IMPROVED PARTIAL DISTORTION SEARCH ALGORITHM FOR RAPID BLOCK MOTION ESTIMATION VIA DUAL-HALFWAY-STOP

Xiaoquan Yi and *Nam Ling*
Department of Computer Engineering, Santa Clara University, Santa Clara, CA 95053, USA
{xyi, nling}@scu.edu

ABSTRACT

Block motion estimation is a critical yet computationally intensive task for video encoding. Many fast block matching algorithms have been developed. Partial distortion search (PDS) algorithms generally produce less video quality degradation of the predicted images than those of conventional fast block matching algorithms (BMAs). However, the speedup gain of PDS algorithms is usually limited. In this paper, we present an enhancement over a normalized PDS (NPDS) algorithm to further reduce block matching motion estimation complexity and improve video fidelity. The novelty of our algorithm is that, in addition to the halfway-stop technique in NPDS, a *dual-halfway-stop* (DHS) method, which is based on a dynamic threshold, is proposed so that block matching is not performed against all searching points. The dynamic threshold is obtained via a linear model utilizing already computed distortion statistics. An adaptive search range mechanism based on inter block distortion further constrains the searching process. Simulation results show that the proposed algorithm has a remarkable computational speedup. Particularly, it requires less computation by 92.0-99.4% when compared to that of full search (FS) and by 8.0-91.0% when compared to that of NPDS. It encounters an average of 0.07 dB video degradation in PSNR performance compared to that of FS whereas it gains 0.01-0.12 dB over that of NPDS algorithm.

1. INTRODUCTION

Motion estimation (ME) is critical to many video coding techniques since it removes interframe temporal redundancy. Many different motion estimation (ME) techniques were investigated and reported in the literature. Among them, block-matching algorithm (BMA) is the most popular method due to its effectiveness and simplicity. BMA is to seek for the best-matched block from the reference frame within a search window. The matching metric is usually called block distortion measure (BDM) such as the sum of absolute differences (SAD). The displacement of the best-matched block is described as the motion vector (MV) to the block in the current frame. The most obvious candidate for a search technique is full search (FS). Nevertheless, the high computational complexity of FS has motivated many suboptimal but faster search strategies. Two approaches can be chosen to reduce computation in BMAs. The first reduces the number of candidate blocks in the search window, e.g., new three-step search (N3SS) [1], diamond search (DS) [2], [3], cross-diamond search (CDS) [4], and kite CDS (KCDS) [5], etc. Nearly all of these algorithms rely on the assumption that the BDM function increases monotonically as the search location moves away from the global minimum. However, this property does not always hold for real-world video sequences. As a consequence, the video quality degradation introduced by these fast BMAs may be relatively high. This limitation has motivated a second approach which reduces the number of pixels involved in each candidate-test block comparison [6]-[9].

A partial distortion search algorithm (PDS) [6] was recommended to be used in MPEG-2 video codec to reduce computational complexity. Liu and Zaccarin [7] proposed an alternating sub-sampling search algorithm (ASSA) which reduces the number of pixels used in each BDM instead of reducing the number of checking points. Cheung and Po [8] proposed a grouping method called normalized partial distortion search (NPDS) for early rejection of impossible candidate motion vectors (CMV). NPDS ensures that each partial distortion not to be localized in a particular region on the block by dividing the SAD into 16 partial distortions, where each partial distortion consists of 16 pixels spaced equally between adjacent pixels. In addition, a halfway-stop technique is employed to reduce the computations in BDM. NPDS achieves an average computation reduction of 12-15 times with a slightly higher video degradation than ASSA (i.e. 0.08-0.22 dB). More recently, Cheung and Po [9] extended the NPDS to an adjustable PDS (APDS) so that it is capable of adjusting the prediction accuracy against searching speed by a quality factor. Simulation results show that APDS reduces computation up to 38 times with a relatively larger degradation in PSNR performance (e.g., 0.93 dB).

Both conventional fast BMAs and PDS's have advantages and disadvantages. In this paper, we aim to develop a simple yet efficient PDS algorithm to further reduce ME searching complexity while maximizing the advantages and avoiding the disadvantages. Our proposed algorithm is based on the following observations. It is widely known that the block motion field of natural video is usually gentle and smooth [1]-[5]. Although this property has been utilized in many fast BMAs, it is not well realized in any PDS algorithms. The halfway-stop technique in NPDS is used only inside the checking points. In other words, all checking points are checked as in the FS algorithm. This approach limits NPDS maximum of 16 times speedup. On the other hand, APDS [9] can theoretically lead up to 256 times faster than FS if the sub-sampling of 1 out of 256 pixels pattern is chosen which would introduce quite high prediction errors. In our proposed algorithm, we adopt NPDS fixed 16-pixel sub-sampling pattern but with relaxed comparisons. Then we introduce a novel *dual-halfway-stop* (DHS) method in addition to the one halfway-stop technique in NPDS [8] and APDS [9]. Our new second halfway-stop is applied only at the checking point level which is based on a dynamic block distortion threshold. The dynamic threshold is obtained via a linear model utilizing already computed block distortion statistics. Tourapis *et al*. [10] proposed adaptive thresholding parameters which are calculated by a function of a few minimum distortions from spatial and temporal neighboring blocks. Our dynamic threshold calculation is different with that of [10] in the sense that our algorithm utilizes the information of the current block whereas [10] does not. Additionally, in our new algorithm, an adaptive search range (ASR) mechanism based on inter block distortion further constrains the searching process. The coupling of DHS and ASR makes our new algorithm well adaptive to the motion content in which a strong correlation holds between searching complexity and video distortion across all the sequences. The next section reviews NPDS algorithm. Section 3 is devoted to our proposed algorithm. Extensive experiments are provided in Section 4. This paper concludes with Section 5.

2. NORMALIZED PDS (NPDS) ALGORITHM

In this section we briefly summarize the NPDS algorithm since it forms the basis of our new algorithm. SAD is chosen as the BDM. In addition, 16×16 block size is used. Suppose $I_n(i, j)$ is the intensity of pixel (i, j) in frame n and (k, l) is the location of the upper left corner of a block. The distortion between the block (k, l) of frame n and the block $(k + u, l + v)$ of frame and $n - \tau$ is given by

$$D = SAD(k,l;u,v) = \sum_{i=0}^{15}\sum_{j=0}^{15} | I_n(k+i,l+j) - I_{n-\tau}(k+i+u,l+j+v) | \ . \tag{1}$$

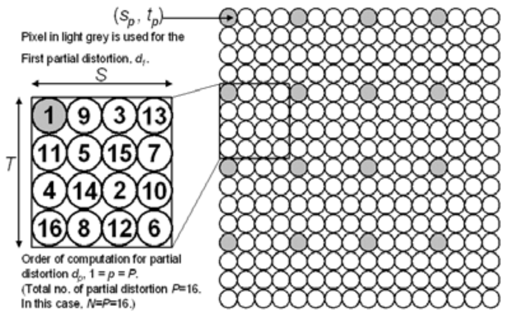
The basic operations of computing SAD are absolution (|·|) and addition (±), and require about $(3(2W + 1)^2 - 1)$ operations per BMD where W is the maximum search window size. To reduce the number of operations, partial distortion is defined as a group of pixels' distortions. Thus, the block distortion D is divided into 16 partial distortions (d_p) [9], where each partial distortion consists of 16 points spaced equally between adjacent points, as shown in Fig. 1. The pth partial distortion is defined as

$$d_p(k,l;u,v) = \sum_{i=0}^{3}\sum_{j=0}^{3} | I_n(k+4i+s_p, l+4j+t_p) - I_{n-\tau}(k+4i+s_p+u, l+4j+t_p+v) | \tag{2}$$

where (s_p, t_p) is the offset of the upper left corner point of the pth partial distortion from the upper left corner point of the block. The order of calculation of d_p is depicted in the left part of Fig. 1. The pth accumulated partial distortion is defined as

$$D_p(k,l;u,v) = \sum_{i=1}^{p} d_i(k,l;u,v). \tag{3}$$

The NPDS matches all the checking points inside the search window as the FS algorithm. The search begins at the origin checking points and moves outwards with a spiral scanning path. During each block matching, the NPDS compares each accumulated partial distortion D_p with the normalized minimum distortion ($pD_{\min}/16$). The comparison starts from p=1 to p=16, and the comparison is stopped if the normalized partial distortion of the CMV is greater than the normalized minimum distortion (NMD). This is called the halfway-stop in NPDS [8]. At the end of comparison (i.e., p=16), if D_{16} is smaller than $pD_{\min}/16$, then this CMV becomes the new current minimum point. By comparing the normalized partial distortion against the NMD, computational complexity is reduced by high rejection of impossible CMV at early stage. However, NPDS results in a limited saving of multiples of 16-pixel matching-operations. As a result, it limits the maximum possible speedup 16 times theoretically. In a more recent work [9], smaller partial distortions are formed to further speedup, however, it results in larger video degradations due to heavy sub-sampling (e.g. 4 out of 256 pixels). In this paper, we adopt NPDS fixed 16-pixel sub-sampling pattern with more relaxed comparisons. Then we introduce a novel *dual-halfway-stop* (DHS) method in addition to the halfway-stop technique in NPDS [8] and APDS [9]. The details about our new algorithm are described in the next section.



(s_p, t_p)
Pixel in light grey is used for the First partial distortion, d_1.
S
T
1 9 3 13
11 5 15 7
4 14 2 10
16 8 12 6
Order of computation for partial distortion d_p, 1 = p = P.
(Total no. of partial distortion P=16. In this case, N=P=16.)

Fig. 1. Computation order of partial distortion d_p of 16 × 16 block. (s_p, t_p) is the offset of the upper left corner of a partial distortion from the upper left corner of the candidate block.

3. PROPOSED DUAL-HALFWAY-STOP NPDS (DHS-NPDS) ALGORITHM

Unlike the conventional PDS algorithms, our new algorithm does not need to match all the checking points inside the search window. It is due to a well-known fact that the block motion field of natural video is usually gentle and smooth [1]-[5]. This implies many motion vectors are stationary or quasi-stationary, hence many checking points can be safely skipped. In our algorithm, to further reduce computational complexity without introducing much video degradation, those redundant checking points are avoided via an intelligent second halfway-stop technique that is based on a dynamic block distortion threshold. The halfway-stop technique in NPDS and the second halfway-stop techniques together are named *dual-halfway-stop-NPDS* (DHS-NPDS) algorithm. The checking points are in an outward spiral order fashion as shown in Fig. 2. In our algorithm, the search can be stopped at any point if necessary. The DHS is coupled with an adaptive search range which is described in the following sub-section.

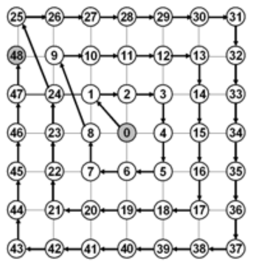


Fig. 2. Spiral scanning path of DHS-NPDS for W = ±3. The search can be stopped at any checking point (e.g., 0, 1, 2, 3, or 10 etc.)

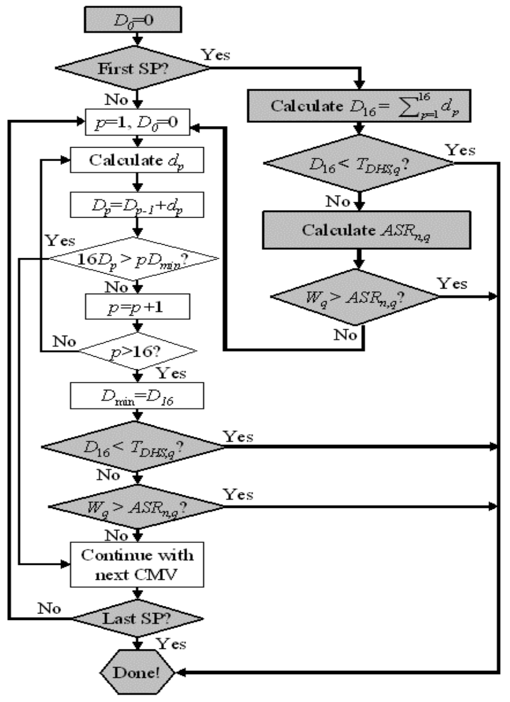


Fig. 3. Flowchart of a complete block matching calculation in the DHS-PDS algorithm. d_p: pth partial distortion, D_p: pth accumulated partial distortion, $D_{\min}$: current minimum distortion, D_{16}: BDM of the CMV, $T_{DHS,q}$: threshold for DHS, and SP: search point. W_q: current search window size. Shaded parts are our algorithm added in addition to the original NPDS algorithm.

3.1. Adaptive Search Range

Although adaptive search range has been used in some BMAs such as [11], where the size is based on the smallest BDM and the second smallest BDM, this approach has the drawback that the size can be only determined after a few searches already performed. In contrast, our adaptive search range (ASR) is based only on the first block matching error and is determined only once in the initial search step. Mathematically,

$$ASR_{n,q} = \left\lceil \frac{SAD_{n,q;0,0}}{16\times 16} \times \left(\frac{W}{\psi} + \theta\right) \right\rceil \tag{4}$$

where $SAD_{n,q;u,v}$ is the SAD of the current qth block in the nth frame and (u,v) is the displacement of the possible matching block in the reference frame, W is the maximum search window size, ψ is set as 128, and θ is set as 0.5 empirically. Note that the ASR is a relaxed upper-bound. Eq. (4) is from empirical results, indicating the general relationship between $ASR_{n,q}$ and $SAD_{n,q;0,0}$, and can be adjusted and relaxed. Finally, $ASR_{n,q}$ is clipped with W. Only two right-shift, one addition, and one multiplication operations are required for ASR computation per block. In the case of zero motion vector, ASR is not required to compute.

3.2. Dynamic Thresholding via a Linear Model

Ideally, the dynamic threshold for the current qth block ($T_{DHS,q}$) should be very close to the final minimum SAD using FS. With this goal in mind, $T_{DHS,q}$ is chosen adaptively with the current block motion content via a linear model. We use two terms to obtain $T_{DHS,q}$, namely, SAD ratio ($SADR_{n,q;0,0}$) and average minimum SAD ($ASAD_{n,q;min}$). The $SADR_{n,q;0,0}$ is the ratio of $SAD_{n,q;0,0}$ to the average of SAD of the same location between the current nth frame and the reference frame up to the qth block in the frame. Mathematically,

$$SADR_{n,k;0,0} = SAD_{n,q;0,0} \Big/ \left(\frac{1}{q}\sum\nolimits_{m=1}^{q} SAD_{n,m;0,0}\right) . \tag{5}$$

Since the minimum SAD of the qth block is not known yet, $ASAD_{n,q;min}$ is calculated using all the blocks prior to the qth block as

$$ASAD_{n,q;\min} = \frac{1}{k-1}\sum\nolimits_{z=1}^{k-1} SAD_{n,q;\min} . \tag{6}$$

The dynamic threshold $T_{DHS,q}$ is computed using a linear model utilizing the current block distortion information with $SADR_{n,q;0,0}$ and $ASAD_{n,q;min}$ as

$$T_{DHS,q} = SADR_{n,q;0,0} * ASAD_{n,q;\min} * \lambda + \varepsilon \tag{7}$$

where λ is a constant scale factor and is set as 0.5, ε is a constant factor and is empirically set as 0. Those parameters are carefully selected so that none or minimal premature endings happen. Plugging (4) and (5) into (6), after a few straightforward manipulations, the threshold is expressed as

$$T_{DHS,q} = SAD_{n,q;0,0} \frac{q}{q-1} \frac{\sum_{z=1}^{q-1} SAD_{n,z;\min}}{\sum_{l=1}^{q} SAD_{n,l;0,0}} \lambda . \tag{8}$$

Approximately, $q/(q-1)$ is close to 1. Eq. (7) is then simplified as

$$T_{DHS,q} = SAD_{n,q;0,0} \frac{\sum_{z=1}^{q-1} SAD_{n,z;\min}}{\sum_{m=1}^{q} SAD_{n,m;0,0}} \lambda . \tag{9}$$

If the $SAD_{n,q,u,v}$ of the current block is less than $T_{DHS,q}$, we immediately stop the search process for this block and return the MV that has the minimum SAD so far. It is noted that the threshold is computed only once per block. Since $SAD_{n,q;min}$ and $SAD_{n,q;0,0}$ are already computed in the original NPDS algorithm, the extra operations are two multiplications, two additions, and one division in (9). The summations are accumulative so only two additions are required for each block. The multiplication and division can be easily translated to combinations of "left-shift" (<<), "right-shift" (>>), and "addition" (±) operations. Thus, the overhead of the DHS-NPDS is negligible compared to the computation required by each partial distortion calculation. Unlike [10], our dynamic threshold calculation is very simple and more accurate since it utilizes the distortion information of the current block. Besides, in our method, there is no need for additional memory for storing the neighboring motion vectors.

3.3. DHS-NPDS Algorithm

The search begins at the origin checking point and then moves outwards with a spiral scanning path as depicted in Fig. 2. For the first block matching in the checking point, in contrast to [8], [9], we always compute the full distortion D_{16} instead of partial distortion since the first distortion will be used as the initial minimum distortion $D_{\min}$ for later comparisons. More importantly it is used for computing the ASR and the threshold as described in the previous sub-sections. We compare the D_{16} with the threshold $T_{DHS,q}$. If D_{16} is less than $T_{DHS,q}$, then we stop immediately. Hence the rest of block matching computation is avoided. Otherwise, we set $ASR_{n,q}$ using Eq. (4). During the remaining block matching process, the DHS-NPDS algorithm compares each accumulated partial distortion D_p with the NMD ($pD_{\min}/16$). The comparison starts from p=1 to p=16, and the comparison is stopped if the normalized accumulated partial distortion of the CMV is greater than the NMD. We observed that NPDS may result in up to 0.22 dB video degradations, mainly due to the normalized comparisons. In our algorithm, we slightly relax the NMD by a factor of 5/4 if $W > 31$ or 9/8 if $W \leq 31$, which can be easily translated to an addition and a right-shift. At the end of comparison (i.e. p=16), if the normalized D_{16} is smaller than the normalized $D_{\min}$, then this CMV becomes the new current minimum point. The NPDS in [7] matches all the checking points inside the search window as the FS algorithm. As pointed out earlier, this approach is inefficient. To overcome the NPDS and APDS's inadequacy, a second halfway-stop technique is applied at the end of the comparison (i.e., p = 16). After the last d_p or d_{16} in our case, we add an extra step in which the final accumulated partial distortion D_{16} is compared with the threshold $T_{DHS,q}$. If D_{16} is smaller than $T_{DHS,q}$ or the search window size reaches $ASR_{n,q}$, then we stop immediately, hence the rest of block matching computation is saved. Otherwise, we continue to the next block matching. Given the fact that more than 80% of the blocks can be regarded as stationary or quasi-stationary blocks for the real-world sequences [1]-[5], our innovative second halfway-stop helps reduce the computation significantly with little performance compromise. The procedures of BDM calculations of a complete checking point in the DHS-NPDS algorithm is summarized in the flow-chart as shown in Fig. 3. Our enhancement in addition to NPDS is in light grey in Fig. 3. With our DHS-NPDS method, the search can be stopped at any point (e.g., 0, 1, 2, or 10 etc). This additional halfway-stop method, coupling with ASR, makes DHS-NPDS the fastest BMA among all the BMAs we know so far, based on the above analysis and our intensive experimental results.

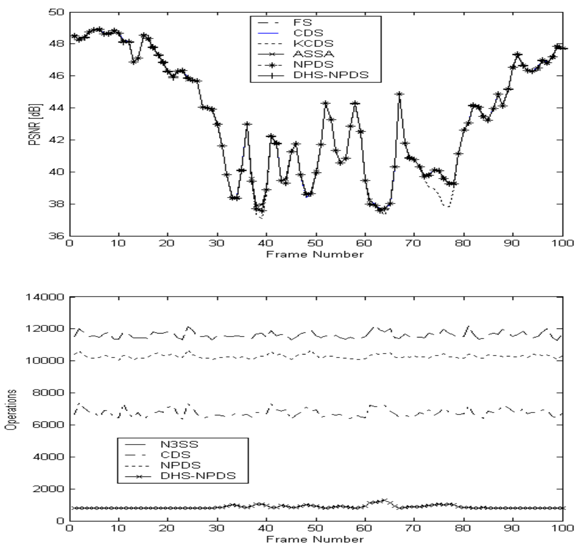


Fig. 4. Frame by frame comparisons for "Grandma" sequence. Top: Average PSNR. Bottom: Average total operations per block.

4. EXPERIMENTAL RESULTS

Numerous experiments have been conducted to justify the performance of our DHS-NPDS algorithm. The PSNR was used as a performance measure. The motion vector search was based on the luminance component with the search block of 16×16, and the maximum search area (SA) was ±7 pixels. The simulation was performed with seven sequences in QCIF and three sequences in CIF (each of 100 frames) which represent a wide range of motion contents. We compare our DHS-NPDS against FS, N3SS [1], DS [2], [3], CDS [4], KCDS[5], ASSA [7], and NPDS [8] in terms of PSNR and operations: absolution, addition, comparison, and shifting, while the computation reduction is based on the total operations of these types of operations. One example, "Miss America" with detailed operations is listed in Table I. The PSNR differences to FS and speedups for all sequences are tabulated in Table II. The speedup is computed via *operations_of_FS / operations_of_BMA*. The speedup of DHS-NPDS is quite significant for all slow-, medium-, and high-motion videos, particularly requiring only 882 operations per block for gentle video, which is 160 times faster than that of FS, 11.3 times faster than that of NPDS. In the mean time, DHS-NPDS only results in an average of 0.07 dB PSNR degradations compared to that of FS. DHS-NPDS shows better PSNR by 0.01-0.12 dB than that of NPDS, which is due to the modified and relaxed normalized minimum distortion in our new algorithm. One example, “Grandma”, with frame by frame comparisons of PSNR and average operations is given in Fig. 4. The subjective video quality of sequence “Football” is reflected in Fig. 5. Another nice property about DHS-NPDS is that the complexity is strongly correlated with its motion content magnitude observed from Table II. DHS-NPDS under other search areas consistently shows its superiority over any other suboptimal BMAs and PDS algorithms. For the cases examined, DHS-NPDS is about 280 times faster than FS for SA = ±15, 939 times for SA = ±32, and 2307 times for SA = ±64, while having an average PSNR loss of only 0.07 dB versus that of FS. DHS-NPDS can be easily extended for variable block size motion estimation such as in H.264/AVC standard.

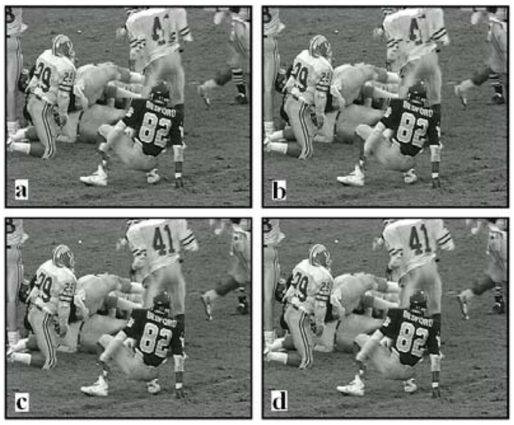


Fig. 5. The 81st reconstructed frame for “Football” sequence using different BMAs: (a) CDS; (b) KCDS; (c) NPDS; and (d) DHS-NPDS.

5. CONCLUSION

In this paper, we propose an effective algorithm to reduce motion estimation computational complexity while retaining video visual quality. Our new algorithm, DHS-NPDS, is simple and robust, easy to implement. It shows distinct improvements in terms of speedup, resulting in smoother motion field, and hence demanding less motion vector difference bits when compared to those of many well-known PDS and fast block matching algorithms. Our algorithm can be applied to a wide range of applications, owning to its superior performance and regularity.

TABLE I – AVERAGE OPERATIONS PER BLOCK FOR “MISS AMERICA”

BMA	Abs.	Add.	Comp.	Shift	Total
FS	47246.22	94307.89	183.56	-	141737.67
N3SS	4035.49	8055.22	14.76	-	12105.47
DS	3092.38	6172.67	11.08	-	9276.13
CDS	2487.13	4964.55	8.72	-	7460.39
KCDS	1727.61	3448.48	3.6	-	5179.68
ASSA	12835.56	25482.56	188.56	-	38506.67
NPDS	3494.93	6814.14	385.99	21.53	10716.58
DHS-NPDS	***363.78***	***739.4***	***15.91***	***20.11***	***1139.19***

REFERENCES

[1] 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.

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

[3] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. Image Processing*, vol. 9, pp. 287-290, Feb. 2000.

[4] 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.

[5] C. W. Lam, L. M. Po, and C. H. Cheung, “A novel kite-cross-diamond search algorithm for fast block matching motion estimation,” *IEEE International Symp. on Circuits and Syst.*, vol. III, pp. 729-732, May 2004.

[6] S. Eckart, C. Fogg, ISO/IEC MPEG-2 software video codec. *Proc. SPIE*, vol. 2419 (1995), pp. 100-118.

[7] B. Liu and A. Zaccarin, “New fast algorithms for the estimation of block motion vectors,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 148-157, Apr. 1993.

[8] C. K. Cheung and L. M. Po, “Normalized partial distortion search algorithm for block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 417-422, Apr. 2000.

[9] C. K. Cheung and L. M. Po, “Adjustable partial distortion search algorithm for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 100-110, Jan. 2003.

[10] A. Tourapis, O. C. Au, and M. L. Liou, “Highly efficient predictive zonal algorithm for fast block-matching motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, pp. 934-947, Oct. 2002.

[11] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, “Dynamic search-window adjustment and interlaced search for block-matching algorithm,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 85-87, Feb. 1993.

TABLE II – SIMULATION RESULTS OF PROPOSED ALGORITHM IN TERMS OF PSNR (DB) AND SPEEDUP TO FS (FS SHOWS ACTUAL PSNR)

Sequence	PSNR (dB)	BMA							
	Speedup	FS	N3SS	DS	CDS	KCDS	ASSA	NPDS	***DHS-NPDS***
Akiyo (QCIF)	PSNR to FS	44.16	0	0	0	-0.2	0	-0.01	***0***
	Speedup	**1**	**12.57**	**16.14**	**22.42**	**39.58**	**3.68**	**14.24**	***160.58***
Claire (QCIF)	PSNR to FS	42.95	0	0	-0.01	-0.19	0	-0.01	***-0.01***
	Speedup	**1**	**12.49**	**16.1**	**22.13**	**38.40**	**3.68**	**14.12**	***152.71***
Table Tennis (QCIF)	PSNR to FS	27.29	-0.28	-0.43	-0.48	-2.49	-0.02	-0.18	***-0.12***
	Speedup	**1**	**10.38**	**13.35**	**14.18**	**17.96**	**3.68**	**12.56**	***27.38***
Stefan (QCIF)	PSNR to FS	25.34	-0.16	-0.50	-0.55	-1.45	-0.02	-0.09	***-0.06***
	Speedup	**1**	**9.50**	**12.54**	**12.50**	**18.12**	***3.68***	**12.43**	***17.99***
Flower Garden (CIF)	PSNR to FS	25.63	-0.01	-0.02	-0.01	-7.28	*-0.01*	-0.11	***-0.05***
	Speedup	**1**	**11.06**	**14.25**	**15.43**	**29.73**	***3.71***	**12.84**	***25.34***
Mobile (CIF)	PSNR to FS	23.89	-0.03	-0.08	-0.05	-1.15	*-0.02*	-0.18	***-0.08***
	Speedup	**1**	**11.36**	**15.42**	**17.79**	**22.08**	***3.71***	**13.32**	***20.4***
Football (CIF)	PSNR to FS	23.36	-0.37	-0.6	-0.69	-2.46	*-0.01*	-0.22	***-0.1***
	Speedup	**1**	**9.76**	**12.37**	**12.65**	**15.04**	***3.71***	**11.96**	***13.05***