COMPARISON AND IMPLEMENTATION OF FAST BLOCK MATCHING MOTION ESTIMATION ALGORITHMS FOR VIDEO COMPRESSSION

D.V.MANJUNATHA[1]
Assistant Professor and Head, Department of Electronics & Communication Engineering,
DON BOSCO Institute of Technology, Visveswaraya Technological University,
Kumbalagodu, Mysore Road, Bangalore, Karnataka state, 560060, India.
dvmanjunatha@gmail.com, http ://< www.dbit.ac.in>

Dr. SAINARAYANAN[2]
Professor, Department of Electrical and Electrical Engineering,
New Horizon College of Engineering, Bangalore.
sai.jgk@gmail.com

Abstract:

In Digital video communication it is not practical, to store the full digital video without processing, because of the problems encountered in storage and transmission, so the processing technique called video compression is essential. In video compression, one of the computationally expensive and resource hungry key element is the Motion Estimation. The Motion estimation is a process which determines the motion between two or more frames of video. In this paper, Four block matching motion estimation algorithms, namely Exhaustive Search (ES), Three Step Search (TSS), New Three Step Search (NTSS), and Diamond Search (DS) algorithms are compared and implemented for different distances between the frames of the video by exploiting the temporal correlation between successive frames of mristack and foreman slow motion videos and proved that Diamond Search (DS) algorithm is the best matching motion estimation algorithm that achieve best tradeoff between search speed (number of computations) and reconstructed picture quality with extensive simulation results and comparative analysis.

Key words: Motion Estimation, ES, TSS, NTSS, DS and PSNR etc.

I. Introduction

In multimedia communication, the important requirement is to achieve high processing speed and a low computing time simultaneously without scarifying in image quality [1]. The video compression has become an important part of the way we create, communicate, and consume visual information" thus Video compression is vital for efficient storage and transmission of digital signal in multimedia. The video compression involves Video coding exploits the high correlation between successive frames to improve coding efficiency, which is usually achieved by using motion estimation (ME) and motion compensation techniques. Hence in video compression one of the computationally expensive and resource hungry key element is the Motion Estimation. Motion estimation is defined as searching the best motion vector, which is the displacement of the coordinate of the best similar block in previous frame for the block in current frame. In a video sequences; there exists a high level of redundancy between consecutive frames which means the changes from one frame to the other are minimal. In temporal redundancy the reduction of redundancy involves encoding first a reference frame and for the consecutive frames encode only the difference between the reference frame and the current frame, while doing so, lot complexity is involved in motion estimation in a video codec in video compression. In effort to reduce the computational complexity of ME algorithms, a variety of methods have been presented by many researchers such as block matching algorithm (BMA) [2]. The purpose of this work consists in a comparative study between block matching search algorithms, used in video compression, exploiting the temporal correlation between successive sequence frames we can reduce enormously the memory space needed for video storage and processing. The major applications of motion estimation algorithms include traffic movement tracking, studying plant root growth [9], hand posture analysis, human posture analysis, lip movement for user authentication, cinematography, robotic heart surgery, breathing motion estimation and many more. This paper is presented as follows. In Section 2, the Exhaustive Search (ES) is explained. In Section 3 the fast block matching techniques such as TSS, NTSS, and Diamond Search DS algorithms are described. In section 4, simulation results for proposed tequniques are compared in terms of computational complexity (number of computations) & PSNR are compared and presented. The Section 5 includes conclusion based on the results obtained in section 4.

II. The Exhaustive Block matching algorithm

Motion estimation is defined as searching the best motion vector, which is the placement of the co-ordinate of the best similar block in previous frame for the block in current frame [3]. Block-based matching algorithms find the optimal motion vectors which minimize the difference between reference block and candidate blocks. Exhaustive search or the Full search algorithm is the simple method for motion estimation. In searching for the best match, the correlation window is moved to each candidate position within the search window. There are a total (2p+1) *(2p+1) positions that need to be examined, where p is the search range for the block. The minimum dissimilarity gives the best match. The full search is brute force in nature and it delivers good accuracy in searching for the best match. But because of a large amount of computation is involved, it is useless in real-time encoding. There are several approaches to reducing the computational complexity, and these algorithms are fast search algorithms, the motion estimation process done using fast search algorithms instead of full search, follows special pattern that uses less computations [4].

III. The Fast block based search motion estimation algorithms

These algorithms estimate the amount of motion on a block by block basis, i.e. for each block in the current frame, a block from the previous frame is found, that is said to match this block based on a certain criterion. Some important search algorithms considered in this paper are TSS, NTSS and Diamont Search algorithms and all these algorithms are briefly explained in this section.

i. Three Step Search (TSS) Algorithm

This algorithm was introduced by Koga et al [5]. It became very popular because of its simplicity and also robust and near optimal performance. It searches for the best motion vectors in a course to fine search pattern. The algorithm is explained as:

Step 1: An initial step size is picked. Eight blocks at a distance of step size from the centre (around the centre block) are picked for the comparison.

Step 2: The step size is halved. The centre is moved to the point with the minimum distortion.

Steps 1 and 2 are repeated till the step size becomes smaller than 1. A particular path for the convergence of this algorithm is shown below:

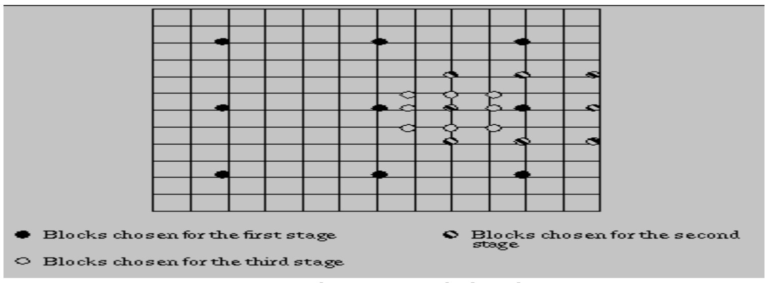
Blocks chosen for the first stage
Blocks chosen for the second stage
Blocks chosen for the third stage

Figure 1: Three step search algorithm

One problem that occurs with the Three Step Search is that it uses a uniformly allocated checking point pattern in the first step, which becomes inefficient for small motion estimation. This algorithm is used in the MPEG video standard.

ii. New Three Step Search (NTSS) algorithm

The improvement introduced by the New Three Step Search (NTSS) is a better estimation of the motion with low amplitude [6]. It has the same steps as the TSS with adding eight points neighboring the center point of block to be checked in the first step. If the minimum is matched in one of these points, we stop the search else we continue like it's done in the TSS. The NTSS needs at a maximum 33 checking points [7].

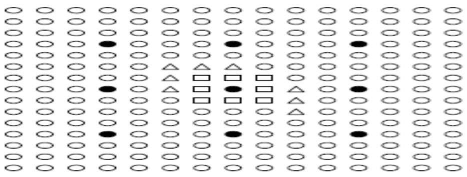
Figure2: New Three step Search (NTSS) algorithm.

iii. The Diamond Search (DS algorithm

DS [8] algorithm is exactly the same as four step search but the search point pattern is changed from a square to a diamond, and there is no limit on the number of steps that the algorithm can take. DS uses two different types of fixed patterns, one is Large Diamond Search Pattern (LDSP) and the other is Small Diamond Search Pattern (SDSP). These two patterns and the DS procedure are illustrated in Figure.3. The first step uses LDSP and if the least weight is at the center location we jump to fourth step. The consequent steps, except the last step, are also similar and use LDSP, but the number of points where cost function is checked are either 3 or 5 and are illustrated in second and third steps of procedure shown in Fig.3. The last step uses SDSP around the new search origin and the location with the least weight is the best match. As the search pattern is neither too small nor too big and the fact that there is no limit to the number of steps, this algorithm can find global minimum very accurately. The end result should see a PSNR close to that of ES while computational expense should be significantly less. The figure 3 shows the large diamond search pattern and the small diamond search pattern. It also shows an example path to motion vector (-4, -2) in five search steps four times of LDSP and one time of SDSP.

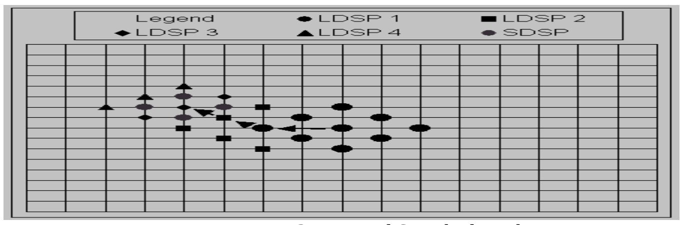

Figure 3 Diamond Search algorithm

IV. The simulation Results

The simulation results of mristack and foreman videos are as shown in the following tables and figures:

1. Simulation results for slow motion Mristack video:

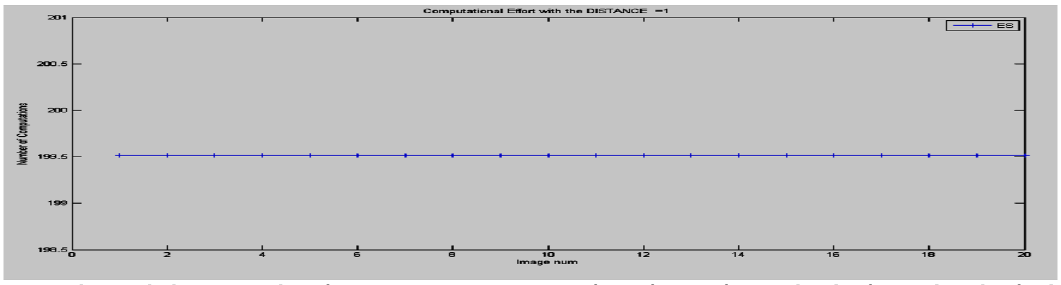

Figure 4: The Graph showing number of computations Vs image Num for 20 frames of mristack video for ES algorithm for d=1.

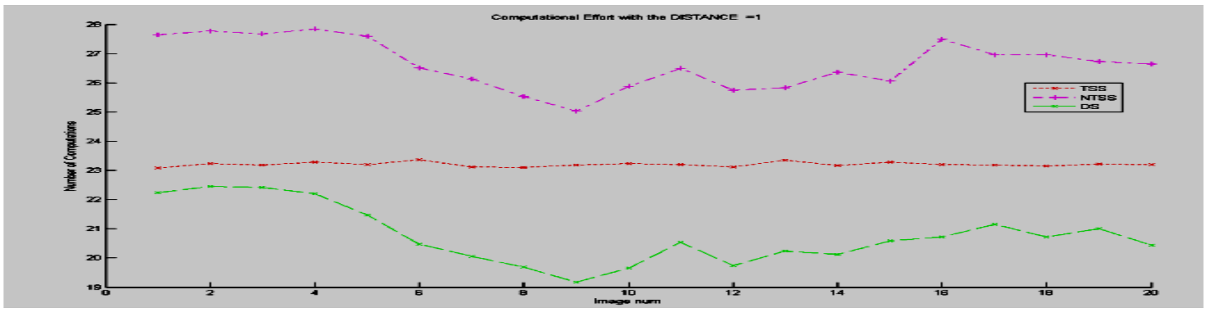

Figure 5: The Graph showing number of computations Vs Image Num for 20 frames of mristack video (for TSS, NTSS and DS algorithms

Sl No.	Name of the Algorithm	Distance d	No of computations	Maximum PSNR
01	Exhaust Search Algorithm (ES or FS)	d=1	199.515625	21.924823
		d=2	199.515625	18.708789
		d=3	199.515625	17.898999
		d=4	199.515625	17.183677
		d=5	199.515625	16.443549
02	Three Step Search Algorithm (TSS)	d=1	23.363281	21.721482
		d=2	23.460938	18.405207
		d=3	23.367188	17.309923
		d=4	23.433594	16.865210
		d=5	23.433594	16.064282
03	New TSS (NTSS) algorithm	d=1	27.835938	21.704972
		d=2	28.378906	18.405207
		d=3	28.324219	17.306259
		d=4	28.605469	16.822931
		d=5	28.304688	16.046599
04	Diamond Search Algorithm	d=1	22.457031	21.566716
		d=2	22.957031	17.904443
		d=3	23.054688	17.129012
		d=4	23.855469	16.503637
		d=5	23.597656	15.918865

Table 1 Number of computations and maximum value of PSNRs of different search algorithms for different values of distance between the frames for Mristack video for 1st 20 frames.

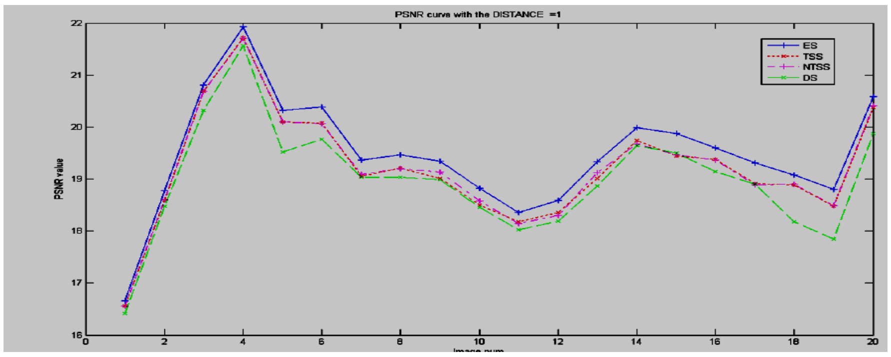

Figure 6: The Graph showing value of PSNR Vs image num for 1st 20 frames of slow motion mristack video for ES, TSS, NTSS and DS algorithms for d=1

2. Simulation results for slow motion Foreman video:

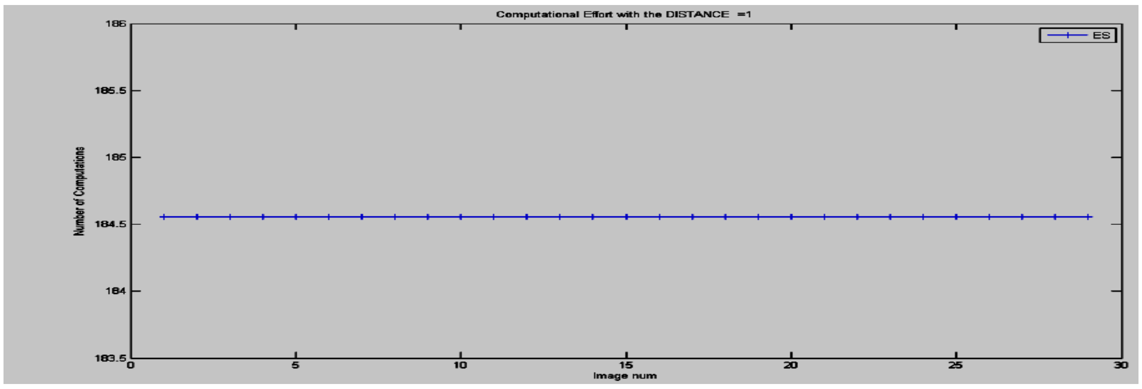

Figure 7: The Graph showing number of computations Vs image Num for 30 frames of Foreman video for ES algorithm for d=1.

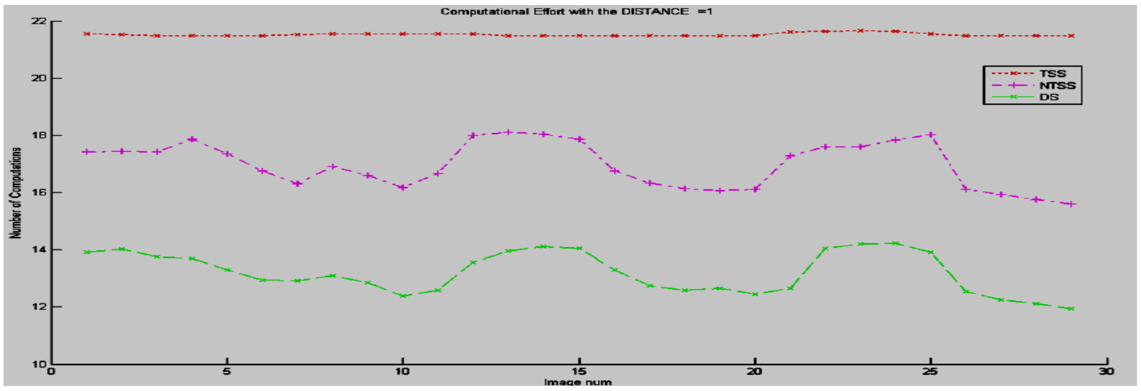

Figure 8: The Graph showing number of computations Vs image Num for 30 frames of Foreman video (for TSS, NTSS and DS algorithms for d=1

Sl No.	Name of the Algorithm	Distance d	No of computations	Maximum PSNR
01	Exhaust Search Algorithm (ES or FS)	d=1	184.555556	37.450451
		d=2	184.555556	34.997737
		d=3	184.555556	33.067133
		d=4	184.555556	31.340607
		d=5	184.555556	28.769263
02	Three Step Search Algorithm (TSS)	d=1	21.666667	37.450451
		d=2	21.989899	34.484299
		d=3	22.151515	32.772751
		d=4	22.202020	31.035942
		d=5	22.262626	28.514775
03	New TSS (NTSS) algorithm	d=1	18.111111	37.450451
		d=2	20.404040	34.997753
		d=3	24.020202	32.488730
		d=4	26.202020	30.522503
		d=5	28.000000	28.426694
04	Diamond Search Algorithm	d=1	14.232323	37.450451
		d=2	16.757576	34.997737
		d=3	19.000000	32.797457
		d=4	20.383838	31.018891
		d=5	21.545455	28.577820

Table 2: Number of computations and maximum value of PSNRs of different search algorithms for different values of distance d between the frames for Foreman video for 1st 30 frames.

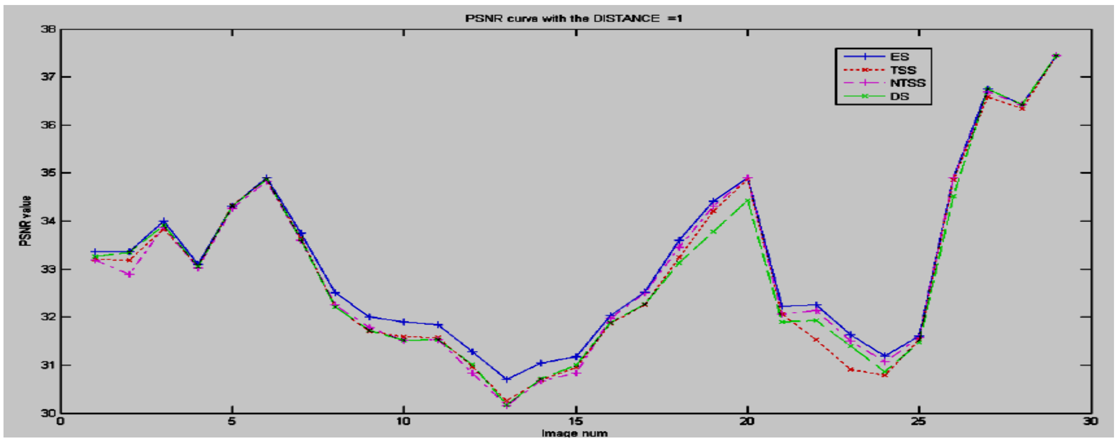

Figure 9: The Graph showing PSNR Vs image Num for 1st 30 frames of slow motion Foreman video for ES, TSS, NTSS and DS algorithms for d=1

During the course of this project all of the above 4 algorithms have been implemented. For 'Mristack' and 'Foreman' video sequence with variable distances from d=1 to d=5 between current frame and reference frame were used to generate the frame-by-frame results of the algorithms as below:

1. **Simulation results of Mristack video**

- Table 1 indicates Number of computations and maximum value of PSNRs of different search algorithms for different values of distance between the frames for Mristack video for 1st 20 frames.
- Figures 4 and 5 indicates The Graph showing number of computations Vs image Num for 20 frames of mristack video for ES , TSS, NTSS and DS algorithms for d=1
- Figure 6 indicates the graph showing values of PSNR Vs Image Num for 1st 20 frames of slow motion mristack video for ES, TSS, NTSS and DS algorithms for d=1

2. **Simulation results of Mristack video**

- Table 2 indicates Number of computations and maximum value of PSNRs of different search algorithms for different values of distance between the frames for foreman video for 1st 30 frames.
- Figures 7 and 8 indicates The Graph showing number of computations Vs image Num for 30 frames of foreman video for ES , TSS, NTSS and DS algorithms for d=1
- Figure 9 indicates the graph showing values of PSNR Vs image Num for 1st 30 frames of slow motion Foreman video for ES, TSS, NTSS and DS algorithms for d=1

V. Conclusion

In this paper we have compared four motion estimation algorithms, out of which diamond search algorithm is giving the best results in terms of number of computations and the image quality, As is shown by table1and 2 the DS algorithm come pretty close to the PSNR results of ES with reduced number of computation, for distance d=1 without scarifying the image quality for mristack and foreman videos. As the distance between the frames is increased the image quality keeps reducing but number of computations remains same for ES and is slightly varying for the other 3 algorithms for Mristack and foreman video sequences. Therefore the constant distance of d=1 is maintained for getting good image quality with reduced number of computations for both the video sequences. Overall the diamond search algorithm is giving the highest PSNR with reduced number of computations for slow video motion of both Mristack and foreman video sequences.

References

[1] Ulas Demir et al,"Motion wavelet compression, " University of Hertfordshire, Hatfield, Herts, UK. (1999-2000).

[2] Fulvio Moschetti Murat Kunt and Eric Debes, "A statical adaptive block matching motion estimation", IEEE transactions on circuits and systems for video technology, Vol 13 N° 4 April 2003

[3] Ahmadi, M. 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, March 2008.

[4] Deepak Tauraga, Mohamed Alkanhal, "Search algorithms for Block Matching Estimation", Mid-term Project, spring 1998.

[5] T. Koga et al, "Motion compensated interframe image coding for video conference," *Proc. NTC81*, pp. G5.3.1, Nov. 1981.

[6] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans.Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438–442, Aug. 1994.

[7] Mei-Juan Chen et al, "One dimentional Full Search motion estimation algorithm for video coding", IEEE transactions on circuits and systems for video technology. Vol 4, N° 5, Octobre 1994.

[8] 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.

[9] P. Vijaykumar et al "Latest Trends, Applications and Innovations in Motion estimation Research", International Journal of Scientific & Engineering Research Volume 2, Issue 7, July-2011 ISSN 2229-5518. Applications