Survey on Block Matching Algorithms for Motion Estimation

Chittaranjan Pradhan
School of Computer Engineering
KIIT University, Bhubaneswar
Odisha, India.

Dipannita Adak
School of Computer Engineering
KIIT University, Bhubaneswar
Odisha, India.

ABSTRACT
Motion estimation technique is the most vital component of any video coding standard. Therefore, development of an efficient method for fast motion estimation is the basic requirement of the video encoder design. Block based motion estimation algorithms are used to reduce the memory requirements of any video file and also decrease computational complexity. Motivated by the specific requirements of motion estimation, a variety of algorithms have been developed. In this paper, we have discussed the commonly used motion estimation algorithms such as- Full Search (FS), Three-Step Search (TSS), New Three-Step Search (NTSS), Four-Step Search (FSS), Diamond Search Algorithm (DS), and Hexagon Based Search Algorithm (HEXBS). We have also analyzed these techniques by using Peak Signal to Noise Ratio (PSNR) values.

Keywords
Motion estimation, block matching, motion vector and block distortion measure.

1. INTRODUCTION
The demand for communications with moving video picture is rapidly increasing. In normal video processing, the system needs to send dozens of individual frames per second to create an illusion of a moving picture. For this reason, several standards [9] for compression of the video such as MPEG-1, MPEG-2, MPEG-4 and MPEG-7 have been developed. Video coding achieves higher data compression rates without significant loss of picture quality.

MPEG-encoding technique uses block-based motion-compensation. This method takes advantage of temporal redundancy between two or more frames. Motion estimation is the process of finding a fixed region of a reference frame of video to search a matching block of pixels of the same size under consideration in the current frame. The advantage of this technique is that the image data for the background and for the objects is stored only in one frame - the following frames contain the motion vectors. This process is widely used in block-matching algorithms.

In block matching approach [7], each image frame is divided into non-overlapping rectangular blocks of equal size of 8x8 pixels, each of which consists of luminance and chrominance blocks. Generally, motion estimation is performed only on the luminance block. For each luminance block, the algorithm finds matching block – a block in the reference frame that matches the current block best. The best candidate block is found and its displacement (motion vector) is recorded. We assume the input images are grayscale only.

Block matching uses a value called "block distortion measure" (BDM) - to rate the similarity between two blocks. The basic idea is to sum up the square differences of the pixel luminance of pixels located at the same position in the two blocks. The "Mean Squared Error" (MSE) uses the sum of squared differences between the two luminance values. The sum is divided by the quantity of the compared pixels to normalize the result.

$$MSE\,(i,j) = \frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}[F(m,n) - G(m+i, j+n)]^2 \quad (1)$$

Here, M×N is the size of the sub block. F (m, n) is the intensity of the pixel located at (m, n) of the current frame. G (m+i, n+j) is the intensity of the pixel located at (m+i, n+j) of the reference frame and (i, j) is the motion vector.

This study represents a survey of various fast existing block-matching motion estimation algorithms- Full Search, Three Step Search, New Three Step Search, Four Step Search, Diamond Search and Hexagon Based Search Algorithm. All of these existing motion estimation algorithms use "Mean Squared Error" (MSE) formula to evaluate the minimum BDM position. Experimental results show the performance of these algorithms mainly based on PSNR (Peak Signal to Noise Ratio) [10]. PSNR value is calculated by the following equation (2):

$$PSNR = 10 * \log\left(\frac{255*255}{error}\right) \quad (2)$$

Here error is the MSE value.

2. EXISTING MOTION ESTIMATION ALGORITHMS
In this section, we have presented some popular existing motion estimation algorithms such as Full Search (FS), Three Step Search (TSS), New Three Step Search (NTSS), Four Step Search (FSS), Diamond Search (DS) and Hexagon Based Search (HEXBS) algorithm.

2.1 Full Search Algorithm (FS)
This is a simple block matching algorithm. Full search [8] or Exhaustive search algorithm compares the current block with all the candidate blocks of the reference frame within the search area, for finding out the best matched block in the reference frame (Figure 1). The minimum BDM point is calculated by MSE (Mean Squared Error) formula.

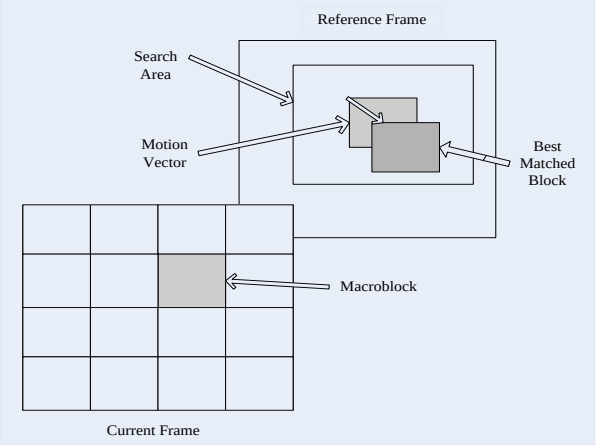

Figure 1: Full Search Algorithm

This algorithm gives the global minimum block distortion position which corresponds to the best matching block. Calculation costs of this algorithm are very high because we have to check all the candidate blocks for minimum BDM point.

Figure 2: Screenshot of Full Search Algorithm

The Figure 2 shows the screenshot of full search algorithm. Here "Foreman" image sequence is used and search range is 7. We have taken image01 and image02- two frames from "Foreman" image sequence. The black window shows MSE values of each compared blocks and corresponding PSNR (Peak signal to Noise Ratio) values. In this technique, number of compared blocks are 225. The yellow dot located at (4, 0) coordinate, denotes the minimum BDM point and green dot denotes the current search position. At (4, 0) point error (MSE) value is 69.80, PSNR (Peak Signal to Noise Ratio) is 68.37. The black rectangle marks the block located in the displayed frame, a red rectangle marks the block in the other frame. The red line is the resulting motion vector.

2.2 Three Step Search Algorithm (TSS)

Another fast and efficient motion estimation algorithm is the three-step search [6]. It mainly used for low bit-rate video compression applications (such as videophone and video conferencing), because of its simplicity and effectiveness. The following steps describe the TSS algorithm:

Step 1: Create a search pattern consisting of 9 coordinates. These are the centre of the search area (0,0) and the 8 coordinates with a horizontal and/or vertical distance of 4 pixels to it as shown in Figure 3. Find the coordinate with the mimimum block distortion (BDM) value.

Step 2: Halve the search range to 2 pixels. Create a new search pattern with the eight coordinates surrounding the current best matching coordinate. Again, find the coordinate with the minimum block distortion (BDM) point.

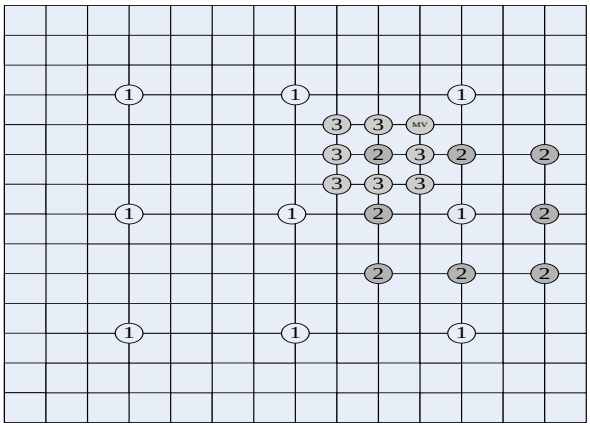

Figure 3: Steps of TSS Algorithm

Step 3: Reduce the search range again to 1 pixel. Then, create a search pattern existing of the eight coordinates surrounding the current best matching position. Finally, search for the minimum BDM. This is the coordinate, the motion vector will point to.

TSS is more efficient to find the global minimum particularly for those sequences with large motion. On the other hand, it becomes inefficient for the estimation of small motions as it will be trapped into a local minimum. It reduces the quality of the motion compensation system.

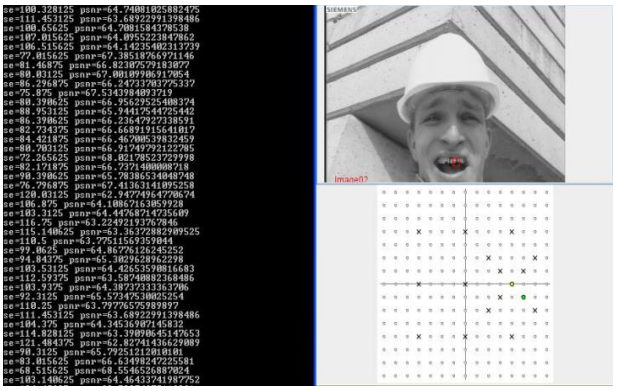

Figure 4: Screenshot of Three Step Search Algorithm

Figure 4 shows the minimum BDM (Block Distortion Measure) position which is (4,0). At that point error value (MSE) is 72.27, psnr (Peak Signal to Noise Ratio) is 68.02. The yellow dot is the minimum BDM position and the green dot is the current search point. This algorithm compares 17 blocks. The BDM point (4,0) is same for both FS and TSS. Therefore, this algorithm is better than FS algorithm when compared with number of search blocks.

2.3 New Three Step Search Algorithm (NTSS)

The NTSS algorithm [1] is a more centre biased variant of TSS algorithm. In the first step, the eight directly adjacent coordinates to the centre coordinate are added to the initial search pattern. The steps of the algorithm in detail:

Step 1: Create the initial search pattern, which consists of the centre coordinate, the eight coordinates surrounding it at a

distance of one pixel and at a distance of four pixels as shown in Figure 5. Search the minimum BDM of this pattern. If the centre coordinate is the minimum BDM, the search is finished. Otherwise: Go to step 2.

***Step 2*:** If the minimum BDM is one of the eight direct neighbours of the centre coordinate, add the eight direct neighbours of this coordinate and search it again for the minimum BDM. After that, the search is done. Otherwise, continue like TSS: add the four coordinates surrounding the BDM like an "X" at a distance of 2 pixels and search for the new minimum BDM.

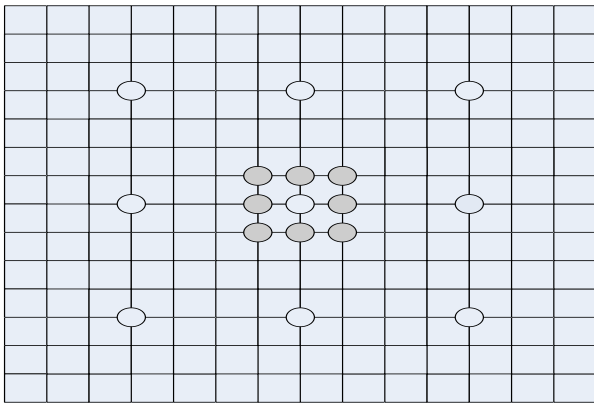

Figure 5: Initial Search Pattern of NTSS Algorithm

Step3: Search the four coordinates surrounding the current minimum BDM like an "X" at a distance of 1 pixel for the final minimum BDM.

The search pattern of this algorithm is fixed and no predefined threshold operations are required. This algorithm does not perform well in worst case condition, because 33 (9+8+8+8) blocks are compared in worst case.

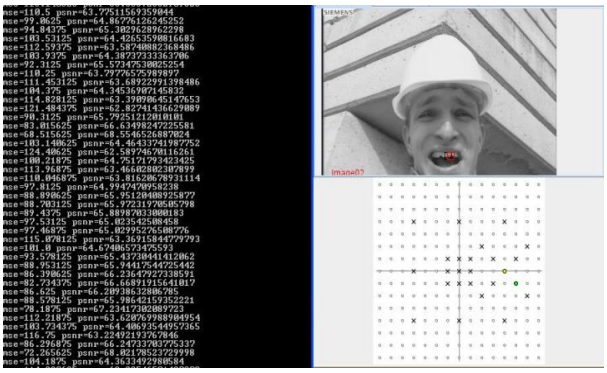

Figure 6: Screenshot of New Three Step Search Algorithm

Figure 6 shows the minimum BDM (Block Distortion Measure) position which is (4,0). At that point error value (mse) is 72.27, psnr (Peak Signal to Noise Ratio) is 68.02. This Algorithm compares 25 blocks. This algorithm performs better than FS algorithm when compared with number of compared blocks. Both TSS and NTSS gives same psnr value, which indicates poor performance than FS algorithm.

2.4 Four Step Search Algorithm (FSS)

FSS [2] algorithm is also based on the TSS algorithm. The intention of this algorithm, was creating an algorithm which achieves better results than the TSS algorithm while having a better worst case performance than the NTSS algorithm. The search range of FSS is fixed to 7 pixels. Steps of the algorithm:

Step 1: Create a 5x5 search pattern around the centre of the search area, consisting of the nine coordinates surrounding the centre. Find the BDM of these coordinates. If the minimum is located at the centre, proceed with *Step 4*. Otherwise: Go to *Step 2*.

Step 2: The search pattern still has the size of 5x5. Depending on the location of the current BDM, new search coordinates are added to the pattern.

a)If the current minimum BDM is located at the corner of the pattern in *Step 1*, add 5 additional coordinates positioned next to the corner as shown in Figure 7(b).

b)If the minimum BDM is located at the centre of a vertical or horizontal axis of the search pattern in *Step 1*, add 3 coordinates to the pattern as shown in Figure 7(c). Search for the new BDM. If its position has not changed after this search, continue with *Step 4*.

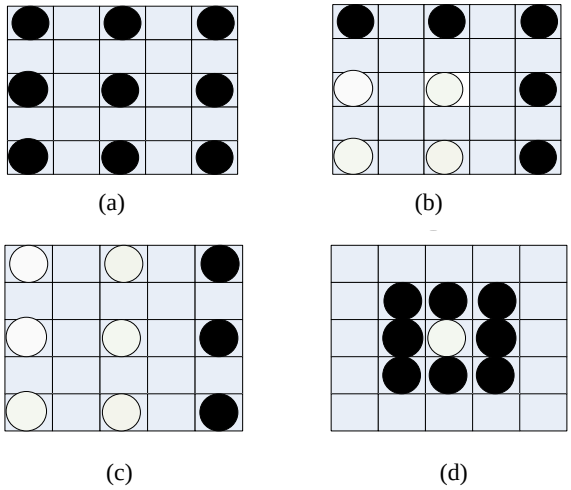

Figure 7: Steps of FSS Algorithm

Step 3: Repeat *Step 2* once and proceed with *Step 4*.

Step 4: The search window size is reduced to 3x3 around the minimum BDM as shown in Figure 7(d). Search all eight new coordinates in this pattern for the final minimum BDM.

This algorithm helps in reducing the number of search points when compared to the TSS and therefore, it is more robust. In worst case situation, it compares 25 (9+5+5+8) blocks. Hence, FSS performs better in the worst case when compared with NTSS algorithm. Computational complexity and picture quality remain almost same when compared to the TSS and NTSS algorithm.

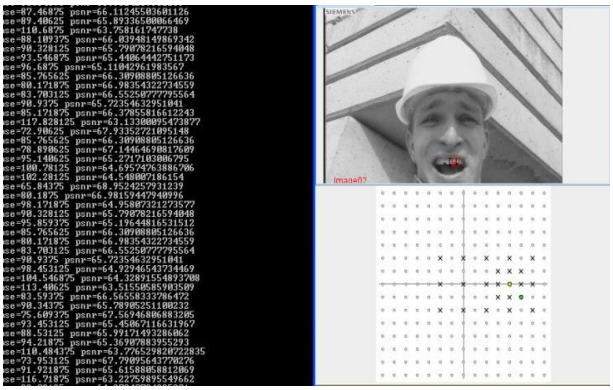

Figure 8: Screenshot of Four Step Search Algorithm

Figure 8 shows the minimum BDM (Block Distortion Measure) position which is (4,0). At that point error value is 63.08, psnr (Peak Signal to Noise Ratio) is 69.38. It compares 23 blocks. This algorithm is better than FS algorithm when compared with number of compared blocks.

2.5 Diamond Search Algorithm (DS)

Diamond Search algorithm takes a diamond-shaped search pattern. The two fixed types of search patterns used in DS [3] algorithm. The first pattern is- large diamond search pattern (LDSP) shown in Figure 9 (a), consists of nine checking points from which eight points surround the center to form a diamond shape. The second pattern with five checking points forms a small diamond shape, called small diamond search pattern (SDSP) shown in Figure 9 (b).

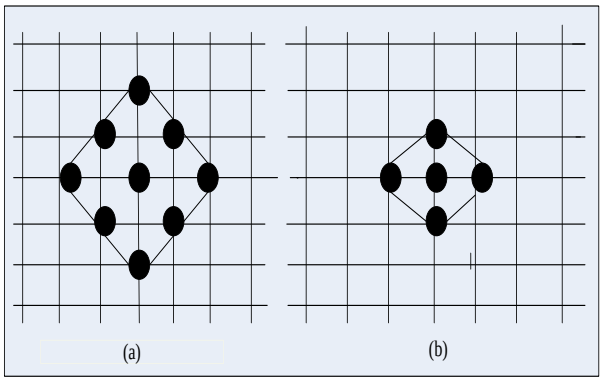

Figure 9: (a) Large Diamond Search Pattern (LDSP), and (b) Small Diamond Search Pattern (SDSP)

The Diamond Search algorithm works as follows:

Step 1: For the initial LDSP, the center is at (0, 0), and the nine search points of LDSP are tested. If the minimum block distortion (BDM) point is present at the center, then go to Step 3; otherwise, go to Step 2.

Step 2: The minimum BDM obtained in the previous step is re-positioned as the new center point to form another LDSP. If the new BDM point found is located at the center position, then go to Step 3; else, recursively repeat this step.

Step 3: The search pattern is switched from LDSP to SDSP. The final result of the motion vector (MV) is obtained by minimum BDM point found in this step. This MV points to the best matching block.

The DS pattern can find large motion blocks with fewer search points. The compact shape of the DS pattern around the center also yields fewer search points than other square-shaped algorithm for finding stationary or small motion vectors. The diamond shape is not approximating sufficient to a circle, which is just 90 rotation of square.

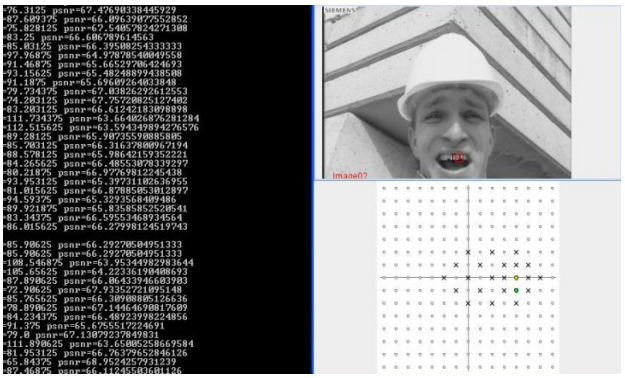

Figure 10: Screenshot of Diamond Search Algorithm

Figure 10 shows the minimum BDM (Block Distortion Measure) position which is (4,0). At that point error value is 65.84, psnr (Peak Signal to Noise Ratio) is 68.95. This algorithm compares 23 blocks. The yellow dot indicates minimum BDM position.

2.6 Hexagon Based Search Algorithm (HEXBS)

A hexagon-based search pattern [4] [5] consists of seven checking points with the center bounded by six endpoints of the hexagon. The HEXBS algorithm is described in the following steps:

Step 1: The large hexagon with seven checking points is centered at (0, 0). If the minimum block distortion (BDM) point is found to be at the center of the hexagon, then proceed to Step 3; otherwise, proceed to *Step 2* as shown in Figure 11.

Step 2: With the minimum BDM point in the previous step as the center, a new large hexagon is formed. Three new points are checked. If the minimum BDM point is still the center point of the newly formed hexagon, then go to *Step3*; otherwise, repeat this step continuously.

Step 3: Switch the search pattern from the large to the small size of the hexagon. The four points covered by the small hexagon are checked and compared with the current BDM point. The new minimum BDM point is the final result of the motion vector.

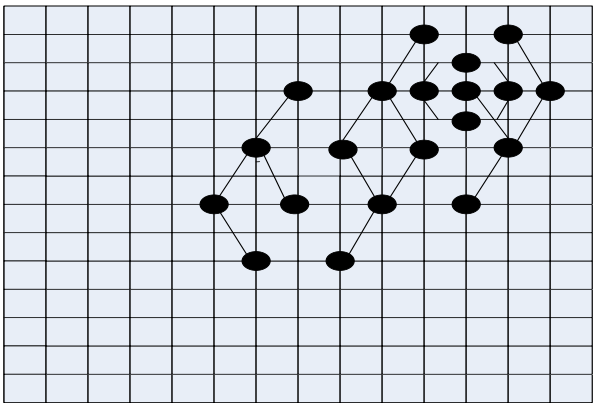

Figure 11: Hexagon-Based Search Algorithm

The hexagon-based search scheme finds any motion vector in motion field with fewer search points than the DS algorithm. This motion estimation algorithm doesn't have significant execution speed to reduce encoding time.

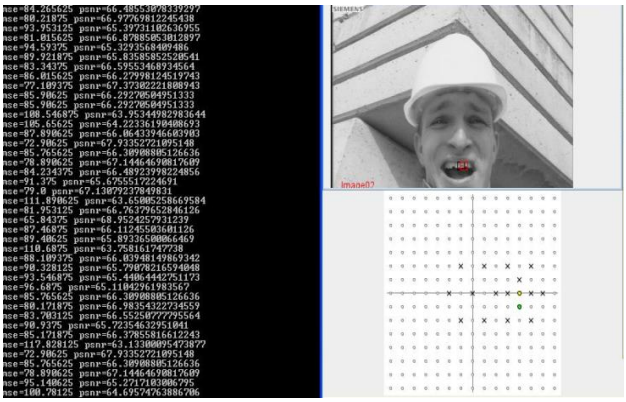

Figure 12: Screenshot of Hexagon-Based Search Algorithm

The above screenshot (Figure 12) shows the minimum BDM (Block Distortion Measure) position which is (4,0). At that point error (MSE) value is 65.84, psnr is 68.95. This algorithm compares 17 blocks. The yellow dot indicates minimum BDM position.

3. COMPARISON STUDY

The algorithms have been tested with frames of the video sequences such as "Foreman", "Garden" and "Container". The Figure 13 shows the graph, based on the PSNR values of these three sequences. The PSNR (Peak Signal to Noise Ratio) value is calculated at minimum block distortion point (BDM) point (4, 0) for each algorithm.

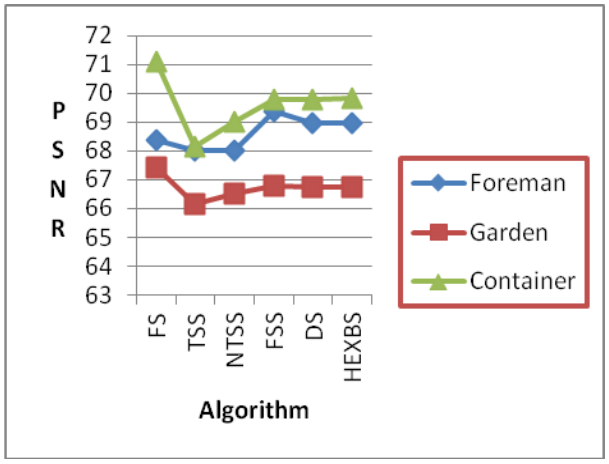

Figure 13: Graph based on PSNR values of the motion estimation algorithms

The full search (FS) algorithm gives better performance for "Garden" and "Container" sequences, but in "Foreman" sequence this algorithm shows poor performance. Four-step search algorithm (FSS) gives better result in "Foreman" sequence than other MEAs. The three-step search (TSS) and new three-step search (NTSS) show poor performance for these three sequences. The diamond search (DS) and hexagon-based search (HEXBS) technique give almost same PSNR values for the above three image sequences, as well as decrease number of compared blocks. Therefore, DS and HEXBS algorithms give overall good performance.

4. CONCLUSION

In this paper, we have discussed and analyzed the most common block matching algorithms. From this analysis, we have found that the full search (FS) technique produces better quality image as it gives better performance in PSNR calculation, but takes larger number of search points; whereas, diamond search (DS) and hexagon-based search (HEXBS) algorithms take a few numbers of search points and also give average performance in PSNR calculation. Other algorithms (i.e. TSS, NTSS, and FSS) take lesser number of search points, but produce distorted image because of poor PSNR performance. As DS and HEXBS algorithms take fewer number of search points, they are faster. The image quality of DS and HEXBS can be improved by increasing the PSNR values. Therefore, getting a better quality image by reducing number of search points remains a goal.

5. REFERENCES

[1] Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 4, No. 4, pp. 438-442, August 1994.

[2] Lai-Man Po, Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation" , IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 3, page 313-317, June 1996.

[3] Jo Yew Tham, Surendra Ranganath, Maitreya Ranganath, and Ashraf Ali Kassim, "A Novel Unrestricted Center-biased Diamond Search Algorithm for Block Motion Estimation", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 8, No. 4, pp. 369-377, August 1998.

[4] Ce Zhu, Xiao Lin, Lap-Pui Chau, Keng-Pang Lim, Hock-Ann Ang, Choo-Yin Ong, "A Novel Hexagon-Based Search Algorithm for Fast Block Motion Estimation ", IEEE, pp. 1593-1596, 2001.

[5] Ce Zhu, Xiao Lin, and Lap-Pui Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No. 5, pp. 349-355, May 2002.

[6] Aroh Barjatya, "Block-Matching Algorithms For Motion Estimation", DIP 6620 Spring 2004 Final Project Paper.

[7] S. Immanuel Alex Pandian, Dr. G. Josemin Bala, Becky Alma George,"A Study on Block Matching Algorithms for Motion Estimation", International Journal on Computer Science and Engineering (IJCSE), Vol. 3, No. 1, pp. 34-44, Jan 2011.

[8] D.V.Manjunatha, Dr. Sainarayanan, "Comparison and Implementation of Fast Block Matching Motion Estimation Algorithms for Video Compression", International Journal of Engineering Science and Technology (IJEST), Vol. 3, No. 10, pp. 7608-7613, October 2011.

[9] John Watkinson, "The MPEG Handbook", 2nd Edition, Chapter 5, Elsevier Ltd. 2004, pp. 230-352.

[10] PSNR calculating formula. Available at http://www.chasanc.com/index.php/Coding/PSNR-Calculating-Difference.html.