. 438-442, August 1994.
[6] Lai-Man Po, and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 6, no. 3, pp. 313-317, June 1996.
[7] Shan Zhu, and Kai-Kuang Ma, " A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", IEEE Trans. Image Processing, vol 9, no. 2, pp. 287-290, February 2000.
[8] Aroh Barjatya, " Block Matching Algorithms For Motion Estimation ", DIP 6620 Spring 2004 Final Project Paper.
[9] Deepak Tauraga, Mohamed Alkanhal, "Search algorithms for Block Matching Estimation", Mid-term Project, spring 1998.