AN ADAPTIVE PARTIAL DISTORTION SEARCH FOR BLOCK MOTION ESTIMATION

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

ABSTRACT

Fast search algorithms for block motion estimation reduce the set of possible displacements for locating the motion vector. All 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 a Hilbert-grouped partial distortion search algorithm (HGPDS) by grouping the representative pixels based on pixel activities in the hilbert scan. By using the grouped information and computing the accumulated partial distortion of the representative pixels before that of other pixels, impossible candidates can be rejected sooner and the remaining computation involved in the matching criterion can be reduced remarkably. In addition, we also suggest a smart search strategy which is an excellent complement of the HGPDS to form an efficient partial distortion search algorithm. The new search strategy rearranges the search order such that the most possible candidates are searched first and this rearrangement will increase the probability of early rejection of impossible motion vectors. Simulation results show that the proposed 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) [3-5]. For the full search algorithm (FSA) of BMA, the sum of absolute difference (SAD) between every block in a search window from the previous frame and the current block is calculated. 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 SAD is a measure of the error between the block at location *(k,l)* of frame *t* and the block at location *(k+u, l+v)* of frame *t-1* in the search window which is given by,

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

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

The FSA is able to find the best-matched block which guarantees that the minimal SAD will be obtained. On the other hand, it also demands an enormous amount of computation. Thus a number of fast search algorithms [3-5] have been proposed, which seek for reducing the computation by searching only a subset of eligible candidates. 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 [3]. Unfortunately, this is usually not true in real-world video signals. As a consequence, the minimum SAD found by these methods is frequently higher than that produced by the FSA.

Apart from the above lossy fast searching algorithms, a partial distortion search algorithm (PDS) [6, 7] is recommended to be used in H.263 [1] and MPEG-2 [2] video codec to reduce the computational complexity of the SAD calculation without introducing any loss. In the conventional PDS [6, 7], an accumulated partial SAD (PSAD) is used to eliminate the impossible candidates of motion vector before the completion of calculating the SAD in a matching block. That is, if an intermediate PSAD is greater than the current minimum SAD at any time, this candidate is rejected and the remaining computation of SAD is unnecessary. We can define the p^{th} PSAD as,

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

In the conventional PDS, $PSAD_{15}(k,l;0,0)$, which is equal to *SAD(k,l;0,0)*, is first computed and this search block at origin is set to the so-far-minimum SAD, $so\text{-}far\text{-}SAD_{min}$. For other search blocks, if the $PSAD_p$ exceeds the $so\text{-}far\text{-}SAD_{min}$ at that time, this candidate is rejected and the remaining computation of the PSAD is saved. This algorithm can greatly reduce computational burden of the block-matching motion estimation. In [8], a grouping method is proposed to ensure that each PSAD not to be localized in a particular region on the block by dividing *SAD(k,l;u,v)* into *16* partial distortions, where each partial distortion consists of *16* pixels spaced equally between adjacent pixels. This sub-sampling PDS achieves a better efficiency than the conventional PDS.

In fact, the speed-up of the PDS depends on both grouping method (i.e., the pixel order for computing the SAD) and search order (i.e., the search order for candidate blocks in the search window). First, this paper is based on the principle that high activities in pixels such as edges and texture contribute most to the SAD matching error. If the PSAD is firstly computed on these representative pixels within a block, the impossible candidates will be removed sooner. However, the SSPDS is not sufficient to reflect the above consideration. In this paper, we propose a new partial distortion search algorithm, 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. Second, we propose a new search strategy to get a good match sooner. The good match implies that a lower *so-far-SAD$_{min}$* is obtained which can save a lot of computation than a higher *so-far-SAD$_{min}$*. It is found that the most possible reference blocks are searched first will be of great help to the PDS. Experimental results show that the proposed algorithm has a significant speed-up when compared to the conventional PDS and the SSPDS.

2. HILBERT-GROUPED PARTIAL DISTORTION SEARCH (HGPDS)

A critical issue in a PDS is how fast the impossible candidates are detected in order to reduce the computation to a minimum. In fact, pixels with high activities such as edges and texture contribute most to the SAD, and this type of pixels is regarded as the representative pixels in the block. By computing the PSAD of the representative pixels before that of the other pixels, we can remove impossible candidates sooner. Consequently, the extraction of the representative pixels plays an important role in the development of an efficient PDS.

The Hilbert scan [9] is the result of scanning a 2-D image through one of its Hilbert curves, as depicted in Figure 1. The Hilbert scan extracts clusters in an image easier than other scanning methods e.g. raster scan, and it preserves 2-D coherence [9]. In [10], 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 HGPDS that we are going to propose groups pixels according to the edge information in the 1-D Hilbert sequence.

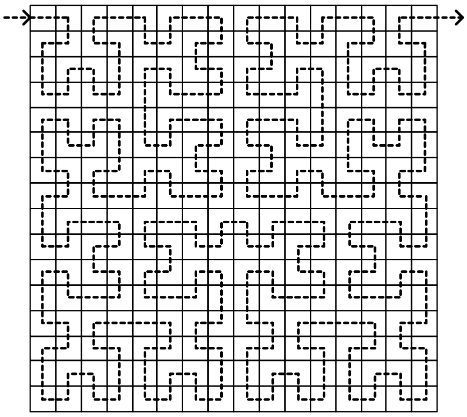


Figure 1. The Hilbert scan.

In the proposed HGPDS, 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 HGPDS. 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 1. The 1-D Hilbert sequence is denoted as *hs*={*hs$_i$* | *i=0,1,…,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$_j$* | *j=1,…,255*}, and *AD$_j$* is written by

$$AD_j = | hs_j - hs_{j-1} | \tag{3}$$

and each *AD$_j$* is associated with a pixel pair, *hs$_j$* and *hs$_{j-1}$*.

3. For sequence *AD*, the *AD$_j$s* are sorted in descending order by counting sort [11].

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'*={ $hs_i^{'}$ | *i=0,1,…,255*}, where $hs_i^{'}$ is a sorted pixel in the 1-D Hilbert sequence. Since each *hs$_i$* is associated with both *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,

$$d_p(k,l;u,v) = \sum_{i=16p}^{16p-1} \Big| I_t(k + hsx_i^{'}, l + hsy_i^{'}) - I_{t-1}(k + hsx_i^{'} + u, l + hsy_i^{'} + v) \Big| \tag{4}$$

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.

3. SEARCH ORDER FOR HGPDS

Ability to reject impossible candidates in the PDS also depends on the search order of possible candidate blocks in the search window, which allows the faster detection of minimum SAD. For this purpose, the spiral search [1-2] is employed in both H.263 and MPEG-2. The search begins at the origin checking point and then moves outward with a spiral-scanning path. This order of searching is to exploit the center-biased property of motion vector. Unfortunately, this is usually not true in videos containing fast moving objects. In order to save more operations for these videos, the search strategy should be designed by reordering the searching of possible blocks such that the candidates containing smaller SAD are searched first. Figure 2 shows a practical example to give an intuitive feeling of how the computation can be saved by reordering the searching of possible blocks. Figure 2(a) gives the error surfaces of the complete SAD (dotted line) and the PSAD using the spiral search with different initial centers (dashed line and solid line) when one of the current block with a size of *16 ´ 16* is matched in the search range, *±7 ´ ±7*, in the reference frame. PSAD is the accumulated partial SAD computed up to the moment when its value just exceeds the *so-far-SAD$_{min}$*. For simplicity, only the search points along the horizontal search line are shown and they are labeled as *s$_i$*, where *i* = (*-7,-6,...,-1,0,1,…,6,7*). If the spiral search starts at the origin (*s$_0$* in Figure 2(a)), the search order is {*s$_0$* , *s$_1$* , *s$_{-1}$* , *s$_2$* , *s$_{-2}$* , *s$_3$* , *s$_{-3}$* , *s$_4$* , *s$_{-4}$* , *s$_5$* , *s$_{-5}$* , *s$_6$* , *s$_{-6}$* , *s$_7$* , *s$_{-7}$*} and the *so-far-SAD$_{min}$* is updated in the order of {*s$_0$* , *s$_{-1}$* , *s$_{-2}$* , *s$_{-3}$* , *s$_{-4}$* , *s$_{-5}$*}, as depicted in Figure 2(a). On the other hand, if the starting point is shifted to *s$_{-5}$*, the search order becomes {*s$_{-5}$* , *s$_{-4}$* , *s$_{-6}$* , *s$_{-3}$* , *s$_{-7}$* , *s$_{-2}$* , *s$_{-1}$* , *s$_0$* , *s$_1$* , *s$_2$* , *s$_3$* , *s$_4$* , *s$_5$* , *s$_6$* , *s$_7$*}. In this case, *s$_{-5}$* initially gives a smaller *so-far-SAD$_{min}$* than the case of using *s$_0$* as the initial center, this means that PSAD for the remaining search points are much smaller. Figure 2(b) shows the number of pixels used for computing the PSAD using spiral search with initial centers *s$_0$* and *s$_{-5}$*, as depicted in dashed line and solid line respectively. The shaded area between the dashed line and the solid line implies the amount of saved operations when an initial center is shifted from s_0 to s_{-5}, which can be quite large. Thus if the starting point is close to the true motion vector, the PDS becomes more efficient. As a consequence, we rearrange the search order such that the most possible candidates are searched first. An outline of the proposed algorithm is as follows.

1. For each search point, only the first PSAD, $PSAD_1$, is computed by using the grouping method proposed in Section 2.
2. *h* search points, with $PSAD_1$ being the minimum, are selected. Since the Hilbert-scan pattern ensures that each $PSAD^1$ is the most contributive to the complete SAD, these *h* search points have large probabilities of being closest to the true motion vector.
3. Among these *h* search points, the one that has the minimum SAD (*so-far-SAD*$_{min}$) is selected as an initial center for the following spiral search.
4. The spiral search begins at the initial center found in step (3). For each search point, its $PSAD^1$ compares with the scaled *so-far-SAD*$_{min}$, for example, *so-far-SAD*$_{min}$ */SF*, where *SF* is the scaling factor. If the $PSAD^1$ is greater than the scaled *so-far-SAD*$_{min}$, its corresponding SAD is likely greater than the *so-far-SAD*$_{min}$. Thus, this position is considered as an invalid search point. Otherwise, the HGPDS is applied at this point and the *so-far-SAD*$_{min}$ is updated if its SAD is smaller than the *so-far-SAD*$_{min}$. Jumping out of the comparison at such an early stage will increase the probability of early rejection of non-possible candidates, however, it may also eliminate some good candidates too early, and thus introduce the distortion of the decoded sequence. Note that early jump-out is disabled when *SF* is set to one.

4. SIMULATION RESULTS

This section reports simulation results using the proposed grouping method and the search order for partial distortion search (PDS) in block motion estimation. Simulations have been carried out using the following sequences: “Flower Garden”, “Table Tennis” and “Football”. Table 1 shows the average numbers of operations and the PSNR of different grouping methods combined with the spiral search (*Spiral*) or the new search strategy (*NSS*) in partial distortion search. The performances of *FS+Spiral* (full search + spiral), *RSPDS+Spiral* (raster-scan PDS + spiral), *SSPDS+Spiral* (subsampling PDS + spiral), *HGPDS+Spiral* (Hilbert-grouped PDS + spiral), *HGPDS+NSS*$_{SF=1}$ (Hilbert-grouped PDS + NSS with SF=*1*, i.e, the early jump-out step is disable) and *HGPDS+NSS*$_{SF=8}$ (Hilbert-grouped PDS + NSS with SF=*8*) were compared. For *HGPDS+NSS*$_{SF=1}$ and *HGPDS+NSS*$_{SF=8}$, the number of search positions (*h*) selected in step (2) is set to *5*. The maximum allowable displacement in both horizontal and vertical directions was *7* with a block size of *16 ´ 16*.

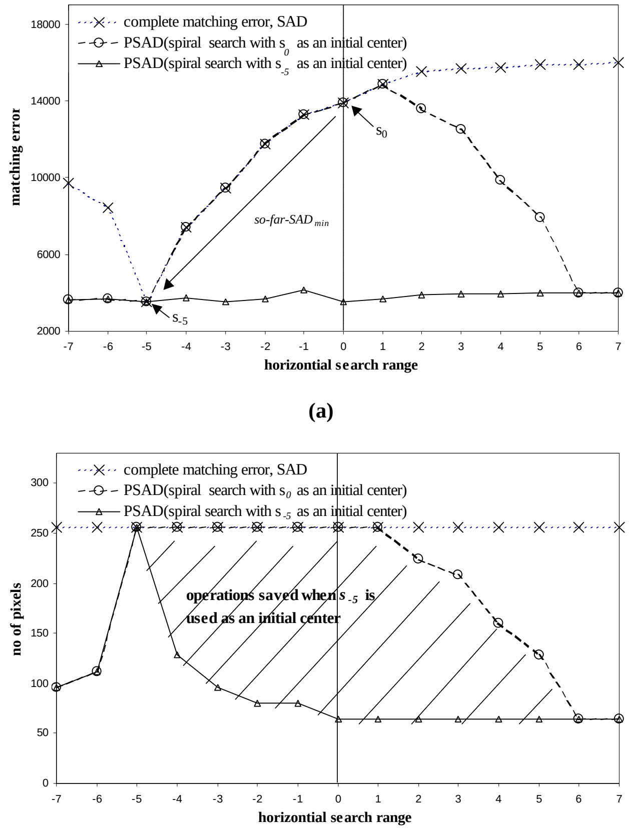
In the following discussion, a fixed-point implementation of different algorithms are compared in terms of absolute conversion, addition/subtraction and integer comparison, while the computational reduction is based on the total operations of these types of operations. For the proposed HGPDS, we suggest the computation of the partial distortion by grouping of pixels according to their activities in the Hilbert scan, which is the major overhead of HGPDS. It mainly consists of two parts: the calculation and sorting of AD_j. The calculation of AD_j includes *255* absolute conversions and *255* subtractions. By employing the counting sort, the amount of computation for sorting of AD_j is *510* increment/decrement operations and *255* additions. Though the increment/decrement in operations requires less processing cycles than the addition/subtraction operations in most digital signal processors, we still assume that this type of operations is equivalent to one addition/subtraction operation for the sake of simplicity. For NSS, it appears that one division operation is required to obtain the scaled *so-far-SAD*$_{min}$ in early jump-out step. However, this can be easily implemented by “right-shift” operations if the SF is carefully selected. In this simulation, the scaled factor *1/8* is used, which is equivalent to three “right-shift” operations. Combining all of these, Table 1 provides a comparison of the operational requirements of *FS+Spiral*, *RSPDS+Spiral*, *SSPDS+Spiral*, *HGPDS+Spiral*, *HGPDS+NSS*$_{SF=1}$ and *HGPDS+NSS*$_{SF=8}$. This table shows that the proposed *HGPDS+Spiral* has a computation reduction of about *2* to *3* times compared with *FS+Spiral*. As compared with the conventional PDS, *RSPDS+Spiral* and *SSPDS+Spiral*, *HGPDS+Spiral* has a remarkable computational reduction, as shown in Table 1. It is due to the fact that the proposed HGPDS can remove the impossible candidates faster by grouping the pixels according to their activities in the Hilbert scan. To further reduce the computational complexity, HGPDS can work with NSS, which rearranges the search order such that the most possible candidates are searched first in order to have a high probability of making an early rejection of the impossible candidates. Table 1 shows that combining the HGPDS with the NNS without early jump out, *HGPDS+NSS*$_{SF=1}$, produces a further computational reduction as compared with that of *HGPDS+Spiral*. In this simulation, we also use the early jump-out technique in the proposed algorithm, *HGPDS+NSS*$_{SF=8}$, in order to increase the probability of early rejection of non-possible candidates. Comparing to *HGPDS+NSS*$_{SF=1}$, it produces about 50% computational reduction.

Table 1 also shows the PSNR performance of all PDS mentioned in this paper. Since all these algorithms are classified as lossless algorithms except *HGPDS+NSS*$_{SF=8}$, the predicted images they produce are the same quality as the images produced by *FS+Spiral*. For *HGPDS+NSS*$_{SF=8}$, it is clearly observed that it can still maintain its PSNR performance very close to *FS+Spiral*. It indicates that the early jump-out technique used in the proposed algorithm has only a small penalty of about 0.02-0.06 dB degradation in PSNR of these reconstructed sequences.

5. CONCLUSIONS

This paper proposes an efficient grouping method for a partial distortion search algorithm (PDS) in which pixels for computing the partial distortion are grouped according to pixel activities in the Hilbert scan. The speed-up of the proposed Hilbert-grouped PDS (HGPDS) depends on how fast the computation of accumulated partial distortion is stopped and it is based on the fact that the pixels with high activities such as edges and texture contribute most to the complete SAD matching error. If the partial distortion is first computed from these representative pixels, 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 HGPDS using many sequences and found that a speed-up of about *2 to 3* times is achievable as compared with the full search algorithm. Furthermore, it is remarkably faster than the conventional PDS approaches.

A novel search strategy has also been suggested to work in conjunction with the HGPDS to further reduce its computational complexity. By locating the search center according to the first Hilbert-grouped partial distortion, the search strategy rearranges the search order such that the most possible candidates are searched first. Experimental results show that this strategy can strengthen the HGPDS especially in the sequences containing high motion. Thus, the proposed algorithm is best suited for a real-time implementation of high quality digital video codecs.



(a)

(b)

Figure 2. (a) Error surfaces of the complete matching error, SAD, (dotted line) and the PSAD using spiral search with s_0 and s_{-5} as an initial center, dashed line and solid line respectively. (b) Number of pixels used for computing the SAD and the PSAD mentioned in (a).

6. ACKNOWLEDGMENTS

This work is supported by the Centre for Multimedia Signal Processing, PolyU and the Internal Competitive Research Grants, PolyU (A-PD64).

7. 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] 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.

[4] 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.

[5] 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.

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

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

[8] 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,

[9] 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.

[10] Y. Wang, Y. 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.

[11] 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.

Table 1. Average operations per block for different sequences.

	Absolute Conv.	Add/ Sub	Comp./ R-shift	Total Op(s)	PSNR (dB)
Flower Garden					
FS+Spiral	51724	103247	201	155172	23.87
RSPDS+Spiral	17460	34717	1075	53252	23.87
SSPDS+Spiral	16386	32569	1008	49963	23.87
HGPDS+Spiral	13304	26915	1001	41220	23.87
HGPDS+NSS$_{SF=1}$	11898	24104	942	36944	23.87
HGPDS+NSS$_{SF=8}$	4932	10172	510	15611	23.85
Table Tennis					
FS+Spiral	51724	103247	201	155172	25.93
RSPDS+Spiral	18308	36414	1329	56051	25.93
SSPDS+Spiral	17492	34782	1278	53552	25.93
HGPDS+Spiral	14535	29378	1078	44991	25.93
HGPDS+NSS$_{SF=1}$	13676	27660	1048	42384	25.93
HGPDS+NSS$_{SF=8}$	5257	10823	525	16605	25.87
Football					
FS+Spiral	51724	103247	201	155172	23.94
RSPDS+Spiral	27995	55787	1935	85717	23.94
SSPDS+Spiral	27961	55720	1933	85614	23.94
HGPDS+Spiral	25655	51617	1772	79044	23.94
HGPDS+NSS$_{SF=1}$	22970	46248	1651	70869	23.94
HGPDS+NSS$_{SF=8}$	12577	25463	1005	39042	23.92