A FAST TWO-STEP SEARCH ALGORITHM FOR HALF-PIXEL MOTION ESTIMATION

Bo Zhou and Jian Chen

Dept. of Electronic Engineering, Shanghai Jiaotong University, Shanghai 200030, P.R. China

ABSTRACT

With the extensive application of integer-pixel fast search algorithms, the half-pixel full search consumes a significant portion of the entire motion estimation time. To reduce its computational complexity, a Two-Step Search (2SS) algorithm for half-pixel motion estimation is proposed in this paper. The 2SS algorithm is based on the fact that the SAD distortion function increases monotonically as the search location moves away from the sub-pixel minimum within the ±1 pixel neighborhood of integer-pixel search result. Simulation results show that the proposed algorithm is simple and efficient. It requires less than one half of the computation compared with that for the full half-pixel search while maintaining almost the same image quality and bit rate.

1. INTRODUCTION

Motion estimation plays an important role in today's video coding and processing systems, because motion vectors are critical information for temporal redundancy reduction. Due to its simplicity and the coding efficiency of motion vectors, the block matching algorithm (BMA) is perhaps the most successful technique for motion estimation, which is vital to many motion-compensated video-coding techniques/standards, such as ISO MPEG-1/2/4 and ITU-T H.261/263 [1]. In BMA, all pixels in a block are assumed to have the same motion. The motion vector is obtained by searching for the minimum point on an error surface determined by the block-distortion measure (BDM), such as the mean squared error (MSE) or the sum of absolute differences (SAD), over candidate motion vectors within the search window.

The full search algorithm (FSA), which exhaustively searches for the best matched block within the search window, can get the optimal solution. FSA, however, requires high computational complexity, making it impractical for a real-time video encoder. Various fast, but sub-optimal, algorithms that reduce the complexity have been developed, such as the conjugate directional search (CDS) [2], [3], the three-step search (TSS) [4], the new three-step search (NTSS) [5], the four-step search (FSS) [6], etc.

It is obviously true that integer-pixel search result cannot represent the actual motion vector, since the frame-to-frame displacements of image contents are completely unrelated to the sampling grid. Therefore a sub-pixel search can be performed to compensate for the error of integer-pixel motion estimation. It can efficiently reduce error between the original frame and the predicted frame. For this reason, half-pixel motion estimation is adopted as standard in H.263 and MPEG-4. However, most of the existing fast algorithms focus on integer-pixel motion estimation and are not applicable to half-pixel search. In order to get the final half-pixel motion vector, an encoder typically performs full search over all candidate half-pixel positions, i.e., searches all 8 interpolated half-pixels around the integer-pixel search result. With the improvement of fast alforithms for integer-pixel, classic ones are able to reduce the number of searched pixels per block to less than 20, with the search window of 15×15. Some novel algorithms can decrease the number further to around 10 [7]. Since the computational complexity of integer-pixel motion estimation has been continually decreasing, the computational complexity of the half-pixel full search, which is made up of the bilinear interpolation and the calculation of the BDM's of 8 half-pixels, has taken a relatively large part of the whole motion estimation [8].

In this paper, by observing the characteristic of error surface within the ±1 pixel neighborhood of the integer-pixel search result, we find that the SAD distortion function increases monotonically as the search location moves away from the sub-pixel minimum within the neighborhood. Then, we propose a new Two-Step Search (2SS) algorithm for half-pixel motion estimation.

Simulation results show that the proposed algorithm is fast and efficient which can significantly reduce more than one half of the computation compared with that for the half-pixel full search while maintaining almost the same image quality and bit rate.

2. FULL SEARCH ALGORITHM FOR HALF-PIXEL MOTION ESTIMATION

In motion compensation, the current block, to be coded, is replaced by one block of the reference frame. Therefore, the predicted block is:

$$\hat{B}_k(x,y) = B_{k-1}(x+dx, y+dy) \qquad (1)$$

The horizontal displacement dx and the vertical displacement dy are defined as motion vector. The

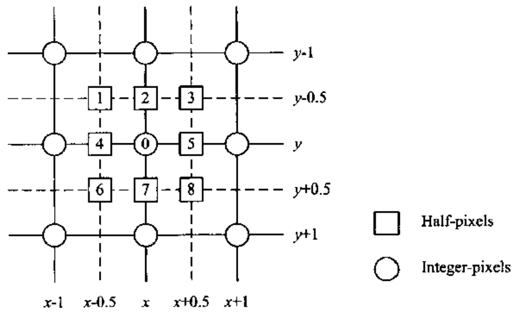

Figure 1. The positions of half-pixels and integer-pixels.

motion vector in the case of half-pixel accuracy can be written as follows:

$$
\begin{aligned}
dx &= dx_i + dx_h \\
dy &= dy_i + dy_h
\end{aligned} \tag{2}
$$

where dx_i and dy_i are integer-pixel components, and dx_h and dy_h are half-pixel components. Half-pixel search is performed in the frame interpolated from its original integer-pixel counterpart. The positions of half-pixels and integer-pixels are shown by Figure 1. For convenience, the integer-pixel and the half-pixels around are numbered in Figure 1. The intensities of the half-pixels are given by

$$
\begin{aligned}
I(x+0.5,y) &= [I(x,y)+I(x+1,y)+1]/2 \\
I(x,y+0.5) &= [I(x,y)+I(x,y+1)+1]/2 \\
I(x+0.5,y+0.5) &= [I(x,y)+I(x+1,y) \\
&\quad +I(x,y+1)+I(x+1,y+1)+2]/4
\end{aligned} \tag{3}
$$

where x and y are the horizontal and the vertical coordinates of integer-pixel, respectively, and $I(\bullet)$ is the intensity of pixels. If the center integer-pixel is the integer-pixel component of motion vector obtained by integer-pixel search, the integer-pixel and the 8 interpolated half-pixels around will be searched in the following half-pixel search so that the half-pixel component of motion vector will be obtained.

Among many BDM's available for motion estimation, SAD is usually preferred for real-time application because it doesn't need any multiplication in implementation and it is not as expensive in computation cost as other BDM's. The SAD of candidate motion vector (i,j) can be expressed as:

$$
SAD(i,j) = \sum_{m=x}^{x+N-1} \sum_{n=y}^{y+N-1} \left| I_k(m,n) - I_{k-1}(m+i,n+j) \right| \tag{4}
$$

where $I_k(m,n)$ denotes the intensity of the (m,n) th pixel in the k th frame, and (x,y) is the location of the current block whose size is $N \times N$.

3. THE CHARACTERISTIC OF ERROR SURFACE WITHIN THE ±1 PIXEL NEIGHBORHOOD OF INTEGER-PIXEL SEARCH RESULT

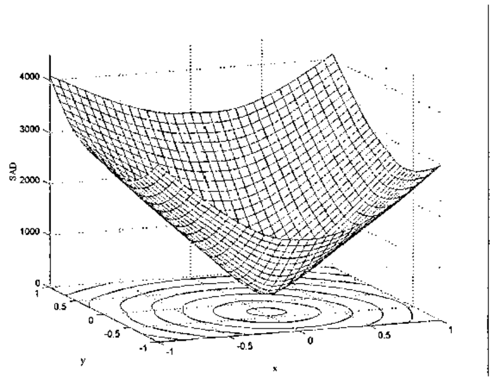

Figure 2. A typical error surface.

Table 1. Statistics of SAD error surface within the ±1 pixel neighborhood of integer-pixel search result.

Sequence	Number of frames	Number of blocks	Number of blocks single valley	Ratio
Claire	494	48 807	47 140	96.58%
News	300	29 601	28 908	97.66%
Salesman	449	44 352	44 106	99.45%
Trevor	150	14 751	14 199	96.26%

Although the integer-pixel result obtained from the fast search algorithm is not always the global minimum in the search window, it is at least a local minimum. Therefore, we can show that SAD (the integer-pixel search result) $\le SAD(i)$, where $i \in$ {the 8 integer-pixels around the integer-pixel search result}.

The intensities of sub-pixels within the ±1 pixel neighborhood of integer-pixel search result are obtained by bilinear interpolation which is given by

$$
\begin{aligned}
I(x_s,y_s) &= I(x,y)(x+1-x_s)(y+1-y_s) \\
&\quad +I(x+1,y)(x_s-x)(y+1-y_s) \\
&\quad +I(x,y+1)(x+1-x_s)(y_s-y) \\
&\quad +I(x+1,y+1)(x_s-x)(y_s-y)
\end{aligned} \tag{5}
$$

where $I(x,y)$ denotes the intensity of the (x,y) th integer-pixel, and (x_s,y_s) is the location of sub-pixel.

SAD error surface is created by computing SAD of all interpolated sub-pixels within the ±1 pixel neighborhood. Figure 2 shows a typical example of SAD error surface, which is derived from the $(5,6)$ th 16×16 block in the 50th frame of Claire sequence in QCIF format. We can observe that SAD error surface within the neighborhood shows a shape of single valley because of the center integer-pixel's characteristic of local minimum. Table 1 gives some sequences' statistics of SAD error surface within the ±1 pixel neighborhood of the integer-pixel search result of all 16×16 blocks, where FSA is used to obtain the integer-pixel component of motion vector and the number of interpolated sub-pixels between integer-pixels is 7. From this table, we can find that for ordinary sequences, especially for head-shoulder ones, the percentage of blocks whose SAD error surface within the ±1 pixel neighborhood is in the shape of single valley is significantly high. For blocks which are not in the shape of single valley, it is observed that they are often without remarkable features, such as monochromatic background, and the corresponding SAD error surface is so flat (the fluctuation is caused mostly by noise) that the variance of SAD is small.

Based on the characteristic of single valley, we can draw the conclusion that the SAD distortion function increases monotonically as the search location moves away from the sub-pixel minimum within the ±1 pixel neighborhood of integer-pixel search result or the SAD error surface is unimodal within the ±1 pixel neighborhood. That's to say, the smaller the SAD distortion function of a sub-pixel, the closer to the minimum within the neighborhood. Almost all existing fast search algorithms for integer-pixel motion estimation rely on this assumption, but it is usually not true over the integer-pixel search window due to such reasons as the aperture problem. However, it is nearly always valid within the ±1 pixel neighborhood.

If we substitute the MSE distortion function as BDM, we will come to a similar conclusion. The only difference is that the gradient of its error surface is not as large as that of SAD's.

4. THE PROPOSED 2SS ALGORITHM

Based on the conclusion of Section 3, 2SS is the algorithm that sequently searches the minimum SAD's within the neighborhood on the horizontal and vertical directions so as to find the half-pixel component of motion vector. It can be summarized in the following detailed steps:

Step 1) Using the SAD distortion function as BDM, the integer-pixel search result, which is the corresponding pixel of the integer-pixel component of motion vector, is taken as the initial search center pixel (x_1, y_1). During the horizontal search, the SAD's of the pixels $(x_1 - 0.5, y_1)$, (x_1, y_1), and $(x_1 + 0.5, y_1)$ are computed and compared with each other, and the pixel with the minimum SAD is taken as the vertical search center (x_2, y_2).

Step 2) While the vertical search is done, the SAD's of the pixels $(x_2, y_2 - 0.5)$, (x_2, y_2), and $(x_2, y_2 + 0.5)$ are computed and compared with each other, and the pixel with the minimum SAD is taken as the half-pixel search result so that the half-pixel component of motion vector is obtained.

Figure 3 shows a typical path of 2SS. The SAD of the integer-pixel search result is known and the SAD of the vertical search center has been computed during the preceding horizontal search, so four half-pixels' SAD's need to be computed in most cases. If the integer-pixel search result is located at the top or bottom of a frame,

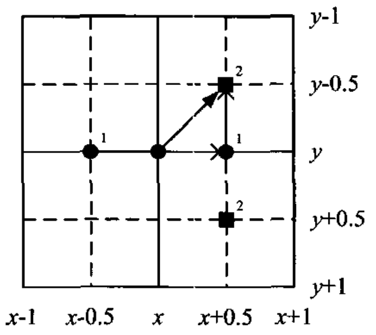

Figure 3. The 2SS process. The half-pixel component of motion vector is (0.5, -0.5) in this example.

there is only one half-pixel's SAD to be computed during the vertical search, and there are three SAD's totally. If the integer-pixel search result is located on the left side or right side of a frame, to ensure precision, we first perform the vertical search rather than the horizontal one. As a result, there are three half-pixels' SAD's to be computed as well. If the integer-pixel search result is located on one of the four corners of a frame, there are only two half-pixels' SAD's to be computed. In a word, the proposed algorithm can reduce the number of half-pixels searched from eight of the full search algorithm to no more than four, which decreases more than one half of the computational complexity.

At the same time, the method can be improved that the entire reference frame is interpolated before half-pixel search. Instead, we only interpolate the necessary half-pixels whenever computing SAD. According to the numbers in Figure 1, because most half-pixels of the block pairs presented by 4 and 5, 1 and 6, 2 and 7, and 3 and 8 are overlapped each other, the outcome of interpolation can be reused to further decrease the computational complexity.

Furthermore, the partial distortion elimination (PDE) algorithm is applied. The computation of current SAD ceases as soon as the accumulation exceeds the minimum computed SAD. To make compromise with computational speed, the accumulation is decided after completing a group of pixels, e.g., one row of a block, instead of every pixel.

5. SIMULATION RESULTS AND DISCUSSION

In our simulation, a codec compatible with H.263 is applied to encode some standard sequences in QCIF format (176×144 pixels/frame), which are Claire, News, Salesman, and Silent. The BDM is defined to be the sum of absolute differences (SAD).

The maximum displacement in the integer-pixel search area is ±7 pixels in both the horizontal and the vertical directions for 16×16 block size. Input/output frame rate is 30 fps/30 fps. Quantization parameter is 10. For simplicity of discussion, the constant bias for null motion vector is set to 0. We used 100 frames of each

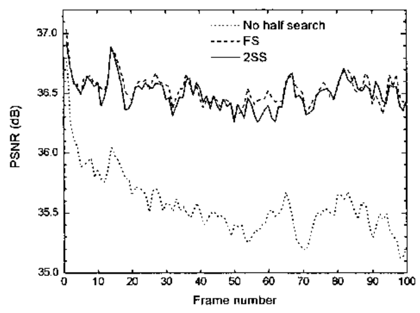

Figure 4. Performance in terms of PSNR of FS, 2SS, and the algorithm with no half-pixel search for Claire.

TABLE 2. Performance comparison between 2SS and FS in terms of average PSNR and bit stream length.

Sequence	$\overline{PSNR}$ (dB)		L (Bytes)	
	FS	2SS	FS	2SS
Claire	36.5346	36.4993	16 415	16 402
News	32.6262	32.6284	26 705	26 808
Salesman	31.7910	31.7917	17 051	17 204
Trevor	32.9476	32.9325	41 843	41 916

TABLE 3. Average number of searched half-pixels of 2SS.

Sequence	Average number of searched half-pixels
Claire	3.465
News	3.453
Salesman	3.455
Trevor	3.456

sequence to test 2SS algorithm and compare it with the half-pixel full search (FS) algorithm and the algorithm with no half-pixel search. FSA is employed as the integer-pixel search technique in all these experiments.

Figure 4 compares the performance of different search algorithms in terms of PSNR between reconstructed frames and original frames of Claire. The average Peak Signal-to-Noise Ratio ($\overline{PSNR}$) and bit stream length (L) comparisons of 2SS and FS are shown in Table 2.

It is observed that 2SS and FS exhibit very close performance which is better than that of the algorithm with no half-pixel search. The maximum decrease of average PSNR for these sequences is only 0.035 dB, which demonstrates that 2SS causes little degradation. And 2SS keeps almost the same bit stream length or bit rate as that of FS, since the greatest increase percentage is 0.90%. In addition, in some cases, the average PSNR increases slightly, such as that of News and Salesman. This can be achieved because the minimum SAD may not be the minimum MSE, on which PSNR is based, for the same motion vector. Table 3 indicates that 2SS can reduce the number of searched half-pixels for each block from 8 of FS to 3.5 or so. From these results, we can see that the proposed algorithm takes full advantage of the conclusion of Section 3 to the extent that it achieves a high accuracy while reducing the number of half-pixels to be searched.

6. CONCLUSION

In this paper, we study the characteristic of single valley on SAD error surface within the ±1 pixel neighborhood of integer-pixel search result, and draw the conclusion that the SAD distortion function increases monotonically as the search location moves away from the sub-pixel minimum within the ±1 pixel neighborhood. Based on this characteristic, we develope a new 2SS algorithm for half-pixel motion estimation. We also improve the process of the bilinear interpolation of reference frames and the calculation of the SAD's. Simulation results, which are performed by using H.263 codec and some standard sequences in QCIF format, demonstrate that the computational complexity of 2SS is lowered by more than 50% in comparison with that of FS. At the same time, the image quality and bit rate remain almost the same. The proposed algorithm can be easily integrated into existing video coding systems to enhance the performance of real-time coding. Hence, there are a wide range of applications such as video phone, video conference.

7. REFERENCES

[1] *ITU-T Recommendation H.263: Video Coding for Low Bit Rate Communication*, ITU-T, 1996.3.

[2] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, pp. 888-896, Aug. 1985.

[3] S. Kappagantula and K. R. Rao, "Motion compensated interframe image prediction," *IEEE Trans. Commun.*, vol. COM-33, pp. 1011-1015, Sept. 1985.

[4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. NTC81*, New Orleans, LA, Nov. 1981, pp. C9.6.1-9.6.5.

[5] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 438-442, Aug. 1994.

[6] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 313-317, June 1996.

[7] C.-H. Cheung and L.-M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 1168-1177, Dec. 2002.

[8] K. Panusopone and D. M. Baylon, "An analysis and efficient implementation of half-pel motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 724-729, Aug. 2002.