BLOCK MOTION ESTIMATION USING ADAPTIVE PARTIAL DISTORTION SEARCH

Yui-Lam Chan, Wan-Chi Siu and Ko-Cheung Hui
Centre for Multimedia Signal Processing
Department of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

ABSTRACT

The conventional search algorithms for block motion estimation reduce the set of possible displacements for locating the motion vector. All of these algorithms produce some quality degradation of the predicted image. To reduce the computational complexity of the full search algorithm without introducing any loss in the predicted image, we propose an adaptive partial distortion search algorithm (APDS) by selecting the most representative pixels with high activities, such as edges and texture which contribute most to the matching criterion. The APDS algorithm groups the representative pixels based on the pixel activities in the hilbert scan. By using the grouped information and computing the accumulated partial distortion of the representative pixels before that of the other pixels, impossible candidates can be rejected sooner and the remaining computation involved in the matching criterion can be reduced remarkably. Simulation results show that the proposed APDS algorithm has a significant computational speed-up and is the fastest when compared to the conventional partial distortion search algorithms.

1. Introduction

Motion estimation is an essential component of all modern video coding standards [1, 2]. It is included in these standards to reduce the redundancy between successive frames of a video sequence. The method adopted to estimate the motion between frames is the block matching algorithm (BMA) [3-6]. For the full search algorithm (FSA) of BMA, a matching criterion between every block in a search window from the previous frame and the current block is calculated. The most commonly used matching criterion is the mean absolute difference (MAD). Suppose that the block size is *16 × 16*, $I_t(i,j)$ is the intensity of the pixel at location *(i,j)* in frame *t* and *(k,l)* is the location of the upper left corner of a *16 × 16* block. The MAD matching criterion is a measure of the error between the block at location *(k,l)* of the frame *t* and the block at location *(k+u, l+v)* of the frame *t-1* in the search window which is given by,

$$MAD(k,l;u,v)=\sum_{i=0}^{15}\sum_{j=0}^{15}\left|I_t(k+i,l+j)-I_{t-1}(k+i+u,l+j+v)\right| \quad (1)$$
$$-D\le u,v\le D$$

where D is the maximum possible displacement of the motion vector *(u,v)*. The motion vector *(u,v)* of the block *(k,l)* is $\arg_{(u,v)}\min MAD(k,l;u,v)$.

The full search algorithm (FS) is the most straightforward BMA, which evaluates the MAD at all possible locations of the search window to find the optimal motion vector. Hence it is able to find the best-matched block which guarantees that the minimal MAD will be obtained. On the other hand, it also demands an enormous amount of computation. Thus a number of fast search algorithms [3-6] have been proposed, which seek to reduce the computation time by searching only a subset of the eligible candidate blocks. These fast block motion estimation algorithms include the three-step search algorithm (3SS) [3], the edge-oriented search algorithm [4, 5], the diamond search algorithm (DS) [6] and many variations of these algorithms. All algorithms reduce the number of computations required by calculating the MAD matching criterion at positions coarsely spread over the search window according to some patterns and then repeating the procedure with finer resolution around the position with the minimum MAD found from the preceding step. Nearly all of these algorithms rely on the assumption that the MAD distortion function increases monotonically as the search location moves away from the global minimum [4]. Unfortunately, this is usually not true in real-world video signals. As a consequence, the minimum MAD found by these methods is frequently higher than that produced by the FS.

Apart from the above lossy fast searching algorithms, a partial distortion search algorithm (PDS) [7, 8] is recommended to be used in H.263 [1] and MPEG-2 [2] video codec to reduce the computational complexity of the MAD calculation without introducing any loss. In the conventional PDS [7, 8], the accumulated partial distortion (MAD_p) is given by,

$$MAD_p(k,l;u,v)=\sum_{i=0}^{p}\sum_{j=0}^{15}\left|I_t(k+i,l+j)-I_{t-1}(k+i+u,l+j+v)\right|, \quad (2)$$
$$p=0,1,\ldots,15$$

The accumulated partial distortion is used to eliminate the impossible candidates of motion vector before a complete calculation of the MAD in a matching block. In other words, if the *p*th accumulated partial distortion is greater than the current minimum MAD at that time, this candidate is rejected and remaining computation of MAD is unnecessary. This algorithm can greatly reduce computational burden of the block-matching motion estimation. In [9], a grouping method is proposed to ensure that each accumulated partial distortion is not localized in a particular region on the block by dividing *MAD(k,l;u,v)* into *16* partial distortions (d_p), where each partial distortion consists of *16* pixels spaced equally between adjacent pixels, as depicted in Figure 1. The *p*th partial distortion is defined as,

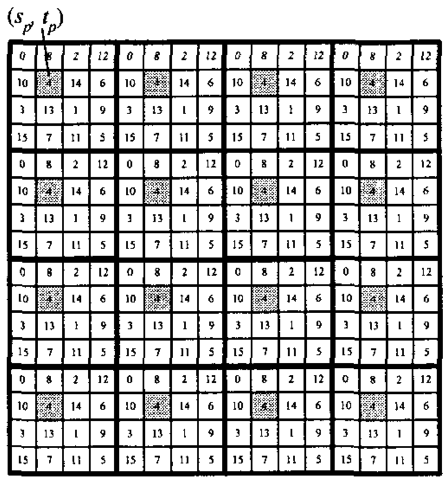
$$d_p(k,l;u,v)=\sum_{i=0}^{3}\sum_{j=0}^{3}\left|I_t(k+4i+s_p,l+4j+t_p)-I_{t-1}(k+4i+s_p+u,l+4j+t_p+v)\right| \quad (3)$$

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on July 05,2010 at 06:47:23 UTC from IEEE Xplore. Restrictions apply.

where s_p and t_p are the horizontal and vertical offsets of the upper left corner point of the pth partial distortion from the upper left corner point of the block, respectively. The order of calculation of the partial distortion d_p is also depicted in Figure 1. The pth accumulated partial distortion in eqn (2) is redefined as

$$MAD_p(k,l;u,v) = \sum_{i=0}^{p} d_i(k,l;u,v) \qquad (4)$$

This sub-sampling PDS (SS-PDS) achieves a better efficiency than the conventional PDS.



0	8	2	12	0	8	2	12	0	8	2	12	0	8	2	12
10	4	14	6	10	4	14	6	10	4	14	6	10	4	14	6
3	13	1	9	3	13	1	9	3	13	1	9	3	13	1	9
15	7	11	5	15	7	11	5	15	7	11	5	15	7	11	5
0	8	2	12	0	8	2	12	0	8	2	12	0	8	2	12
10	4	14	6	10	4	14	6	10	4	14	6	10	4	14	6
3	13	1	9	3	13	1	9	3	13	1	9	3	13	1	9
15	7	11	5	15	7	11	5	15	7	11	5	15	7	11	5
0	8	2	12	0	8	2	12	0	8	2	12	0	8	2	12
10	4	14	6	10	4	14	6	10	4	14	6	10	4	14	6
3	13	1	9	3	13	1	9	3	13	1	9	3	13	1	9
15	7	11	5	15	7	11	5	15	7	11	5	15	7	11	5
0	8	2	12	0	8	2	12	0	8	2	12	0	8	2	12
10	4	14	6	10	4	14	6	10	4	14	6	10	4	14	6
3	13	1	9	3	13	1	9	3	13	1	9	3	13	1	9
15	7	11	5	15	7	11	5	15	7	11	5	15	7	11	5

Figure 1. Grouping pixel locations for the calculation of the partial distortion d_p, and the number inside the pixel represents its corresponding order of computing the partial distortions.

The speed-up of the PDS depends on how fast the computation of the accumulated partial distortion is stopped. This paper is based on the fact that high activities in pixels contribute most to the MAD criterion. If the partial distortion is firstly computed on these high active pixels within a block, the impossible candidates will be eliminated sooner. However, the SS-PDS is not sufficient to reflect the above consideration. In this paper, we propose an adaptive partial distortion search algorithm (APDS) which groups the pixels for computing the partial distortion according to their activities in a hilbert scan to further reduce the computational complexity of the PDS. Experimental results show that the proposed APDS has a significant speed-up when compared to the conventional PDS and the SS-PDS.

2. Adaptive Partial Distortion Search (APDS)

A critical issue in PDS is how fast the impossible candidates are detected in order to reduce the computation to a minimum. In fact, the pixels with high activities such as edges and texture contribute most to the MAD criterion, and are regarded as the representative pixels in the block. By computing the accumulated partial distortion of the representative pixels before that of the other pixels, we can remove the impossible candidates sooner. Therefore, we can reduce the computational complexity of the BMA without any quality degradation of the predicted image. Consequently, the extraction of the representative pixels plays an important role in the development of an efficient PDS, which is the subject of discussion in the following sections.

2.1 Hilbert scan and its features

In 1891, the great German mathematician Hilbert found a family of Hilbert curves that pass through all the grid points only once in a 2-D space [10]. The Hilbert scan is the result of scanning a 2-D image through one of its Hilbert curves, as depicted in Figure 2. The Hilbert scan extracts clusters in an image easier than other scanning methods e.g. raster scan, and it preserves 2-D coherence [11]. In [12], Wang *et al* showed that the edge information in a 2-D image is preserved in its 1-D Hilbert-scan sequence more effectively than the raster scan which may miss edges due to its scanning direction. The adaptive PDS algorithm (APDS) that we are going to propose groups pixels according to the edge information in the 1-D Hilbert sequence.

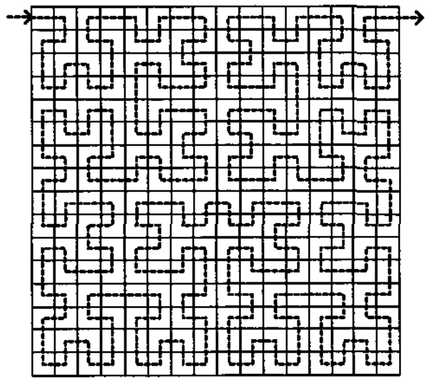


Figure 2. The Hilbert scan.

2.2 Description of the proposed APDS algorithm

After performing a Hilbert scan on a 2-D block, the block is converted into a 1-D Hilbert sequence. In the proposed APDS, pixels on the 1-D Hilbert sequence are sorted according to the absolute difference between a pixel and its preceding pixel. The sorted pixels are grouped for computing the accumulated partial distortion. Let us summarize our proposed algorithm as follows:

1. Convert the block of interest in the current frame into a 1-D Hilbert sequence, as depicted in Figure 2. The 1-D Hilbert sequence is denoted as $hs=\{hs_i \mid i=0,1,\ldots,255\}$, where hs_i is an ordered pixel in the 1-D Hilbert sequence.
2. Compute the absolute difference of the 1-D Hilbert sequence, $AD=\{AD_j \mid j=1,\ldots,255\}$, and AD_j is written by

   $$AD_j = | hs_j - hs_{j-1} | \qquad (5)$$

   and each AD_j is associated with a pixel pair, hs_j and hs_{j-1}.
3. For the sequence AD, the AD_js are sorted in descending order by counting sort [13].
4. Arrange the pixel pairs according to the sorted sequence of AD, and form the final sorted sequence of hs which is denoted as $hs'=\{hs'_i \mid i=0,1,\ldots,255\}$, where hs'_i is a sorted pixel in the 1-D Hilbert sequence. Since each hs_i is associated with both

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on July 05,2010 at 06:47:23 UTC from IEEE Xplore. Restrictions apply.

AD_i and AD_{i+1}, it may be duplicated in the *hs'*. As a consequence, when the pixel pair is placed on the *hs'*, its associated pixels must be checked. If the pixel is already on the *hs'*, this pixel will not be placed on the *hs'*; otherwise, this pixel is appended at the *hs'*.

5. Group *16* partial distortions, each of which consists of *16* pixels according to the order of the *hs'*. The *p*th partial distortion is defined as,

$$d_p(k,l;u,v)=\sum_{i=16p}^{16p-1}\Big|I_t(k+hsx_i^{'},l+hsy_i^{'}) - I_{t-1}(k+hsx_i^{'}+u,l+hsy_i^{'}+v)\Big| \tag{6}$$

where $hsx_i^{'}$ and $hsy_i^{'}$ are the horizontal and vertical offsets of the pixel $hs_i^{'}$ from the upper left corner point of the block, respectively. The proposed grouping method ensures that each accumulated partial distortion is based on the representative pixels which contribute most to the MAD criterion. The accumulated partial distortion, as defined in eqn (4), can reject the impossible candidates faster.

Figure 3 shows an example of how to obtain the final sorted sequence, *hs'*, from the 1-D Hilbert sequence, *hs*. For the sake of simplicity, *8* pixels have been used for our illustration. To compute the AD_j, its corresponding pixel pair is involved. For example,

$$\begin{aligned} AD_1 &= \mid hs_1 - hs_0 \mid = 20 \\ AD_2 &= \mid hs_2 - hs_1 \mid = 50 \\ AD_3 &= \mid hs_3 - hs_2 \mid = 40 \\ &\vdots \end{aligned} \tag{7}$$

After computing the absolute difference, the AD_js are arranged in descending order as shown in Figure 3, and the resulting sorted sequence is $\{AD_2, AD_3, AD_7, \ldots\}$. First, we place the edge pair $\{hs_2, hs_1\}$ of AD_2 at the beginning of the final sorted sequence, $hs' = \{hs_2, hs_1, \ldots\}$. Then, the second edge pair $\{hs_3, hs_2\}$ of AD_3 is checked. In this example, since hs_3 has not been included in *hs'* in the first step, it is appended at *hs'* such that *hs'* becomes $\{hs_2, hs_1, hs_3, \ldots\}$. Note that, the other pixel, hs_2, of the edge pair AD_3 has already been included in *hs'*, hence no further action is required for hs_2. The process is continued until all edge pairs of AD_js are checked.

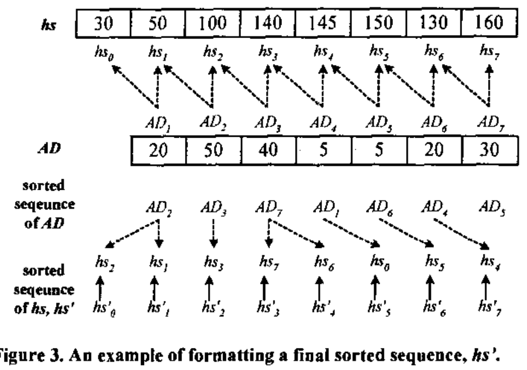

hs
30
50
100
140
145
150
130
160
hs0 hs1 hs2 hs3 hs4 hs5 hs6 hs7
AD1 AD2 AD3 AD4 AD5 AD6 AD7
AD
20
50
40
5
5
20
30
sorted seqeunce of AD
AD2 AD3 AD7 AD1 AD6 AD4 AD5
sorted sequence of hs, hs'
hs2 hs1 hs3 hs7 hs6 hs0 hs5 hs4
hs'0 hs'1 hs'2 hs'3 hs'4 hs'5 hs'6 hs'7

Figure 3. An example of formatting a final sorted sequence, *hs'*.

3. Simulation Results

The performance of the proposed APDS has been tested for a large variety of real image sequences, including "Salesman", "Flower Garden", "Table Tennis" and "Football". The performance results of the block motion vector estimation of the new APDS were compared with the results of some of the conventional methods which were the full search algorithm (FS), the conventional PDS and the sub-sampling PDS (SS-PDS) [9]. Note that the ability of the PDS to reject impossible candidates depends also on how well the search strategy allows the minimum matching error to be detected faster. To meet this purpose, the spiral search method [7, 8] was employed in the conventional PDS, the SS-PDS and the proposed APDS. The search began at the original checking point and then moved outward in a spiral-scanning path. The purpose of this order of searching was to exploit the center-biased motion-vector distribution characteristics of the real-world video sequence. The maximum allowable displacement in both the horizontal and vertical directions was *7* with a block size of *16 x 16*.

Since all partial distortion search algorithms mentioned in this paper are classified as lossless algorithms, the predicted images they produced have the same quality as the images produced by the FS. In the following discussion, the computational complexities of our proposed APDS are compared with those of the conventional algorithms including the FS, the conventional PDS and the SS_PDS. In general, several factors need to be taken into account in comparing the costs associated with various algorithms. These factors include speed, chip area and power, and they can usually be traded with each other depending upon the architecture to be used, hence comparing the costs associated with various algorithms is not an easy task. Nevertheless, it is possible to choose a simple way to define complexity. The fixed-point implementation of the different algorithms are now compared in terms of absolute conversion, addition/subtraction and integer comparison, while the computational reduction is based on the total operations of these types of operations. For our proposed APDS, we realized the partial distortion by adaptively grouping the pixels according to their activities in the hilbert scan for a further reduction of the computational complexity of the PDS, which is the major overhead of our proposed PDS. Its computational complexity is now examined. It mainly consists of two parts: the calculation and sorting of AD_j. From eqn(5), the calculation of AD_j includes *255* absolute conversions and *255* subtractions. By employing the counting sort [13], the amount of computation for sorting AD_j is reduced drastically in which *510* increment/decrement by one operations and *255* additions are required to sort AD_j in descending order. Though the increment/decrement by one operation requires less processing cycles than a general addition/subtraction operation in most digital signal processors, we still assume that this type of operation is equivalent to the addition/subtraction operation for the sake of simplicity. Combining all of these, Table 1 provides a comparison of the operational requirements of the FS, the conventional PDS, the SS-PDS and the proposed APDS. The table shows that the proposed ADPS has a greater computational reduction than the FS; it has a speed-up of about *1.96* to *4.73* times the speed of the FS. Figure 4 shows the average operations per block for each frame of the "Salesman" sequence using different PDS algorithms. These figures demonstrate that the proposed APDS has a significant computational reduction as compared with those of the conventional PDS and the SS-PDS. From the

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on July 05,2010 at 06:47:23 UTC from IEEE Xplore. Restrictions apply.

simulation results, we can conclude that the APDS is effective because of the fact that the proposed APDS can remove the impossible candidates faster by grouping the pixels according to their activities in the hilbert scan.

4. Conclusions

This paper proposes an adaptive partial distortion search algorithm (APDS), which groups the pixels for computing the partial distortion based on the pixel activities in the hilbert scan in order to reduce the computational complexity of the PDS. The speed-up of the APDS depends on how fast the computation of the accumulated partial distortion is stopped. The algorithm is based on the fact that the pixels with high activities such as edges and texture make the most contribution to the MAD criterion. If the partial distortion is firstly computed from these representative pixels within a block, the probability of making an early rejection of the impossible candidates will be high which introduces no quality degradation of the predicted image. We have tested the proposed APDS against some of the related conventional algorithms using many sequences and found that a speed-up of about *1.96 to 4.76* times is achievable as compared with the full search algorithm. Furthermore, it is faster than the conventional BMA using the PDS approach.

5. Acknowledgments

This work is supported by the Centre for Multimedia Signal Processing, HKPolyU. K.C. Hui acknowledges the research studentships provided by the University.

6. References

[1] ITU-T H.263, Video coding for low bit rate communication, Mar. 1996.

[2] ISO/IEC CD 14496-2, Information technology -- coding of audio-visual objects: visual, 1997.

[3] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," Proceedings of the National Tel. Conference, Nov. 29-Dec. 3, 1981, G.5.3.1-5.3.5.

[4] Y. L. Chan and W. C. Siu, "An efficient search strategy for block motion estimation using image features," IEEE Trans. on Image Processing, pp. 1-16, Aug. 2001.

[5] Y. L. Chan and W. C. Siu, "Edge oriented block motion estimation for video coding," IEE Proceedings: Vision, Image and Signal Processing, vol. 144, pp. 136-144, Jun. 1997.

[6] J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, A novel unrestricted center-biased diamond search algorithm for block motion estimation, IEEE Trans. on Circuits and Syst. for Video Tech., vol. 8, pp. 369-377, Aug. 1998.

[7] ITU-T recommendation H.263 software implementation, Digital Video Coding Group, Telenor R&D, 1995.

[8] S. Eckart and C. Fogg, "ISO/IEC MPEG-2 software video codec," Proc. SPIE, vol. 2419, pp. 100-118, 1995.

[9] C. K. Cheung and L. M. Po, "Normalized partial distortion search algorithm for block motion estimation," IEEE Trans. on Circuits and Syst. for Video Tech., vol. 10, pp. 417-422, Apr. 2000,

[10] D. Hilbert, "Über die stetige abbildung einer linie aufein flächenstück," Mathematische Annalen, vol. 38, pp. 459-460, 1891.

[11] R. J. Stevens, A. D. Lehar and F. H. Preston, "Manipulation and presentation of multidimensional image data using the Peano scan," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 5, pp. 520-526, Sept. 1983.

[12] Yankang Wang, Yanqun Wang and H. Kuroda, "A globally adaptive pixel-decimation algorithm for block-motion estimation," IEEE Trans. on Circuits and Syst. for Video Tech., vol. 10, pp. 1006-1011, Sept. 2000.

[13] A. Andersson, T. Hagerup, "S. Nilsson and R. Raman, Sorting in linear time," Journal of Computer and System Sciences, vol. 57, pp. 74-93, 1998.

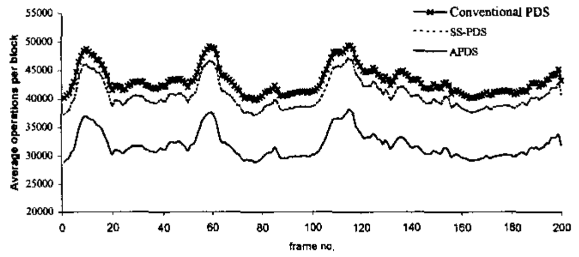


Figure 4. Frame-by-frame comparison of average operations per block in the "Salesman" sequence.

Table 1. Average operations per block for different sequences.

	Absolute Conversions	Additions/ Subtractions	Compariso n	Total
		Salesman		
FS	52296	104389	203	156888
Conventional PDS	14104	28003	1069	43176
SS-PDS	13267	26330	1016	40613
APDS	10657	21620	837	33114
		Flower Garden		
FS	51724	103247	201	155172
Conventional PDS	17460	34717	1276	53453
SS-PDS	16386	32569	1209	50164
APDS	13304	26915	1001	41220
		Table Tennis		
FS	51724	103247	201	155172
Conventional PDS	18308	36414	1329	56051
SS-PDS	17492	34782	1278	53552
APDS	14535	29378	1078	44991
		Football		
FS	51724	103247	201	155172
Conventional PDS	27995	55787	1935	85717
SS-PDS	27961	55720	1933	85614
APDS	25655	51617	1772	79044

Authorized licensed use limited to: Hong Kong Polytechnic University. Downloaded on July 05,2010 at 06:47:23 UTC from IEEE Xplore. Restrictions apply.