Transactions Letters

Fast Variable Block Size Motion Estimation by Adaptive Early Termination

Esam A. Al Qaralleh and Tian-Sheuan Chang, *Member, IEEE*

***Abstract*—This paper presents a fast motion estimation algorithm by adaptively changing the early termination threshold for the current accumulated partial sum of absolute difference (SAD) value. The simulation results show that the proposed algorithm can provide the similar quality while saving 77.9% and 50.6% of SAD computation when comparing with early termination methods in MPEG-4 VM18.0 and H.264 JM9.0, respectively.**

***Index Terms*—Early termination, H.264, MPEG-4, variable block size motion estimation.**

I. Introduction

MOTION ESTIMATION (ME) has been adopted by all of the existing international standards related to video coding [1], [2] to remove temporal redundancy between frames. However, ME is also the most computationally intensive part in a typical video encoder (50%–90%) of the entire system [5]. Thus, many fast ME algorithms have been proposed to reduce the complexity by searching only a subset of eligible candidates, for example, three step search algorithm [4], global elimination algorithm (GEA) [5] and quarter-pel motion estimation (QME) [1]. GEA, derived from successive elimination algorithm (SEA) [7], is proposed to remove the problems of SEA by using techniques similar to pel averaging and multiple candidates search. QME subsamples the search window by four to speed up the process.

Other fast algorithms employ early termination scheme to reduce the number of block distortion measure, which is our concern here. One early termination scheme, partial distortion search algorithm (PDS), is recommended to be used in MPEG-4 [3] and H.264 [10] reference software to reduce the computational complexity of the SAD without introducing any loss in PSNR quality. In the PDS, the accumulated *partial SAD* (PSAD) is used to eliminate the impossible candidates of motion vector (MV) before the completion of calculating the SAD in a matching block. Thus, if the PSAD is greater than the current minimum SAD at anytime, this candidate is rejected and the remaining SAD computation is skipped. The PSAD is computed and accumulated by calculating the SAD for one line of the block at a time. Though this scheme can skip some unnecessary SAD computations, there is still room to be improved. For example, this scheme cannot efficiently skip the SAD computations when the candidate SAD is similar to the current minimum SAD. This situation will happen especially at the search points near to the predicted or the final MV location.

Adjustable partial distortion search (APDS)[9] is a normalized partial distortion comparison method capable of adjusting the prediction accuracy against searching speed by a quality factor. It uses the halfway-stop technique with progressive partial distortions (PPD) to increase early rejection rate of impossible candidate MVs at very early stages. APDS divides each matching block into 16 equal sized groups, each group is subsampled in difference patterns, and each pattern has its own impact on the speed and the quality. Matching process starts by accumulating the distortion measure for one group after another. So by comparisons of the normalized partial distortion against normalized minimum block distortion, it can save more computational complexity.

Hilbert-grouped partial distortion search (HGPDS) [8] first employs the Hilbert scan to extract the representative pixels according to the edge information in the one-dimensional (1-D) Hilbert sequence, groups them into 16 16-pixel groups, sorts these groups in descending order, and finally computes the partial distortion according to the order of the groups. It uses a new search strategy by scanning the search window twice. In the first scan phase, it computes the distortion for the first group of the 16-groups established above. Then n-search points with the lowest PSAD are located and tested later to choose the best search center among them for the spiral scan which is the second scan phase. The overhead of the HGPDS is the grouping of pixels according to their activities in the Hilbert scan which consists of extra calculations and sorting also two scan phases for the new search strategy. So by doing the comparisons using the representative pixels first, this algorithm can save more computational complexity.

Both APDS and HGPDS exhibits irregular data flow due to extracting pixels in irregular pattern, which hinders their use in the hardware design. The PDS still consumes extra SAD computations especially for search points which are closer to the final MV location. H.264 introduces a variable block size ME, which reduces the efficiency of the early termination algorithms mentioned above. Small size blocks (i.e., 4×4) contain small number of pixels, in which the coherency between those pixels is high. So when employing the early termination algorithm, each block will consume more SAD computations till the termination happens. Repeating this process for all small size blocks means more SAD computations when compared with a normal 16×16 macroblock (MB).

To solve the above problems, we propose an early termination algorithm by adaptive threshold instead of using nearly constant

Manuscript received Septmber 18, 2005; revised February 22, 2006. This work was supported in part by the National Science Council, Taiwan, R.O.C., under Grant NSC-93-2200-E-009-028. This paper was recommended by Associate Editor C. N. Taylor.

The authors are with the Department of Electronics Engineering, National Chiao Tung University, Hsinchu 160, Taiwan, R.O.C.

Digital Object Identifier 10.1109/TCSVT.2006.879103

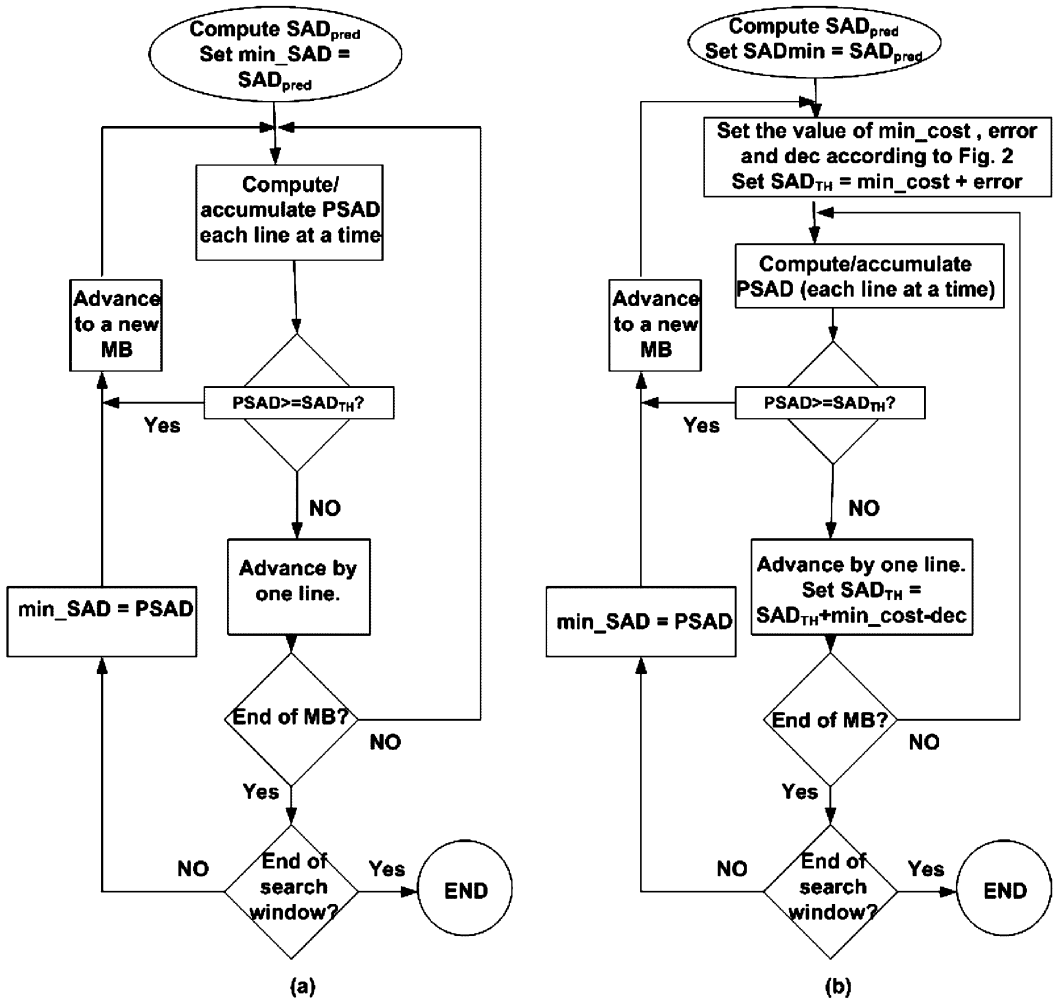


Fig. 1. (a) Algorithm flow for the PDS. (b) Our proposed algorithm.

minimum SAD value during the SAD accumulation. The results show that it can reduce the SAD computation significantly with negligible quality degradation. The proposed algorithm shows good efficiency with variable block size. Also it exhibits regular data flow, since it tests the MB line by line.

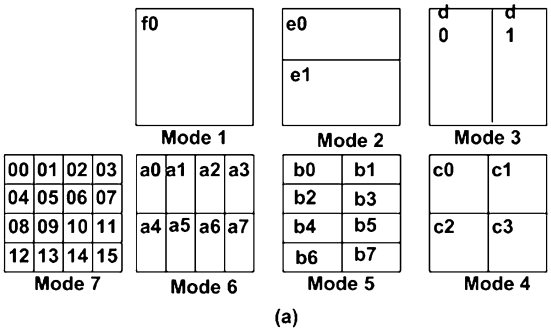
The rest of the paper is organized as follows. In Section II, we will introduce the proposed early termination scheme. Section III will show the experimental results. Comparison with other fast ME algorithms is shown in Section IV. Finally, the conclusion is made in Section V.

II. Early Termination Algorithm

Fig. 1(a) shows the flowchart of computing the PDS. It assumes a minimum SAD ($\mathrm{SAD_{min}}$), by computing the $\mathrm{SAD_{pred}}$ at the predicted location in the search window, and sets that value to be $\mathrm{SAD_{min}}$ for later comparison. Then it starts the SAD computation for each search point. During the SAD computation, if the partial SAD is equal to or greater than $\mathrm{SAD_{min}}$, it terminates the remaining partial SAD computation and jumps to the next search position. For such early termination purposes, the partial SAD is computed one line at a time and accumulated to the total SAD for that MB. This method can quickly skip the unnecessary SAD computation and preserve the search quality. However, if the SAD of the candidate MB is similar to the current minimum SAD, there is little room to save more computations.

Fig. 1(b) shows the flow of the proposed algorithm. Our proposed algorithm adopts the similar approach as the PDS. We still compute the partial SAD one line at a time and add it to the total SAD. However, during the accumulation of SAD, we use a threshold value $\mathrm{SAD_{TH}}$ instead of the minimum SAD ($\mathrm{SAD_{min}}$) as an early termination condition. If the accumulated SAD value is larger than the threshold, we skip the remaining SAD computation and proceed to the next search point. Otherwise, we continue the SAD computation/accumulation, and update the $\mathrm{SAD_{TH}}$ accordingly. When a match occurs, which means lower SAD value than the current minimum SAD, $\mathrm{SAD_{TH}}$ is modified according to the new value of minimum SAD.

To help adaptively changing the threshold value, three parameters are introduced in the flow diagram, "$\mathrm{SAD_{TH}}$," "error," and "dec." The values for $\mathrm{SAD_{TH}}$, "error," and "dec" for H.264 and MPEG-4 are depicted in Fig. 2(b). These parameters have considered the effect of variable block size and thus have different values according to their block size. "$\mathrm{SAD_{TH}}$" represents the base threshold value used in early termination process, which is a summation of current minimum SAD and extra error margin



H.264

```
if(mode == 1)      {min_cost=SADmin/16; error = 64; dec=4;}
if(mode== 2)       {min_cost=SADmin/8; error = 32; dec=4;}
if(mode== 3)       {min_cost=SADmin/16; error = 32; dec=2;}
if(mode== 4)       {min_cost=SADmin/8; error = 16; dec =2;}
if(mode== 5)       {min_cost=SADmin/4; error= 8; dec =2;}
if(mode== 6)       {min_cost=SADmin/8; error= 8; dec = 1;}
if(mode== 7)       {min_cost=SADmin/4; error= 4; dec=1;}
SADTH = min_cost + error;
```

MPEG-4

```
min_cost = SADmin/16; error = 64; dec=4;
SADTH = min_cost + error;
```

(b)

Fig. 2. (a) Seven MB modes in H.264/AVC. (b) Values for $\mathrm{SAD_{TH}}$, "error," and "dec" for different block sizes in H.264 and for MPEG-4.

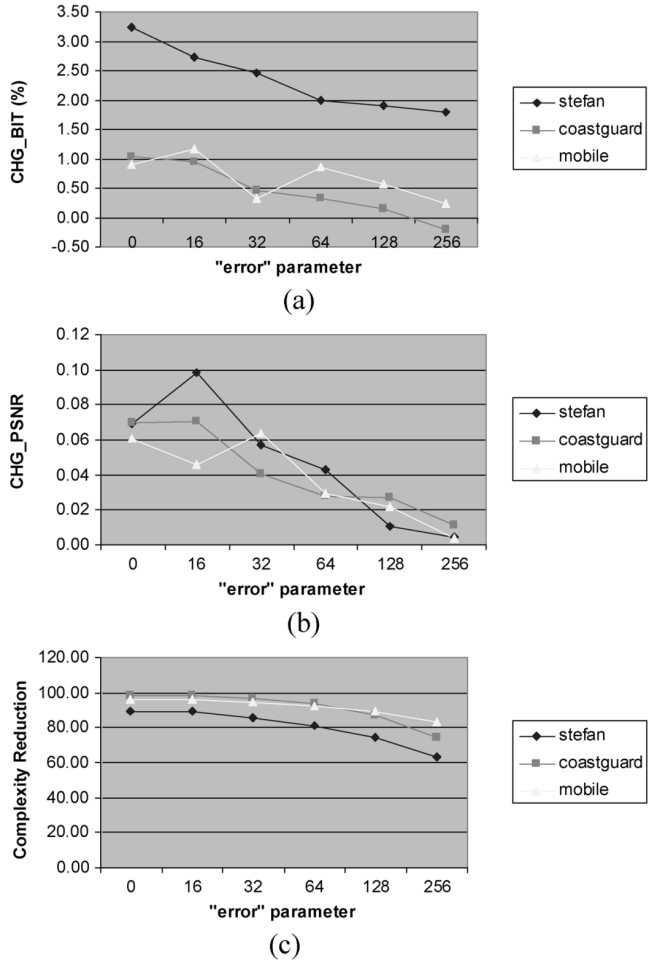


Fig. 3. Effect of "error" parameter variation on (a) generated bit rate, (b)PSNR, and (c) complexity reduction.

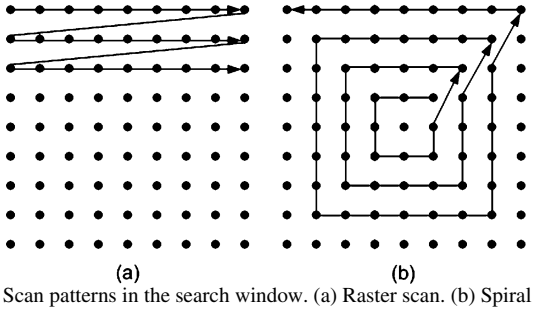


Fig. 4. Scan patterns in the search window. (a) Raster scan. (b) Spiral scan.

TABLE I
EXPERIMENTAL RESULTS FOR THE PROPOSED METHOD WITH $Q_P = 16$

	CHG_BIT (%)		CHG_PSNR		CHG_COMPLEXITY (%)	
Test sequence	raster scan	spiral scan	raster scan	spiral scan	raster scan	spiral scan
Foreman	1.77	1.04	-0.01	-0.01	-75.3	-86.1
Stefan	2.53	2.35	0.00	0.00	-76.2	-87.4
News	1.30	1.02	0.00	0.02	-74.9	-91.9
Coastguard	0.96	0.51	-0.01	-0.01	-78.4	-91.1
M&D	1.28	0.07	-0.04	0.01	-70.8	-86.6
Mobile	2.21	0.90	-0.01	-0.01	-81.8	-91.8
Total/Average	1.68	0.98	-0.01	0.00	-76.23	-89.15

(error). The reason to introduce the "error" parameter is explained as follows. Since we process every candidate MB line by line, each line consists of 16 pixels in the case of 16×16 block size. Some lines may generate large PSAD and others may generate low PSAD. The accumulated PSAD for the 16 lines will represent the SAD value for that candidate position. While comparing the PSAD for every line with the normalized $\mathrm{SAD_{TH}}$, it could be the case of PSAD $>$ $\mathrm{SAD_{TH}}$ for these lines and terminates the SAD process. Meanwhile, if we proceeded to compute the PSAD for the remaining lines (those lines may generate low PSAD), the total SAD for that candidate position is lower than the global min_SAD. To give the chance for those candidate positions with inhomogeneous PSAD distribution among the MB lines, we introduce an error margin. If the generated PSAD for those lines are less than [$\mathrm{SAD_{TH}}$+ "error"], we give them more chance to proceed in the SAD process. Otherwise terminate the process if the generated PSAD is larger than [$\mathrm{SAD_{TH}}$+ "error"].

The "error" parameter represents a margin of allowance for the candidate MB to pass the test in case that MB is similar or near to the target MB. The "error" parameter gives it the chance to continue the SAD test unless it really generates PSAD much greater than $\mathrm{SAD_{TH}}$. This "error" parameter should be decreased line by line by using "dec" parameter (decrement). Every time we process a new line we subtract "dec" from $\mathrm{SAD_{TH}}$. Thus, when processing the last line in the candidate MB, we will compare the true min_cost with the accumulated SAD for that position.

The value for "error" is set to the difference of 8 pixels. If this value increases, more accurate MV will be generated on the expense of increasing the SAD computations. On the other hand, lower "error" value will generate more error in the generated MV, but less SAD computations. Fig. 3 shows the effect of changing the "error" parameter on the generated bit rate, the peak signal-to-noise ratio (PSNR), and the reduction in complexity, applied on three test sequences Stefan, Coastguard, and

TABLE II
EXPERIMENTAL RESULTS FOR MPEG-4 WITH SPIRAL SCAN PATTERN, $Q_P = 16$, AND 4 MV ENABLED

	Bit rate			PSNR		
Test sequence	VM 18.0	Proposed	CHG_BIT (%)	VM 18.0	Proposed	CHG_PSNR
Foreman	1806074	1818842	0.71	31.8644	31.8641	-0.0003
Stefan	8023412	8157877	1.68	28.4128	28.4085	-0.0043
News	1109574	1113001	0.31	31.9215	31.9281	0.0066
Coastguard	3522424	3534403	0.34	29.1339	29.1293	-0.0046
M&D	668805	669789	0.15	34.0386	34.0355	-0.0031
Mobile	7941210	7970559	0.37	26.3518	26.3463	-0.0055
Total/Average	27349747	27521063	0.63			-0.0018
	Complexity			Timing (ME time)		
	VM 18.0	Proposed	CHG_COMPLEXITY(%)	VM 18.0	Proposed	CHG_Time (%)
Foreman	7.19E+08	2.02E+08	-71.919	114	58	-49.1
Stefan	8.02E+08	2.66E+08	-66.853	173	74	-57.2
News	1.48E+09	1.65E+08	-88.882	156	49	-68.6
Coastguard	8.37E+08	2.12E+08	-74.716	116	36	-69.0
M&D	1.38E+09	2.75E+08	-80.062	192	94	-51.0
Mobile	6.06E+08	1.39E+08	-77.081	117	32	-72.6
Total/Average	6.43E+09	1.42E+09	-77.914	868	343	-60.5

TABLE III
EXPERIMENTAL RESULTS FOR H.264 WITH SPRIRAL SCAN PATTERN, $Q_P = 28, \pm 32$ SR, AND RDO DISABLED

	Bit rate			PSNR		
Test sequence	JM	Proposed	CHG_BIT (%)	JM	Proposed	CHG_PSNR
Foreman	3223016	3240832	0.55	37.01	36.97	-0.04
Stefan	13694704	13710984	0.12	35.4	35.37	-0.03
News	2228896	2244720	0.71	38.09	38.07	-0.02
Coastguard	11279688	11231760	-0.42	34.52	34.51	-0.01
M&D	1382624	1382808	0.01	38.93	38.92	-0.01
Mobile	17495736	17458936	-0.21	33.76	33.74	-0.02
Total/Average	49304664	49270040	-0.07			-0.02
	Complexity			Timing (ME time)		
	JM	Proposed	CHG_COMPLEXITY (%)	JM	Proposed	CHG_Time (%)
Foreman	3.07E+10	1.66E+10	-45.89	1841	1111	-39.7
Stefan	5.43E+10	2.74E+10	-49.47	3686	1986	-46.1
News	2.83E+10	1.52E+10	-46.35	1694	1118	-34.0
Coastguard	6.78E+10	3.08E+10	-54.49	4165	1981	-52.4
M&D	2.86E+10	1.40E+10	-50.92	1697	1099	-35.3
Mobile	5.37E+10	2.60E+10	-51.59	3835	2044	-46.7
Total/Average	2.63E+11	1.30E+11	-50.60	16918	9339	-44.8

Mobile, which exhibit high and low motion. In Fig. 3(a), it is clear that as the "error" value increases, the change in bit rate will decrease, which means more accurate generated MV. In Fig. 3(b), when the "error" parameter increases the quality will be better and closer to that of the full search without termination. Finally, the complexity reduction variation due to the "error" parameter is shown in Fig. 3(c), in which more computations could be saved when the "error" parameter is smaller. The "dec" parameter is simply the result of dividing the "error" by the number of lines in each sub-MB according to each mode we are working on.

In summary, the advantage of our algorithm is that with the adaptive threshold value instead of the constant one allows us to skip more unnecessary SAD computation, as shown later in the simulation results.

III. EXPERIMENTAL RESULTS

In the following, we will show the simulation results of the proposed algorithm by integrating it into the MPEG-4 verification model V18.0 [3], and H.264 JM 9.0 [10]. All the following test sequences are in CIF format with 300 frames and $\pm$ 32 search range, unless otherwise specified.

TABLE IV
COMPARISON FOR DIFFERENT EARLY TERMINATION ALGORITHMS

	Ref. [8]	Ref.[9]	proposed
PSNR	-0.0033	-0.885	-0.0018
Speed up	1.19 – 2.82	18.7	4.52-65

Like other early termination algorithms, the efficiency of the proposed algorithm depends on the scan pattern in the search window, and the initial $\mathrm{SAD_{TH}}$ value that affects how fast we can reach the minimum SAD point and save more computations. In the following test, we test two scan patterns as depicted in Fig. 4, the spiral scan adopted by MPEG-4 VM18.0 and the normal raster scan used for most of the hardware implementations. The complexity saving is due to the adaptive threshold mechanism during the SAD accumulation. $\mathrm{SAD_{TH}}$ should be set to a value that ensures fast early termination and gives accurate results as well. When $\mathrm{SAD_{TH}}$ adapted to a lower

TABLE V
COMPARISON FOR DIFFERENT NONEARLY TERMINATION ALGORITHMS

	CHG_BIT			CHG_PSNR			CHG_COMPLEXITY		
Test sequence	GEA (Δ%) [5]	QME (Δ%) [6]	Proposed (%)	GEA (ΔdB)[5]	QME (ΔdB)[6]	Proposed (Δ dB)	GEA (Δ%) [5]	QME (Δ%) [6]	Proposed (Δ%)
Foreman	0.77	0.41	1.20	-0.0222	-0.0059	-0.0046	-66.41	-60.49	-83.88
Stefan	0.85	0.15	1.87	-0.0317	-0.0123	-0.0077	-69.40	-67.41	-85.37
News	0.45	0.34	0.67	-0.0384	-0.0176	-0.0062	-39.09	-66.86	-91.31
Coastguard	0.95	0.09	0.31	-0.0018	-0.0003	-0.0058	-72.92	-65.21	-87.92
M&D	0.67	0.48	0.61	-0.0665	-0.0234	0.0112	-58.99	-39.86	-85.56
Mobile	0.29	0.12	0.48	-0.0027	0	-0.0078	-73.24	-59.36	-90.17

value, more line skipping will occur. Only those MBs near to the lowest SAD point will consume more SAD computations because the error between those MBs and the current MB will be smaller, and thus many lines will pass the threshold detection. Setting $\mathrm{SAD_{TH}}$ to large value will slow down the termination process, especially for the raster scan method. In the following testing results, the first min_SAD is set to be $\mathrm{SAD_{pred}}$, which is the SAD calculated at the predicated MV position.

Table I shows the results for both scanning methods, relative to the search algorithm implemented in VM18.0 without any early termination. The comparison results are produced and tabulated according to the parameters as follows.

CHG_BITy	Change of bits used for the whole sequence.
CHG_PSNR	Change of PSNR.
CHG_COMPLEXITY	Change of SAD computations for the whole sequence.

It is clear that the spiral search pattern achieves better results in terms of total bits and PSNR, and also saves more SAD computations than the raster scan method. This is due to the raster scan pattern will pass by many local minimum, changing $\mathrm{SAD_{TH}}$ accordingly and may skip the targeted MB. This effect is shown clearly in the low and moderate motion test sequences, such as M&D and Mobile, where the MV is more likely to be near to the predicated position.

The proposed early termination method is also applied to the 8×8 subblock ME. Table II shows the performance results of the proposed algorithm compared with MPEG-4 VM 18.0 enabling the 4 MV option. Our method can save 77.91% of complexity with negligible quality loss. The last column shows the time consumed by the ME subroutine to find the MV. This is computed with the *time* function implemented by Visual C++ 6.0, on a P4 2 GHz, 768 MB memory and windows XP system.

The same tests are applied to H.264. Table III shows the results for the proposed method compared to the early termination method employed in JM 9.0. The proposed method can save 50.6% of complexity with similar quality. The difference in percentage between the time saving and the complexity saving is due to the accuracy of the *time* function, since many search positions will be skipped after few lines tests, and the *time* function will not capture this time difference. The difference appears more in H.264 since it works with small block sizes, and more error in capturing the time.

The result for H.264 shows less complexity reduction and more PSNR degradation but less bit rate increase compared to MPEG-4. This is due the variable block size ME of H.264. The small block size tends to consume more SAD computations compared to that of large block size. For example, when testing 4×4 block, at least we need to go through one line test, which in total equivalent to 4 lines test in a 16×16 block. Meanwhile, if the tests only applied on 16×16 block, we can stop SAD computations after the first line test. The PSNR degradation is due to reducing the chances to select modes with smaller size blocks, which gives better quality when selected. When dealing with small size blocks, the "error" parameter also will be small, hence early termination will occur frequently, and will result in ignoring more small size blocks from the mode selection later by H.264.

IV. PERFORMANCE COMPARISONS

The comparison hereafter is based on the SAD computational complexity and the PSNR value, when compared with other early termination algorithms such as [8] and [9]. The algorithm presented in [8] is considered as lossless due to no degradation in PSNR with complexity reduction of 16.6% compared to PDS. However, when it employs the early jump out technique it will result in PSNR reduction of (-0.033 dB) in average and 64.5% complexity reduction. While the algorithm presented in [9] can achieve a speedup of 18.7 times compared to PDS by testing one pixel at a time, which results in a high PSNR degradation of (-0.885 dB). The presented algorithm is superior to the above two algorithms in terms of negligible PSNR loss (in average -0.0018 dB), and also in higher complexity reduction (77.914% compared to PDS). Comparing to [9], our algorithm can achieve a theoretical speedup of 65 times by testing 4 pixels at a time rather than 16 pixels (skipping the SAD computations after testing 4 pixes only). Table IV summarizes this comparison.

For other nonearly termination algorithms, Table V presents a comparison between our proposed algorithm with recently proposed GEA and QME. It is clear that the proposed method can save more computations with negligible quality loss.

The simplicity and regularity of the proposed algorithm makes it suitable for hardware implementation, with low control circuitry, regular data flow, and no special circuitry to read pixels in certain form (as Hilbert Scan or different block patterns employed in [9]). Also it can be integrated with other fast algorithms such as QME to achieve more computation reduction since its PSNR degradation is very low.

V. Conclusion

In this paper, we present a simple but efficient early termination method. It reduces the complexity of ME module by skipping the unnecessary SAD computations with adaptively changed threshold. Both MPEG-4 and H.264 can benefit from this method. The reduction in SAD computations achieved saving of 77.9% and 50.6 % of that in MPEG-4 VM18.0 and H.264 JM9.0, respectively, with negligible quality loss.

References

[1] *Draft ITU-T Rec. Final Draft International Standard of Joint Video Specification*, JVTG050, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Joint Video Team (JVT) of ISO/IEC MPEG&ITU-T VCEG.

[2] *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification*, ITU-T Rec. H.264/ISO/IEC 14 496-10 AVC, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, JVT-G050, 2003.

[3] *MPEG-4 Video Verification Model Version 18.0*, ISO/IEC JTC1/SC29/WG11 N3908, Jan. 2001.

[4] T. Koga, "Motion-compensated interframe coding for video conferencing," in *Proc. Nat. Telecommun. Conf.*, Nov./Dec. 1981, pp. G 5.3.1–G 5.3.5.

[5] Y. W. Huang, S. Y. Chien, B. Y. Hsieh, and L. G. Chen, "An efficient and low power architecture design for motion estimation using global elimination algorithm," in *Proc. Acoustics, Speech, Signal Process.*, May 2002, vol. 3, pp. 3120–3123.

[6] K. B. Lee, H. Y. Chin, H. C. Hsu, and C. W. Jen, "QME: An efficient subsampling-based block matching algorithm for motion estimation," *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 2, pp. II-305–II-308, May 2004.

[7] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *IEEE Trans. Image Process.*, vol. 4, no. 1, pp. 105–107, Jan. 1995.

[8] Y. L. Chan and W. C. Siu, "An adaptive partial distortion search for block motion estimation," *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process.*, vol. 3, pp. 153–156, Apr. 2003.

[9] C. H. Cheung and L. M. Po, "Adjustable partial distortion search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 1, pp. 100–110, Jan. 2003.

[10] JVT Reference Software Ver. 9.0. [Online]. Available: http://bs.hhi.de/~suehring/tml/download/