A Block-Based Gradient Descent Search Algorithm for Block Motion Estimation in Video Coding

Lurng-Kuo Liu and Ephraim Feig

Abstract—A block-based gradient descent search (BBGDS) algorithm is proposed in this paper to perform block motion estimation in video coding. The BBGDS evaluates the values of a given objective function starting from a small centralized checking block. The minimum within the checking block is found, and the gradient descent direction where the minimum is expected to lie is used to determine the search direction and the position of the next checking block. The BBGDS is compared with full search (FS), three-step search (TSS), one-at-a-time search (OTS), and new three-step search (NTSS). Experimental results show that the proposed technique provides competitive performance with reduced computational complexity.

I. INTRODUCTION

The high correlation between successive frames of a video sequence makes it possible to achieve high coding efficiency in a video coding system by reducing the temporal redundancy. Motion compensated video coding technique, which predicts current frame from previous frame (or reference frame), has been used to exploit the temporal redundancy between successive frames. Motion estimation plays an important role in such an interframe predictive coding system. Among many types of motion estimation algorithms, block-matching technique has been adopted in many video compression standards, such as MPEG 1, 2, H.261, and H.263, due to its simplicity. In block-matching technique, frames are divided into blocks and one motion vector is associated with each block. For each block in the current frame, the motion estimation searches for a motion vector which points to the best match block in the reference frame. The best match block is then used as the predictor for the current block.

The full search (FS) block-matching algorithm is the simplest, but computationally very intensive. It provides an optimal solution by exhaustively evaluating all the possible candidates within the search range in the reference frame. Several fast algorithms, such as the three-step search (TSS) [5], the 2-D logarithmic search (LOGS) [6], one-at-a-time search (OTS) [7], and the new three-step search (NTSST) [8] have been developed to reduce the computational complexity by reducing the number of checking points. Of these, the first three can be easily trapped into a local minimum, thereby degrading performance; see, for example [9]. The NTSS takes into account the fact that the distribution of global minimum in real world video sequences is centered at zero. This is especially true for head-and-shoulder sequences typical in video conferencing. The NTSS adds checking points which are centered about zero to the first step of the TSS. But usually the search stops after the first step, and then the second most likely scenario has the search ending after a second step which searches close to the first. So, on average, NTSS is somewhat faster than TSS. More significantly, for head-and-shoulder sequences, NTSS yields significant SNR gains over TSS.

In the present work we push the idea of utilizing the statistical nature of the motion even further. We further assume that the global minimum has a monotonic distortion in its neighborhood. In our first step we only search around the center point. If the optimum is found at the center, the procedure stops. This will be more than 80% of the time. Otherwise, we proceed to search around the point where the minimum was found. The procedure continues until the winning
TABLE I
THE PERFORMANCE COMPARISON OF THE ALGORITHMS WITH SEARCH RANGE ±7 PIXELS IN BOTH HORIZONTAL AND VERTICAL DIRECTIONS

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Foreman</th>
<th>Salesman</th>
<th>Miss America</th>
<th>Car Phone</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>Complexity</td>
<td>MSE</td>
<td>Complexity</td>
<td>MSE</td>
</tr>
<tr>
<td>FS</td>
<td>26.82</td>
<td>100%</td>
<td>6.52</td>
<td>100%</td>
<td>4.99</td>
</tr>
<tr>
<td>BBGDS</td>
<td>28.80</td>
<td>5.40%</td>
<td>6.66</td>
<td>4.22%</td>
<td>5.05</td>
</tr>
<tr>
<td>TSS</td>
<td>34.23</td>
<td>11.43%</td>
<td>6.97</td>
<td>11.36%</td>
<td>5.50</td>
</tr>
<tr>
<td>OTS</td>
<td>36.62</td>
<td>3.75%</td>
<td>7.06</td>
<td>2.96%</td>
<td>5.20</td>
</tr>
<tr>
<td>NTSS</td>
<td>28.63</td>
<td>8.93%</td>
<td>6.62</td>
<td>7.83%</td>
<td>5.04</td>
</tr>
</tbody>
</table>

TABLE II
THE PERFORMANCE COMPARISON OF THE ALGORITHMS WITH SEARCH RANGE ±15 PIXELS IN BOTH HORIZONTAL AND VERTICAL DIRECTIONS

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Foreman</th>
<th>Salesman</th>
<th>Miss America</th>
<th>Car Phone</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MSE</td>
<td>Complexity</td>
<td>MSE</td>
<td>Complexity</td>
<td>MSE</td>
</tr>
<tr>
<td>FS</td>
<td>25.78</td>
<td>100%</td>
<td>6.51</td>
<td>100%</td>
<td>4.97</td>
</tr>
<tr>
<td>BBGDS</td>
<td>28.79</td>
<td>1.33%</td>
<td>6.66</td>
<td>1.04%</td>
<td>5.05</td>
</tr>
<tr>
<td>TSS</td>
<td>37.25</td>
<td>6.98%</td>
<td>7.33</td>
<td>6.96%</td>
<td>5.61</td>
</tr>
<tr>
<td>OTS</td>
<td>36.67</td>
<td>0.93%</td>
<td>7.08</td>
<td>0.73%</td>
<td>5.20</td>
</tr>
<tr>
<td>NTSS</td>
<td>28.23</td>
<td>6.39%</td>
<td>6.63</td>
<td>6.09%</td>
<td>5.02</td>
</tr>
</tbody>
</table>

Fig. 2. Performance comparison on Foreman sequence with search ranges (a) ±7 pixels and (b) ±15 pixels in both horizontal and vertical directions.

point is a center point of the checking block or the checking block hits the boundary of the predefined search range. Hence, we call our method the block based gradient descent search (BBGDS) algorithm. Of course, various modifications suggest themselves, like stopping after $K$ number of searches (say $K = 3$), or after the improvement in the objective function is below some fixed threshold. It is also clear that the larger size of checking block can reduce the chance of being trapped in a local minimum.

The details of the proposed algorithm are described in Section II. Experimental results are given in Section III. Finally, conclusions are made in Section IV.

II. BLOCK-BASED GRADIENT DESCENT SEARCH ALGORITHM

The search procedure of the BBGDS algorithm is illustrated in Fig. 1. Checking blocks are squares of 3 × 3 pixels. The BBGDS starts by initializing the checking block so that its center pixel is at the origin.
1) Evaluate the objective function for all nine points in the checking block.
2) If the minimum occurs at the center, stop; the motion vector points to the center. Otherwise, reset the checking block so that its center is the winning pixel, and go to step 1).

Note that except for the first iteration, most of the pixels in the checking block have already been checked in a previous pass. This is highlighted in Fig 1(a), where the second pass is centered either around corner pixel (2) or edge pixel (3). In the first case, all eight pixels must be visited; in the second case, only three.

The search procedure of the BBGDS always moves the search in the direction of optimal gradient descent. This is the direction where one expects the objective function to approach its minimum. The procedure is illustrated in Fig. 1, where the motion vector $(-2, -4)$ is found.

III. EXPERIMENTAL RESULTS

Four QCIF video sequences, "Foreman," "Salesman," "Miss America," and "Car Phone" are used in our simulations. The block size is fixed at $8 \times 8$. Two different search ranges $\pm 7$ and $\pm 15$ pixels, in both horizontal and vertical directions, are used in our simulations. We use the mean absolute difference (MAD) as the objective function. For a given displacement $(x, y)$, the MAD between block$(m, n)$ of current frame and block$(m+x, n+y)$ of reference frame is defined as

$$MAD_{(m,n)}(x,y) = \frac{1}{64} \sum_{i=0}^{7} \sum_{j=0}^{7} |f_k(m+i,n+j) - f_{k-1}(m+x+i,n+y+j)|$$

where $f_k(i,j)$ is the pixel intensity at position $(i,j)$ of frame $k$, and the block$(m, n)$ is the block with its upper left corner at position $(m, n)$ of a frame.

The first 100 frames of the video sequence are used in our simulations. We use the mean square error per pixel as the measure of performance. The required number of search points of each
block is used as the measure of computational complexity. Each video sequence is processed by five algorithms: full search (FS), three-step search (TSS), one-at-a-time search (OTS), and the proposed block-based gradient descent search (BBGDS). The degree of computational complexity of each algorithm with respect to full search algorithm and MSE is calculated. The simulations show that both NTSS and BBGDS outperform TSS and OTS. The distortions of BBGDS are very slightly greater than those of NTSS, but the computational complexity of BBGDS is significantly less than that of NTSS.

IV. CONCLUSION

In this paper, the BBGDS algorithm is proposed to perform block motion estimation in video coding. Based on the observation that global minimum distribution is centralized in real world video sequence, the BBGDS takes this advantage by using a center-biased checking block in its initial search step. The BBGDS also employs the concept of checking block instead of checking point in each of its search steps. The BBGDS searches for the motion vector along the block-based gradient descent direction where the minimum is expected to lie. The BBGDS has reduced susceptibility to the local minimum because of the use of the checking block. Experimental results show that the proposed technique provides competitive performance with reduced computational complexity.

REFERENCES


A Note on “Block Wavelet Transforms for Image Coding”

A. Sharaf and P. Marvasti

Abstract—We note that the arrangement of the rows of the block wavelet transform (BWT) matrix as given by (3) in the above paper is in an increasing order of frequency as implied by the equations and figures in the same paper. The rows of the matrix are rearranged to follow an increasing order of frequency.

I. THE REARRANGEMENT AND ITS JUSTIFICATION

The arrangement of the rows of the block wavelet transform (BWT) matrix corresponding to the Daubechies’ 8-tap filter as given by (3) in the above paper is not in an increasing order of frequency as implied by (1) and Fig. 1 of the same paper. Equation (1) states implicitly that the transform domain coefficients are to be ordered

Manuscript received October 17, 1995; revised April 23, 1996. This paper was recommended by Associate Editor J. Brailean. This work was supported by Signals and Software (Ltd) and the EPSRC. The authors are with the Department of Electrical and Electronic Engineering, King’s College London, Strand, London, WC2R 2LS U.K.

Publisher Item Identifier S 1051-8215(96)08459-6.