International Journal of Engineering Research & Management Technology ISSN: 2348-4039

March- 2015 Volume 2, Issue-2

Email: editor@ijermt.org www.ijermt.org

A Comparative Review on Block Motion Estimation Using New Four Step Search Algorithm

Shruti Saxena, Ankit Khetan
B.Tech Student
Department Of Information Technology
SRM University, NCR Campus
Modinagar

ABSTRACT—
This paper is a review of the block matching algorithms used for motion estimation in video compression. It implements and compares 5 different types of block matching algorithms that range from the very basic Exhaustive Search to the recent four step search algorithm.. The paper also presents a very brief introduction to the entire flow of video compression. Vector distribution, a new four-step search (4SS) algorithm with center-biased checking point pattern for fast block motion estimation is proposed in this paper. Halfway-stop technique is employed in the new algorithm with searching steps of 2 to 4 and the total number of checking points is varied from 17 to 27. Simulation results show that the proposed 4SS performs better than the well-known three-step search and has similar performance to the new three-step search (N3SS) in terms of motion compensation errors. In addition, the 4SS also reduces the worst-case computational requirement from 33 to 27 search points and the average computational requirement from 21 to 19 search points as compared with N3SS.

Keywords: NSS, Frames, Video Compression

1. INTRODUCTION

Video communication has found to have a wide range of application in modern era. The application domains may be of Conversational video such as video telephony, video conferencing through wired or wireless medium and Streaming Video such as Video on demand, HDTV etc. Digital video communication requires a high bandwidth and storage space. A video is a continuous stream of images taken by camera and it has temporal aspect of the signal that not only varying in space but also in time [1]-[12].

.

2. INTRA AND INTER CODED FRAMES

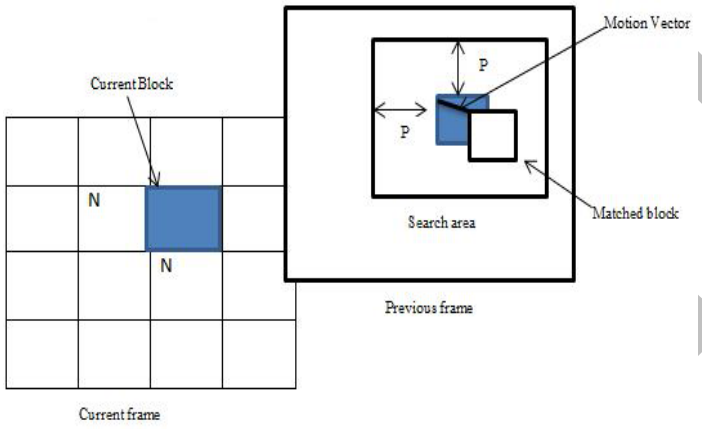
The question now arises that how can we predict and encode the first frame as it has no past frame for reference? Then the concept of Intra-frame coding has come. Then we have to consider the first frame of each video as a still image which will have only spatial or intra-frame redundancy. The frames that contains all of its own information and results less error when encoded is known as Intra coded frame or I-frame [2]. All subsequent frames have both temporal and spatial redundancy i.e. they use inter code redundancy for compression. These are called inter coded frames. In intra coded more bits are needed because temporal redundancy is not considered

3. MOTION ESTIMATION AND BLOCK MATCHING

In video between successive frames there is great deal of similarity. To find the displacement vector for the candidate block of current frame in reference frame (past frame or future frame) one has to perform matching between two consecutive frames [11]. Comparing pixel to pixel intensity between frames

We cannot be sure that this pixel corresponds that pixel in two frames because two different pixels may have same value [3]. So instead of matching pixel to pixel value, a region or block

Based matching is done over past frames. The matching is done by searching the position corresponding to the minimum value of matching criteria which gives the motion vector [9]. This whole process of finding the best match is known as Motion Estimation. Since it is done per block basis it is called Block based Motion Estimation Technique (BMA). For BMA we subdivide the image into NxN non overlapping blocks.



3.1 Forward and backward prediction: (FP &BP):

In encoding frame k, we can use the past frame i.e. (k-1)th frame as a reference to predict what frame number k is going to be. As we are predicting future it is known as Forward prediction and for finding out MV and for doing FP that we are going back in time so it is called backward motion estimation. This is one type of unidirectional prediction. In the reverse case if we use future frame for reference to encode Current frame we are predicting the past going ahead in time so the process is called backward prediction or forward motion estimation technique. The combination of both i.e. in some cases both past frame and future frame are used as a reference. This type of prediction is called bidirectional prediction and the frame used is known as Bi-directional

Predicted frame called B-frame.

4. FAST MOTION ESTIMATION ALGORITHM

Generally, there is a relationship between the displacement of current block and the displacement of previous block which is known as motion vector. This is particularly true in low bit-rate video applications, including videophone and video conferencing, where fast and complex movements are involved rarely. The FTSS algorithm makes use of the directional information from adjacent previous block motion vector. The current motion vector is estimated by the directional information of the previous motion vector. The adjacent motion vector for the current frame helps in the prediction of the direction of the current motion vector [4]. The motion vector of the left neighbor i.e. the sign of previous motion vector can be determine from the current motion vector.

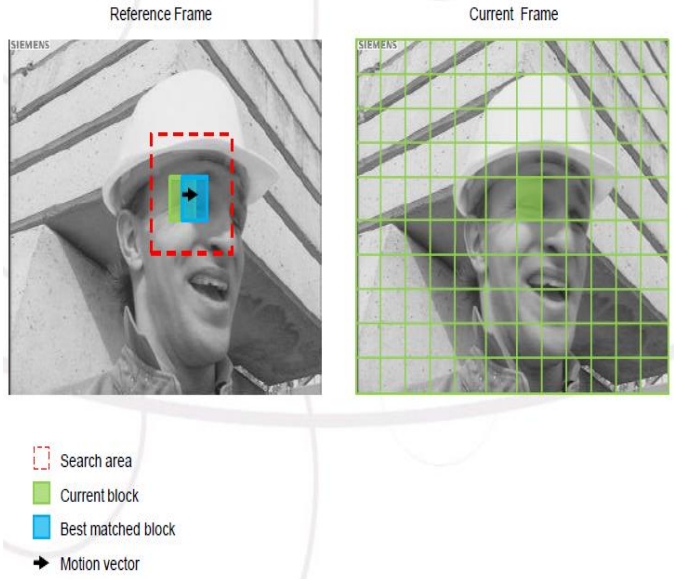
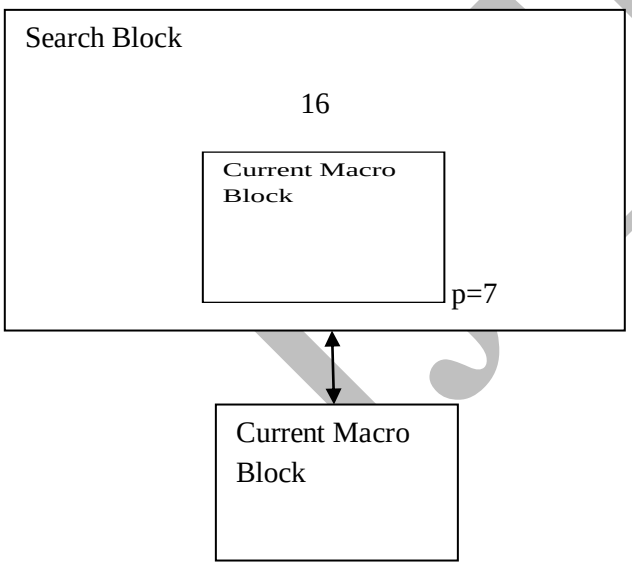


Fig 1- Block Matching

5. BLOCK MATCHING ALGORITHMS

The underlying supposition behind motion estimation is that the patterns corresponding to objects and background in a frame of video sequence move within the frame to form corresponding objects on the subsequent frame. 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 [9]. 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 [10].



The idea is represented in Fig 4. 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[2]. There are various cost functions which can decide the complexity of the various algorithms like block distortion mean.

5.1 FULL SEARCH ALGORITHM

This algorithm, also known as Exhausted 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 [1]. 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.

Since complexity of the implementation mostly depends on the complexity of the algorithm itself, full search algorithm needs a huge number of gates for its implementation.

The following table shows the comparison of the Block Distortion Mean for the next comparison, next 10 comparisons and all comparisons [5]. The Full search algorithm calculates the BDM (when search range was taken 23) by searching the whole search window and hence computationally expensive Table 3.1 BDM values using Full Search algorithm

FULL SEARCH ALGORITHM			
Block Distortion Mean	Next comparisn	Next 10 comparisons	All comparisons
	30.02	20.65	33.56

5.2 FULL SEARCH ONE DIMENSIONAL ALGORITHM

It is an alternative for Full Search Algorithm and achieves a good compromise between computational complexity and performance .This algorithm calculates the block distortion mean and cost function along a single dimension in the search window. All the searching steps are same as that of the Full Search Algorithm except that in Full Search One Dimensional the search is done along a single dimension in the search window[1]-[8].

The following table shows the comparison of the Block Distortion Mean for the next comparison, next 10 comparisons and all comparisons. The Full search one dimensional algorithm calculates the BDM (when search range was taken 23) by searching along only one dimension of the search window and hence less computationally expensive in comparison to full search algorithm.

Table 3.2 BDM values using Full Search one dimensional algorithm

FULL SEARCH ONE DIMENSIONAL ALGORITHM			
Block Distortion Mean	Next comparison	Next 10 comparisons	All comparisons
	40.4	14.87	17.59

5.3 THREE STEP SEARCH ALGORITHM

Originally proposed by Koga et al., this is a fine-coarse search mechanism. The search pattern of TSS is shown in Figure. The first step involves search based on 4-pixel/4-line resolution at nine locations i.e. 9x9 search window, with the center point corresponding to zero MV[3]. The second step involves search based on 2-pixel/2-line resolution i.e. 5x5 search window around the location determined by the first step. This is repeated in the third step with 1-pixel/1-line resolution and a search window of 3x3. The last step yields the MV. The TSS is one of the most popular BMAs. For a maximum displacement window of 7 i.e. d=7, the number of checking points required is

(9+8+8)=25. For larger search window (i.e. larger d), TSS can be easily extended to n-steps using the same searching strategy with the number of checking points required equals to [1+8{log2(d+1)}].
It starts with the search location at the center and sets the 'step size' S = 4, for a usual search parameter value of 7. It then searches at eight locations +/- S pixels around location (0,0). From these nine locations searched so far it picks the one giving least cost and makes it the new search origin. It then sets the new step size S = S/2, and repeats similar search for two more iterations until S = 1. At that point it finds the location with the least cost function and the macro block at that location is the best match. The calculated motion vector is then saved for transmission.
The following table shows the comparison of the Block Distortion Mean for the next comparison, next 10 comparisons and all comparisons [6]. The Three Step Search algorithm calculates the BDM by searching the best match block in three steps using the 9 check points. In this algorithm no search range is defined. It's computational cost is very less.

Table 3.3 BDM values using Three Step search algorithm

THREE STEP SEARCH ALGORITHM			
Block Distortion Mean	Next comparison	Next 10 comparisons	All comparisons
	27.35	27.81	17.99

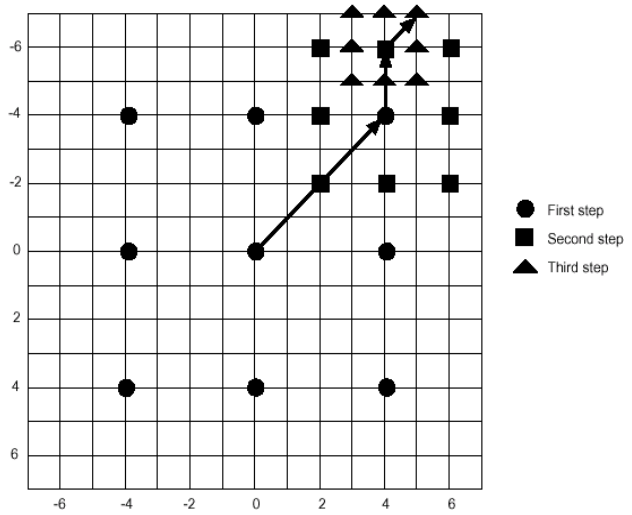


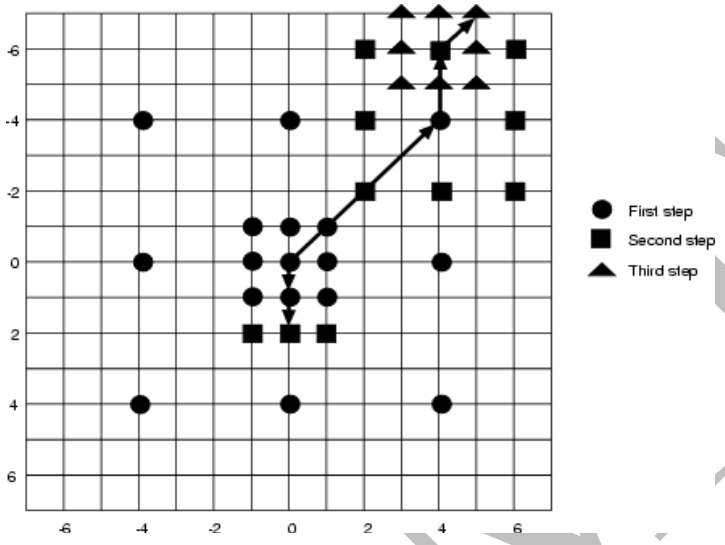
Fig 5- TSS Algorithm

5.4 NEW THREE STEP SEARCH ALGORITHM

The new three step search algorithm (NTSS) has been proposed by Li, Zeng and Liou in 1994. It is a modified version of the three step search algorithm for searching small motion video sequences. For these video sequences, the motion vector distribution is highly center biased. Therefore, additional 8 neighboring checking points are searched in the first step of NTSS. Figure shows two search paths with d=7. The center path shows the case of searching small motion. In this case, the minimum BDM point of the first step is one of the 8 neighboring checking points. The search is halfway stopped with matching three more neighboring checking points of the first step's minimum BDM point. The number of checking points required is (17+3)=20. The upper right path shows the case of searching large motion[8]. In this case, the minimum BDM point of the first step is one of the outer eight checking points. Then the searching procedure proceeds in the same way as in the TSS algorithm. The number of checking points required in this step is (17+8+8) =33.
Table 3.3 BDM values using New Three Step search

ALGORITHM

NEW THREE STEP SEARCH ALGORITHM			
Block Distortion Mean	Next comparison	Next 10 comparisons	All comparisons
	18.76	26.42	20.32



5.5 FOUR-STEP SEARCH ALGORITHM

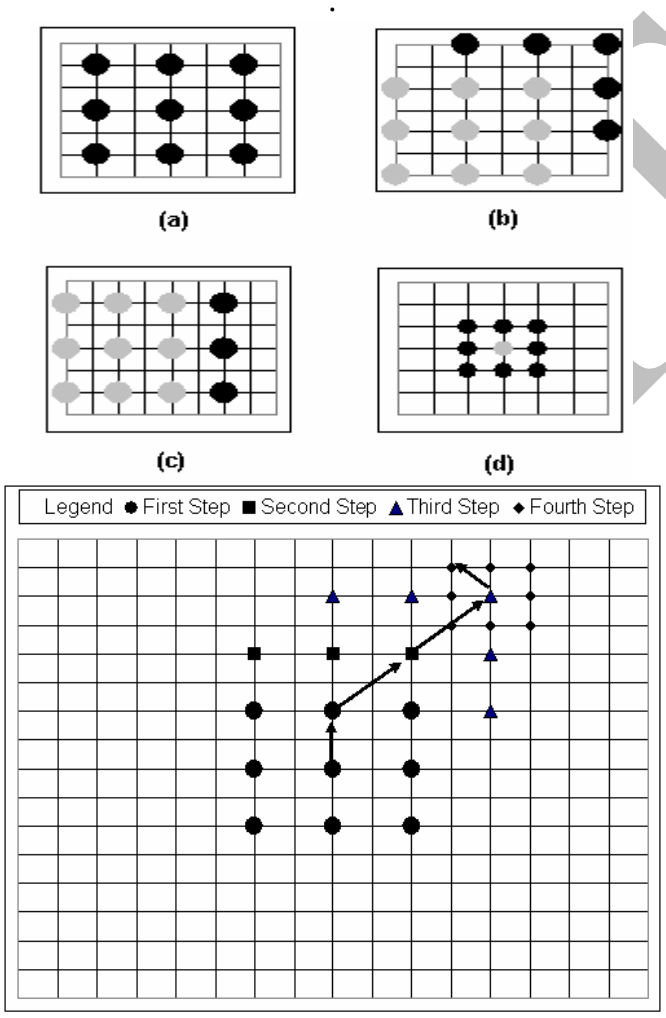
For the maximum motion displacements of ±7, the proposed 4SS algorithm utilizes a center-biased search pattern with 9 checking points on a 5´5 window in the first step instead of a 9´9 window in the 3SS. The center of the search window is then shifted to the point with minimum block distortion measure (BDM)[5]. The search window size of the next two steps is depended on the location of the minimum BDM points. If the minimum BDM point is found at the center of the search window, the search will go to the final step (Step 4) with 3´3 search window. Otherwise, the search window size is maintained in 5´5 for step 2 or step 3. In the final step, the search window is reduced to 3´3 and the search stops at this small search window. The 4SS algorithm is summarized as follows:

Step 1: A minimum BDM point is found from a 9 checking points pattern on a 5´5
Window located at the center of the 15´15 searching area as shown in Fig.
2a. If the minimum BDM point is found at the center of the search window,
go to Step 4; otherwise go to Step 2.

Step 2: The search window size is maintained in 5´5. However, the search pattern Will depend on the position of the previous minimum BDM point.
a) If the previous minimum BDM point is located at the corner of the previous search window, 5 additional checking points as shown in Fig.2b are used.
b) If the previous minimum BDM point is located at the middle of horizontal or vertical axis of the previous search window, 3 additional checking points as shown in Fig. 2c are used. If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 3.

Step 3: The searching pattern strategy is the same as Step 2, but finally it will go to Step 4.

Step 4: The search window is reduced to 3´3 as shown in Fig. 2d and the direction of the overall motion vector is considered as the minimum BDM point among these 9 searching points.
From the algorithm, we can find that the intermediate steps of the 4SS may be skipped and then jumped to the final step with a 3´3 window if at any time the minimum BDM point is located at the center of the search window. Based on this four-step search pattern, we can cover the whole 15´15 displacement window even only small search windows 5´5 and 3´3 are used[4]. There are overlapped checking points on the 5´5 search windows in the step 2 and step 3, thus the total number of checking points is varied from (9+8) = 17 to (9+5+5+8) = 27. The worst case computational requirement of the 4SS is 27 block matches, it will happen for the estimation of large4 - movement.



CONCLUSION

In this review paper, the block motion estimation (for low bit-rate video compression) was formulated as an optimization problem in which the objective function is defined by an average distance measure and a center-biased constraint has been taken into consideration. As a feasible solution, we proposed a new four-step search algorithm for fast motion estimation. We constructed a center-biased search point pattern in the first step by adding 8 extra checking points which are the neighbors of the window center. This has significantly reduced the average distance. In the mean time, we introduced the use of a halfway stop technique to keep the NFSS algorithm compatible to FSS in terms of computational complexity. Experimental results showed that, compared to FSS, NFSS always provides smaller motion compensation errors and in particular is much more robust, no matter it is used alone or combined with motion field sub sampling techniques.

Based on the center-based global minimum motion vector distribution characteristic of real world image sequence, a new 4SS algorithm for fast block-based motion estimation is proposed in this paper[7]. Experimental results show that the proposed 4SS algorithm performs better than the well-known 3SS and have similar performance to the N3SS in terms of mean-square error measure with smaller computational requirement. In addition, 4SS is more robust as compared with 3SS and N3SS. It is because the performance of 4SS is maintained for image sequence that contains complex movement such as camera zooming and fast motion. On the other hand, the 4SS also possesses the regularity and simplicity of hardware-oriented features.

REFERENCES

1. R. Li, B. Zeng, and M. L. Liou, “A new three-step search algorithm for block motion IEEE Trans. on Circuits and Systems for Video Technology, vol. 4, No. pp. 438-442, Aug. 1994.
2. CCITT SGXV, “Description of reference model 8 (RM8),” Document 525, Working Party XV/4, Specialists Group on Coding for Visual Telephony, June 1989.
3. ISO/IEC 11172-2 (MPEG-1 Video), “Information Technology – Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to about 1.5 Mbit/s: Video,” 1993.
4. ISO/IEC 13818-2 | ITU-T H.262 (MPEG-2 Video), “Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Video,” 1995.
5. 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.
6. 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.
7. R. Srinivasan and K. R. Rao, “Predictive coding based on efficient motion estimation,” IEEE Trans. Commun., vol. COM-33, pp. 888-896, Aug. 1985.
8. S. Kappagantula and K. R. Rao, “Motion compensated interframe image prediction,” IEEE Trans. Commun., vol. COM-33, pp. 1011-1015, Step. 1985.
9. M. Ghanbari, “The cross-search algorithm for motion estimation,” IEEE Trans. Commun., vol. COM-38, No. 7, pp. 950-953, July 1990.
10. L.W. Lee, J.F. Wang, J.Y. Lee, and J.D. Shie, “Dynamic search-window adjustment and interlaced search for block-matching algorithm,” IEEE Trans. Circuits and Systems for Video Tech., vol. 3, pp. 85-87, Feb. 1993.
11. S. C. Kwatra, C-M Lin and W. A. Whyte, “An adaptive algorithm for motion compensated color image coding,” IEEE Trans. Commun., col. COM-35, pp. 747-754, July 1987.
12. B. Liu and A. Zaccarin, “New fast algorithm for estimation of block motion vectors,”
13. IEEE Trans. Circuits and Systems for Video Tech., vol. 3, pp. 148-157, Apr. 1993