

Adaptive Square-Diamond Search(ASDS) Algorithm for Fast Block Matching Motion Estimation

M K Pushpa

*Dept. of Instrumentation Technology,
M S Ramaiah Institute of Technology,
Bangalore, India*

Dr.S.Sethu Selvi

*Dept. of Electronics & Communication Engg.,
M S Ramaiah Institute of Technology,
Bangalore, India*

Abstract—This paper presents an efficient search algorithm for fast block matching motion estimation(ME) for video applications. The algorithm uses two patterns for initial search and refined local search. For initial search, the proposed algorithm uses a square pattern adaptively by selecting the step size based on Maximum Absolute Value of predicted motion vector. This search pattern is aimed to reduce the computational complexity of the ME block and find the least error. If the least error point is in the middle of the pattern, it shows that image is still and terminates the search. If the least error point is other than the middle point, then it becomes a new origin for subsequent refined local search with the pattern as small diamond. This is iteratively continued until the final motion vector is found. The efficacy of the proposed algorithm is verified by comparing with existing search algorithms in terms of PSNR, computations/search points and elapsed time for motion estimation. Simulation results when compared with Exhaustive Search(ES) algorithm show that the proposed algorithm reduces number of search points by 96%. The PSNR performance is close to the ES search method by 99.8% and the elapsed time saving is 95%. This savings ensure that the proposed algorithm is suited for real-time applications.

Keywords— *Motion estimation, Search algorithm, Square pattern, Small Diamond, Block matching.*

I. INTRODUCTION

Motion Estimation is an important technique needed in any video image analysis. The video application areas like transmission, surveillance, data storage, conferencing and object tracking contain images of moving objects which needs large data to be transmitted or storage. The motion captured in a multiframe sequence of images includes translation and rotation of objects with respect to the camera movements. Hence using motion estimation techniques the actual displacement from one frame to the next can be easily identified. For interframe image coding, large levels of compression could be achieved if only one knew the trajectories traversed by the various objects[1]. The overwhelming complexity of motion estimation(ME) using a full search(ES) based brute-force approach has led to explosive research in ME. Most ME algorithms exhibit tradeoffs between quality and speed. Since ME is highly scene dependent, and, no one technique can be fully relied to generate good visual quality for all kinds of video scenes.

The variety of techniques, such as motion starting point, motion search patterns, adaptive control to curb the search and avoidance of search stationary regions, etc. makes ME algorithm robust and efficient across the board. The performance of the full search method is considered to be “optimal” and its complexity is prohibitively high for software implementation. Furthermore, since the full search method aims to find the minimum sum of absolute differences(SADs), the presence of noise in a video can lead to suboptimal motion vectors(MVs). The presence of noise can also cause the full search to produce chaotic motion field for a smooth motion video, costing more bits to encode MVs with fewer bits left for encoding DCT coefficients with a given bit budget[2,3]. The most common ME method is the block matching technique, in which a video frame is divided into macroblock (MB) of 16×16 pixels or blocks of 8×8 pixels and a search window(P) is defined. Each MB of the current frame is compared with the blocks of the reference frame within a search window. The displacement with the maximum correlation or the minimum distortion between the current block and the reference blocks within the search window is selected as the MV.

A vast number of block matching algorithms(BMAs) have been proposed in[4]. Some of the algorithms are: block pixel decimation[1], two dimensional logarithmic search algorithm(2D-LOG)[1] as well as their variations[5,6], three step search algorithm(TSS)[4], new three step search algorithm(NTSS)[3], Simple and efficient three step search(SESTSS), four step search algorithm(FSS)[7], diamond search (DS)[14,15], Adaptive rood pattern search(ARPS)[30], conjugate directional search[8,9,10], orthogonal direction search algorithm(OSA)[11], cross search[13], dynamic search window adjustment (DSWA)[12], gradient-based search[16], zone-based search[17], refined zone-based search[18] and parallel hierarchical one-dimensional search algorithm(PHODS)[19]. Hierarchical search[20,21,6,22] and multiresolution algorithms[23,24] perform ME at multiple levels successively, starting with the lowest resolution level using low-pass filtering or sub sampling[25]. Motion estimations are computationally very expensive but intelligent search strategies can reduce computational burden. Designing fast and accurate ME algorithm remains an open research problem. Hence author

has implemented a new search algorithms like Joint Adaptive Block Matching Search(JABMS) algorithm[28] and Adaptive Hexa-Diamond Search(AHDS) algorithm[29]. In a similar direction this new algorithm is proposed as an efficient search method.

The paper is organized as follows: In Section 2, the proposed Adaptive square diamond search(ASDS) algorithm for motion estimation is discussed in detail. In Section 3, the performance of the proposed algorithm is demonstrated with the comparison results and analysis. Conclusions are drawn in Section 4. The ASDS algorithm is compared with existing search algorithms in terms of Peak signal to noise ratio(PSNR), number of computations in terms of search points and elapsed processing time.

II. METHODOLOGY

A. Adaptive Square diamond search(ASDS) algorithm for fast block matching motion estimation

The most common ME method is the block matching algorithm(BMA), in which a video frame is divided into MBs of 16×16 pixels and a search window is defined. Each MB of the current frame is compared with the blocks of the reference frame within a search window. The displacement with the maximum correlation or the minimum distortion between the current block and the reference blocks within the search window is selected as the MV. To estimate MV using a BMA, an image is first divided into blocks of 16×16 pixels. For simplicity, the blocks resulting from the partition of the current frame and the previous frame are called the current(C) and the reference(R) blocks, respectively. For each current block(C), the algorithm will find a representative block corresponding to it within a search area(P= ±7) surrounding the block in the previous frame with the same spatial position as C. The search area usually extends pixels in both the horizontal and vertical directions. The MV of C is then defined as the direction from the representative block to it. The optimality of the pattern based search depends on size of the search pattern and the magnitude of the predicted MV. Therefore different search patterns are used in accordance with the computed motion behavior for the current block. Hence importance is given to two issues. 1) To predetermine the motion behavior of the current block for effective motion estimation and 2) To adapt size and shape of the search pattern. For the first issue coherency of the motion in a frame is considered and for the second issue, two types of search patterns are used. i.e an adaptive rood pattern for initial search and a fixed size search pattern for repetitive fine search until the final MV is found. The algorithm uses a pre processing stage as in Figure.1 as region of support(ROS) which is only one block that is situated at immediate left to the current block to predict the MV using conventional search[30] method as

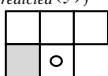
$$\{MV_{predicted}(x), MV_{predicted}(y)\}.$$


Fig.1: A Type D Region of Support (ROS) to predict MV.

The objective of the algorithm is to find a good initial starting point for the initial search so as to avoid unnecessary intermediate search and reduce the risk of being trapped into a local minimum in the case of long

search path. The new starting point identified is as close to the global minimum as possible. Since the usage of more blocks involves higher computational complexity, the search will be done in a selected area(P). Figure.2 shows the search pattern used in the algorithm. For initial search, the proposed algorithm uses a square pattern adaptively by selecting the step size(S) based on maximum absolute value of predicted motion vector using Eq.1. Initially select the positions of points where the checking has to be done and initialized to zero. As one point is checked, it sets the corresponding element in the matrix to one. Algorithm starts from the top left of the image and move with stepsize of macro block. Macroblock count will keep track of how many blocks have been evaluated. If the macroblock is in the left most column then algorithm uses square pattern with Step Size = 2, else step size is maximum of absolute value of predicted motion vector values in x and y directions as in Eq.1.

Is macroblock is leftmost block, then S=2,
else

$$S = \max \{ |MV_{predicted}(x)|, |MV_{predicted}(y)| \} \tag{1}$$

where $MV_{predicted}(x)$ and $MV_{predicted}(y)$ are the x-coordinate and y-coordinate of the predicted motion vector respectively. Initial pattern is symmetric with four search points located at vertices to form a square pattern, including one at the centre point there are five search points. The step size refers to the distance between the centre and the search points. Since the initial search pattern includes all the directions it can quickly detect any motion and the search will be directly jump to a region in the direction of the predicted MV. The major goal of initial search is to detect the moving object direction. Since the pattern is symmetrical it is beneficial, robust and easy to implement in hardware.

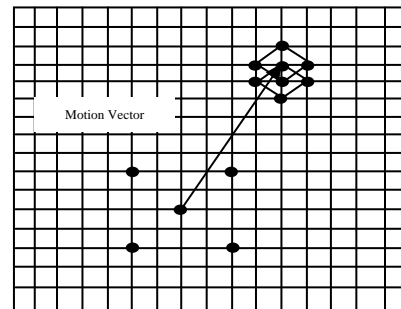


Fig.2: Adaptive Square-Diamond Search Pattern.

From[26,27] it is observed that at position(0,0) about 98% of the stationary blocks have their SAD less than 512 for MB size of 16×16. The algorithm exploits an optional phase called early elimination of search as its first step. If SAD at (0,0) is less than threshold(T=512), then consider that MB as stationary block. The MV is assigned as (0,0) then in the phase of early elimination of search, the search for a MB will be terminated immediately. i.e. SAD at (0,0) < T, then MV=(0,0)

In terms of block distortion method, the sum of absolute differences (SAD) is commonly used and is defined by Eq.2

$$SAD = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}| \tag{2}$$

Where C_{ij} and R_{ij} are the pixels being compared in current block and reference block respectively and N×N is

image size. If the least error point is in the middle of the square pattern, it shows that image is still and terminates search. In addition to the square pattern it is desirable to add predicted MV into the search as it is likely to be similar to the target MV. Hence this will increase the probability of accurately detecting the motion in the initial stage. The initial idea is to decide the pattern size to use only one of the two components of the predicted MV that has larger magnitude because the magnitude of the MV's component with greater absolute value is nearly close to the length of the MV. If the least error point is other than the middle point and if it is one of the points of the square pattern, then skip that point and the least error point will be a new origin for subsequent refined local search with the pattern as small diamond search pattern(SDSP).

If $|MV_{predicted}(x)| = S$ and $MV_{predicted}(y) = 0$

Or

$|MV_{predicted}(y)| = S$ and $MV_{predicted}(x) = 0$ then algorithm need to check only five points otherwise six points has to be checked(including new origin).

The SDSP comprising of five search points of unit step and is iteratively continued until the least error is found to be at the centre of the SDSP pattern. Then the final motion vector is found.

III. RESULTS AND ANALYSIS

The proposed algorithm uses MATLAB 7. Macroblock size is 16x16 and search area is ±7. For slow motion, medium motion and fast motion, six video sequences are considered. The results are tabulated in Table. I to VI. The performance analysis is done by considering average of 30 frames. Quality of image is measured in terms of average PSNR. Computations are indicated as average number of search points. Time consumption during motion estimation and compensation is represented as average elapsed time in seconds. Time is calculated using Intel® Core™2Duo Processor @1.80GHz. The ES is producing good image quality in block based ME hence the proposed algorithm is compared with ES algorithm and ARPS algorithms. From the analysis it is found that the proposed ASDS algorithm works well for all motion image sequences. The ASDS algorithm offers high performance both in quality and speed.

The performance is evaluated as in Eq.3

$$\frac{ES - ASDS}{ES} * 100 \tag{3}$$

TABLE I. COMPARISON RESULTS FOR MOM_DAUGHTER SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Mom daughter	ARPS	8.58758	31.5014	0.169233
	DS	15.72943	31.52167	0.260467
	ES	202.05	31.6837	2.585933
	NTSS	20.39527	31.57977	0.288467
	SESTSS	16.29927	31.14073	0.241267
	FSS	17.8706	31.49847	0.259367
	TSS	23.07893	31.49477	0.3093
	AHDS	8.556663	32.6078	0.169267
	PROPOSED ASDS	8.344343	32.6043	0.1657

In Table.I “Mom_daughter” image sequence is tested on different algorithms and results are tabulated. The proposed ASDS algorithm when compared with ES algorithm, shows PSNR gain as 0.93dB i.e. 2.91% improvement, 95.87% saving in search points and 93.6% saving in elapsed time. When compared with ARPS algorithm it shows 2.83% savings in terms of search points, 3.5% improvement in PSNR and 2.08% savings in elapsed time.

TABLE II. COMPARISON RESULTS FOR MOM SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Mom	ARPS	8.60424	34.36097	0.169333
	DS	16.11123	34.3175	0.2645
	ES	202.05	37.3752	2.540133
	NTSS	20.93767	34.39137	0.293833
	SESTSS	16.36767	34.2703	0.2417
	FSS	18.0896	34.34103	0.264067
	TSS	23.031	34.35077	0.310867
	AHDS	8.32939	36.92953	0.163567
	PROPOSED ASDS	7.973423	37.0342	0.155767

In Table.II, the results are tabulated for “Mom” image sequence. When compared with ES algorithm, the proposed ASDS shows 96.05% reduction in search points, 99.09% of PSNR and 93.87% saving in elapsed time. When compared with ARPS algorithm, it shows 7.34% saving in search points, PSNR gain of 2.67dB i.e. 7.78% improvement and 8.01% savings in elapsed time.

TABLE III. COMPARISON RESULTS FOR ALEX SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Alex	ARPS	7.361103	42.17527	0.188033
	DS	13.6704	42.19653	0.285
	ES	204.28	42.2197	3.082733
	NTSS	16.85263	42.1739	0.294833
	SESTSS	16.679	41.77367	0.303667
	FSS	16.8455	42.07307	0.298333
	TSS	23.50027	42.00563	0.3854
	AHDS	8.15934	42.17127	0.206333
	PROPOSED ASDS	8.01944	42.205	0.196333

In Table.III, “Alex” image sequence is tested on different algorithms. The proposed ASDS algorithm when compared with ES algorithm shows 96.07% savings in search points, 0.0147dB gain in PSNR i.e. 0.03% improvements and 93.63% savings in elapsed time. When compared with ARPS algorithm, the proposed ASDS requires additional 8.94% search points, 99.93% of PSNR and 4.41% additional elapsed time is required.

In Table.IV, “Diskus” image sequence is tested on different algorithms. The proposed ASDS algorithm when compared with ES algorithm shows 94.51% savings in search points, 98.65% of PSNR and 91.7% savings in elapsed time. When compared with ARPS algorithm, the proposed ASDS requires 16.8% additional search points, 0.1091dB gain in PSNR, i.e. 0.34% improvement and 0.03% additional elapsed time is required.

TABLE IV. COMPARISON RESULTS FOR DISKUS SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Diskus	ARPS	9.608593	31.9674	0.217533
	DS	17.5385	31.79357	0.342733
	ES	204.28	32.294	3.003533
	NTSS	21.8714	31.85833	0.368233
	SESTSS	16.20297	30.9394	0.290133
	FSS	19.2059	31.73823	0.338033
	TSS	23.41857	31.8811	0.3745
	PROPOSED ASDS	11.22493	31.85833	0.2495

In Table.V, “Flower garden” image sequence is tested on different algorithms. The proposed ASDS algorithm when compared with ES algorithm shows 93.1% savings in search points, PSNR of 98.06% and 89.6% savings in elapsed time. When compared with ARPS algorithm, the proposed ASDS requires 30.68% additional search points, 0.092dB gain in PSNR i.e. 0.48% gain and 28.8% additional elapsed time is required.

TABLE V. COMPARISON RESULTS FOR FLOWER GARDEN SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Flower Garden	ARPS	10.76347	19.05197	0.2026
	DS	20.12033	18.276	0.325433
	ES	202.05	19.52187	2.508267
	NTSS	26.0641	19.1431	0.365167
	SESTSS	15.43793	18.7972	0.2334
	FSS	20.5266	18.47623	0.310333
	TSS	23.35027	18.94893	0.314533
	PROPOSED ASDS	14.06617	19.1431	0.261

TABLE VI. COMPARISON RESULTS FOR TENNIS SEQUENCE.

Sequence	Search Algorithm	Average Search Points	Average PSNR in dB	Average Elapsed time in sec
Tennis	ARPS	7.292757	24.6744	0.146267
	DS	14.9618	24.79253	0.249
	ES	202.05	25.5635	2.504167
	NTSS	19.30943	24.93657	0.273967
	SESTSS	16.4579	24.43373	0.2453
	FSS	17.58963	24.78617	0.2563
	TSS	23.17957	24.82097	0.3115
	PROPOSED ASDS	7.773667	24.98413	0.1526

In Table.VI, “Tennis” image sequence is tested on different algorithms. The proposed ASDS algorithm when compared with ES algorithm shows 96.2% savings in search points, PSNR of 97.75% and 93.91% savings in elapsed time. When compared with ARPS algorithm, the proposed ASDS requires 6.58% additional search points, gain in PSNR is 0.31dB i.e. 1.26% improvement and 4.32% additional elapsed time is required.

Figure.3, 4 and 5 show small, medium and fast motion images respectively and the image sequences are shown with i^{th} frame as reference frame, $i+2$ frame as current frame, also the compensated images and the residual difference image using different existing algorithms like ES, TSS, SESTSS, NTSS, FSS, DS, ARPS, AHDS and the proposed ASDS algorithm.



Fig.3: Alex image sequence (small motion) with reference frame, current frame, compensated images and its residual images using existing algorithms.



Fig.4: Mom_daughter image sequence (medium motion) with reference frame, current frame, compensated images and its residual images using existing algorithms.

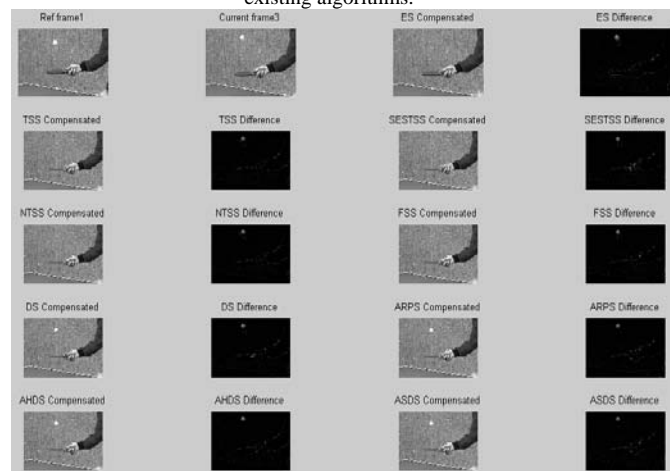


Fig.5: Tennis image sequence (fast motion) with reference frame, current frame, compensated images and its residual images using existing algorithms.

Figure 6, 9, 12, 15, 18 and 21 show comparison of number of search points for all six image sequences. Figure 7, 10, 13, 16, 19 and 22 show comparison results of PSNR for all six image sequences. Figure 8, 11, 14, 17, 20 and 23 show comparison plots of elapsed time per frame for all six image sequences.

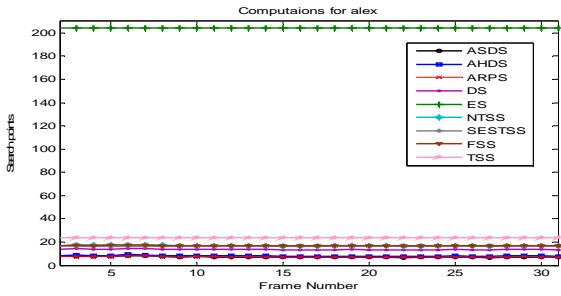


Fig.6: Comparison of number of search points for Alex image sequence.

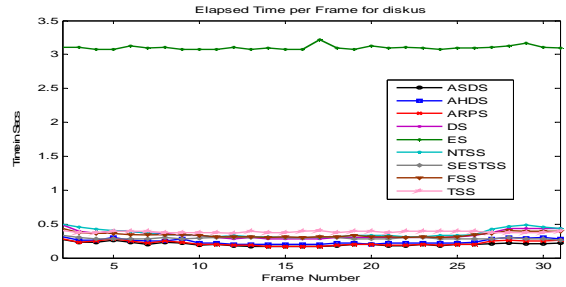


Fig.11: Comparison of elapsed time per frame for Diskus image sequence.

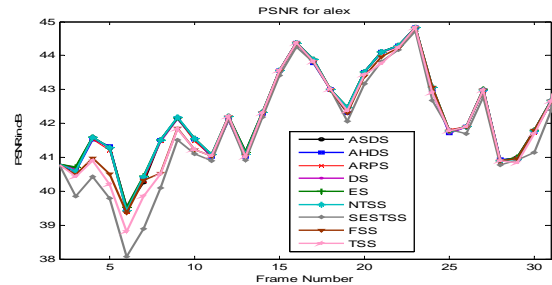


Fig.7: Comparison of PSNR for Alex image sequence.

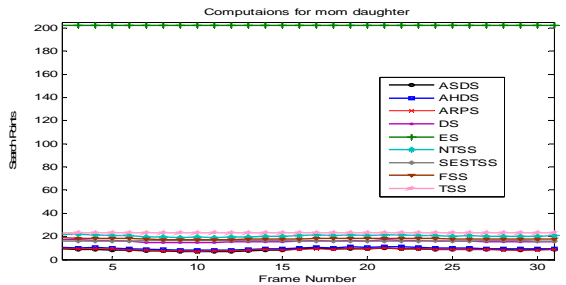


Fig.12: Comparison of number of search points for Mom_daughter image sequence.

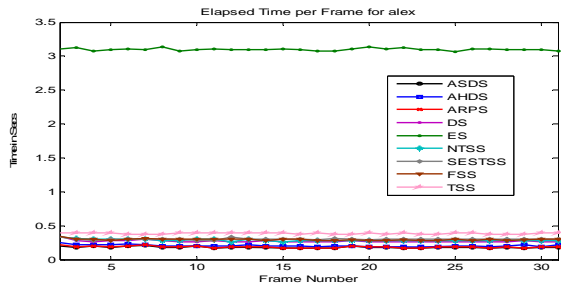


Fig.8: Comparison of elapsed time per frame for Alex image sequence.

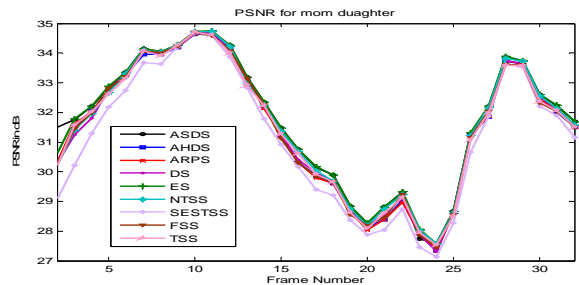


Fig.13: Comparison of PSNR for Mom_daughter image sequence.

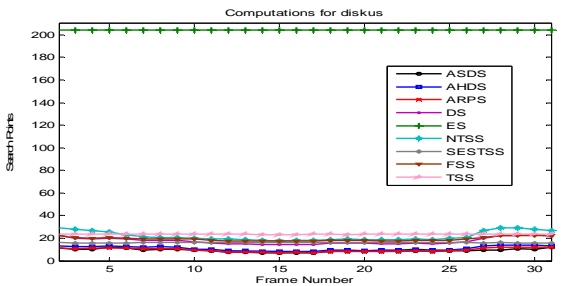


Fig.9: Comparison of number of search points for Diskus image sequence.

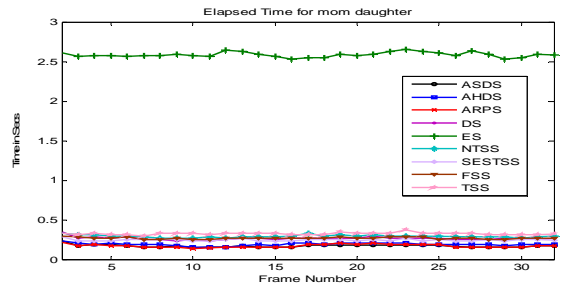


Fig.14: Comparison of elapsed time per frame for Mom_daughter image sequence.

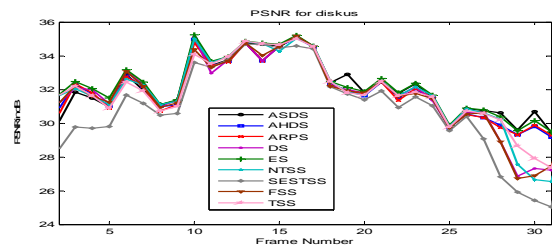


Fig.10: Comparison of PSNR for Diskus image sequence.

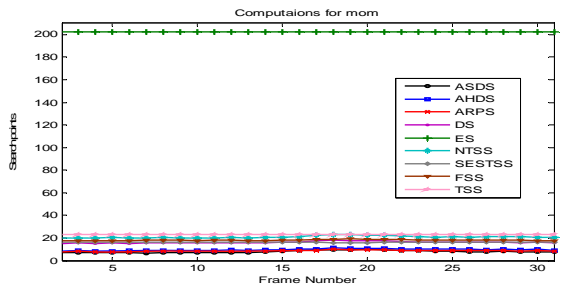


Fig.15: Comparison of number of search points for Mom image sequence.

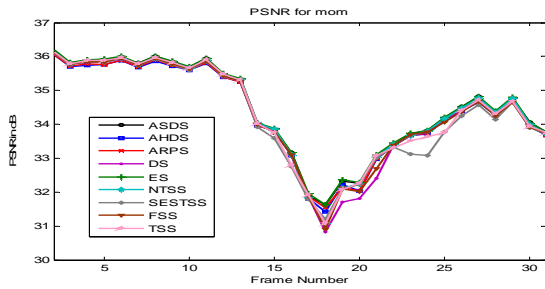


Fig.16: Comparison of PSNR for Mom image sequence.

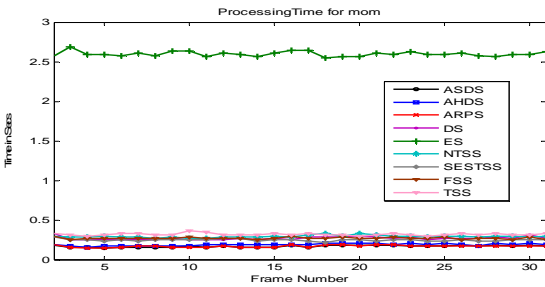


Fig.17: Comparison of elapsed time per frame for Mom image sequence.

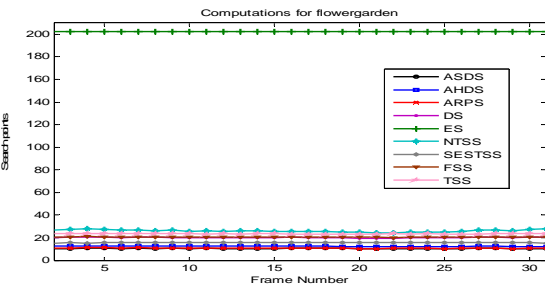


Fig.18: Comparison of number of search points for Flower Garden image sequence.

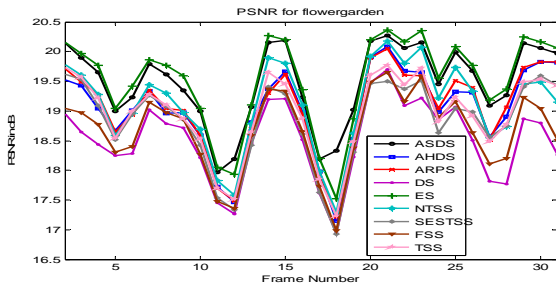


Fig.19: Comparison of PSNR for Flower Garden image sequence.

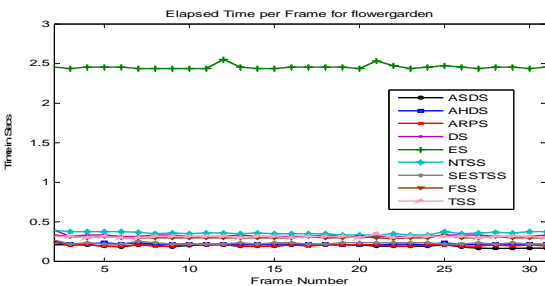


Fig.20: Comparison of elapsed time per frame for Flower Garden image sequence.

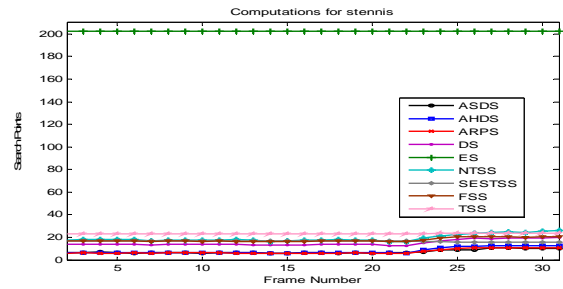


Fig.21: Comparison of number of search points for Tennis image sequence.

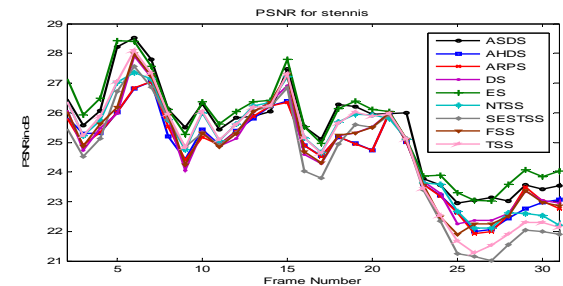


Fig.22: Comparison of PSNR for Tennis image sequence.

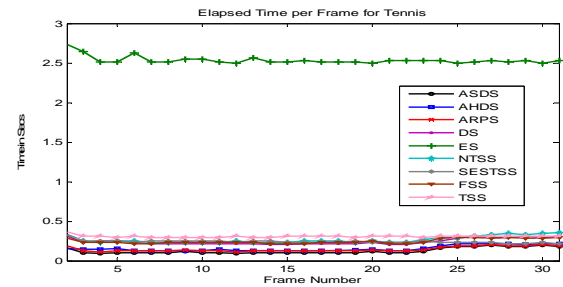


Fig.23: Comparison of elapsed time per frame for Tennis image sequence.

IV. CONCLUSIONS

The proposed algorithm is suited for all kinds of image sequences like small, medium and fast motion. The efficacy of our ASDS algorithm shows that an average 96% savings in search points, 99.8% achievement in PSNR and 95% savings in elapsed time with respect to ES algorithm. Compared with ARPS the proposed ASDS algorithm requires average of 8% additional search points, 1.55% gain in PSNR and 4.3% additional elapsed time. Hence the algorithm can be used where time saving and computations complexity reduction is needed.

REFERENCES

- [1] Jaswant R. Jain and Anil K. Jain, "Displacement Measurement and Its Application in Interframe Image Coding", *IEEE Transactions on communications*, Vol. COM-29, No. 12, Pg. 1799-1808, Dec 1981.
- [2] P. A. A. Assuncao and M. Ghanbari, "A frequency-domain video transcoder for dynamic bitrate reduction of MPEG-2 bit streams," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 8, pp. 953-967, Dec.1998.
- [3] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for fast motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol.4, no. 4, pp. 438-442, Aug. 1994.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. Nat Telecommun. Conf.*, New Orleans, LA, Nov. 1981, pp. G5.3.1-G5.3.5.
- [5] A. Netravali and B. Haskell, *Digital Pictures Representation and Compression*. New York: Plenum, 1988.

- [6] X. Lee and Y. Q. Zhang, "A fast hierarchical motion-compensation scheme for video coding using block-feature matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 6, pp. 627–635, Dec. 1996.
- [7] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast blockmatching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313–317, Jun. 1996.
- [8] B. Liu and A. Zaccartin, "New fast algorithms for estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, Apr. 1993.
- [9] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. Com- 33, no. 8, pp. 888–896, Aug. 1985.
- [10] B. Zeng, R. Li, and M. L. Liou, "Optimization of fast block motion estimation algorithms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 6, pp. 833–844, Dec. 1997.
- [11] A. Puri, H. M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion compensated coding," in *Proc. IEEE ICASSP*, 1987, pp. 2.4.1–2.5.4.4.
- [12] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, "Dynamic searchwindow adjustment and interlaced search block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 1, pp. 85–87, Feb. 1993.
- [13] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, no. 7, pp. 950–953, Jul. 1990.
- [14] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 369–377, Aug. 1998.
- [15] S. Zhu and K. K. Ma, "A new diamond search algorithm for fast block matching," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.
- [16] L. K. Liu and E. Feig, "A block-based gradient descent search algorithm for block-based motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 4, pp. 419–422, Aug. 1996.
- [17] H. M. Jung, D. D. Hwang, C. S. Park, and H. S. Kim, "An annular search algorithm for efficient motion estimation," in *Proc. Int. Picture Coding Symp.*, 1996, pp. 171–174.
- [18] Z. He and M. L. Liou, "A high performance fast search algorithm for block matching motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 5, pp. 826–828, Oct. 1997.
- [19] L. G. Chen, W. T. Chen, Y. S. Jehng, and T. D. Chuieh, "An efficient parallel motion estimation algorithm for digital image processing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 1, no. 4, pp. 378–384, Dec. 1991.
- [20] Y. L. Chan and W. C. Siu, "Adaptive multiple-candidate hierarchical search for block matching algorithm," *IEE Electron. Lett.*, vol. 31, no. 19, pp. 1637–1639, Sep. 1995.
- [21] C. K. Cheung and L. M. Po, "A hierarchical block motion estimation algorithm using partial distortion measure," in *Proc. ICIP'97*, vol. 3, 1997, pp. 606–609.
- [22] X. Song, T. Chiang, and Y. Q. Zhang, "A scalable hierarchical motion estimation algorithm for MPEG-2," in *Proc. ICIP*, 1998, pp. IV126–IV129.
- [23] M. Bierling, "Displacement estimation by hierarchical blockmatching," *SPIE Vis. Commun. Image Process.*, pp. 942–951, May 1998.
- [24] Y. L. Chan and W. C. Siu, "Adaptive multiple-candidate hierarchical search for block matching algorithm," *IEE Electron. Lett.*, vol. 31, no. 19, pp. 1637–1639, Sep. 1995.
- [25] Y. Q. Shi and X. Xia, "A thresholding multiresolution block matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 2, pp. 437–440, Apr. 1997.
- [26] P. I. Hosur and K. K. Ma, "Motion vector field adaptive fast motion estimation," presented at the *Second Int. Conf. Inf., Commun., Signal Process.*, Singapore, Dec. 1999.
- [27] *Optimization Model Version 1.0*, ISO/IEC JTC1/SC29/WG11 N3324, Mar. 2000.
- [28] M K Pushpa and V K Anathashayana "Joint Adaptive Block Matching Search(JABMS) Algorithm for Motion Estimation", *International Journal of Recent Trends in Engineering(IJRTE)*, Vol 2, No. 2, ACEEE, Academy publishers, Finland, pp.212-216, Nov 2009.
- [29] M K Pushpa and Dr. S.Sethu Selvi "Adaptive Hexa-Diamond search(AHDS) Algorithm for Fast Block Matching Motion Estimation," *Proceedings of ICAdC 2012, Advances in Intelligent Systems and Computing*, Vol.174, pp. 85–93, Springer Series 2012, July. 2012.
- [30] Yao Nie, and Kai-Kuang Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", *IEEE Transactions on Image Processing*, Vol. 11, No. 12, pp. 1442-1449, Dec 2002.