AN EFFICIENT SEARCH ALGORITHM FOR BLOCK MOTION ESTIMATION

Jae-Yong Kim and Sung-Bong Yang
Department of Computer Science, Yonsei University
Seoul, 120-749 Korea
{kimjy,yang}@mythos.yonsei.ac.kr

ABSTRACT

Many suboptimal motion vector search algorithms have been proposed because the full search algorithm, which is an optimal method, requires huge computational cost. Most of these algorithms find motion vectors simply from the center of the search window. In this paper, we propose an efficient motion vector search algorithm which, in order to predict the initial search point, exploits the global motion information obtained from the previous three frames and the local motion information regarding the motion vectors of the neighboring blocks of the current block. Our proposed algorithm searches for a motion vector from this initial search point, instead of from the center of the search window, using either the diamond search algorithm [8,9] or the unrestricted small diamond search algorithm which performs its search always with a smaller diamond search pattern. Experimental results show that our proposed algorithm is faster than other suboptimal block matching algorithms while it maintains lower average block distortion.
I. INTRODUCTION

It is unbearable to use uncompressed video signals because of huge amount of information. The high correlation between temporally adjacent frames of a video sequence allows a reasonable compression of the signals. Motion estimation plays a critical role in compressing video sequences. Well-estimated motions result in small errors as well as high compression. Recent video compression standards such as ITU-T H.261[1], H.263[2], ISO/IEC MPEG-1[3], and MPEG-2[4] make use of block matching algorithms (BMAs) for motion estimation. Among the known BMAs, the most accurate is the full-search (FS) algorithm that compares every block in the search window for finding the optimally matched block. However, FS is not practical at all because of its immense computational requirement. Many suboptimal algorithms have been proposed to reduce the computational cost. Among them are the three-step search (TSS)[5], the new three-step search (NTSS)[6], the four-step search (4SS)[7], and the diamond search(DS)[8, 9].

TSS compares the eight points on the boundary of the 9 x 9 grid as well as the center point of the search window in the first step. Then it compares the eight points of the 5 x 5 grid and the eight points of the 3 x 3 grid from the minimum point of the previous step in the second and the third steps, respectively. Hence TSS always searches 25 points per block. It is more adaptable for images with large motions because it has a large search pattern in the first step. But the motions in typical image sequences are usually small and center-biased [6-11]. NTSS, 4SS, and DS have been proposed using this center-biased
characteristics of real-world image sequences. NTSS allows the search to terminate after the first or the second step. If the search stops after the first step, NTSS compares 17 points, otherwise it has to compare 33 points. But NTSS searches only 20 to 23 points in typical image sequences. Even though NTSS is faster than TSS for typical image sequences, the computational cost of NTSS is higher than that of TSS for the sequences with large motions. 4SS has been proposed to speed up both the worst and the average cases. 4SS searches the 5 x 5 grid in the first step. 4SS is more suitable for center-biased motions because it has a smaller search pattern and keeps the 5 x 5 grid up to the third step for large motions. The minimum and maximum numbers of search points of 4SS are 17 and 27, respectively. 4SS searches 6 points less than NTSS in the worst case while its performance is as good as NTSS.

DS algorithm reduces the average search points by searching a center-biased motion vector very quickly and a large motion vector unrestrictedly. One of the special features of DS is that its search pattern has a diamond shape. DS finds a center-biased motion vector faster than 4SS because the area of a 3 x 3 diamond shape is smaller than that of a 3 x 3 square. Therefore, the minimum number of search points for DS is only 13 while its maximum number is more than that of 4SS. In the simulations we performed, DS searches more than 40 points in the worst case. But its average number of search points is only 15 to 18 for typical image sequences. The motion vectors found by DS are, on the average, more accurate than those found by both 4SS and NTSS.

We defined the unrestricted small diamond search algorithm (USDS) as the method that performs
search always with a smaller diamond search pattern that is used in the last step in DS. Therefore, it is a very highly center-biased search method. The expansion of USDS occurs only at a vertex (a corner of the diamond shape). Three new search points are added for the next search. The maximum number of search points of USDS is less than 60, but the minimum number is only 5 in the simulations. This method works very well for the background or stall images. However, its search may stop easily at a local minimum because the size of its search pattern is very small. This prematurity in the search may occur frequently for image sequences with large motions. In our simulations, USDS shows a great performance in the search speed, but its accuracy is not good enough for typical image sequences. USDS is applicable only to video sequences with small motions such as the videophone or video-conference applications.

This paper proposes an efficient motion vector search algorithm, called the Global_Local Search (GLS) algorithm, using the global and local motion information with respect to the current block. GLS predicts the initial search point using either the global or the local motion information and then it proceeds its search from the predicted point using either DS or USDS. DS is chosen for its fast search speed and USDS for its smaller search pattern. In our simulations, more than 97% of the motion vectors in the “Tennis” test sequence are located within the 3 x 3 area around the predicted initial search point obtained by the global motion information (see Table 2). Our experimental results show that GLS is faster than other suboptimal BMAs while lower average block distortion is maintained.

The rest of this paper is organized as follows. The GLS algorithm is described in Section 2. The
The motion of an object in an image sequence is usually continuous and an object consists of several blocks. Thus, the motion vectors have temporal and spatial correlation between successive frames in an image sequence. The motion vector of a block is, therefore, highly related to those of the neighboring blocks even though it is a background. Figure 1 depicts the neighboring blocks of the current block. In the figure, MV denotes the motion vector that we have to find for the current block, and MV1 through MV4 indicate the motion vectors of the neighboring blocks that have already been found. Since MV4 is highly related to both MV1 and MV2, we do not consider MV4 in finding MV [10]. In [10] and [11], the average of MV1, MV2, and MV3 is used in predicting motion vectors. Table 1 shows the performance comparison for various ways of utilization of MVs. The median of MV1, MV2, and MV3 used in [2], shows the best performance. This median is defined as the local motion vector (LMV).

It is known that most real-world image sequences have a center-biased motion vector distribution. More than 70 to 80% of the motion vectors are normally stationary or quasi-stationary [6-11]. For example, as camera panning takes place, the highest peak in the motion vector distribution shifts according to the amount of the camera movement. When most of the objects in a frame move toward the
same direction, an effect like camera panning also takes place. The highest peak point in a frame is called

*global motion vector* (GMV). If the location of GMV is known and motion vectors in the previous frames
are the same as GMV, then we can find each motion vector quite easily. For example, if the majority of
motion vectors in the previous frames are the same as GMV, the motion vector in the current frame is
likely to be very near the initial search point obtained from GMV.

There are several ways to obtain GMV. One is to use the average value of motion vectors in the
previous frames. But in this paper, we look into the three previous frames to obtain GMV. If more than
1/3 of all the motion vectors in the three frames are the same, the motion vector is regarded as GMV;
otherwise, it is said that MV does not exist. In determining the existence of GMV, 1/3 is chosen as a
threshold through various experiments. When the value of the threshold is greater than 1/3, GMV hardly
exists. In this case, we could not take advantage of the global information at all. When the threshold is
smaller than 1/3, the motion vector searched from the initial search point obtained from GMV is likely to
be inaccurate. After a number of experiments, we decided that three previous frames with respect to the
current frame are enough to obtain GMV. Using less than three frames, GMV becomes so sensitive to a
scene change or a local movement that an accurate GMV cannot be found. Using more than three frames,
almost the same GMV was obtained as the one using three frames.

GLS predicts the initial search point using either GMV or LMV. Then GLS starts from this predicted
search point. Figure 2 shows two types of initial search points predicted from the center of the search
window. The search patterns in Figure 2 (a) and (b) are for DS and USDS, respectively. The GLS algorithm is described in detail as follows:

**The GLS Algorithm**

```
if more than 1/3 of all motion vectors in three previous frames are the same
    GMV ← the same motion vector;
else
    GMV ← NULL;
for each block in the current frame
    LMV ← median (MV1, MV2, MV3); /* where MV1, MV2, and MV3 are the motion vectors of the
    neighboring blocks of the current block as shown in Figure 1 */
    if GMV = NULL
        Perform DS from the initial search point predicted from LMV;
    else if GMV = LMV
        Perform USDS from the initial search point predicted from GMV;
    else
        Perform DS from the initial search point predicted from LMV;
```

If GMV is NULL, the motions of the objects in the current frame may be nonuniform or large. In this case, GLS performs DS, which has a larger search pattern than USDS, from the initial search point predicted from LMV. If GMV is different from LMV, the motion vector of the object involved with the current block differs from GMV. Therefore, GLS performs DS because the motion vector may be located far away from the initial search point. From Table 2, it can be observed that when GMV and LMV are different each other, the initial search point should be predicted from LMV. The table also shows that if GMV equals LMV, USDS should work well because more than 90% of motion vectors are located within the 3 x 3 window area of the predicted initial search point in all the test sequences.
III. SIMULATION RESULTS

For the simulations, the “Flower Garden,” the “Tennis,” and the “Football” test sequences have been used, where the first 150 frames have been chosen for each test sequence. These test sequences are chosen because they typically represent different characteristics. The “Flower Garden” has simple camera panning. The “Tennis” has zooming, panning, and scene changes. The “Football” includes large local motions and fast camera panning. The size of each frame in all the test sequences is 352 x 288 and each frame is quantized to 8 bits uniformly. The maximum displacement in a search region is 15 x 15 pixels in both horizontal and vertical directions for 16 x 16 block size. The performances of various BMAs have been evaluated, including GLS under the following three categories:

1. average number of search points per block to find the motion vector;
2. average sum of absolute difference (SAD) which is the absolute criterion of block distortion; and
3. average probability of finding the optimum motion vector by a BMA.

Table 3 presents the performance comparison of the aforementioned search algorithms using the three test sequences. The difference between the minimum (Min.) and the maximum (Max.) numbers of search points of DS (also GLS) is quite large because of its unrestricted search strategy. Even though the difference for GLS is slightly larger, the minimum number of search points for GLS is smaller than that of DS, since GLS starts its search from the initial search point predicted from the global or the local motion
information. The average (Avg.) number of search points of GLS is smaller than those of other BMAs for all three test sequences.

The average distance (Avg. dist.) and the average probability (Avg. prob.) in Table 3 show how the motion vectors found by the BMAs are close and similar to those obtained by FS. The average SAD provides the quality of the motion vector found by a BMA. GLS shows the best results in all three categories for the "Flower Garden" test sequence. It means that GLS finds the motion vectors with the highest accuracy especially for the frames that have camera panning or small motions. In the other two test sequences, GLS shows better speed than other BMAs while it maintains lower average block distortion.

Figure 3 shows the performance comparison of various BMAs on a frame-by-frame basis using the "Flower Garden" sequence. GLS performs quite well throughout the whole test sequence uniformly, regardless of various characteristics of frames in the sequence. Figure 4 (c)-(h) show the differences between the 16th frame of the "Flower Garden" sequence and the frames compensated only with the motion vector information obtained by FS, TSS, NTSS, 4SS, DS, and GLS, respectively. A perfectly well estimated frame should show virtually nothing in the picture. The difference made by FS in Figure 4 (c) gives the best result among the BMAs. The difference made by GLS in Figure 4 (h) is much closer to that by FS than those by other suboptimal BMAs. Figure 4 (h) shows that GLS can find motion vectors well for small motions just as 4SS or DS (e.g., the roof of the house) as well as for large motions like NTSS.
(e.g., the small trees on the middle-left portion of the scene).

**IV. CONCLUSION**

In this paper, we have proposed an efficient motion vector search algorithm using the global and local motion information with respect to current block. We exploited the fact that the motion vector of a block is highly related to those of its neighboring blocks and the motion vectors of most image sequences are centrally biased. We predicted the initial search point using either the global or the local motion information. When a motion vector is searched from this predicted initial search point, the chances that the optimal motion vector is located at the initial search point or within the 3 x 3 window of the initial search point are very high, if GMV and LMV are the same. In this case, GLS employs USDS for its search. If not, GLS utilizes DS for the search. Our simulations have shown that GLS finds motion vectors faster than other BMAs while lower average block distortion is maintained. This enhancement comes from the fact that GLS predicts the initial search point according to the global or the local motion information and is based on a hybrid method using DS and USDS.

**V. ACKNOWLEDGEMENTS**

We are grateful to two anonymous reviewers of the earlier version of this paper whose incisive comments helped improve the presentation.
REFERENCES


Table 1. Performance comparison for various ways of utilization of the motion vectors of the neighboring blocks

<table>
<thead>
<tr>
<th>Utilization of the motion vectors of the neighboring blocks</th>
<th>Probability that the obtained motion vector is located within the 3x3 area</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tennis</td>
</tr>
<tr>
<td>MV1</td>
<td>0.7357</td>
</tr>
<tr>
<td>MV2</td>
<td>0.7684</td>
</tr>
<tr>
<td>MV3</td>
<td>0.7374</td>
</tr>
<tr>
<td>MV4</td>
<td>0.7447</td>
</tr>
<tr>
<td>Average (MV1, MV2)</td>
<td>0.7478</td>
</tr>
<tr>
<td>Average (MV1, MV2, MV4)</td>
<td>0.7366</td>
</tr>
<tr>
<td>Average (MV1, MV2, MV3)</td>
<td>0.7447</td>
</tr>
<tr>
<td>Average (MV1, MV2, MV3, MV4)</td>
<td>0.7470</td>
</tr>
<tr>
<td>Median (MV1, MV2, MV3)</td>
<td><strong>0.7765</strong></td>
</tr>
</tbody>
</table>
Table 2. Accuracy comparison of various methods using global and local motion information

<table>
<thead>
<tr>
<th>Methods for predicting the initial search point</th>
<th>Probability that MV exists within the 3x3 area of the predicted point</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMV = LMV using GMV</td>
<td>“Flower Garden”</td>
</tr>
<tr>
<td></td>
<td>0.9738</td>
</tr>
<tr>
<td>GMV ≠ LMV using GMV</td>
<td>0.7314</td>
</tr>
<tr>
<td>GMV ≠ LMV using LMV</td>
<td>0.8804</td>
</tr>
<tr>
<td>GMV ≠ LMV using average (GMV, LMV)</td>
<td>0.8002</td>
</tr>
</tbody>
</table>

Table 3. Performance comparison of FS, TSS, NTSS, 4SS, DS, and GLS using the three test sequences

<table>
<thead>
<tr>
<th>BMAs</th>
<th>The “Flower Garden” sequence</th>
<th>The “Football” sequence</th>
<th>The “Tennis” sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>FS</td>
<td>225 225 225 1.00</td>
<td>2368.14</td>
<td></td>
</tr>
<tr>
<td>TSS</td>
<td>25 25 25 9.00</td>
<td>2581.02</td>
<td>0.8682</td>
</tr>
<tr>
<td>NTSS</td>
<td>17 33 22.15 10.15</td>
<td>0.204</td>
<td>2403.57</td>
</tr>
<tr>
<td>4SS</td>
<td>17 27 19.32 11.64</td>
<td>0.342</td>
<td>2492.19</td>
</tr>
<tr>
<td>DS</td>
<td>13 41 17.05 13.19</td>
<td>0.221</td>
<td>2429.47</td>
</tr>
<tr>
<td>GLS</td>
<td>5 42 11.97 18.79</td>
<td>0.137</td>
<td>2387.42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BMAs</th>
<th>The “Tennis” sequence</th>
<th>The “Tennis” sequence</th>
<th>The “Tennis” sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>FS</td>
<td>225 225 225 1.00</td>
<td>1283.45</td>
<td>0.8548</td>
</tr>
<tr>
<td>TSS</td>
<td>25 25 25 9.00</td>
<td>1371.57</td>
<td>0.8035</td>
</tr>
<tr>
<td>NTSS</td>
<td>17 33 19.96 11.27</td>
<td>0.578</td>
<td>1283.45</td>
</tr>
<tr>
<td>4SS</td>
<td>17 27 18.48 12.17</td>
<td>0.651</td>
<td>1279.21</td>
</tr>
<tr>
<td>DS</td>
<td>13 46 15.53 14.48</td>
<td>0.471</td>
<td>1245.17</td>
</tr>
<tr>
<td>GLS</td>
<td>5 51 11.06 20.34</td>
<td>0.485</td>
<td>1261.66</td>
</tr>
</tbody>
</table>
Figure 1. The motion vectors of the current block and its neighboring blocks

Figure 2. Two types of predicted initial search points

Figure 3. Performance comparison of FS, TSS, NTSS, 4SS, DS, and GLS for the three criteria using the “Flower Garden” sequence

Figure 4. Differences made by FS, TSS, NTSS, 4SS, DS, and GLS for the 16th frame of the “Flower Garden” sequence
Figure 1. The motion vectors of the current block and its neighboring blocks
Figure 2. Two types of predicted initial search points
Figure 3. Performance comparison of FS, TSS, NTSS, 4SS, DS, and GLS for the three criteria using the “Flower Garden” sequence.
Figure 4. Differences made by FS, TSS, NTSS, 4SS, DS, and GLS for the 16th frame of the “Flower Garden” sequence