Journal of Advances in Science and Technology
Vol. 12, Issue No. 25, (Special Issue) December-2016, ISSN 2230-9659

Analysis of Motion Estimation and Compensation for Video Compression

V. M. Bastawadi[1]*, S. R. Daddimani[2], S. S. Ittannavar[3]

[1]ECE Department, HIT, Nidasoshi

[2]ECE Department, HIT, Nidasoshi

[3]Assistant Professor, ECE Dept, HIT, Nidasoshi, India

Abstract – This paper focuses on comparative study of the different motion estimation techniques for full search algorithms that have been proposed for inter frame coding in moving video sequences taking into consideration the different kinds of motion of the moving objects in successive frames. This is very important because it helps in bandwidth conservation by reducing temporal redundancy and reduction in power consumption by reducing computation complexity. It implements and compares 5 different types of block matching algorithms that range from the very basic Exhaustive Search to the recent fast adaptive algorithms like Adaptive Rood Pattern Search. The algorithms that are evaluated in this paper are widely accepted by the video compressing community and have been used in implementing various standards, ranging from MPEG1 / H.261 to MPEG4 / H.263. The paper also presents a very brief introduction to the entire flow of video compression. Keywords - Block matching, motion estimation, video compression, MPEG, H.261, H.263

----------♦----------

INTRODUCTION

The temporal prediction technique used in MPEG video is based on motion estimation. The basic premise of motion estimation is that in most cases, consecutive video frames will be similar except for changes induced by objects moving within the frames. In the trivial case of zero motion between frames and no other deference caused by noise, etc., it is easy for the encoder to efficiently predict the current frame as a duplicate of the prediction frame. When this is done, the only information necessary to transmit to the decoder becomes the syntactic overhead necessary to reconstruct the picture from the original reference frame. When there is motion in the images, the situation is not as simple. Another goal of Motion Estimation is to reduce the total amount of bits required for transmission or storage of the frames of an image sequence. Motion compensated image sequence coding primarily focuses on the reduction of the high temporal correlation of the signal to be stored or transmitted. To that end, motion information has to be extracted from the sequence in order to relate locations in consecutive frames that correspond in gray level. This motion is represented by so called corresponding or displacement vectors.

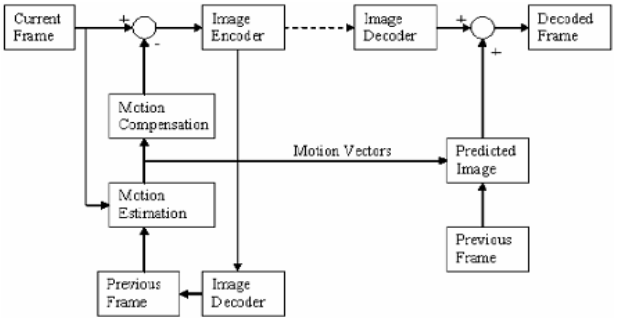

Fig. 1 MPEG/H.26x video compression process flow.

The basic flow of the entire compression decompression process is largely the same and is depicted in Fig. 1 The encoding side estimates the motion in the current frame with respect to a previous frame. A motion compensated image for the current frame is then created that is built of blocks of image from the previous frame. The motion vectors for blocks used for motion estimation are transmitted, as well as the difference of the compensated image with the current frame is also JPEG encoded and sent. The encoded image that is sent is then decoded at the encoder and used as a reference frame for the subsequent frames. The decoder reverses the process and creates a full frame. The whole idea behind motion estimation based video compression is to save on bits by sending JPEG encoded difference

www.ignited.in

images which inherently have less energy and can be highly compressed as compared to sending a full frame that is JPEG encoded.

2. BLOCK MATCHING ALGORITHMS

The idea behind block matching is to divide the current frame into a matrix of 'macro blocks' that are then compared with corresponding block and its adjacent neighbors in the previous frame to create a vector that stipulates the movement of a macro block from one location to another in the previous frame. This movement calculated for all the macro blocks comprising a frame, constitutes the motion estimated in the current frame. The search area for a good macro block match is constrained up to p pixels on all fours sides of the corresponding macro block in previous frame. This 'p' is called as the search parameter. Larger motions require a larger p, and the larger the search parameter the more computationally expensive the process of motion estimation becomes. Usually the macro block is taken as a square of side 16 pixels, and the search parameter p is 7 pixels. The matching of one macro block with another is based on the output of a cost function. The macro block that results in the least cost is the one that matches the closest to current block. There are various cost functions, of which the most popular and less computationally expensive is Mean Absolute Difference (MAD). We can also use Sum of Absolute Difference (SAD). Another cost function is Mean Squared Error (MSE). Equations are given below.

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}|Cij - Rij|$$

$$SAD = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1}|Cij - Rij|$$

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(Cij - Rij)^2$$

Where N is the side of the macro bock, Cij and Rij are the pixels being compared in current macro block and reference macro block, respectively. Peak-Signal-to-Noise-Ratio (PSNR) is given by equation. It characterizes the motion compensated image that is created by using motion vectors and macro clocks from the reference frame.

$$\text{PSNR} = 10\log_{10}\frac{[(\text{peak to peak value of original data})^2]}{MSE}$$

A. Exhaustive Search (ES)

In selecting a suitably matched block, the ES algorithm searches the entire search region for a block such that the BDM is a global minimum. If more than one block generates a minimum BDM, the ES algorithm selects the block whose motion vector has the smallest magnitude, in order to exploit the centre-biased motion-vector distribution characteristics of a real-world video sequence. To achieve this, checking points are used in a spiral trajectory starting at the centre of the search region. If the maximum displacement of a motion vector in both the horizontal and vertical directions is + d or -d pixels, the total number of search points used to locate the motion vector for each block can be as high as (2d+1)2. This algorithm, also known as Full Search, is the most computationally expensive block matching algorithm of all. This algorithm calculates the cost function at each possible location in the search window. As a result of which it finds the best possible match and gives the highest PSNR amongst any block matching algorithm. Fast block matching algorithms try to achieve the same PSNR doing as little computation as possible. The obvious disadvantage to ES is that the larger the search window gets the more computations it requires.

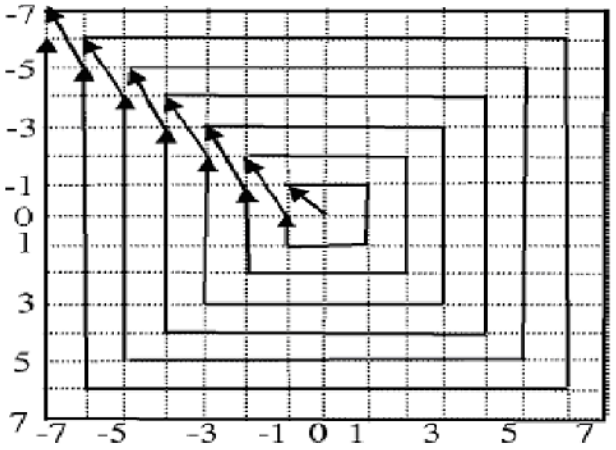

Fig. 2 The spiral trajectory of the checking points in the ES algorithm

B. Three Step Search (TSS)

Three step search (TSS), employs rectangular search patterns with different sizes. Koga introduced this algorithm in 1981. It became very popular because of its simplicity and also robust and near optimal performance. Figure 3 shows TSS algorithm. It searches for the best motion vectors in a coarse to fine search pattern. Algorithm of three step search as follows.

- **Step 1:** An initial step size is picked. Eight blocks at a distance of step size from the

www.ignited.in

centre (around the center block) are picked for 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.

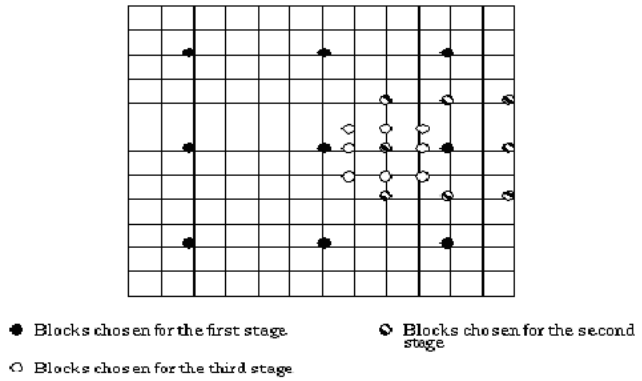

Fig. 3 Three Step Search Algorithm

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.

C. Four Step Search (4SS)

This algorithm also exploits the centre-biased property in its search. The computational complexity of the four-step search is less than that of the three step search, while the performance in terms of quality is as good. It is also more robust than the three step search and it maintains its performance for image sequences with complex movements like camera zooming and fast motion. Hence it is a very attractive strategy for motion estimation. The algorithm starts with a nine point comparison and then the other points for comparison are selected based on the following algorithm:

Step 1: Start with a step size. Pick nine points around the search window centre. Calculate the distortion and find the point with the smallest distortion If this point is found to be the centre of the searching area go to step 4, otherwise go to step 2.

Step 2: Move the centre to the point with the smallest distortion. The step size is maintained at 2. The search pattern however depends on the position of the previous minimum distortion.

a. If the previous minimum point is located at the corner of the previous search area, five points are picked as shown in the Figure 4.3.

b. If the previous minimum distortion point is located at the middle of the horizontal or vertical axis of the previous search window, three additional checking points are picked. as shown in the Figure 4. Locate the point with the minimum distortion. If this is at the centre, go to step 4 otherwise go to step 3.

Step 3: The search pattern strategy is the same, however it will finally go to step 4.

Step 4: The step size is reduced to 1 and all nine points around the centre of the search are examined.

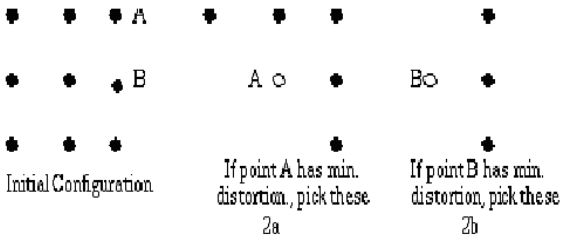

Fig. 4 Four Step Search Algorithm-1

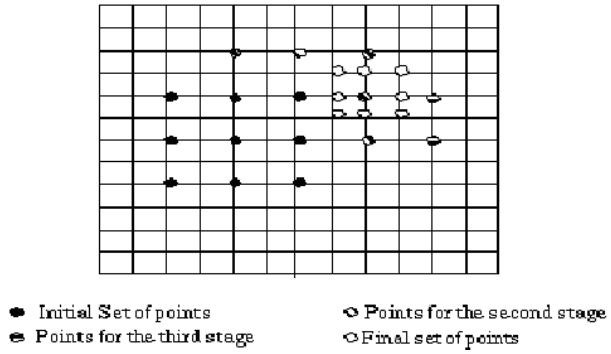

Fig. 5 Four Step Search Algorithm-2

The computational complexity of the four step search is less than that of the three step search, while the performance in terms of quality is as good. It is also more robust than the three step search and it maintains its performance for image sequences with complex movements like camera zooming and fast motion. Hence it is a very attractive strategy for motion estimation. *D. Diamond Search (DS)*

The diamond search is based on MV distribution of real world video sequences. It employs two search patterns in which the first pattern, called Large Diamond Search Pattern (LDSP) comprises nine checking points and forms a diamond shape. The second pattern consists of five checking points make a Small Diamond Search Pattern (SDSP). The search starts with the LDSP and is used repeatedly until the minimum BDM point lies on the search centre. The search pattern is then switched to SDSP. The position yielding minimum error point is taken as the final MV. The search process is shown in Figure 6. DS is an outstanding algorithm adopted by MPEG-4

verification 4 model (VM) due to its superiority to other methods in the class of fixed search pattern algorithms.

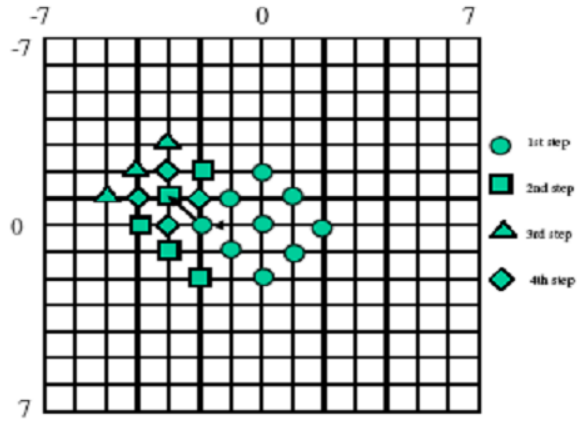

Fig. 6 Diamond Search Algorithm

In DS the search basic point pattern is the diamond shape, and there is no limit on the number of steps that the algorithm can take. There are two different types of fixed patterns, one is Large Diamond Search Pattern (LDSP) and the other is Small Diamond Search Pattern (SDSP).

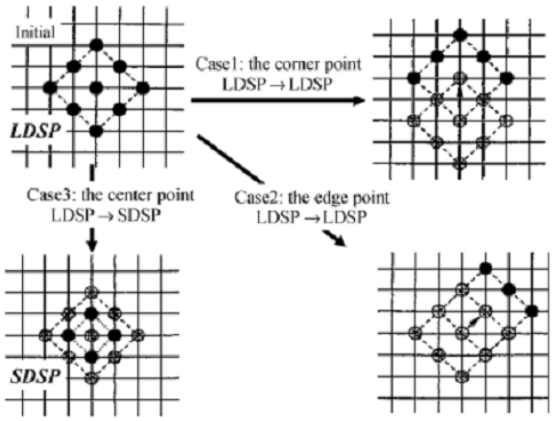

Fig.7 Different cases of diamond search algorithm

These two patterns and the DS procedure are illustrated in Fig.6 The first step uses LDSP and if the least cost search point is at the centre location we jump to fourth step. The consequent steps, except the last step, are also similar and use LDSP, but the number of search points where cost function is checked is reduced to either 3 or 5 as shown in Fig.7. The last step uses SDSP around the new search origin and the location with the least weight is the best match.

Algorithm of Diamond Search as follows.

Step 1: The initial LDSP is centred at the origin of the search window, and the 9 Checking points of LDSP are tested. If the MBD point calculated is located at the centre position, go to **Step 3**; otherwise, go to **Step 2**.

Step 2: The MBD point found in the previous search step is re-positioned as the centre point to form a new LDSP. If the new MBD 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 SDSP. The MBD point found in this Step is the final solution of the motion vector which points to the best Matching block.

E. Adaptive Rood Pattern Search (ARPS)

ARPS algorithm makes use of the fact that the general motion in a frame is usually coherent, i.e. if the macro blocks around the current macro block moved in a particular direction then there is a high probability that the current macro block will also have a similar motion vector. This algorithm uses the motion vector of the macro block to its immediate left to predict its own motion vector. The predicted motion vector points to (3, -2). In addition to checking the location pointed by the predicted motion vector, it also checks at a rood pattern distributed points, as shown in Fig 8, where they are at a step size of S = Max (|X|, |Y|). X and Y are the x-coordinate and y-coordinate of the predicted motion vector. This rood pattern search is always the first step. It directly puts the search in an area where there is a high probability of finding a good matching block. The point that has the least weight becomes the origin for subsequent search steps, and the search pattern is changed to SDSP.

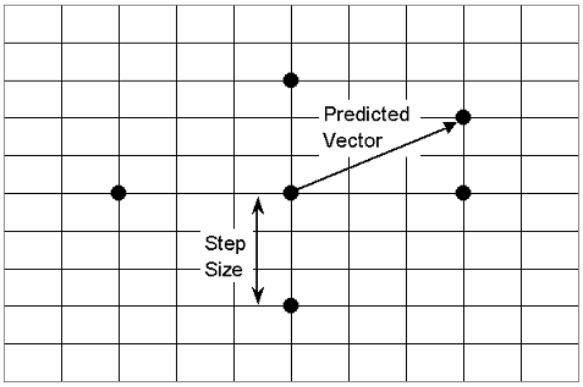

Fig 8. Adaptive Root Pattern : The predicted motion vector is (3,-2) ,and the step size S = Max(|3|, |-2|) = 3.

The procedure keeps on doing SDSP until least weighted point is found to be at the centre of the SDSP. A further small improvement in the algorithm can be to check for Zero Motion Prejudgment using which the 5 search is stopped half way if the least

www.ignited.in

weighted point is already at the centre of the rood pattern. The main advantage of this algorithm over DS is if the predicted motion vector is (0, 0), it does not waste computational time in doing LDSP, it rather directly starts using SDSP. Furthermore, if the predicted motion vector is far away from the centre, then again ARPS save on computations by directly jumping to that vicinity and using SDSP, whereas DS takes its time doing LDSP. Care has to be taken to not repeat the computations at points that were checked earlier. Care also needs to be taken when the predicted motion vector turns out to match one of the rood pattern location. We have to avoid double computation at that point. For macro blocks in the first column of the frame, rood pattern step size is fixed at 2 pixels.

3. SIMULATION RESULTS

Exhaustive Search(ES), Three step search (TSS), Four Step Search (FSS), Diamond Search (DS) and Adaptive Rood Pattern Search (ARPS) are implemented and the comparison of average number of search for macro-block of full search, computed by different algorithms. From the results It clearly indicates that Adaptive Rood Pattern Search has minimum average number of search for macro-block and exhaustive search has maximum average number of search for macro-block. Now in comparison with ES of full search algorithms, TSS, FSS, DS and ARPS are better in macro-block comparison and take less time to search macro-blocks. DS and FSS drop that number by more than an order of magnitude. ARPS further drops by a factor of 2 compared to DS. ARPS takes up less number of search point computations amongst all. But exhaustive search algorithm finds accurate macro-block than all others.

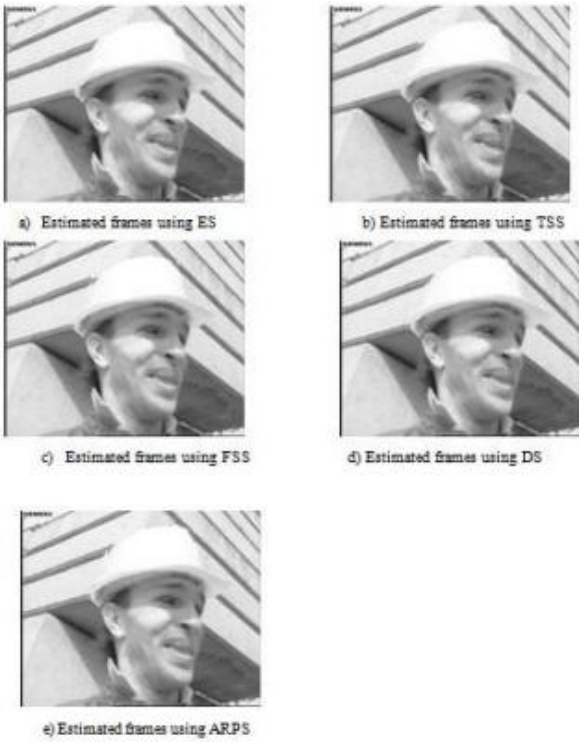

Fig 9. The 60th estimated frames from the "Foreman" sequence, Full Search using different searching Algorithms

Table 1. Comparison of Motion Estimation Algorithms with fast full search

Sequence	Algorithm	Average Number Of Search Points	PSNR in dB	Time in sec.
Formen_ 352x288 _90frs	ES	204.2828	32.7725	100.8935
	DS	21.9848	32.6534	13.7832
	TSS	23.5480	32.2452	11.7402
	FSS	22.5758	32.5461	10.8090
	ARPS	12.7828	32.5378	9.9883

4. CONCLUSION

This work has explored the theory of all motion estimation algorithms and examined basic features of motion estimation algorithms with full search. Five different algorithms with full search for motion estimation are tested and compared Average number of search points and PSNR ratio. It is concluded that Adaptive Rood Pattern Search (ARPS) motion estimation algorithm gives better result compared to all other algorithms. In full search the PSNR Performance of ARPS is worst. Although PSNR performance of FSS, DS, and ARPS is relatively the same. In this case ARPS takes less computation and hence is the best of the fast block matching algorithms. The past two decades have seen wide acceptance of multimedia. Video compression plays an important role in archival of entertainment based video (CD/DVD) as well as real-time reconnaissance / video conferencing applications. While ISO MPEG sets the standard for the former types of application,

ITU sets the standards for latter low bit rate applications. In the entire motion based video compression process motion estimation is the most computationally expensive and time-consuming process. The research in the past decade has focused on reducing both of these side effects of motion estimation.

5. FUTURE SCOPE

In future this algorithm can be further improved by developing new search technique so that it can take less time to search macro-blocks. Other evolutionary computing techniques also can be tried for the better results. There are many other ME methods, but Block matching ME is normally preferred due to its simplicity and good compromise between prediction quality and motion overhead. In full search motion estimation we can implement new techniques which will further reduce the complexity of MPEG video coding. Further we can find methods to estimate the compression loss in relation to time of compression and its suitability to the types of images. 6

6. REFERENCES

Gary J.Sullivan and Thomas Wiegand, "Rate Distortion Optimization for Video Compression," IEEE Signal Processing Magazine. 1053-5888, November 1998.

M. Kemal Gullu, Member, IEEE "Weighted Constrained One-Bit Transform Based Fast Block Motion Estimation", IEEE Transactions on Consumer Electronics, Vol. 57, No. 2, May 2011..

Frederic Dufaux, Fabrice Moscheni, "Motion Estimation Techniques for Digital TV", Proceedings of IEEE Vol 83 N0, 6 June 1995.

Robert B. Fisher, "Sub-Pixel Estimation", University of Edinburgh.

S. Dhahri, A. Zitouni, H. Chaouch, and R. Tourki, "Adaptive Motion Estimator Based on Variable Block Size Scheme" World Academy of Science, Engineering and Technology 50 2009.

Aroh Barjatya, "Block Matching Algorithms For Motion Estimation" DIP 6620 Spring 2004.

Deepak S. Jain, Dr. Sanjay L. Nalbalwar, "A Fast Fractional Pel Motion Estimation Algorithm" Published in International Journal of Advanced Engineering & Applications, Jan. 2010.

Weiyao Lin, Krit Panusopone, David M. Baylon, Ming-Ting Sun and Hongxiang Li "A Fast Sub-Pixel Motion Estimation Algorithm for H.264/AVC Video Coding", IEEE Transactions on circuits and systems for video technology, vol. 21, no. 2, February 2011.

Lai-Man Po, Wing-Chung Ma "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation" IEEE Trans. Circuits Syst. Video Technol., vol. 6, No. 3, pp. 313-317, Jun. 1996

Yun-Gu Lee, Jae Hun Lee, and Jong Beom Ra "Fast half-pixel motion estimation based on directional search and a linear model" Proceedings of SPIE Vol. 5150 (2003).

Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim "A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation" IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 8, NO. 4, AUGUST 1998.

Iain E.G Richardson "H.264 and MPEG-4 Video Compression" The Robert Gordon University, Aberdeen, UK.

H. Mahdavi-Nasab and Shohreh Kasaei "New Half-Pixel Accuracy Motion Estimation Algorithms for Low Bitrate Video Communications"

M. Ezhilarasan and P. Thambidurai "Simplified Block Matching Algorithm for Fast Motion Estimation in Video Compression" 282-289, 2008 ISSN 1549-3636, 2008 Science Publications.

Corresponding Author

V. M. Bastawadi*

ECE Department, HIT, Nidasoshi

E-Mail – vidyamb4@gmail.com