

# Performance Analysis of Fast Block Matching Motion Estimation Algorithms

Nasir Mahmud Khokhar

Department of Aeronautical Engg, Faculty of Engineering, King Abdul-Aziz University, Jeddah, Kingdom of Saudi Arabia

Wail Harasani

Department of Aeronautical Engg, Faculty of Engineering, King Abdul-Aziz University, Jeddah, Kingdom of Saudi Arabia

## ABSTRACT

Motion Estimation (ME) is an integral part of any video encoder and a large number of Block Matching Motion Estimation (BMME) Algorithms are proposed to cope the computational complexity and increase quality of ME process requirement. Therefore, it is necessary to evaluate the performance of these ME algorithms for different motion activities. In this paper five fast famous BMME algorithms are considered to evaluate their performance on the basis of ME time, search points, PSNR and Means Square Error (MSE). The algorithms evaluated in this paper are considered for state of the art video compression standards like MPEG 1, to MPEG4 and H.261 to H.264. Results show that the PSNR of Diamond Search (DS) is best for all test video sequences, whereas, Hardware Modified DS takes maximum number of search points to calculate motion vector. Moreover, hexagon search algorithm takes minimum number of search points but its PSNR is considerably lower than the other algorithms.

## General Terms

Video compression, Motion estimation algorithm, Block matching algorithms

## Keywords

Motion estimation, Motion vector, MBD, DS, FHS, MDS, HexBS,

## 1. INTRODUCTION

H.264/AVC video compression standard has a wide range of application from low data rate like video streaming to High Definition Television (HDTV) broadcast. Motion Estimation (ME) is a critical and computationally expensive part of any video encoder which reduces temporal redundancy between two or more successive video frames. ME algorithms are used to compute the displacement between reference and current frame in the video encoder. Normally, previous frame is considered as a reference frame. The luminance of pixel in the current frame and in reference frame has some correlation with its neighborhood, which is used to determine the best matching position of pixels intensity values of current frame in reference frame. Where accurate match is found, difference in the positions of current and reference frame is calculated which is called Sum of Absolute Difference (SAD). The displacement vector for this SAD is defined as the Motion Vector (MV) [1–2].

There are two basic types of ME process [3], the first one is called pixel based ME, whereas, the second is block based ME. The pixel based ME technique is also known as the optical flow method. It works on the basic assumption of brightness constancy which states that the intensity of a pixel remains constant when it is displaced in the consecutive video frame. In this scheme, MVs are determined for every pixel in

the frame. In block-based ME approach, the current frame is divided into nonoverlapping blocks (for example, the seven possible modes of a block like  $16 \times 16$ ,  $16 \times 8$ ,  $8 \times 16$ ,  $8 \times 8$ ,  $8 \times 4$ ,  $4 \times 8$ ,  $4 \times 4$  are used in H.264) [4]. These blocks are called Macro Blocks (MB). One frame consist of 396 MBs for case of CIF (352x288) video if MB size is  $16 \times 16$ . One best MV is calculated in the reference frame for every MB of current frame [5]. The algorithms which use block-based ME technique are known as Block Matching ME (BMME) algorithms. BMME algorithms remove temporal redundancy between two or more adjacent video sequences, and are essential part of most video coding standards like MPEG-4 or H.264 / HEVC [6].

The best match or MV is calculated by using a cost function in BMME algorithms. This cost function is based on a Minimum Block Distortion (MBD). The cost function may be the SAD or Mean Absolute Differences (MAD) and Mean Squared Error (MSE) [7, 8]. All these cost functions are given in the following equations respectively.

$$SAD = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}| \quad (1)$$

$$MSE = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (C_{ij} - R_{ij})^2 \quad (2)$$

Where  $N^2$  is the total numbers of pixels of a macro block,  $C_{ij}$  and  $R_{ij}$  are the intensity values at the pixel  $(i, j)$  being compared in current MB and reference MB, respectively.

The ME is the most expensive part of video compression process in term of computational complexity and it takes greater than fifty percent of overall encoding time when consider only one reference frame. Moreover, computational cost increases relatively as the number of reference frames increases gradually. For example, for four reference frames computational cost of ME is about 70% of encoding cost [5]. Due to such high computational cost, this field has gone through a tremendous research activity. There are many type of ME algorithms such as Three Step Search (TSS) [9], New Three Step Search (NTSS) [10], Simple and Efficient Search (SES) [11], Four Step Search (FSS) [12], Diamond Search (DS) [13], Hardware Modified Diamond Search (MDS) [14], Adaptive Rood Pattern Search (ARPS) [15] and Cross Diamond Search (CDS) [16]. Moreover, a merge and split procedures based fast variable block-size motion estimation algorithm is proposed in [17] for MPEG-4/ H.264 and to increase the coding efficacy of latest video encoder like ITU-T H.263 and MPEG-4 visual, a multi-frame prediction based algorithm is proposed in [18].

Among these large number of ME algorithms it is necessary to determine which algorithm performs better than the others for a specific type of motion activity. Therefore, in this paper, I have evaluated the performance of various fast ME algorithms like DS, MDS, FSS, HexBS and Flatted HexBS. Moreover, HD videos (720p) are also used for performance evaluation of ME algorithms. The rest of the paper is organized as follows: Section 2 reviews the some famous BMME algorithms. Results are discussed in Section 3 and on the basis of results, conclusion is made in Section 4.

## 2. REVIEW OF FAST ME ALGORITHMS

A large number of BMME algorithms are proposed to decrease computational cost and increase quality of ME process requirement in any video encoder. Some fast famous BMME algorithms are discussed below.

### 2.1 Full-Search BMME Algorithm

Full search (FS) is very basic technique for ME, which compute best MV by searching all possible locations in the search window exhaustively. Therefore, it leads to a very high computation during ME process. FS gives best PSNR among any other BMME algorithm because it computes the MBD at every possible search location in the predefined search window, as a result a best MV is computed [19]. The whole process illustrate in Fig. 1 in which shows that best MV is computed by search sixteen possible locations in the search window.

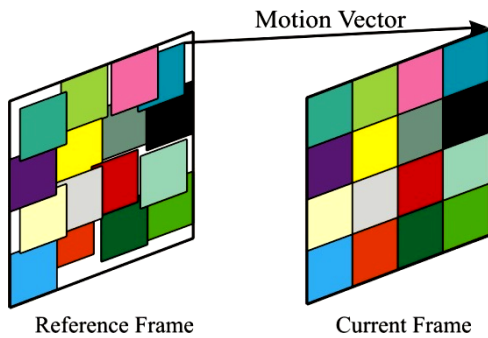
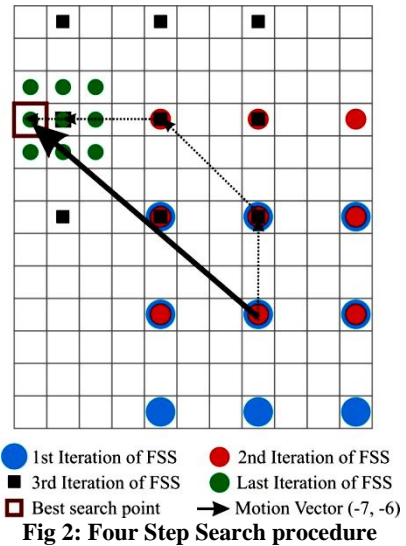


Fig 1: FSBMME, Reference and Current Frame

### 2.2. Four Step Search (FSS)

The general idea of Four Step Search (FSS) is represented in Fig 2. It uses a fixed pattern having size is equal to 2 for the first step and not based on the value of search parameter. It complete search in four steps. In first step it searches nine locations in a window of size 5x5 pixel. If MBD is available at the middle of 5x5 window then the search procedure moves to 4th step and stop the search after finding MBD by using small nine point search pattern, which is known as halfway stop. In other case it moves to the second step by setting the MBD point as center of 5x5 search window. If the MBD point is at center of 5x5 window in step two then stop search by performing fourth step otherwise third step is performed exactly the same as 2nd step. In the fourth step the pattern size decrease from 2 to 1 which causes to reduce the window size up to 3x3. At the end of fourth step search the MBD point is considered as the best corresponding MB and the MV is send for this location. This algorithm takes 17 checking points as a best case while 27 points as worst case [12]. The search flow of FSS is shown in Fig. 2 in which best MV is computed by searching 24 points.



### 2.3. Diamond Search (DS)

DS is very famous algorithm and accepted in the MPEG-4 Video Verification Model version 14.0 [20]. The DS algorithm uses two different types of fixed diamond shaped search patterns, one is large diamond search pattern (LDSP) and the other is small diamond Search pattern (SDSP).

DS algorithm starts by applying LDSP on the macroblock, and then there are two cases, in first case if the MDB occurs at center point of LDSP and in other case MDB occurs at outer corner of demand. For first case, the search pattern is switched from LDSP to SDSP and finds MBD. Whereas in other cases, if MDB occurs at outer corner of demand then LDSP is applied again until MDB occurs at center point. Finally SDSP finds the MBD as the last step of DS algorithm. The MBD provides the best MV for best matching block among all five points in SDSP [13].

The search procedure of diamond search is illustrated in Fig 3, which shows that LDSP is applied two times and at second iteration the MDB obtained at center of LDSP. Then SDSP is applied and final MBD point set as best MV. So, the whole process takes sixteen search points within three iterations to compute final MV.

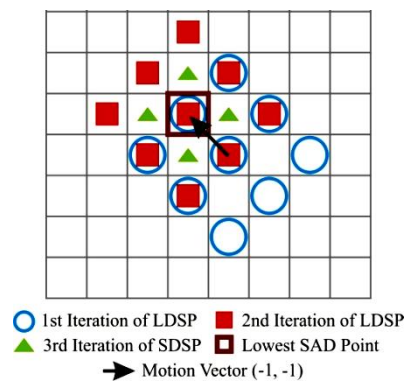


Fig 3: Search flow of DS

### 2.4. Hexagon Based Search (HexBS)

The Hexagonal Based Search (HexBS) algorithm is implemented in H.264/AVC [21] and use two types of search patterns with different sizes like as DS algorithm but the pattern shape is hexagonal instead of diamond. The larger

search pattern consists of 7 check points, called Large Hexagonal Search Pattern (LHSP) while the smaller consist of five check points like SDSP which is named as Small Hexagonal Search Pattern (SHSP) as shown in Fig 4.

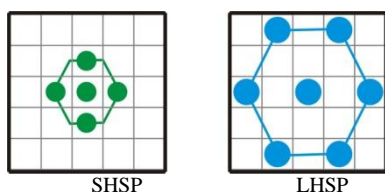


Fig 4: Hexagon Search Patterns

The search process of HexBS algorithm is exactly the same as DS algorithm only the difference is that in first step it checks seven points by using LHSP. In step one if the MBD at the middle point of LHSP then jump to last step otherwise, in next step make the least position as origin and repeat previous iteration until the least position becomes at center. Moreover, leave all the points that were searched in previous step. In last step use the SHSP by setting new origin around MBD point and find the best math location which is set as MV [22, 23].

### 2.5. Novel Flated Hexagon Based Search (FHS)

The FHS uses flatted shaped hexagon search pattern (FHSP) instead of uniform hexagon and a small hexagon search pattern same as SDSP, as shown in Fig 5.

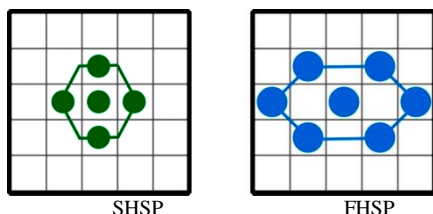


Fig 5: Flatted Hexagon Search Pattern

The FHS algorithm find MV by using flatted hexagon search pattern [24]. The Search flow of FHS algorithm is shown in Fig 6 in which best MV is found in four iterations.

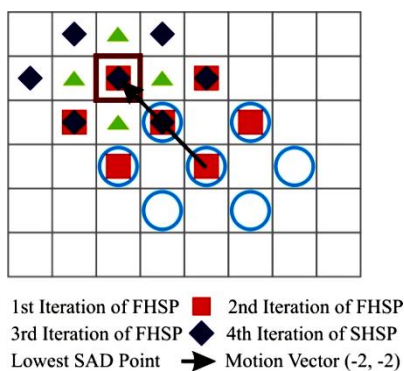


Fig 6: Search process of FHS ME algorithm

The FHS algorithm starts search by setting FHSP at the origin (0, 0) of the search window. If MBD is found at the center of the FHSP, then apply SHSP and stop search, otherwise again apply FHSP by setting minimum MBD as origin until minimum point found at center position. Then apply SHSP

and terminate the search procedure by setting minimum MBD position as best MV.

### 2.6. Hardware Modified Diomand Search (MDS)

MDS ME algorithm is designed to reduce the complexity of algorithm when implemented on the hardware and uses two types of search pattern like Modified Diamond Search Pattern (MDSP) which consist of seventeen points and LDSP having thirteen search points as shown in Fig 7.

The MDS ME algorithm compute best MV in three steps and start search by applying MDSP at center of search window and find the MBD point in step 1. In second step the MBD point set as new origin and apply LDSP to identify new MBD for last third step. Apply LDSP again in last step and MBD point declared as best MV. Therefore, it can be noted that the MDS algorithm always use a fixed 41 search points in each step to calculate each MV [14].

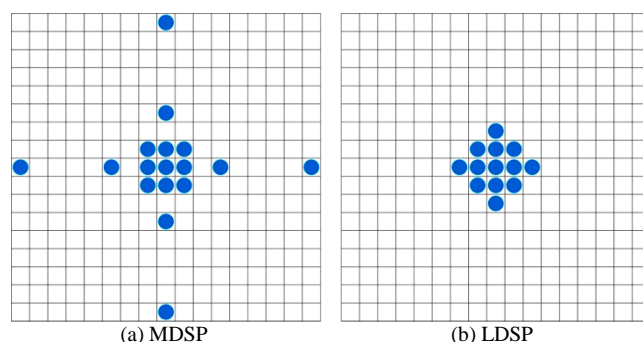


Fig 7: MDS search patterns

## 3. RESULTS AND DISCUSSION

The compilation and simulation is done by using MATLAB 2014a software tool. The specifications of the computer system which is used for the compilation and simulation are: Intel® Core™2 Duo CPU T6570 @ 2.10G processor, 2 MB L2 Cache, 800 MT/s FSB, 4 GB DDR3 RAM, video card of 512 MB memory and 32bit Windows®7 operating system. Various parameters were used for simulation like video format is “yuv”, MB size is 16 and search range is +8 etc. Moreover four performance measuring parameters like Number of Search Points (SP), ME time, PSNR and MSE were used for performance evaluation of five famous fast BMME algorithms like DS, HexBS, FHS, FSS and MDS.

Eleven different video sequences [25], each having 100 video frames were used during the experiment for better analysis of each ME algorithm. Moreover, I also include some HD (720p) videos in my analysis by keeping in mind the upcoming HD video broadcasting requirements. The test video sequences are selected on the basis of different motion activities [26]. For example, InToTree and OldTownCross are the 720p videos having unidirectional and low motion contents. Similarly, for CIF videos, News, Akyio and Miss America are the video sequence having very low motion contents, but Foreman, Soccer, Football and Bridge Close are having multi-directional and high amount of motion activity. In addition with the HD videos like DucksTakeOff and Parkjoy having moderate unidirectional motion and high bidirectional motion respectively.

The number of SP for computing MV by any algorithm directly effect on computational complexity and ME time. The large number of SP means that the algorithm is more

computationally complex and takes excessive ME time for MV calculation. Therefore, number SP is very important parameter for performance evaluation. The average number of SP of one MB for 100 video frames from different test video sequences to calculate the MVs by each algorithm is shown in Fig 8. It can be observed that HexBS takes minimum number of search points whereas MDS still takes maximum number of search points for all types of video sequence. FHS ME algorithm takes approximately same number of search points where there is low unidirectional motion contents.

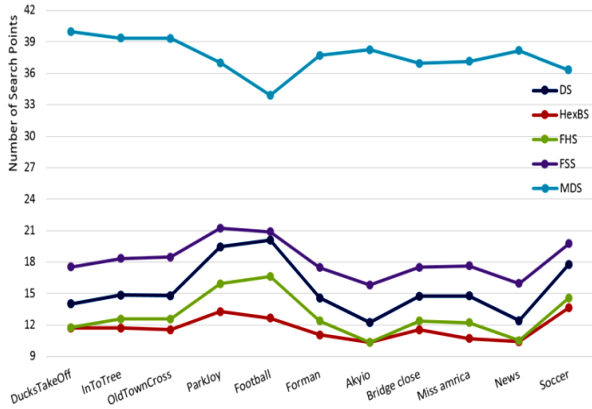


Fig 8: Average number of SP of 100 HD and CIF video frames

Fig 9 shows the results for average ME time of 100 frames of different test video sequences taken by each algorithm to calculate the MVs.

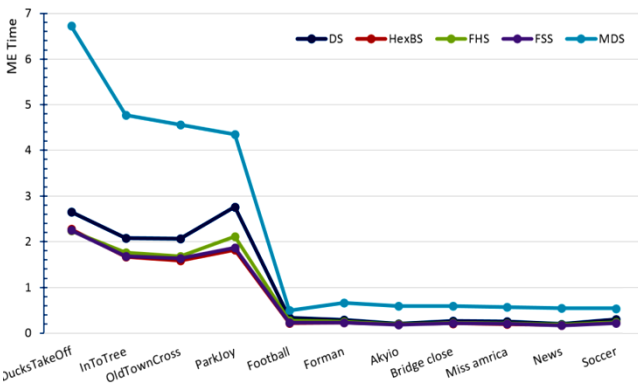


Fig 9: Average ME time of 100 HD and CIF video frames

It can be observed that ME time is quite high for MDS for all types of video sequences because it also takes more SP than other, however, HexBS performs well for approximately all the video sequence in term of ME Time. While the FSS is at fourth number in term of ME time.

Peak signal to noise ratio (PSNR) and MSE performance parameters gives the information about quality measurement. Higher the PSNR, more accurate calculated MV. The PSNR can be computed by using equation 3.

$$PSNR = 10 \log_{10} \left[ \frac{(\text{peak to peak value of original data})^2}{MSE} \right] \quad (3)$$

Fig 10 shows the average PSNR taken by each algorithm to calculate the MVs of 100 videos frames of different test video sequences.

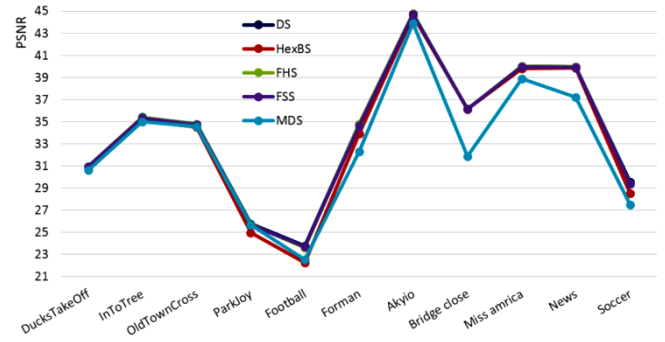


Fig 10: Average PSNR of 100 HD and CIF video frames

PSNR of DS, FHS and FSS is approximately same for all of the video sequences. The PSNR of MDS algorithm is worst for all the video sequences except Parkjoy and Football. PSNR of HexBS is less than the MDS for these two video sequences.

The results for average MSE of 100 frames from different test video sequences taken by each algorithm to calculate the MVs is shown in Fig 11.

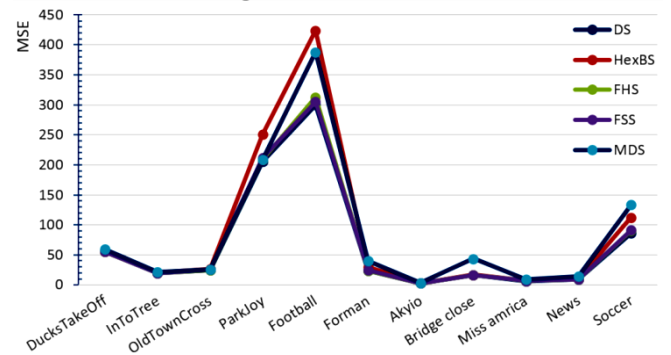


Fig 11: Average MSE of 100 HD and CIF video frames

As PSNR is a function of MSE, so greater the PSNR, lower the MSE. MSE of DS is approximately same as the other ME algorithms, but DS has the lowest MSE and outperforms all the ME algorithms in case of MSE. Whereas, MSE of MDS and HexBS is quite high as compare to the rest of the algorithms.

The ME algorithms can be categorized from the following expression, in term of SPs taken to compute MVs:

$$MDS \succ FSS \succ DS \succ FHS \succ HexBS$$

The overall performance of ME algorithms in term of PSNR can be categorized as follows:

$$DS \succ FHS \succ FSS \succ HexBS \succ MDS$$

The PSNR of MDS and HexBS vary for some video sequences, but always remain less than the other algorithms.

#### 4. CONCLUSION

This paper evaluates the performance of five fast BMME algorithms by considering four critical performance measuring parameters. The results show that HexBS takes minimum number of SP and ME Time for both, HD and CIF videos, to calculate MVs with some loss of PSNR as

compared to other algorithms. FHS is at second number in term of SPs HexBS and gives improved results for PSNR but less than DS. FSS and MDS are at fourth and fifth number in term of SP and ME time, whereas, PSNR of MDS has also degraded for multidimensional sequence. Moreover, DS computes more accurate MVs than the other ME algorithms. Therefore, it can be concluded that DS is the recommended algorithm when high quality of accuracy is required like video surveillance etc. Whereas, HexBS can be used for the scenarios where quality can be compromised but fast encoding is the key requirement, like videos conferencing and live streaming applications.

## 5. ACKNOWLEDGEMENT

This project was funded by Deanship of Scientific Research (DSR) King Abdulaziz University, Jeddah, under the grant no. (431/009), the authors therefore, acknowledge with thanks DSR technical and financial support, furthermore, the authors would like to express their gratitude and appreciation to Tokai University for their technical help and support.

## REFERENCES

- [1] I. Ali, M. Muzammil and G. Raja, 2012 "performance analysis of motion estimation algorithms based on motion activity in video sequences" Pakistan Journal of Science Vol. 64 No. , pp no. 39-45
- [2] I. Ali, G. Raja, M. Muzammil; A.K. Khan, 2014 "Adaptive Modified Hexagon Based Search Motion Estimation algorithm," 2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin), pp.147-148.
- [4] A.Ahmadi, M.M.Azadfar, 2008 "Implementation of fast motion estimation algorithms and comparison with full search method in H.264" IJCSNS International Journal of Computer Science and Network Security, Vol.8 No.3, pp 139-143.
- [3] Multimedia Processing, Video Coding and Motion Estimation (module 7, Lesson 21) IIT Kharagpur course material, available on-line at, [http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT\\_Kharagpur/Multimedi Processing/pdf/ssg\\_m7l21.pdf](http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT_Kharagpur/Multimedi%20Processing/pdf/ssg_m7l21.pdf)
- [5] Aroh Barjatya, 2004 "Block Matching Algorithms for Motion Estimation". DIP 6620, Final project paper.
- [6] S.Radacke, J.-U.Hahn, Wang Qi and C.Grecos, 2014, "Bi-predictive motion estimation for HEVC on a graphics processing unit (GPU)," *IEEE Transactions on Consumer Electronics*, vol.60, no.4, pp.728-736.
- [7] S.R. Subramanya Hiral Patel Ilker Ersoy, 2004, "Performance Evaluation of Block-Based Motion Estimation Algorithms and Distortion Measures" IEEE Int. Conf. Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04). Vol. 2 , pp. 2-7.
- [8] R. A. Manap, S. S. S. Ranjit, A. A. Basari, and B. H. Ahmad, 2010, "Performance Analysis of Hexagon-Diamond Search Algorithm for Motion Estimation" IEEE Int. Conf. Computer Engineering and Technology (ICCET) Vol. 3, pp no. V3-155 - V3-159.
- [9] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, 1981, "Motion compensated inter frame coding for video conferencing," in Proc. NTC 81, pp. C9.6.1-9.6.5, New Orleans, LA.
- [10] Renxiang Li, Bing Zeng, and Ming L. Liou, 1994, "A New Three-Step Search Algorithm for Block Motion Estimation" IEEE Transactions on Circuits and Systems for Video Technology, Vol. 4, No. 4.
- [11] Jianhua Lu and Ming L. Liou, 1997, "A Simple and Efficient Search Algorithm for Block-Matching Motion Estimation" IEEE Transactions on Circuits and Systems for Video Technology, Vol. 7, No. 2, pp no 429-433.
- [12] Lai-Man Po, and Wing-Chung Ma, 1996, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", IEEE Trans. Circuits and Systems for Video Technology, Vol. 6, no. 3, pp. 313-317.
- [13] Shan Zhu, and Kai-Kuang Ma, 2000, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", IEEE Trans. Image Processing, vol 9, no. 2, pp. no. 287-290.
- [14] G. Sanchez, D. Noble, M. Porto and L. Agostini, 2011, "A Real-Time HDTV Motion Estimation Architecture for the New MPDS Algorithm," in *EUROCON International conference on computer as a tool*, Lisbon.
- [15] Yao Nie, and Kai-Kuang Ma, 2002, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", IEEE Trans. Image Processing, vol. 11, no. 12, pp. 1442-1448.
- [16] Chun-Ho Cheung, and Lai-Man Po, 2002, "A Novel Cross-Diamond Search Algorithm for Fast Video Coding and Video Conferencing Applications", IEEE Transactions on Circuits and Systems for Video Technology, Vol.12, No. 12, pp 1168-1177.
- [17] Efficient Architecture for Variable block size Motion Estimation of H.264 Video Encoder, 2012, International Conference on Solid-State and Integrated Circuit (ICSIC 2012) IPCSIT vol. 32, IACSIT Press, Singapore
- [18] Ljubomir Jovanov, Aleksandra Pižurica, Stefan Schulte, 2009, "Combined Wavelet-Domain and Motion-Compensated Video Denoising Based Video Codec Motion Estimation Methods" IEEE transactions on circuits and systems for video technology, Vol. 19, No. 3.
- [19] Vasiljevic, J.; Ye, A.G., 2012, "Effect of scaling on the area and performance of the H.264/AVC full-search fractional motion estimation algorithm on field-programmable gate arrays," *IET. Computers & Digital Techniques*, vol.6, no.2, pp. 95-104.
- [20] Coding of moving pictures and audio, 1999, ISO/IEC JTC1/SC29/WG11N2932.
- [21] Sourabh Rungta, Dr. Neeta Tripathi and Prof Anupam Shukla, 2011, "Hexagonal Based Search Pattern for Motion Estimation in H.264/AVC," World of Computer Science and Information Technology Journal (WCSIT), vol. 1, no. 2221-0741, pp. 162- 166.
- [22] Ce Zhu, Xiao Lin, Lap-Pui Chau, Keng-Pang Lim, Hock-Ann Ang, Choo-Yin Ong, 2001, "A Novel Hexagon-Based Search Algorithm for Fast Block Motion Estimation" IEEE Conference on Acoustics, Speech, and Signal Processing vol.3 pp no. 1593-1596.
- [23] Ce Zhu, Xiao Lin, and Lap-Pui Chau, 2002, "Hexagon-Based Search Pattern for Fast Block Motion Estimation"

IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, NO. 5, pp no 340-355.

- [24] Thou-Ho Chen, Yi-Fan Li, 2004, "A novel flatted hexagon search pattern for fast block motion estimation" *International Conference on Image Processing, (ICIP 2004)*, vol.3, no., pp.1477-1480 Vol. 3, 24-27.

[25] Available online at: <https://media.xiph.org/video/derf/>

- [26]G.Raja, M. J. Mirza and T. Song, 2008, H.264/AVC deblocking filter based on motion activity in video sequences. *IEICE Electronics Express J.*, 5:809-814.