A New Square-Diamond Search Algorithm for Fast Block Motion Estimation

Peng-Peng Jiao*, Xing Wei, Xia Lu
Nanjing Normal University Taizhou College, Taizhou 225300, Jiangsu, China

DOI: 10.23956/ijarcsse/SV7I5/0167

Abstract—Block motion estimation is a key technology of video compression coding. Considering the center-biased distribution characteristics and high space-time correlation of motion vectors in video sequences, a square-diamond search strategy based on a successive elimination algorithm (SEA-SDS) is designed. Then, a predictive starting point and adaptive square-diamond search algorithm based on successive elimination (PASEA-SDS) is proposed. The experimental results show that the algorithm can achieve almost the same results as the full search algorithm with very low computational cost, and it is better than the diamond and square-diamond fast motion estimation algorithms with respect to search times and accuracy.

Keywords—motion estimation, block match, square-diamond search, successive elimination algorithm

I. INTRODUCTION

The proportion of computation needed for motion estimation is the largest in the video coding process. For instance, the proportion of computation needed for motion estimation is 63% of the overall video compression computation in H.261 using the three-step search method. Motion estimation also takes up 42% of the computation in H.263. Therefore, improving the efficiency of motion estimation has become an important issue in video research [1-2].

Block matching motion estimation is easier to implement in software and hardware, hence it is used by many video compression coding standards. Classical block matching algorithms include the full search algorithm (FS), three-step search algorithm (TSS), new three-step search algorithm (NTSS) [3], four step search algorithm (FSS) [4], cross-search algorithm (CSA) [5-6], block-based gradient descent search algorithm (BBGDS) [7-8] among others. The FS algorithm gives the best results, but its computation is slow, restricting the applications of the algorithm; other fast search algorithms achieve higher speeds through different search templates and search modes, but their quality is lower than FS because they cause a “block effect” at different levels when they reduce the search points.

The search for a more efficient block matching motion estimation algorithm has attracted much attention by scholars, and the diamond search algorithm (DS) is one of the highest performing block matching algorithms [9-10]. The diamond search algorithm was adopted and used as test model by the MPEG-4 international standard in 1999. Later, the square-diamond search algorithm (SDS) was proposed as an extension of the DS algorithm.

Taking into account the cross-center-biased distribution characteristics and high space-time correlation of motion vectors in image sequences [11-12], a square-diamond search strategy based on the successive elimination algorithm [13-14] (SEA-SDS) is designed in this paper. In addition, by fusing starting point prediction and adaptive search techniques, a predictive adaptive square-diamond search algorithm using successive elimination (PASEA-SDS) is proposed. A comparison of this algorithm with the FS, TSS, DS, SDS, and other classic algorithms is given in the experiment section. The experimental results show that the algorithm further reduces the number of motion search below that of the SDS algorithm while its image quality is close to the FS algorithm.

II. METHODS

2.1. Static block detection

In general, the motion vectors of video sequences have center-biased distribution characteristics, namely, most of the motion vectors are the zero vector. Hence, to construct a threshold T, the vector (0, 0) is first detected before the search. When measuring the degree of block distortion, the sum of absolute differences (SAD) is used as the error criterion and is defined as follows:

\[ SAD(i, j) = \sum_{m=1}^{M} \sum_{n=1}^{N} |f_k(m, n) - f_{k-1}(m + i, n + j)| \]

Here, \((i, j)\) is the displacement vector, \(f_k\) and \(f_{k-1}\) are the grey values of the current and previous frames, respectively, and \(M \times N\) is the size of the macro block. If SAD \((0, 0) < T\), then the block is a static block, the optimal motion vector is \((0, 0)\), and the search is immediately suspended.
Because of the complexity of motion sequences, threshold T is not universal. In Fig.1, the current block has the highest correlation with the blocks left, above, and above right in the frame as well as the block at the same position in the reference frame. Its correlation to the other blocks is weaker. Hence, we can use the median SAD of the blocks left and above in the frame as the new threshold \( T \) and use it to determine if the macro block is a static block, namely

\[
T = \text{median}(SAD_1, SAD_2),
\]

where \( SAD_1 \) and \( SAD_2 \) are the SAD values of the blocks to the left and above, respectively. The corresponding threshold can change adaptively; hence, this threshold-setting method is more flexible than a fixed threshold method.

2.2. Determination of movement type

Assuming that the motion vectors of the current and four space-time adjacent blocks are \( V_0, V_1, V_2, V_3, \) and \( V_4 \), respectively, and that \( V_0 = (0,0) \) and \( V_i = (x_i, y_i), i = 1, 2, 3, 4 \), where \( x_i \) is the x direction component of the motion vector and \( y_i \) is the y direction component of the motion vector. The absolute value distance \( l_i \) of the motion vector is defined as follows:

\[
l_i = |x_i| + |y_i|
\]

For all candidate vectors of the motion vector set, we define

\[
L = \max\{l_0, l_1, l_2, l_3, l_4\},
\]

such that \( L \) reflects the amplitude of the macro block movement. The thresholds needed to determine and classify movement types are \( L_1 \) and \( L_2 \), and, assuming that \( L_1 \leq L_2 \), then:

1. When \( L \leq L_1 \), the current block is a small motion block, and we only perform a small step-size intensive search near the search start point.
2. When \( L_1 < L \leq L_2 \), the current block is a medium motion block, and we perform a large step-size search near the search start point.
3. When \( L > L_2 \), the current block is a large motion block, and we perform a small step-size search near the prediction search start point.

2.3. Predicting the search start point

In Figure 1, we assume that the motion vectors of the current and four space-time adjacent blocks are \( V_0, V_1, V_2, V_3, \) and \( V_4 \) respectively. First, the corresponding minimum block distortion (MBD) of the five motion vectors are calculated, and \( SAD_0, SAD_1, SAD_2, SAD_3, \) and \( SAD_4 \) are recorded. SAD is used to measure the MBD. The predictive motion vector \( V \) of the current search start point is then calculated as follows:

\[
V = \text{Arg}\{\min\{SAD_0, SAD_1, SAD_2, SAD_3, SAD_4\}\}
\]

Namely, the motion vector with the minimum SAD value is the predicted initial motion vector of the current block, and its location is the search start point.

2.4. SEA algorithm

The successive elimination algorithm (SEA) (Li and Safari, 1995) is an excellent lossless fast algorithm that effectively accelerates the search speed without loss of performance. SEA reduces the number of SAD calculations of the match point using a concise inequality, avoiding further calculation for points that will not become the best match point. A point \((i, j)\) that could be the best match point in SEA has the following prerequisite:

\[
\Delta S\text{um}0 - \text{MinSAD} \leq \Delta S\text{um}(i, j) \leq \Delta S\text{um}0 + \text{MinSAD}
\]

where \( \Delta S\text{um}0 \) is the sum of the gray scale absolute value of all coding block points, \( \text{MinSAD} \) is the minimum SAD of the searched points, and \( \Delta S\text{um}(i, j) \) is the sum of the gray scale absolute value at all points of the block that matches \((i, j)\).

Whether the matching calculation process of a point \((i, j)\) is terminated depends on whether the above formula is met. When considering a new match point, we first determine whether it meets the formula. If it does, \( \text{SAD}(i, j) \) is calculated,
otherwise the process skips this point. If SAD(i, j) is smaller than the current MinSAD, then MinSAD = SAD(i, j) and the corresponding vector is updated. The final search result is MinSAD and its corresponding motion vector.

A typical successive elimination algorithm is shown in Fig. 2.

![Flow chart of SEA](image)

**Fig. 2 Flow chart of SEA**

### 2.5. PASEA-SDS algorithm

#### 2.5.1. SEA-SDS search strategy

Two search templates, a large squares pattern (LSP) and small diamond pattern (SDP), were used in the algorithm; they are shown in Fig. 3.

![Search templates](image)

**Fig. 3 Search templates**

The algorithm process is as follows:

**Step 1:** The center of the current block is used as search starting point and the LSP template is used. First, the SAD of the LSP search area center is calculated, then the surrounding four points are matched using the SEA algorithm. If these points cannot meet the conditions of the SEA algorithm, or if at least one point meets the conditions and the MBD point is located in the center, then the algorithm goes to Step 3; otherwise it goes to Step 2.

**Step 2:** The MBD point is used as the center, and the new LSP and SEA algorithm are used to evaluate the next points. If the surrounding four points cannot meet the conditions of the SEA algorithm or the MBD point is located in the center, then the algorithm goes to Step 3; otherwise, it repeats Step 2.

**Step 3:** The MBD point is used as the center, LSP is replaced with SDP, and the five new points are evaluated using SEA. If the surrounding four points cannot meet the conditions of the SEA algorithm, then the location of the SDP center corresponds to the optimal motion vector and the search ends; otherwise it finds the MBD point and the algorithm goes to Step 4.

**Step 4:** The MBD point is used as the center, and the SDP and SEA are continuously used for evaluation. A new MBD point is determined and the location of this point corresponds to the optimal motion vector. The search then ends.

#### 2.5.2. PASEA-SDS algorithm

The basic SEA-SDS algorithm is further improved with starting point prediction and an adaptive search pattern, and is called the PASEA-SDS algorithm.

The steps of the algorithm are as follows:

**Step 1:** Static blocks are detected. The zero vector position (0,0) is determined and the SAD values of the point are denoted by SAD0. If SAD0 < T0, the block is static, the location of (0,0) is the optimal motion vector, and the algorithm ends; otherwise the algorithm goes to Step 2.

**Step 2:** Motion types are detected. The prediction vector L of the current block is determined according to the four time-space adjacent blocks, if L ≤ L1, the current block is a small motion block, and the algorithm goes to Step 4; if L1 < L ≤ L2, the current block is a medium motion block, and the algorithm goes to to Step 5; if L > L2, the current block is a large motion block, and the algorithm goes to Step 3.

**Step 3:** The starting point is predicted. The motion vector with the minimum error among the four time-space adjacent blocks is selected as the motion estimation starting point of the current block and the algorithm goes to Step 4.

**Step 4:** The SDP search is performed. The current MBD point is used as the center and the SDP is used for matching. If the MBD point is located in the center, then the current point is the optimal motion vector and the algorithm ends; otherwise the MBD point is used as the starting point, and Step 4 is repeated.

**Step 5:** The SEA-SDS search is performed. The current point is used as the search point origin, and the SEA-SDS search method is used to search. The optimal motion vector is then determined and the algorithm ends.
The flow chart of the PASEA-SDS algorithm is shown in Fig.4:

Fig.4 PASEA-SDS algorithm flow chart

**III. EXPERIMENTAL RESULTS AND ANALYSIS**

**3.1. Experimental data**

In these experiments, four representative CIF format video sequences (Claire, Football, Tennis, and Mobile) were selected and various representative motion estimation algorithms (FS, TSS, DS, and SDS) were used to compare with the new algorithm under the same conditions. The sub-block size used was 16 × 16, the size of the search window was ±15 pixels, and the matching criterion used SAD.

**3.2. Comparison of search speed and precision**

The pros and cons of a motion estimation algorithm depend on the time complexity of the search and the matching results. Generally, the former is measured by the average number of search points and the latter is measured by the average peak signal to noise ratio (PSNR) of the image. The values for these metrics are shown in Tables I and II, respectively.

<table>
<thead>
<tr>
<th>algorithm</th>
<th>Claire</th>
<th>Football</th>
<th>Tennis</th>
<th>Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>average search points</td>
<td>acceleration multiple</td>
<td>average search points</td>
<td>acceleration multiple</td>
</tr>
<tr>
<td>FS</td>
<td>225</td>
<td>1.00</td>
<td>225</td>
<td>1.00</td>
</tr>
<tr>
<td>TSS</td>
<td>25</td>
<td>9</td>
<td>25</td>
<td>9</td>
</tr>
<tr>
<td>DS</td>
<td>13.12</td>
<td>17.14</td>
<td>15.35</td>
<td>13.51</td>
</tr>
<tr>
<td>SDS</td>
<td>2.65</td>
<td>84.90</td>
<td>11.52</td>
<td>19.53</td>
</tr>
<tr>
<td>PASEA-SDS</td>
<td>1.90</td>
<td>118.42</td>
<td>8.83</td>
<td>25.48</td>
</tr>
</tbody>
</table>

Table I shows that the new algorithm can reduce the average number of search points needed to find the optimal vector for videos containing any motion type.
Table II: Comparison of the average PSNR

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Claire PSNR</th>
<th>Δ PSNR</th>
<th>Football PSNR</th>
<th>Δ PSNR</th>
<th>Tennis PSNR</th>
<th>Δ PSNR</th>
<th>Mobile PSNR</th>
<th>Δ PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>FS</td>
<td>42.00</td>
<td>0.00</td>
<td>22.48</td>
<td>0.00</td>
<td>29.49</td>
<td>0.00</td>
<td>22.72</td>
<td>0.00</td>
</tr>
<tr>
<td>TSS</td>
<td>41.72</td>
<td>-0.28</td>
<td>21.37</td>
<td>-1.11</td>
<td>27.37</td>
<td>-2.12</td>
<td>22.40</td>
<td>-0.32</td>
</tr>
<tr>
<td>DS</td>
<td>41.90</td>
<td>-0.10</td>
<td>21.67</td>
<td>-0.81</td>
<td>28.72</td>
<td>-0.77</td>
<td>22.64</td>
<td>-0.08</td>
</tr>
<tr>
<td>SDS</td>
<td>41.94</td>
<td>-0.06</td>
<td>21.86</td>
<td>-0.62</td>
<td>28.68</td>
<td>-0.81</td>
<td>22.67</td>
<td>-0.05</td>
</tr>
<tr>
<td>PASEA-SDS</td>
<td>41.98</td>
<td>-0.02</td>
<td>22.00</td>
<td>-0.48</td>
<td>28.89</td>
<td>-0.60</td>
<td>22.70</td>
<td>-0.02</td>
</tr>
</tbody>
</table>

Table II shows that the search precision of the new algorithm is higher than other fast algorithms for all sequences and slightly lower than the FS algorithm. Hence, the new algorithm can improve the search speed of motion estimation while guaranteeing search quality.

3.3. Subjective comparison

In order to compare the subjective effect of an image reconstructed by various algorithms, the standard video sequence “football” was selected for this experiment. The 16th frame is reconstructed using the 15th frame in the sequence, and the motion estimation of the new algorithm was compared using FS, TSS, DS, and SDS. The reconstructed images for five motion estimation algorithms are shown in Fig.5.

The images show that the figures and background have loss to different degrees in the reconstructed images using the TSS, DS, and SDS algorithms, while the reconstructed image of the algorithm in this paper is more complete and closer to the results of the FS algorithm. This indicates that the proposed algorithm is superior to other fast search algorithms with respect to subjective evaluation.

![Reconstructed images using different motion estimation algorithms](image_url)

**IV. CONCLUSION**

A predictive starting point and adaptive square-diamond search algorithm based on successive elimination is proposed in this paper. The algorithm adaptively selects different search strategies according to the motion type of the video sequence. For a large motion block, we perform a small step-size search after predicting the search start point; for a static block, we immediately end the search; for a medium motion block, we perform a SEA-SDS search; for a small motion block, we only perform a small step-size search. The experiments show that the algorithm can guarantee search quality and improve search speed, and has strong adaptability and robustness compared to typical fast estimation algorithms such as TSS, DS, and SDS.

**REFERENCES**


