Abstract- Motion estimation is a significant process in video compression. In the last two decades, several block matching algorithms were proposed for motion estimation. The objective of block matching algorithms is to reduce the computational time while maintaining the quality of the video sequence. This paper presents the performance comparison of recent block matching algorithms used in video compression. The role of various search patterns including diamond, hexagon, square and octagon search patterns are analyzed. Experiments are carried out for large, medium motion and slow motion video sequences. It is observed that Cross Octagon Search (COS) algorithm performs well for small motion video sequences and OCTSS perform reasonably well for all types of motion in video sequences.

Index Terms- Block matching algorithm, temporal redundancy, Motion estimation

I. INTRODUCTION

In video compression, temporal redundancies are identified by comparing two frames. This process is termed as Motion Estimation. There are two types of Motion Estimation methods: pixel-based and block-based motion estimation methods. The pixel-based motion estimation method seeks to determine motion vectors (MV) for every pixel in the image. This is also referred to as the ‘optical flow method’, which works on the fundamental assumption of brightness constancy, that is, the intensity of a pixel remains constant when it is displaced. However, no unique match for a pixel in the reference frame is found in the direction normal to the intensity gradient. It is for this reason that an additional constraint is also introduced in terms of the smoothness of velocity (or displacement) vectors in the neighborhood. The smoothness constraint makes the algorithm interactive and requires excessively large computation time, making it unsuitable for practical and real-time implementation. In the block-based motion estimation method, the candidate frame is divided into non-overlapping blocks (of size 16 x 16, or 8 x 8, or even 4 x 4 pixels in the recent standards) and for each such candidate block, the best MV is determined in the reference frame [1]. The block matching algorithm decides how the candidate block is compared with a block in the reference frame. The blocks are compared within a search area in the frame.

The block matching algorithms [1] that have been implemented earlier are Full Search (FS), Three Step Search (TSS), New Three Step Search (NTSS), Simple and Efficient TSS (SES), Four Step Search (4SS), Cross Search Pattern, Diamond Search (DS), Cross Diamond Search (CDS), Hexagon-based Search (HEXS), Hexagon-Diamond Search (HDS), Octagon-based Search (OCTBS) etc.

Full search (FS) [3] is the initial block searching technique in which the best match in the reference frame is found by checking the current block with the candidate blocks. The other block matching algorithms are explained in [1].

In 2005, cross-diamond-hexagonal search (CDHS) [4] algorithms is introduced. This algorithm differs from HDS by its size of search patterns. CDHS consists of two cross-shaped search patterns consecutively in the very beginning steps and it switches to diamond-shaped patterns. To further reduce the computational time, two pairs of hexagonal search patterns are introduced in conjunction with candidates found located at diamond corners.

In 2005, Cross Octagon Search (COS) algorithm is developed [5]. This algorithm uses cross pattern, octagon pattern and diamond pattern for searching. Cross pattern is used in the first step to know the direction of motion in the video sequence.

In 2006, a fast algorithm Unsymmetrical-cross Multi-Hexagon-gird Search (UMHexagonS) [6] is developed. UMHexagonS algorithm adaptively adopt different searching pattern according to similarity measure used. It also has early termination strategy to reduce the searching time. Although UMHexagonS algorithm performs well, there is information redundancy between the search points.

In 2009, Cross hexagon based algorithm (CHS) [7] is developed by Li Hong ye. In this algorithm cross pattern & hexagonal algorithm is combined & it gives the better result than the hexagonal search. In 2010 R.A. Manap et al proposed hexagon-diamond search. The HDS algorithm gives the better result than the NTSS and FS algorithms.

In 2012, a fast Octagon and triangle search pattern (OCTTS) [8] is developed by C. Duanmu and Yu Zhang.
The multi-octagon-grid search pattern is used in this algorithm. This pattern has much less computational complexity than that of the multi-hexagon-grid search pattern in the UMHexagonS algorithm and this pattern has good search efficiency. In 2014, we have introduced the Octagon and Square Pattern (OCTSS) for searching [9]. Our algorithm uses Octagon pattern for large motion and square pattern for small motion video sequences.

The block matching algorithms uses a measure called Block Distortion Measure (BDM) to calculate the similarity between blocks. It is best to calculate the sum of the squared differences of the pixels located at the same position in the two blocks. Mean Squared Error is the measure which calculates the squared differences of the pixels. It is given by

$$\text{MSE} = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(f(M,N) - f'(M,N)\right)^2$$  \hspace{1cm} (1)

where $f(M,N)$ represents the current frame and $f'(M,N)$ is the reconstructed frame with frame size as M x N.

This paper presents the review of search algorithms for various motion estimations in a video. We have analyzed the performance of the conventional FS, CDHS, CDS, UMHexagonS, CHS, OCTTS, OCTSS algorithms. The following metrics have been used to evaluate the effectiveness of these algorithms: PSNR, computational time, Probability of achieving true Motion Vector (MV), number of search points per macroblock, Distance, Average MSE per pixel.

II. SYSTEM ARCHITECTURE

The process of how the block matching algorithm works is shown in Fig. 1. The full process consists of 2 phases: Generation and Reconstruction. The function of Generation phase is to generate motion vectors for all blocks in the current frame using reference frame. The current frame is initially divided into macroblocks of size 8 x 8 in this paper. For each macroblock in the current frame, the best macroblock is identified using Block Matching Algorithm (BMA). From the Generation phase, MVs are generated. In the Reconstruction phase, the current frame is reconstructed using MV and the reference frame.

III. BLOCK MATCHING ALGORITHMS

A. Full Search Algorithm

The first algorithm developed for block-based motion estimation is full search algorithm [3], which searches all positions in the search area of the reference frame to find the best block. The FS performs well if the search range is correctly defined; it is guaranteed to determine the best matching position. As FS algorithm searches all positions in the search area, it takes very large computational time.

Fig. 2 Search Patterns in CDHS algorithm

B. Cross Diamond Hexagon Search Algorithm

The CDHS algorithm [4] uses cross-center-biased search with Small Cross-Shaped Pattern (SCSP) in the first step. In further steps, the search involves different patterns: Large Cross-Shaped Pattern (LCSP), Large Diamond Shaped Pattern (LDSP), Large Hexagon Search Pattern (LHSP) and Small Hexagon Search Pattern (SHSP). The searching pattern is shown in Fig. 2 and 3.

1. SPI (i) Starting: A minimum BDM point is found from the five checking points of SCSP at the center of search area. If the minimum BDM occurs at SCSP center, the search stops.

2. SPI (ii) Large Cross Searching: The four outermost points of the central LCSP are evaluated. This step guides another possible correct direction for the subsequent steps.

3. SPI (iii) Half-diamond Searching: Two additional points of the central LDSP closest to the current minimum BDM of the central LCSP are checked. If the minimum BDM found in previous steps is at any endpoint of SCSP, and the new minimum BDM found in this step still coincides with this point, the search stops.

4. SPI (iv) Searching:
   a. Case (1): If LDSP is used in the previous step and the minimum BDM is found to be located at any point on a diamond edge, a new LDSP is formed by repositioning the previous minimum BDM point as the center of LDSP.
Case (2): If LDSP is used in the previous step and the minimum BDM is found to be located at either of the horizontal or vertical diamond corners, a new horizontal or vertical LHSP is formed by repositioning the previous minimum BDM as the center of LHSP.

Case (3): Otherwise, a new LHSP of the same shape is formed by repositioning the previous minimum BDM as the center of LHSP.

For any case above (LDSP→LDSP, LDSP→LHSP, or LHSP→LHSP), three new check points are evaluated.

C. Cross Octagon Search Algorithm

The searching pattern of Cross Octagon Search algorithm [5] is shown in Fig 4. The steps involved in COSA are as follows:

- Initially, COSA uses cross pattern to find the motion direction. If minimum BDM is at the center point, the small diamond pattern is used. Otherwise two points most close to the minimum BDM point in the octagonal pattern are evaluated.

- Even if the minimum BDM point is found, the point (p) which is close to the minimum BDM point on the line passing through the center and the minimum BDM point are evaluated. If p is the minimum BDM point, then small diamond pattern is used to perform refinement search. Otherwise the point in the corresponding direction is evaluated until the minimum BDM point doesn’t change. The last step is to use the small diamond pattern to perform refinement search.
Fig. 4 Search patterns of COSA

D. UMHexagonS Algorithm

In this algorithm, lot of search patterns such as cross pattern, multi-hexagon-grid, 24 adjacent points, classical hexagon pattern and small diamond pattern are used. This algorithm combines the merits of TSS, NTSS, FSS, DS, HexagonS. UMHexagonS is depicted in Fig. 5. The algorithm steps are given below.

Start: Initial search point decision
Step 1: Unsymmetrical-cross search
Step 2: Small rectangular full search
Step 3: Uneven Multi-hexagon-grid Search
Step 4: Extended Hexagon-based Search, until the winning point is in the center of the hexagon

E. Cross Hexagon Search Algorithm (CHEX)

The CHEX search [7] consists of two patterns: cross-based and hexagon-based patterns. The search patterns are shown in Fig. 2 and 6.

The CHEXS algorithm is summarized as follows.
Step 1. A minimum BDM is found from the 5 search points of the SCSP located at the center of the search window. If the BDM point occurs at the center of the pattern, the search stops. Otherwise go to Step 2.
Step 2. A new SCSP is formed by using the vertex in the first SCSP as the center. If the BDM point occurs at the center of this pattern, the search stops. Otherwise go to Step 3.
Step 3. The three unchecked outermost search points of the LCSP and the two unchecked points near the center (radius = ±2) are checked. This step directs to the HEXS.
Step 4. A new Large Hexagon Search Pattern (LHEXSP) is formed by repositioning the minimum BDM found in the previous step as the center of LHEXSP. If the new BDM point is still at the center of the newly formed LHEXSP, then go to Step 5; otherwise repeat step 4.
Step 5. Switch the search pattern from the large size of hexagon to the small size of hexagon (SHEXSP). The four points covered by the small hexagon are evaluated to compare with the current BDM point. The new BDM point is the final solution of the motion vector.

F. Octagon and Triangle Search Patterns

The algorithm [8] classifies the blocks into two categories according to the block motion estimation mode and motion characteristics of the block. This algorithm uses octagon search pattern and triangle search pattern and the multi-octagon-grid search to enhance the search efficiency and to reduce the computational complexity. The OCTTS pattern is shown in Fig. 7.
A. Octagon and Square Search Algorithm

OCTSS uses 13 search points for large motion and 8 search points for small motion. The search patterns are shown in Fig. 8. The algorithm is as follows.

Step: 1 Calculate BDM for center point. If it is zero, go to Step 4.

Step: 2 Calculate BDM for all points (13 for octagon & 8 for square). Move the center to the point which has minimum BDM.

Step: 3 The search pattern strategy is square from this step. Calculate BDM and move to the point with minimum BDM. Repeat Step 3 until the center point has the minimum BDM.

Step: 4 Stop the search.

IV. EXPERIMENTAL RESULTS

The block matching algorithms are tested with slow, medium and large motion video sequences such as mobile, soccer and foreman sequence respectively. The quality of the reconstructed sequence should be estimated by subjective tests. One of the subjective metrics is MSE which is given in Eq. 1. The lesser the value of MSE, the better is the prediction quality.

Another widely used metric for comparing various image compression techniques is the peak-signal-to-noise-ratio (PSNR). The mathematical formulae for PSNR is

\[ PSNR = 10 \log_{10} \left( \frac{(2^b - 1)^2}{MSE} \right) \]  (2)

where \( b \) is the number of bits in a pixel. For 8-bit uniformly quantized video sequence \( b = 8 \). The higher the value of PSNR, the better is the quality of the compensated image.

In addition to Average PSNR per frame and Average MSE/pixel, Computational time (Speed), Number of search points per macroblock (Ns), Distance and Probability are also calculated. The computation time has been calculated on Intel core-2-duo processor (3 GHz). Distance is calculated as the average distance between MV of the appropriate method and the true MV. The probability of achieving the true MV, which is obtained using FS is also calculated. The results obtained for Soccer and Mobile video sequences are shown in Table I.

The Distance, Probability, Average MSE/pixel and Average PSNR measures obtained by the FS algorithm are used as the ground truth values. The objective of other algorithms is to reduce the computation time and the number of search points. The time taken by the OCTSS algorithm is very much lesser than other compared algorithms for all type of video sequences.

It shall be observed from Table I, that OCTSS algorithm takes 0.09ms whereas the traditional FS algorithm takes 0.9ms for the Soccer sequence. The number of search points is reduced in UMHexagonS algorithm for the Mobile sequence whereas in the Soccer sequence OCTSS algorithm has less number of search points. Distance and probability of getting true MV achieved by COS algorithm is the same as that of the FS algorithm. In terms of Average MSE/pixel and PSNR, OCTSS performs better than other algorithms.

Further, it shall also be observed that the COS algorithm performs well for slow motion video sequence. OCTSS algorithm consumes less time for processing while preserving the quality of the video sequence. The number of search points reduces the computational complexity. In all type of video sequences, OCTSS algorithm uses less number of search points per macroblock to find the matching block in the reference frame.

The number of search points used to find the matching block by different block matching algorithms for all types of video sequences is shown in Fig. 9.

The average PSNR obtained for foreman sequence (large motion) for the existing methods and the proposed method is shown in Fig. 10. From the Fig. 10, it is observed that the average PSNR obtained by the proposed method is little bit higher than the existing method but it does not achieve the PSNR obtained by the FS method.

<table>
<thead>
<tr>
<th>Method</th>
<th>Speed (ms)</th>
<th>Ns</th>
<th>Distance</th>
<th>Probability</th>
<th>Average MSE/pixel</th>
<th>Average PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Octagon search points</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Square Search points</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table I Results obtained for medium (Soccer) and slow (Mobile) Video Sequences
### Soccer Sequence

<table>
<thead>
<tr>
<th>Method</th>
<th>FS</th>
<th>CDHS</th>
<th>COS</th>
<th>UMHexagonS</th>
<th>CHS</th>
<th>OCTTS</th>
<th>OCTSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>0.9295</td>
<td>0.2356</td>
<td>0.2248</td>
<td>0.2581</td>
<td>0.2221</td>
<td>0.1589</td>
<td><strong>0.0947</strong></td>
</tr>
<tr>
<td>Number of Search Points</td>
<td>194.65</td>
<td>23.41</td>
<td>26.12</td>
<td>22.36</td>
<td>20.36</td>
<td>19.44</td>
<td><strong>18.09</strong></td>
</tr>
<tr>
<td>Cost</td>
<td>6.10</td>
<td>6.30</td>
<td>6.21</td>
<td>7.19</td>
<td>6.44</td>
<td>6.26</td>
<td><strong>6.21</strong></td>
</tr>
</tbody>
</table>

### Mobile Sequence

<table>
<thead>
<tr>
<th>Method</th>
<th>FS</th>
<th>CDHS</th>
<th>COS</th>
<th>UMHexagonS</th>
<th>CHS</th>
<th>OCTTS</th>
<th>OCTSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>0.2163</td>
<td>0.0202</td>
<td>0.0276</td>
<td>0.0183</td>
<td>0.0249</td>
<td>0.0269</td>
<td><strong>0.0158</strong></td>
</tr>
<tr>
<td>Number of Search Points</td>
<td>184.56</td>
<td>14.58</td>
<td>11.48</td>
<td><strong>4.96</strong></td>
<td>9.69</td>
<td>11.13</td>
<td>7.55</td>
</tr>
<tr>
<td>Cost</td>
<td>0.6854</td>
<td>0.6854</td>
<td>0.6854</td>
<td>0.6857</td>
<td>0.6927</td>
<td>0.69</td>
<td><strong>0.6854</strong></td>
</tr>
<tr>
<td>Index</td>
<td>40.6629</td>
<td>40.6629</td>
<td>40.6629</td>
<td>40.6589</td>
<td>40.5817</td>
<td>40.6087</td>
<td><strong>40.663</strong></td>
</tr>
</tbody>
</table>

---

**Fig. 9 Number of Search Points Comparative Analysis**

---

129
V. CONCLUSION

Block matching algorithms play a vital role in video compression, video summarization and so on. This paper analyzed several recent block matching algorithms and their performance is compared with the basic FS algorithm. Experimental results show that OCTSS performs well for all types of motion sequences, irrespective of the motion vectors of the neighboring blocks of the current block. It is also concluded that COS algorithm performs little better than the other block matching algorithms especially for slow motion videos.

REFERENCES
