International Journal of Scientific Engineering and Technology
Volume No.5 Issue No.6, pp: 352-357

ISSN:2277-1581
01 June 2016

Optimization of Block base Searching Methods for Motion Estimation in Video Compression

Om Prakash, Prof. Nar Singh
Department of Electronics and Communication, University of Allahabad, Allahabad U.P.Indai
Email:opsir1@gmail.com, nsjk53@rediffmail.com

Abstract- In area of video compression, Motion Estimation is one of the most important modules to design and implementation of any the video encoder. It consumes more than 85% of video encoding time due to searching of a candidate block in the search window of the reference frame. To minimize the search time on block matching, a simplified and efficient block matching method is required to improve Motion Estimation. Today, various searching methods have been proposed to perform motion estimation. This paper has presented comparative picture of various searching methods like Exhaustive Search (ES), Three Step Search (TSS), New Three Step Search (NTSS), Four Step Search (4SS), and Diamond Search (DS), Adaptive Rood Pattern Search (ARPS)

***Keywords: ME*, BMM, ES, TSS, NTSS, 4SS, SES, DS, ARPS.**

I-Video Compression

Video compression technologies are about reducing and removing redundant video data so that a digital video file can be effectively sent over a network as well as stored on computer disks in limited space. With efficient compression techniques, a significant reduction in file size can be achieved with little or no adverse effect on the visual quality. The video quality, however, can be affected if the file size is further reduced by raising the compression level for a given compression technique.

Now these days, compression technologies are available. Most network video vendors today use standard compression techniques. Standards are important in ensuring compatibility and interoperability. They are particularly relevant to video compression since video may be used for different purposes and, in some video surveillance applications, needs to be viewable many years from the recording date. By deploying standards, end users are able to pick and choose from different vendors, rather than be tied to one supplier when designing a video surveillance system.

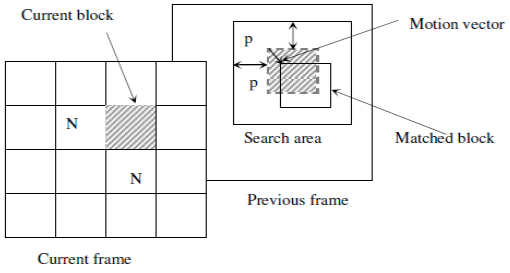
II-Motion Estimation

In video editing motion estimation is a type of video compression scheme. The motion estimation process is done of

Displacement vector is assigned to all pixels within a block. The motion model assumes that an image is usually composed major drawback in the presence of zoom but the block matching technique is able to estimate closely the true zooming motion. And hence the block matching ME results globally in motion fields more representative of true motion in the scene [4]. The basic concept of block base ME is depicted in figure 1 as below.

III- Block Base Motion Estimation

rigid objects in translational motion. Although the assumption of translational motion is often considered by the coder to find the motion vector pointing to the best prediction macroblock in a reference frame or field. For compression redundancy between adjacent frames can be exploited where a frame is selected as a reference and subsequent frames are predicted from the reference using motion estimation. The motion estimation process analyzes previous or future frames to identify blocks that have not changed, and motion vectors are stored in place of blocks. The process of video compression using motion estimation is also known as interframe coding.



]

Figure 1: Basic concept of block motion estimation

Motion estimation (ME) techniques have been successfully applied in motion compensated predictive coding for reducing temporal redundancies. They belong to the class of nonlinear predictive coding techniques. An efficient representation of motion is critical in order to reach high performance in video coding. ME techniques should on one hand provide good prediction, but on the other hand should have low computational load. The purpose of ME is indeed to globally minimize the sum of these two terms. As a compromise, block matching ME, even though not optimal, has been universally used [1]-[3] in interframe motion compensated (MC) predictive coding since its computational complexity is much lower than optical flow and pel-recursive methods. In block based ME image is partitioned into blocks and the same

The block base matching is a temporal compression technique used in the video encoding. The main purpose of this method is to determine the displacements of each block of pixels between two successive frames. This technique, performed in the step of motion estimation, occupies the majority of the total time of video coding. The object of this work is to give a comparative picture of various search methods using Block Matching. This picture does not focus only on the complexity and computation time of each method, but it also gives objective

International Journal of Scientific Engineering and Technology
Volume No.5 Issue No.6, pp: 352-357
ISSN:2277-1581
01 June 2016

and subjective quality assessment of decoded video frame by means of each search method.

The motion of a block of MxN pixels centered at point (x, y) within a frame interval is estimated as shown in Figure 2. The goal is to find the best match or the least distortion block between the MxN blocks in the frame k (current frame) and a corresponding block in the frame (k-1) (previous frame) within a search area of size $(M+2m_2)x(N+2n_1)$ within the previous frame, in a given search window. The range of the motion vector is constrained by the search window. Block matching methods (BMMs) ignore rotational motion and assume that all pixels within the MxN block have the same uniform motion. FS is the most straightforward and optimal block matching method (BMM) which searches exhaustively inside the search window to find the motion vector. It searches the best match block of the current frame from the candidate blocks inside the search window in the previous frame. The candidate that gives the best match is chosen as the estimated motion vector (MV). Despite the very heavy computations required in FS it is still widely used in video coding applications due to its simplicity and ease of hardware implementation. Block matching methods differ in:
1. The matching criteria (e.g. MSE, MAD, SAD)
2. The search method (e.g. FS, TSS, FSS)
3. The determination of block size (e.g., hierarchical, adaptive)

In order to measure the similarity between current frame block and a candidate block of the reference frame, various criteria can be used as a measure for the block matching between the two blocks, such as minimum MSE (mean square error), minimum MAD (mean absolute difference) or minimum SAD (sum of absolute differences). Among these SAD is often chosen because it achieves the same performance as the others, without requiring any multiplication in the calculation. The distortion measure between the block in the present frame and the displaced block in the previous frame can be defined as follows:

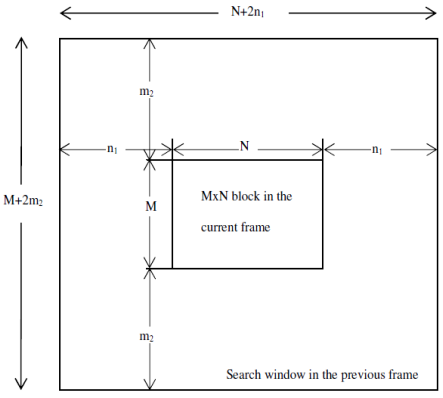


Figure 2: Full search BMM

Mean Square Error

Mean square error (MSE) can be defined as follows:

$$MSE(i,j) = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left(X_{m,n} - X_{m}^{R}\right) \quad \ldots\ldots (1)$$

is the pel intensity at row and column in the present frame and refers to row and column in the reference frame. The minimum MSE criterion is rarely used in very large scale integration (VLSI) implementation because it is difficult to realize the square operation in hardware.

Mean Absolute Difference

Mean absolute difference (MAD) is defined as follows:

$$MAD(i,j) = \frac{1}{MN}\sum_{m=1}^{M}\sum_{n=1}^{N}\left|X_{m,n} - X_{m+i,n+j}^{R}\right| \quad \ldots\ldots (2)$$

It is well known that the performance of the MAD criterion decreases, as the search area becomes large due to the presence of local minimum, still MAD is widely used for VLSI implementations.

Sum of Absolute Difference

Sum of absolute difference (SAD) is defined as:

$$SAD(i,j) = \sum_{m=1}^{M}\sum_{n=1}^{N}\left|X_{m,n} - X_{m+i,n+j}^{R}\right| \quad \ldots\ldots (3)$$

This is often used because of its simplicity

Matching Performance Measuring Parameters

Mean square error (MSE) and peak signal to noise ratio (PSNR) are used to evaluate the subjective quality of a reconstructed video sequence. PSNR is defined as follows:

PSNR=10log10 (2552/MSE) ……. (4)

For performance comparison PSNR difference is also calculated. It is defined as the difference in PSNR of the proposed method with respect to FS method. Within a video codec it is also advisable to calculate bit rate at different quantization parameters for the rate-distortion (bit rate versus PSNR) comparison. Apart from the prediction error criterion, computational complexity is also a key criterion for the performance evaluation of fast BMMs. For fast fixed search pattern BMAs that reduce computational complexity by limiting the number of checking points, such as TSS, FSS, NTSS,4SS etc., the computational complexity can be directly compared by counting the number of checking points required.

VI. Block base Searching Methods

A. Full or Exhaustive Search(FS or ES)

This method is the most computationally expensive block matching method of all. It 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 method. Fast block matching methods try to achieve the same PSNR doing as little computation as possible. The obvious disadvantage to FS is that the larger the search window gets the more computations it requires. The basic idea of FS is given in figure 3 as below.

International Journal of Scientific Engineering and Technology
Volume No.5 Issue No.6, pp: 352-357
ISSN:2277-1581
01 June 2016

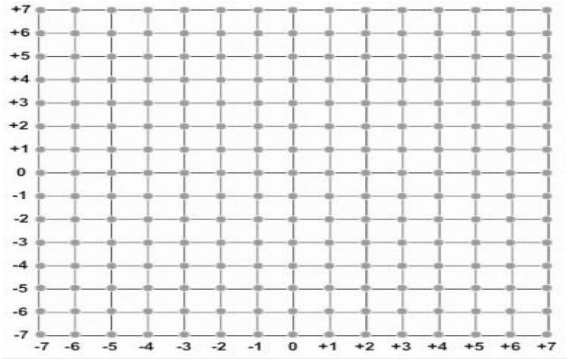


Figure.3: Full Search or Exhaustive Search.

B. Three Step Search(TSS)

This is one of the earliest attempts at fast block matching methods and dates back to mid 1980s. The general idea is represented in Figure 4. 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 and the macro block at that location is the best match. At that point it finds the location with the least cost function.

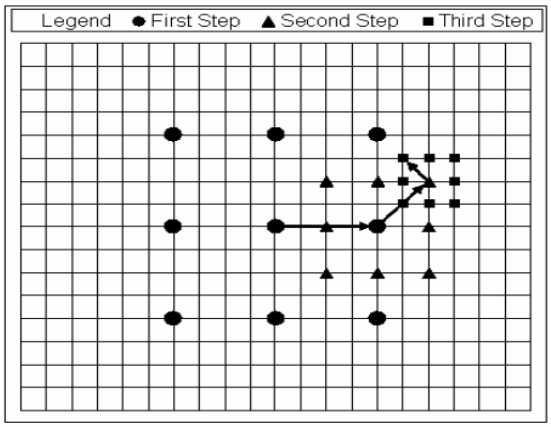


Figure 4: Three step search methods.

The calculated motion vector is then saved for transmission. It gives a flat reduction in computation by a factor of 9. So that for p = 7, ES will compute cost for 225 macro blocks whereas TSS computes cost for 25 macro blocks. The idea behind TSS is that the error surface due to motion in every macro block is unimodal. A unimodal surface is a bowl shaped surface such that the weights generated by the cost function increase monotonically from the global minimum.

C. Simple and Efficient TSS (SES)

SES [9] is another extension to TSS and exploits the assumption of unimodal error surface. The main idea behind the method is that for a unimodal surface there cannot be two minimums in opposite directions and hence the 8 point fixed pattern search of TSS can be changed to incorporate this and save on computations.

The method still has three steps like TSS, but the innovation is that each step has further two phases. The search area is divided into four quadrants and the method checks three locations A, B and C as shown in Figure 5. A is at the origin and B and C are S = 4 locations away from A in orthogonal directions. Depending on certain weight distribution amongst the three the second phase selects few additional points drawn in figure 5. The rules for determining a search quadrant for seconds phase are as follows:

If MAD (A) _ MAD (B) and MAD (A) _ MAD(C),

Select (b);

If MAD (A) _ MAD (B) and MAD (A) _ MAD(C), select (c);

If MAD (A) < MAD (B) and MAD (A) < MAD(C), select (d);

If MAD (A) < MAD (B) and MAD (A) _ MAD(C), select (e);

Once we have selected the points to check for in second phase, we find the location with the lowest weight and set it as the origin. We then change the step size similar to TSS and repeat the above SES methods again until we reach S = 1. The location with the lowest weight is then noted down in terms of motion vectors and transmitted. The basic idea of this method is illustrated in Fig 6. Although this method saves a lot on computation as compared to TSS, it was not widely accepted for two reasons. Firstly, in reality the error surfaces are not strictly unimodal and hence the PSNR achieved is poor compared to TSS. Secondly, there was another method, Four Step Search, that had been published a year before that presented low computational cost compared to TSS and gave significantly better PSNR.

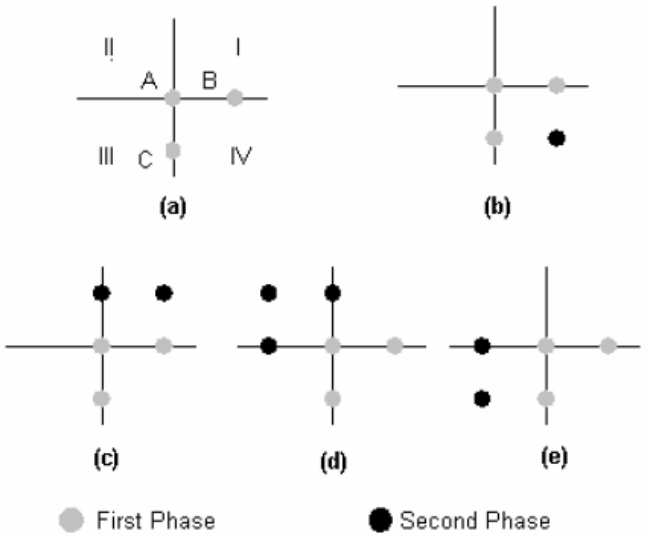


Figure: 5. Search patterns corresponding to each selected quadrant: (a) Shows all quadrants (b) quadrant I is selected Quadrant II is selected

International Journal of Scientific Engineering and Technology
Volume No.5 Issue No.6, pp: 352-357
ISSN:2277-1581
01 June 2016

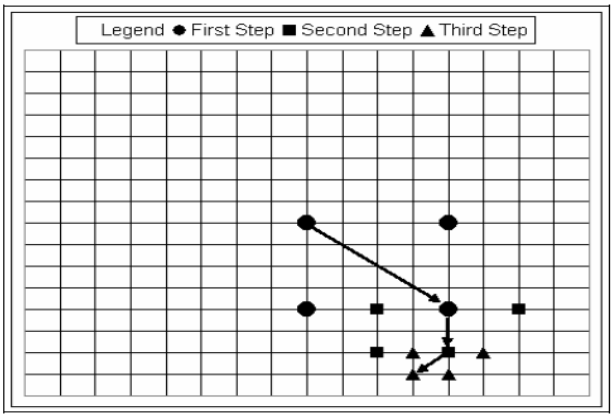


Figure 6: SES method.

D. New Three Step Search(NTSS)

NTSS [8] improves on TSS results by providing a center biased searching scheme and having provisions for half way stop to reduce computational cost. It was one of the first widely accepted fast methods and frequently used for implementing earlier standards like MPEG 1 and H.261. The TSS uses a uniformly allocated checking pattern for motion detection and is prone to missing small motions. The NTSS process is illustrated graphically in Fig 7. In the first step 16 points are checked in addition to the search origin for lowest weight using a cost function of these additional search locations, 8 are a distance of S = 4 away (similar to TSS) and the other 8 are at S = 1 away from the search origin. If the lowest cost is at the origin then the search is stopped right here and the motion vector is set as (0, 0). If the lowest weight 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 weights adjacent to it. Depending on which point it is we might end up checking 5 points or 3 points (Fig 7(b) & (c)). The location that gives the lowest weight is the closest match and motion vector is set to that location. On the other hand if the lowest weight after the first step was one of the 8 locations at S = 4, then we follow the normal TSS methods. Hence although this process might need a minimum of 17 points to check every macro block, it also has the worst-case scenario of 33 locations to check.

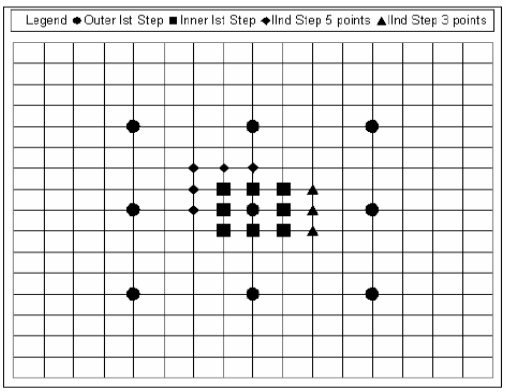


Figure: 7. New Three Step Search block matching: Big circles are checking points in the first step of TSS and the squares are the extra 8 points added in the first step of NTSS. Triangles and diamonds are second step of NTSS showing 3 points and 5 points being checked when least weight in first step is at one of the 8 neighbors of window center.

E. Four Step Search(4SS)

Similar to NTSS, 4SS [10] also employs center biased searching and has a halfway stop provision. 4SS sets a fixed pattern size of S = 2 for the first step, no matter what the search parameter *p* value is. Thus it looks at 9 locations in a 5x5 window. If the least weight is found at the center of search window the search jumps to fourth step. If the least weight is at one of the eight locations except the center, then we make it the search origin and move to the second step. The search window is still maintained as 5x5 pixels wide. Depending on where the least weight location was, we might end up checking weights at 3 locations or 5 locations. The patterns are shown in figure. 8. Once again if the least weight location is at the center of the 5x5 search window we jump to fourth step or else we move on to third step. The third is exactly the same as the second step. IN the fourth step the window size is dropped to 3x3, i.e. S = 1. The location with the least weight is the best matching macro block and the motion vector is set to point 0 that location. A sample method is shown in Fig 9. This search method has the best case of 17 checking points and worst case of 27 checking points.

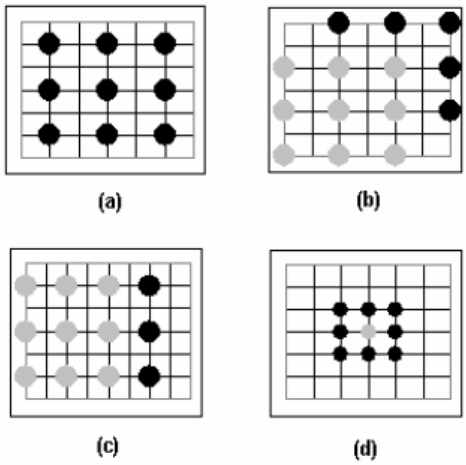


Figure 8: Search patterns of the 4SS. (a) First step (b) Second/Third step(c) Second/ Third Step (d) Fourth Step.

F. Diamond Search(DS)

DS [11] method is exactly the same as 4SS, 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 method 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 methods are illustrated in Fig. 10. Just like in FSS, the first step uses LDSP and if the least weight is at the center location we jump to fourth step. The

ISSN:2277-1581
01 June 2016

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 methods shown in Fig.10. The last step uses SDSP around the new search origin and the location with the least weight is the best match.

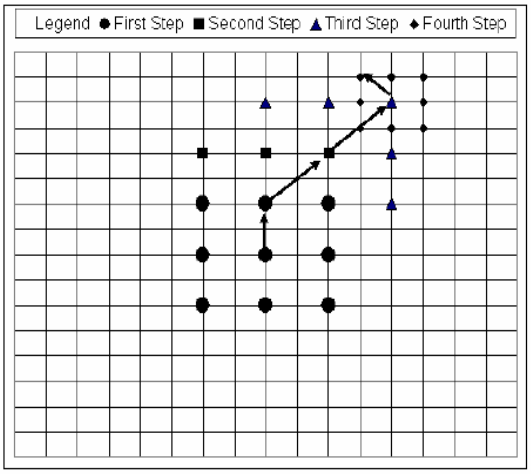


Figure 9: Four Step Search method

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 method 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.

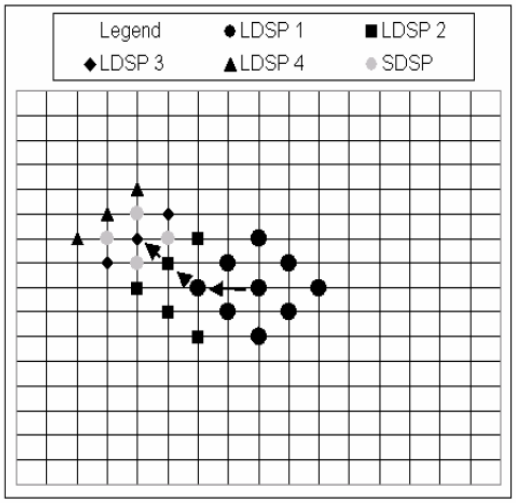


Figure: 10. Diamond Search methods. This figure shows the large diamond search pattern and the small diamond search pattern

G. Adaptive Rood Pattern Search(ARPS)

ARPS [12] method 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 method uses the motion vector of the macro block to its immediate left to predict its own motion vector. An example is shown in figure. 11. 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 11, 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. The methods keep on doing SDSP until least weighted point is found to be at the center of the SDSP. A further small improvement in the method can be to check for Zero Motion Prejudgment [12] using which the search is stopped half way if the least weighted point is already at the center of the rood pattern.

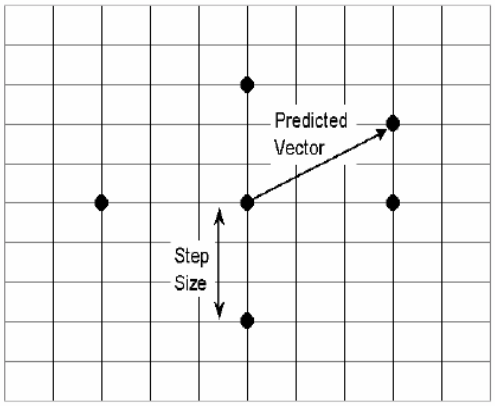


Figure 11: Adaptive Root Pattern Search: The predicted motion vector is (3,-2), and the step size S = Max (|3|, |-2|) = 3.

The main advantage of this method 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 center, then again ARPS save on computations by directly jumping to that vicinity and using SDSP, whereas DS takes its time doing LDSP. The precaution has to be taken not only to repeat the computation at point that were checked but also needs to Care has to be taken to not repeat the computations at points that were earlier but also needs to be taken when the predicted motion vector turns 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.

V. Simulation and Discussion

In this paper, various block matching searching methods for motion estimation are implemented in MATLAB 2013b for Garden video sequences having SIF (352x240) format with distance of 2 between current frame and reference frame is used to generate the frames by frame and compared with well known parameters like peak signal to noise ratio (PSNR) and computations have shown in figure 12,. There are total 20 (data

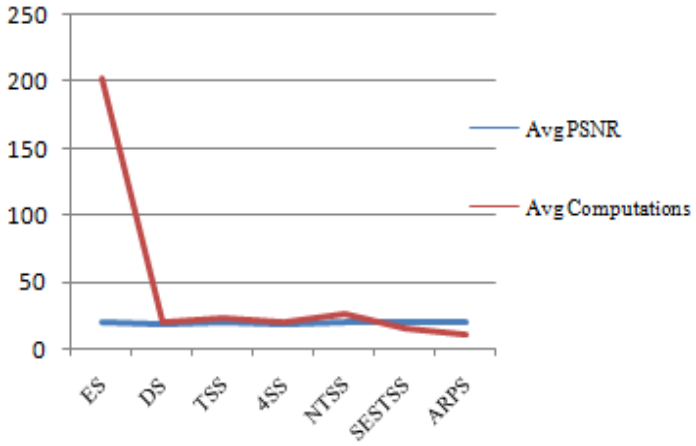
ISSN:2277-1581
01 June 2016

set) frames with block size 16x16 having search parameter p=7 are taken for testing and experimentation purpose.

The optimization of average value of PSNR and its average value of computations for various searching methods has plotted and shown in figure 12. The average PSNR performance of most of the methods is nearly close in their nature except ES. While the average value of computations for ARPS method is least relatively others. In other word , we can say that ARPS is computationally least expensive method have optimum value.

VI. Conclusion

As per above discussion of result and its analysis, we are in position to say that the ARPS method has optimum, performing better result in terms of computation with slight compromisation of PSNR. Hence, this method can be recommended for better motion estimation.



VII. Future Aspects

These searching algorithms are limited by search speed and pattern so that very quick moments it can't find exact motion vector. So we can try different search patterns and In future other evolutionary computing techniques also can be tried for the better results. Three important factors Block size, search area, matching criteria can be varied such as Variable block size, large search area for complex motions and small search area for low complex motions and bidirectional motion estimation also we will try to implement new techniques which will further reduce the complexity of MPEG4 and H.264 video coding.

References

i. *Y. Nakaya and H. Harashima, "Motion compensation based on spatial transformation," IEEE Trans. on Circuits and Systems for Video Technology, Vol. 4, No. 3, pp. 339-356, Jun. 1994.*

ii. *P. Cicconi and H. Nicolas, "Efficient region-based motion estimation and symmetry oriented segmentation for image sequence coding," IEEE Trans. on Circuits and Systems for Video Technology, Vol. 4, No. 3, pp. 357-364, Jun. 1994.*

iii. *Y. Yokoyama, Y. Miyamoto and M. Ohta, "Very low bit rate video coding using arbitrarily shaped region-based motion compensation," IEEE Trans. on Circuits and Systems for Video Technology, Vol. 5, No. 6, pp. 500-507, Dec. 1995. 7*

iv. *F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," Proc. of IEEE, Vol. 83, No. 6, pp. 858-876, Jun. 1995*

v. *Borko Furht, Joshua Greenberg, Raymond Westwater, Motion Estimation Methods For Video Compression. Massachusetts: Kluwer Academic Publishers, 1997. Ch. 2& 3.*

vi. *M. Ghanbari, Video Coding, An Introduction to Standard Codecs, London: The Institute of Electrical Engineers, 1999. Ch.2, 5, 6, 7 & 8*

vii. *Iain E. G. Richardson, Video Codec Design, West Sussex: John Wiley & Sons Ltd., 2002, Ch. 4, 5, & 6.*

viii. *Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Method for Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 4., no. 4, pp. 438-442, August 1994.*

ix. *Jianhua Lu, and Ming L. Liou, "A Simple and Efficent Search Method for Block-Matching Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 7, no. 2, pp. 429-433, April 1997*

x. *Lai-Man Po, and Wing-Chung Ma, "A Novel Four-Step Search Method for Fast Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 6, no. 3, pp. 313-317, June 1996.*

xi. *Shan Zhu, and Kai-Kuang Ma, " A New Diamond Search Method for Fast Block-Matching Motion Estimation", IEEE Trans. Image Processing, vol 9, no. 2, pp. 287-290, February 2000.*

xii. *Yao Nie, and Kai-Kuang Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", IEEE Trans. Image Processing,vol 11, no. 12, pp. 1442-1448, December 2002.*

xiii. *Chun-Ho Cheung, and Lai-Man Po, "A Novel Small Cross-Diamond Search Method for Fast Video Coding and Video Conferencing Applications", Proc. IEEE ICIP, September 2002.*

xiv. *Chun-Ho Cheung, and Lai-Man Po, "A Novel Cross-Diamond Search Method for Fast Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, vol 12., no. 12, pp. 1168-1177, December 2002.*

xv. *C. W. Lam, L. M. Po and C. H. Cheung, "A New Cross-Diamond Search Method for Fast Block Matching Motion Estimation", Proceeding of 2003 IEEE International Conference on Neural Networks and Signal Processing, pp. 1262-1265, Dec. 2003, Nanjing, China. Fig. 13. Cross Search Pattern: This pattern is used by CDS, SCDS, and NCDS as their search start pattern. While CDS uses all 9 points, SCDS and NCDS use only the inner 5 points.*