I.J. Image, Graphics and Signal Processing, 2013, 10, 9-18
Published Online August 2013 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2013.10.02

Evaluation and Comparison of Motion Estimation Algorithms for Video Compression

Avinash Nayak and Bijayinee Biswal
Ajay Binay Institute of Technology, Odisha, India
nayak.av@gmail.com

S. K. Sabut
M.S. Ramaiah Institute of Technology, Bangalore, India
sukanta207@gmail.com

Abstract — Video compression has become an essential component of broadcast and entertainment media. Motion Estimation and compensation techniques, which can eliminate temporal redundancy between adjacent frames effectively, have been widely applied to popular video compression coding standards such as MPEG-2, MPEG-4. Traditional fast block matching algorithms are easily trapped into the local minima resulting in degradation on video quality to some extent after decoding. In this paper various computing techniques are evaluated in video compression for achieving global optimal solution for motion estimation. Zero motion prejudgment is implemented for finding static macro blocks (MB) which do not need to perform remaining search thus reduces the computational cost. Adaptive Rood Pattern Search (ARPS) motion estimation algorithm is also adapted to reduce the motion vector overhead in frame prediction. The simulation results showed that the ARPS algorithm is very effective in reducing the computations overhead and achieves very good Peak Signal to Noise Ratio (PSNR) values. This method significantly reduces the computational complexity involved in the frame prediction and also least prediction error in all video sequences. Thus ARPS technique is more efficient than the conventional searching algorithms in video compression.

Index Terms — Video Compression, Motion Estimation, Full Search Algorithm, Adaptive, Rood Pattern Search, Peak Signal to Noise Ratio

I. INTRODUCTION

Importance of digital video coding has increased significantly since the 90s when MPEG-1 first came to the picture. Compared to analog video, video coding achieves higher data compression rates without significant loss of subjective picture quality which eliminates the need of high bandwidth as required in analog video delivery to a large extent. Digital video is immune to noise, easier to transmit and is able to provide a more interactive interface to the users [1]. The specialized nature of video applications has led to the development of video processing systems having different size, quality, performance, power consumption and cost.

A major problem in a video sequence is the high requirement of memory space for storage. A typical system needs to send dozens of individual frames per second to create an illusion of a moving picture. For this reason, several standards for compression of the video have been developed. Each individual frame is coded to remove the redundancy [2]. Furthermore, between consecutive frames, a great deal of redundancy is removed with a motion compensating system. Motion estimation and compensation are used to reduce temporal redundancy between successive frames in the time domain.

A number of fast block matching motion estimation algorithms were considered in different video coding standards because massive computation were required in the implementation of exhaustive search (ES). In order to speed up the process by reducing the number of search locations, many fast algorithms have been developed, such as the existing three-step search (TSS) algorithm [3]. The Three Step Search method is based on the real world image sequence's characteristic of centre-biased motion vector distribution, and uses centre-biased checking point patterns and a relatively small number of search locations to perform fast block matching. In order to reduce the computational complexity for motion estimation and improve the reliability of the image sequences for super-resolution reconstruction, an effective three-step search algorithm is presented. Based on the center-biased characteristic and parallel processing of the motion vector, the new algorithm adopts the multi-step search strategy [4].

A simple, robust and efficient fast block-matching motion estimation **(**BMME**)** algorithm called diamond search, which employs two search patterns. The first pattern, called large diamond search pattern (LDSP), comprises nine checking points from which eight points surround the center one to compose a diamond shape. The second pattern consisting of five checking points forms a smaller diamond shape, called small diamond search pattern (SDSP).

A simple fast block-matching algorithm (BMA), called adaptive rood pattern searches (ARPS), which consist of two sequential search stages: 1) initial search and 2) refined local search. The initial search is performed only once at the beginning for each MB. This removes unnecessary intermediate search. For the initial search stage, ARP is proposed, based on the available motion vectors (MVs) of the neighboring MBs. In the next stage, a unit-size rood pattern (URP) is exploited repeatedly, and unrestrictedly, until the final MV is found. In this paper we have evaluated the following four algorithms: Exhaustive Search (ES), Three Step Search (TSS), Diamond Search (DS), and ARPS.

II. METHODS

In a conventional predictive coding [5-6], the difference between the current frame and the predicted frame is encoded. The prediction is done using any of the BMA. BMA are used to estimate the motion vectors. Block-matching consumes a significant portion of time in the encoding step.

A. Block matching algorithm

Block matching algorithm (BMA) is widely used in many motion-compensated video coding systems such as H.261 and MPEG standards to remove interframe redundancy and thus achieve high data compression [7, 8]. The process of block-matching algorithm is illustrated in Fig.1. Motion estimation is performed on the luminance block in which the present frame is matched against candidate blocks in a search area on the reference frame for coding efficiency. The best candidate block is found and its motion vector is recorded. Typically the input frame is subtracted from the prediction of the reference frame, thus interframe redundancy is removed and data compression is achieved. At receiver end, the decoder builds the frame difference signal from the received data and adds it to the reconstructed reference frames. This algorithm is based on a translational model of the motion of objects between frames [9].

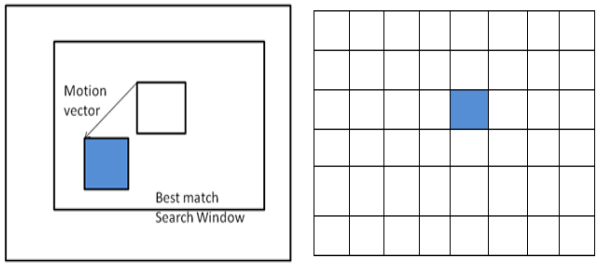


Figure 1. Block-matching motion estimation

The block-based motion vectors can be estimated by using block matching, which minimizes a measure of matching error. The destination of motion estimation is to find macro-block (MB) in the reference frame which has the smallest difference from the MB in the current frame. The difference is denoted by Sum of Absolute Difference (SAD), as shown in equation (1). The SAD is the most popular matching criteria used for block-based motion estimation.

$$SAD(m,n)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|c(i,j)-s(i+m,j+n)\right|,\omega\le m,n\le\omega \quad (1)$$

Where SAD (m, n) is the distortion of the candidate block at search position (m, n), {c (x, y) | $0 \le x \le N-1$, $0 \le y \le N-1$} means current block data, {s (x, y)| $-w \le x \le w+N-1$, $-w \le y \le w+N-1$} stands for search area data, the search range is [-w, w], the block size is N×N. $(2w+1)^2$ motion vectors to be checked .

Consider a block of pixels of size N × N in the reference frame, at a displacement of, where I and j are integers with respect to the candidate block position. SAD makes the error values as positive, but instead of summing up the squared differences, the absolute differences are summed up. The SAD criterion shown in equation (1) requires N^2 computations of subtractions with absolute values and additions N2 for each candidate block at each search position. The absence of multiplications makes this criterion computationally more attractive and facilitates easier hardware implementation. Peak-Signal-to-Noise-Ratio (PSNR) given by equation (2) characterizes the motion compensated image that is created by using motion vectors and macro blocks from the reference frame.

$$\Gamma = Round\ |\overrightarrow{MV}_{predicted}| = Round\ \sqrt{[\mathrm{MV}^2{}_{\mathrm{predicted}}(x)+\mathrm{MV}^2{}_{\mathrm{predicted}}(y)]} \quad (2)$$

Where $MV_{predicted}(x)$ and $MV_{predicted}(y)$ are the horizontal and vertical components of the predicted MV, respectively. Operator "Round" performs rounding operation, which takes the nearest integer value of the argument.

B. Exhaustive search (ES) algorithm

The exhaustive search (ES) algorithm also known as Full Search is the most computationally expensive block matching algorithm. This algorithm calculates the cost function at each possible location in the search window. It gives the highest PSNR amongst any block matching algorithm by the best possible match [10]. 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.

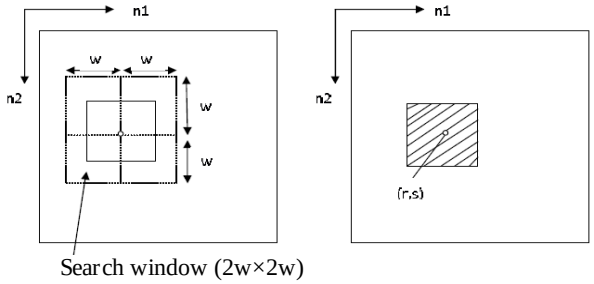


Figure 2. Exhaustive Search Motion Estimation

Consider a block of N × N pixels from the candidates frame at the coordinate position (r, s) as shown in Fig.2 above and then consider a search window having a range ±w in both the directions in the references frame, as shown. For each of the $(2w + 1)^2$ search position (including the current row and the current column of the reference frame), the candidate block is compared with a block of size N × N pixels, according to one of the matching criteria and the best matching block, along with the motion vector is determined only after all the $(2w+1)^2$ search position are exhaustively explored. However, it is highly computational intensive.

C. Three step search (TSS) algorithm

Koga et al introduced this algorithm [11]. It became very popular because of its simplicity and also robust and near optimal performance. It searches for the best motion vectors in a coarse to fine search pattern. The algorithm has steps as described with the help of Fig.3.

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

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

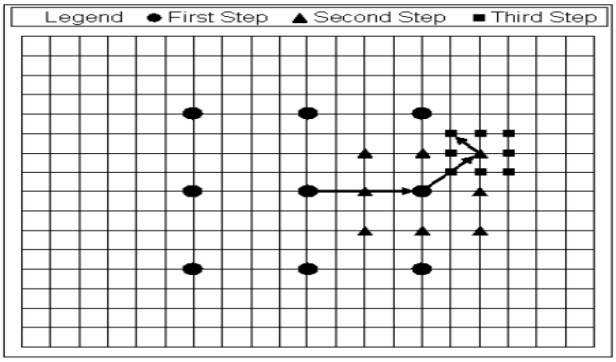


Figure 3. Three Step Search procedures

The point which gives the smallest criterion value among all tested points is selected as the final motion vector m.TSS reduces radically the number of candidate vectors to test, but the amount of computation required for evaluating the matching criterion value for each vector stays the same. TSS may not find the global minimum (or maximum) of the matching criterion; instead it may find only a local minimum and this reduces the quality of the motion compensation system. On the other hand, most criteria can be easily used with TSS [12].

D. Diamond search (DS) algorithm

By exhaustively testing on all the candidate blocks, full search (FS) algorithm gives the global minimum SAD position which corresponds to the best matching block at the expense of highly computation. To overcome this defect, many fast block matching algorithms (BMAs) are developed such as diamond search [13].

The proposed DS algorithm employs two search patterns as illustrated in Fig. 4.1 (a), (b) which are derived from the crosses (×) in Fig. 4. The first pattern, called large diamond search pattern (LDSP), comprises nine checking points from which eight points surround the centre one to compose a diamond shape. The second pattern consisting of five checking points forms a smaller diamond shape, called small diamond search pattern (SDSP) [14,15]. The 13 crosses show all possible checking points within the circle.

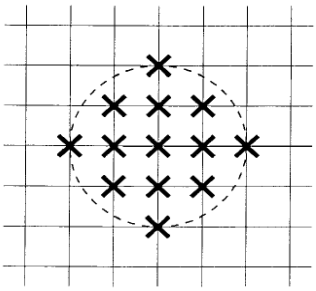
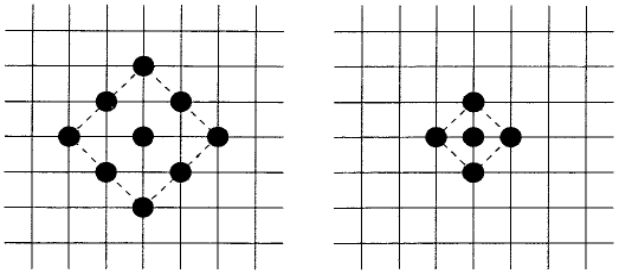


Figure 4. An appropriate search pattern support-circular area with a radius of 2 pels. The 13 crosses show all possible checking points within the circle.



(a) Large Diamond Search Pattern (b) Small Diamond Search Pattern

Figure 4.1. Two search patterns derived from the last figure are employed in the proposed DS algorithm.

Among the five checking points in SDSP, the position yielding the MBD provides the motion vector of the best matching block [16]. The DS algorithm is performed 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 centred 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 4: 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) algorithm

As we have observed, a small search pattern made up by compactly spaced search points is more suitable than a large search pattern containing sparsely spaced search points in detecting small motions, because only a small number of positions around the search window center are necessary to be checked [17]. The speed and accuracy of

pattern-based search algorithms intimately depend on the size of the search pattern and the magnitude of the target MV. Therefore, it is highly desirable to use different search patterns according to the estimated motion behavior for the current block. This boils down to two issues required to be addressed: 1) How to pre-determine the motion behavior of the current block for performing efficient ME? 2) What are the most suitable size and shape of the search pattern(s)?

Regarding the first issue, the current block's motion behavior can be predicted by referring to its neighboring blocks' MVs in the spatial domain. For the second issue, two types of search patterns are used. One is the adaptive rood pattern (ARP) with adjustable rood arm which is dynamically determined for each MB according to its predicted motion behavior. Note that ARP will be exploited only once at the beginning of each MB search [18]. Getting a good starting point for the remaining local search alleviates unnecessary intermediate search and reduces the risk of being trapped into local minimum in the case of long search path. A small, compact, and fixed-size search pattern would be able to complete the remaining local search quickly.

1) Prediction of the target MV

In order to obtain an accurate MV prediction of the current block, two factors need to be considered: 1) Choice of the region of support (ROS) that consists of the neighboring blocks whose MVs will be used to calculate the predicted MV, and 2) Algorithm used for computing the predicted MV.

The spatial ROS is limited to the neighboring block(s) with four promising scenarios as shown in Fig.5. Type A covers all the four neighboring blocks, and +type B is the prediction ROS adopted in some international standards such as H.263 for differential coding of MVs. Type C is composed of two directly adjacent blocks, and type D has only one block that situates at the immediate left to the current block. Calculating the statistical average of MVs in the ROS is a common practice to obtain the predicted MV. The mean and median prediction has been tested in our experiments. Others (such as the weighted average) are either too complex in computation or involving undetermined parameters, they are therefore not considered in our work. Extensive experiments are performed with all four types of ROS and two types of prediction criteria- mean and median. Our experimental results show that these ROSs and prediction criteria yield fairly similar performance in terms of PSNR and the total number of checking points required. Therefore, we adopt the simplest ROS (i.e., type D) in our method, which has the least memory requirement, because only one neighboring MV needs to be recorded.

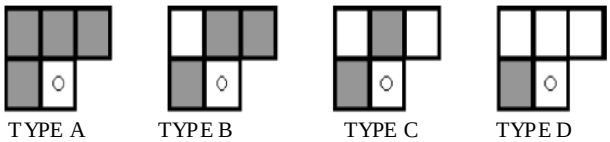


Figure 5. Four types of ROS, depicted by the shaded blocks. The block marked by "○" is the current block.

2) 2) Selection of search patterns

Adaptive Pattern - For the Initial Search: The shape of our rood pattern is symmetrical, with four search points locating at the four vertices, as depicted in Fig.5. The main structure of ARP has a rood shape, and its size refers to the distance between any vertex point and the center point. The choice of the rood shape is first based on the observation of the motion feature of real-world video sequences. The rood shape pattern includes all the horizontal and vertical directions, so it can quickly detect such motion, and the searches will directly – 'jump" into the local region of the global minimum.

Secondly, any MV can be decomposed into one vertical MV component and one horizontal MV component. For a moving object which may introduce MV in any direction, rood-shaped pattern can at least detect the major trend of the moving object, which is the desired outcome in the initial search stage. Furthermore, ARP's symmetry in shape not only benefits hardware implementation, but also increases robustness. It shows that even if the predicted MV could be inaccurate and its magnitude does not match the true motion very well, the rood-shaped pattern which takes all horizontal and vertical directions into consideration can still track the major direction and favor the follow-up refinement process. In addition to the four-armed rood pattern, it is desirable to add the predicted MV into our ARP because it is very likely to be similar to our target MV as shown in Fig.6. By doing so, the probability of detecting the accurate motion in the initial stage will be increased.

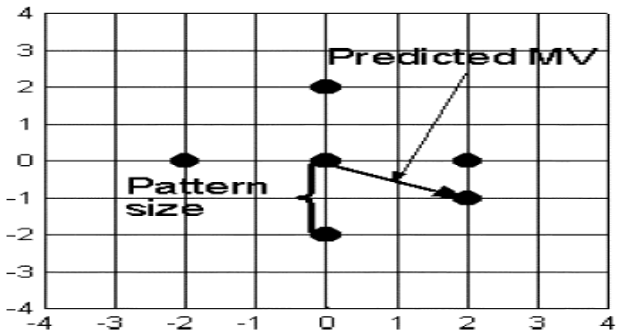


Figure 6. Adaptive rood pattern (ARP)

In our method, the four arms of the rood pattern are of equal length. The initial idea in deciding the ARP's size is to make it equal to the length of the predicted MV (i.e., the MV of the immediate left block of the current block). That is, the size of ARP, Γ, is

$$\Gamma = Round\ |\overrightarrow{MV}_{predicted}| = Round\ \sqrt{[\mathrm{MV}^2{}_{\mathrm{predicted}}(\mathrm{x}) + \mathrm{MV}^2{}_{\mathrm{predicted}}(\mathrm{y})]} \quad (3)$$

where $MV_{predicted}(x)$ and $MV_{predicted}(y)$ are the horizontal and vertical components of the predicted MV, respectively. Operator "Round" performs rounding operation, which takes the nearest integer value of the argument.

Note that parameter Γ defined in (2) involves square and square-root operations; thus, increasing difficulty on hardware implementation. Instead, we use only one of the two components of the predicted MV that has the larger absolute value (or magnitude) to determine the size of our ARP.

That is:

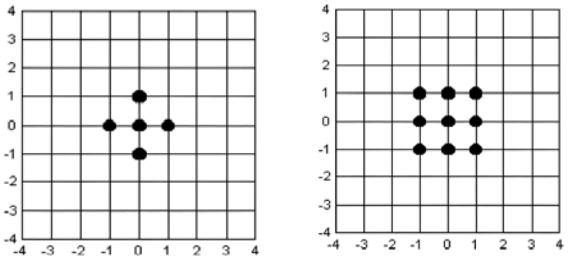
$$\Gamma = Max\ \{MV_{predicted}(x), MV_{predicted}(y)\} \quad (4)$$

From the mathematical standpoint, [19] the magnitude of MV's component with larger absolute value is fairly close to the length of MV, and thus Equation (4) is a good approximation of measurement about motion magnitude. Experimental results show that the second definition of Γ using (4) is, in fact, slightly superior to the first one using (3) in terms of higher PSNR and less total number of checking points.

The second method equation (4) for the rest of ARPS development was adopted. We observed that the chosen ROS (type D) is not applicable to all the leftmost blocks in each frame. For those blocks, we do not utilize any neighboring MVs, but adopted a fixed-size arm length of 2 pixels for the ARP, since this size agrees to that of LDSP which has fairly robust performance as reported in. Also, longer arm lengths are not considered because the boundary MBs in a frame usually belongs to static background.

3) Fixed pattern-for refined local search

In the initial search using ARP as described earlier, the adaptive rood pattern leads the new search center directly to the most promising area which is around the global minimum; thus, effectively reducing unnecessary intermediate searches along the search path. We use a fixed, compact and small search pattern to perform local refined search unrestrictedly for [MV (x) MV (y)] predicted identifying the global minimum. When a fixed pattern is used, the matching motion estimation (MME) point found in the current step will be re-positioned as the new search center of the next search iteration until the MME point is incurred at the center of the fixed pattern. Two types of most compact search patterns have been investigated in our experiments. One is the five-point unit-size rood pattern which is the same as SDSP used in DS, and the other is a 3x3 square pattern as shown in Fig. 7 (a, b). This demonstrates the efficiency of using URP in local motion detection, and it is therefore adopted in our proposed method [20].



(a) (b)

Figure 7. Two fixed search patterns under consideration

Image segmentation is generally defined as the basic image processing that subdivides a digital image $f(x, y)$ into its continuous, disconnect and nonempty subset $f_1, f_2, f_3, \cdots f_n$, which provides convenience to extraction of attribute [3]. In general, Image segmentation algorithms are based on two basic principles [4]: the trait of pixels and the information in nearby regions. Most of segmentation algorithms are based on two characters of pixels gray level: discontinuity around edges and similarity in the same region. As is shown in Table I, there are three main categories in image segmentation [5]: *A.* edge-based segmentation; *B.* region-based segmentation; *C.* special-theory-based segmentation. And some sub-classes are included in the main categories too.

III. Simulation Results and Discussion

In this paper we have evaluated the concepts of motion estimation, video compression, BMA using Exhaustive Search, Three-step Search, Diamond search and ARPS algorithm for 16 × 16 block size images of Chemical Plant, Toy Vehicle and Walter Cronkite. In ME, the search process can be modified to suit the needs of a particular algorithm. The search area is typically restricted to lower the computational cost associated with block matching. In most of the video sequences, the objects in the scene do not have large translational movements between a frame and the next. That is, the fact that frames in a video sequence are taken at small intervals of time and exploited. All tested video scenes are used to generate the frame by frame motion vectors.

A. Exhaustive search

Figure 8 shows the simulation results of ES algorithm for a chemical plant where the number of searches throughout the sequence remains the same and the performance of PSNR is higher.

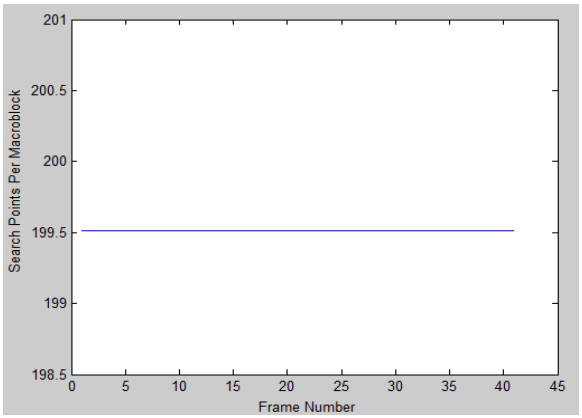
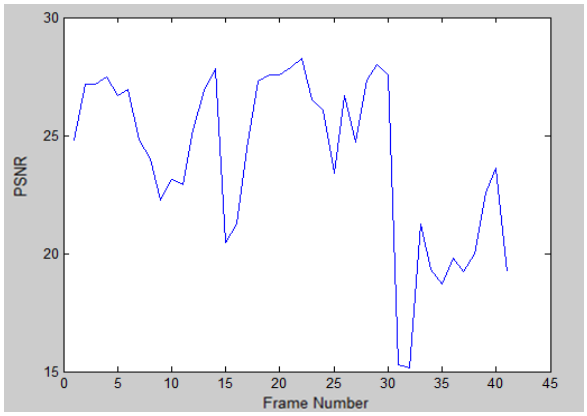


Figure 8. a) Frame based number of searches



b) Frame based PSNR performance of ES in Chemical Plant.

B. *Three step search*

Figure 9 shows the simulation results of TSS algorithm for a chemical plant where the number of searches throughout the sequence varies and the performance of PSNR is lower than ES.

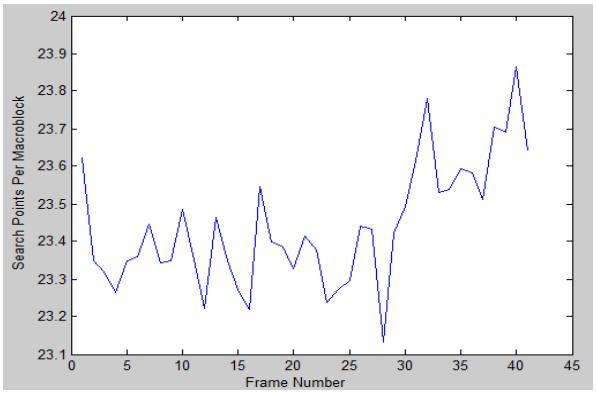
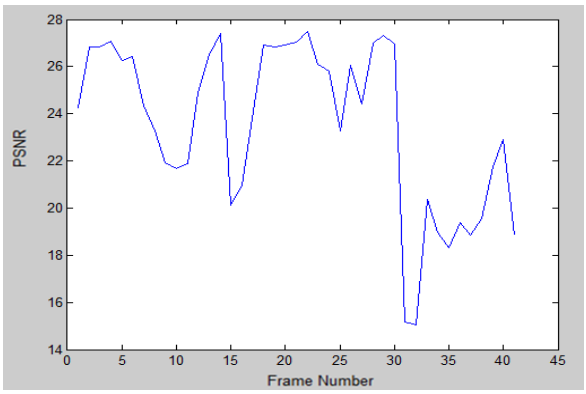


Figure 9. a) Frame based number of searches



b) Frame based PSNR performance of TSS in Chemical plant

C. *Diamond search*

Figure 10 shows the simulation results of DS algorithm for a chemical plant where the number of searches is less than ES and TSS but the performance of PSNR is lower than TSS.

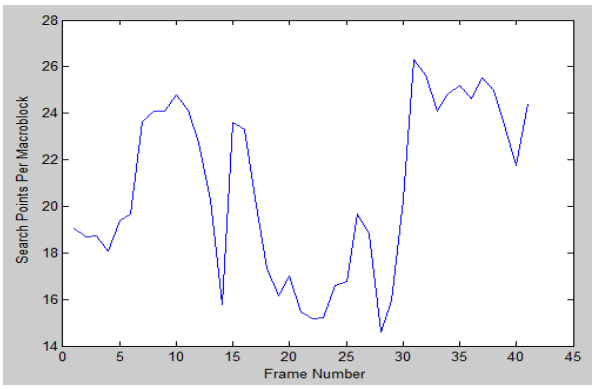
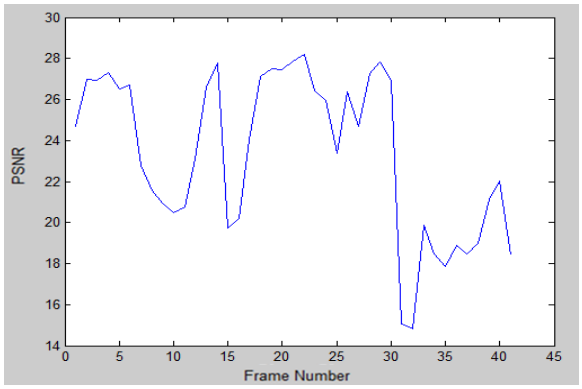


Figure 10. a) Frame based number of searches



b) Frame based PSNRof DS in Chemical plant

D. *Adaptive rood pattern search*

Figure11 shows the simulation results of ARPS algorithm for a chemical plant where the number of searches is less than ES, TSS , DS and the performance of PSNR is higher than TSS, DS and comparable to ES.

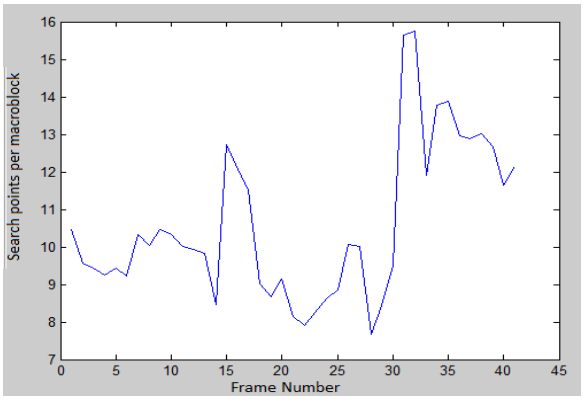
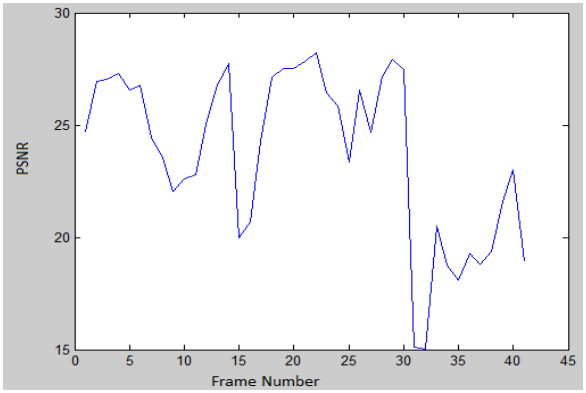


Figure 11. a) Frame based number of searches



b) Frame based PSNR of ARPS

E. Comparison of algorithms

Number of search points per macro-block obtained for various frames of chemical plant video sequences is depicted in Fig.12 and Fig. 13 shows PSNR value with respect to corresponding frame for various algorithms.

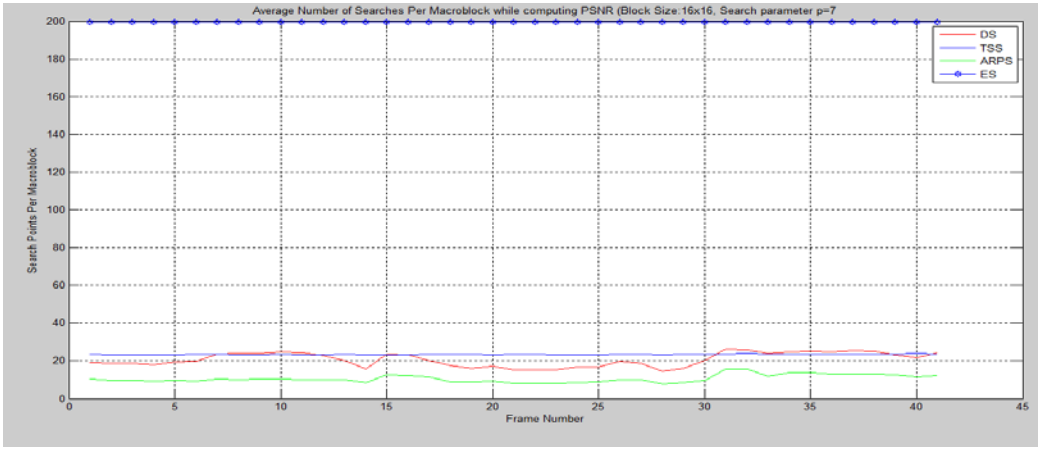


Figure 12. No. of frames versus search points per macroblock for video sequences

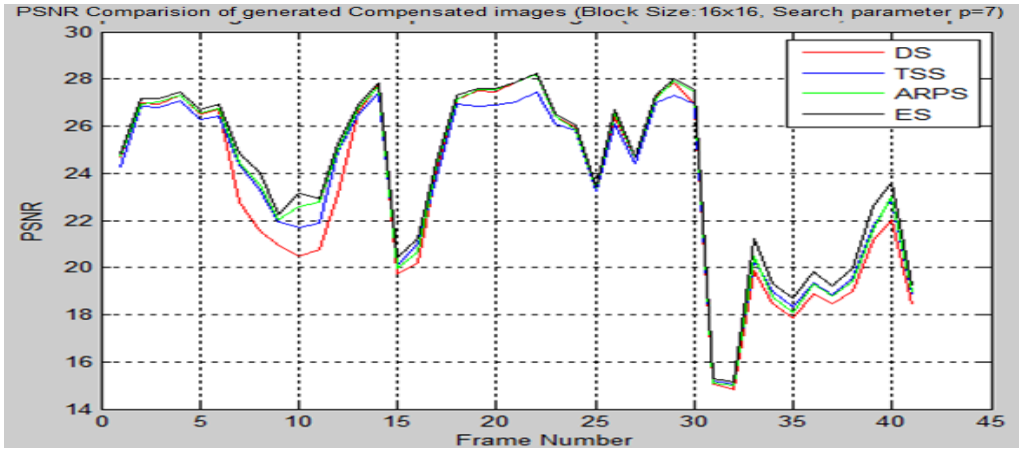


Figure 13. No. of frames versus PSNR for various methods for chemical plant

F. Motion compensated image comparison

The motion compensated images, obtained by using DS, TSS, ARPS and ES algorithm based on number of search points is shown in Fig.14. The images for different algorithms differ in picture clarity. ARPS and ES yield the best motion compensated images but ARPS does it more efficiently, since the number of searches per macroblock is very less as compared to ES.

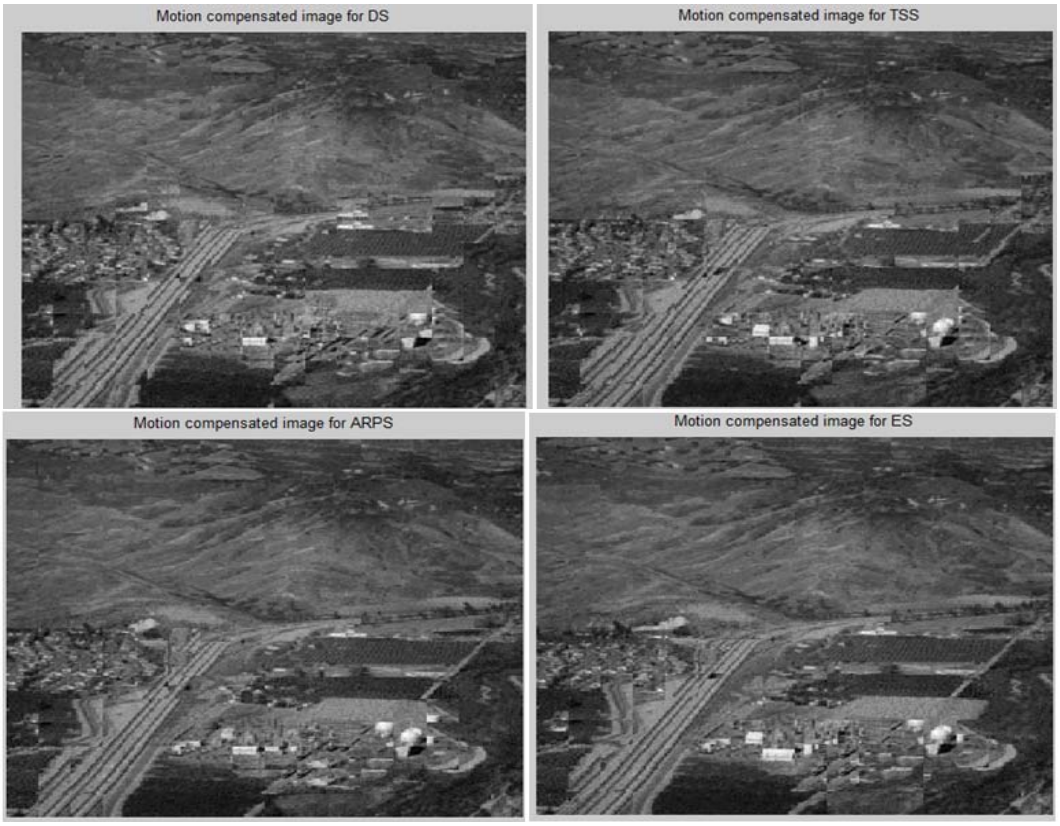


Figure 14. Comparison of the performance of DS, TSS, ARPS and ES

G. Comparison of the Performance of Algorithms

The performances of various algorithms based on number of searches and PSNR values for video compression are compared and the results are presented in Table 1 and Table 2.The simulated results showed that, out of the four algorithms, the values for the average number of searches, for ARPS, were 10.5757, 7.735 and 10.664, similarly for DS, the values were 20.7326, 15.9175 and 18.1875, for ES were 199.5116, 212.0664 and 199.5156, for TSS were 23.4384, 23.9713 and 23.0717 for chemical plant, toy vehicle and Walter Cronkite respectively.

It proved that though ES algorithm finds the best match between the block of the current frame but searches all possible positions thereby making it computationally more expensive. Still this algorithm is considered optimal because it is capable of generating improved motion vectors, because of a high PSNR value, resulting in better quality of videos, where as the ARPS searches the least number of search points for MV generation, less than TSS and DS. Very good picture quality can be achieved using ARPS with very few numbers of searches using ARPS. Similarly the PSNR results for ES yielded values of 24.0595, 27.6936 and 35.1491, for DS were 23.3651, 27.5863 and 34.5904, for TSS were 23.5551, 27.629 and 34.5217, and for ARPS were 23.7884, 27.5995 and 34.5904 for chemical plant, toy vehicle and Walter Cronkite respectively.

The evaluated result shows that the PSNR values are somewhat similar in all the four algorithms. The ES scores are more over the rest algorithms, closely followed by ARPS, suggesting that ARPS maintains similar PSNR performance of ES in most sequences and achieves a superior PSNR than DS and TSS. Exhaustive Search algorithm finds the best match between the block of the current frame and all possible positions inside the search are set in the reference frame. Though this algorithm is computationally more expensive than other proposed algorithms, still this algorithm is considered optimal because it is capable of generating improved motion vectors, with a high PSNR value, resulting in better quality of videos.

TABLE 1. AVERAGE NUMBER OF SEARCH POINTS FOR ARPS, DS, ES AND TS

AVERAGE NUMBER OF SEARCH POINTS PER MV GENERATION				
VIDEO	ARPS	DS	ES	TSS
Chemical Plant	10.5757	20.7326	199.5156	23.4389
Toy Vehicle	7.735	15.9175	212.0664	23.9713
Walter Cronkite	10.664	18.1875	199.5156	23.0717

TABLE 2. AVERAGE PSNR FOR ARPS, DS, ES AND TS

AVERAGE PSNR PERFORMANCE OF ARPS,DS,ES,TSS				
VIDEO	ARPS	DS	ES	TSS
Chemical Plant	23.7884	23.3651	24.0595	23.5551
Toy Vehicle	27.5995	27.5863	27.6936	27.629
Walter Cronkite	34.8236	34.5904	35.1491	34.5217

The tradeoff between performance, simplicity and the fact that the improvement resulted from the adaptability of our search pattern, and more importantly, avoiding local minimum matching error points. This is done by tracking the major trend of the motion at the initial stage, since complex motions and unevenness of the objects cause large number of local minimums on the matching error surface, checking points in all directions (as being done by Large Diamond Search Pattern of Diamond Search Algorithm) at the initial step increases the risk of being trapped into the local minima and thus degrades the search accuracy. Thus we conclude that ARPS is the best block matching algorithm for video compression using motion estimation.

REFERENCES

[1] Tekalp A. Digital video processing, Prentice Hall, PTR, 1995.

[2] Richardson I.G. H.264 and MPEG-4 video compression. Wiley, Chichester, England, 2003.

[3] Jain J. and Jain, A. Displacement Measurement and its Application in Interframe Image Coding. IEEE Trans on Communication, COM- 29, 1981: 1799-1808.

[4] Ning-ning, S., Chao, F., Xu, X. An Effective Three-step Search Algorithm for Motion Estimation. IEEE Int Symposium on IT in Medicine & Education, 2009:400- 403.

[5] Netravali A.N. and Robbins J.D. Motion compensated television coding: Part I. ‖ Journal of Bell Syst Technical, 1979, 58: 631-670.

[6] Lin Y.C. and Tai S.C. Fast Full-Search Block-Matching Algorithm for Motion- Compensated Video Compression. IEEE Trans on Communications, 1997, 45(5): 527-531.

[7] CCITT SG XV. Recommendation H.261-Video codec for audiovisual services at p*64 kbits/s. Tech. Rep. COM XV-R37-E, 1990.

[8] MPEG ISO CD 11172-2: Coding of moving pictures and associated audio for digital storage media at up to about 1.5 M bits/s, 1991.

[9] Kappagantula S. and Rao K.R. Motion compensated interframe image Prediction. IEEE Trans Commun. COM, 1985, 33: 1011-l015.

[10] Ahmed Z., Hussain, A. J. and Al-Jumeily D. Fast Computations of Full Search Block Matching Motion Estimation (FCFS). IEEE Trans on Communications, 2011: 1-6.

[11] Reoxiang L., Bing Z. and Liou M.L. A new three-step search algorithm for block motion estimation. IEEE Trans on Circuits and Systems for Video Technology,1994, 4: 438-442.

[12] Barjatya A. Block Matching Algorithms for Motion Estimation. DIP 6620 Final Project Paper, 2004.

[13] Cheung C.H and Po L.M. A novel cross-diamond search algorithm for fast block motion estimation. IEEE Trans Circuits Syst Video Technol, 2002, 12: 1168- 1177.

[14] Zhu S. and Ma K.K. A new diamond search algorithm for fast block-matching motion estimation. Proc Int Conf Information Communications and Signal Processing (ICICS), 1997, 9(1): 292–296.

[15] Zhu S. Fast motion estimation algorithms for video coding. M.S. thesis, School Elect Electron Eng, Nanyang Technol Univ, Singapore, 1998.

[16] Zhu S. and Ma K.K. A New Diamond Search Algorithm for Fast Block Matching Motion Estimation. IEEE Trans on Image Processing, 2000, 9(2): 287-290.

[17] Shen J.F., Chen L.G., Chang H.C., Wang T.C. Low power full-search block matching motion estimation chip for H.263+. IEEE International Symposium on Circuits and Systems, 1999, 4:299-302.

[18] Nie Y. and Ma K.K. (2002). Adaptive rood pattern search for fast block-matching motion estimation. IEEE Trans Image Processing, 2002, 11 :1442-1448.

[19] Kaushik M. Comparative Analysis of Exhaustive Search Algorithm with ARPS Algorithm for Motion Estimation. International Journal of Applied Information Systems (IJAIS), 2012: 1(6), 16-19.

[20] Kiran S.V., Prasad K.V., Lalitha N.V. and Rao D.S. Architecture for the ARPS Algorithm. International Journal of Advances in Computer Networks and its Security, 2002: 443-448.

Nayak A. was born in Cuttack, Orissa, India on June 20th 1981. Nayak is now working as lecturer in the Department of Electronics and Telecommunication, Ajay Binay Institute of Technology, Cuttack. He received his B.E. degree in Electronics & TeleCommunication in 2003 from CVRCE, M. Tech in Electronics & TeleCommunication from KIIT University, India in 2012. He has over 8 years of experience in teaching and research in Electronics & Communication engineering. His research interest includes signal and image processing. He is a member of ISTE(INDIA).

Biswal B. was born in Keonjhar, Orissa, India on November 10th 1985. Biswal is now working as lecturer in the Department of Electronics and Telecommunication, Ajay Binay Institute of Technology, Cuttack. She received her B. Tech degree in Electronics &Telecommunication Engineering in 2007 from ABIT, M. Tech in Electronics and Communication Engineering from IIT, Kharagpur, India in 2013. She has 6 years of experience in teaching and research in the field of Electronics & Communication engineering. Her research interest includes communication, signal and image processing. She is a member of Orissa Information Technology Society and ISTE, India.

Sabut S. K. was born in Puri, Orissa, India on April 5th 1973. Sabut is now working as associate professor in the Department of Instrumentation Technology, MS Ramaiah Institute of Technology, Bangalore. He received his B.E. degree in Electronics & Communication in 2001, M. Tech in Biomedical Instrumentation from VTU, India in 2005 and Ph. D in Medical Science & Technology from IIT, Kharagpur, India in 2010. He has over 13 years of experience in teaching and research in the field of biomedical & rehabilitation engineering. His research interest includes neural prosthesis and rehabilitation engineering, biomedical instrumentation, biomedical signal and image processing. He has published various research papers in journals and presented research paper in national/ international conferences. He is a member of IFESS, IEEE, Rehabilitation council of India and the Institution of Engineers (India).

