A Content Adaptive Fast PDE Algorithm for Motion Estimation Based on Matching Error Prediction

Sangkeun Lee and Eunjeong Park

Abstract: **This paper introduces a new fast motion estimation based on estimating a block matching error (i.e., sum of absolute difference (SAD)) between blocks which can eliminate an impossible candidate block much earlier than a conventional partial distortion elimination (PDE) scheme. The basic idea of the proposed scheme is based on predicting the total SAD of a candidate block using its partial SAD. In particular, in order to improve prediction accuracy and computational efficiency, a sub-sample based block matching and a selective pixel-based approaches are employed.**

In order to evaluate the proposed scheme, several baseline approaches are described and compared. The experimental results show that the proposed algorithm can reduce the computations by about 44% for motion estimation at the cost of 0.0005 dB quality degradation versus the general PDE algorithm.

Index Terms: **Block matchng, motion estimation (ME), partial distortion elimination (PDE), sub-sampling.**

I. INTRODUCTION

Motion estimation (ME) is defined by finding motion vectors referring pixels from reference(s) to a current frame (detail review reports can be found in [1]–[3]). The ME is widely used to compress the motion pictures for removing the temporal redundancies effectively. Therefore, many researchers have been focused on the approaches to find motion vectors fast and accurate while they keep the quality of pictures when they are decoded. The so-called "block-matching algorithms" are the most important of estimation methods, especially in coding schemes based on discrete cosine transform including MPEG-x and H.26x.

A full search (FS) algorithm [4] is an optimal block matching method to find best motion vectors but it requires heavy computations. Therefore, several alternative and faster techniques have been developed to reduce computational complexity by pruning the search space (fast searching) with respect to the FS method or reducing the number of pixels considered (fast matching) during block matching between original and candidate blocks. In this paper, we only focus on fast matching algorithms.

In fast matching algorithms, the lossy approach aims at reducing the sum of absolute difference (SAD) computation time for each candidate block by testing only part of pixels in the block at the cost of coding quality [5]–[7]. Recently, a prediction-based algorithm has been introduced [8] and reported that it could save the computation cost considerably. However, it is not guaranteed that the prediction is correct because a prediction at a candidate block depends on its neighboring blocks instead of its own contents. The lossless fast matching approach is more attractive because it is fast processing without any loss in quality of the coded image. Partial distortion elimination (PDE) [9] is a good example of the fast matching scheme. Main advantage of this approach is to discard impossible candidates earlier before complete SAD computation for the whole candidate blocks. For this efficiency, many improved approaches [10]–[12] have been proposed and reported good results. More detailed descriptions can be found in Section IV. However, the improved PDE approaches still requires heavy load for SAD computations and time to eliminate a candidate block at the final stage if it is very similar to a original block but not the optimal block.

This paper is based on a previous work [8] of a prediction-based fast motion estimation. However, we employ two new main factors which are a sub-sampling based error prediction and a selective pixel-based error comparison for improving the prediction accuracy but reducing the computational cost, respectively. A sub-sampling based block, which is a subblock, in this paper consists of the pixels values selected from the sub-sampled pixel locations in a given block. The sub-sampling scheme is used for stably estimating a matching error prediction that may be caused by performing sequential neighbor pixels, lines, or sub-blocks in a block. And the selective pixel-based matching error comparison is used for avoiding the unnecessary computations with line or block-based comparisons. In this paper, we focus on lossy matching procedures even though their quality degradation is negligible and introduce a method to reduce computations for a conventional PDE algorithm without significant changes.

The main advantages of this scheme are that 1) it can eliminate an impossible candidate block even though a total SAD between an original and a candidate blocks is not completed; 2) the algorithm can reduce the computational cost considerably for SAD calculation; and 3) conventional approaches can be improved by using the proposed scheme without changing the algorithms significantly.

In this paper, conventional PDE approach for ME is reviewed in Section II. In Section III, an error prediction based PDE algorithm for improving the prediction accuracy and for speeding up the PDE is introduced. Competitive experimental results are illustrated and compared with four baseline systems including our previous approach in Section IV. And we conclude the fast PDE scheme in Section V.

II. BACKGROUND

The basic concepts used in this paper are briefly reviewed. More details can be found in [8].

Manuscript received June 20, 2008; approved for publication by Hanseok Ko, Division I Editor, July 14, 2009.

This research was supported by the Chung-Ang University Research Scholarship Grants in 2009.

The authors are with the Graduate School of Advanced Imaging Science, Multimedia & Film, Chung-Ang University, Seoul, Korea, email: sangkny@cau.ac.kr, tomatonim@hanmail.net.

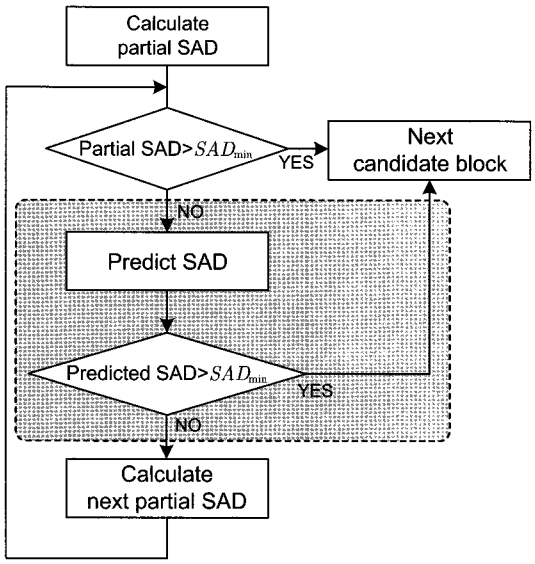

Fig. 1. Overall procedures for PDE and the proposed algorithms.

A. Motion Estimation

ME processing for an original block, whose size in this paper is $N \times N$, located at any position $p = [p_x \ p_y]^T$ in an image is to find out the best matching candidate block located at $mv = [mv_x \ mv_y]^T$ in a given search range $\boldsymbol{R}$ in another image by using SAD criterion as follows:

$$SAD(p, mv') = \min_{mv \in \boldsymbol{R}} SAD(p, mv) \tag{1}$$

where mv' is an optimal motion vector in $\boldsymbol{R}$, and the SAD is expressed by

$$\begin{aligned} SAD(p, mv) &= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |f_t(i + p_x, j + p_y) \\ &\quad - f_{t-1}(i + p_x + mv_x, j + p_y + mv_y)| \\ &= \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |f_t^p(i, j) - f_{t-1}^{p+mv}(i, j)| \end{aligned} \tag{2}$$

where f_t and f_{t-1} indicate current and reference frames, respectively.

B. PDE Algorithm

A block diagram without blocks in the solid dashed box is illustrated for a conventional PDE algorithm in Fig. 1. It is noted that this paper follows a spiral scanning path for searching a matching block [10] and sets the SAD of a candidate block located at the center of an given search range ($mv = [0 \ 0]^T$) to the initial matching error (SAD) because the SAD has a high possibility to be a minimum error $SAD_{\min}$. The partial SAD accumulated up to kth line of a block by using the spiral scanning order can be defined by

$$pSAD^k(p, P(mv)) = \sum_{i=0}^{k} \sum_{j=0}^{N-1} \left| f_t^p(i, j) - f_{t-1}^{p+P(mv)}(i, j) \right| \tag{3}$$

where $P(mv)$ is a reordered location according to the spiral scanning path. Therefore, the ME processing with PDE approach can be formulated by

$$SAD(p, mv') = \min_{mv \in \boldsymbol{R}} pSAD^{N-1}(p, P(mv)). \tag{4}$$

The $SAD_{\min}$ is updated if necessary until a given search range $\boldsymbol{R}$ is scanned completely for an original block.

III. PREDICTION BASED FAST PDE ALGORITHM

The disadvantage of the conventional PDE scheme is that the algorithm has to compute partial SAD of an impossible candidate block, which will be eliminated, until the SAD reaches the minimum SAD. It is obvious that predetermination of the elimination will help in reducing the computations and in accelerating the speed of ME. For this purpose, we proposed a prediction scheme of the total block matching error $bSAD$ from the partial SAD $pSAD^k$ by using Taylor series expansion [8]. The prediction scheme is based on the relationship between a partial block SAD and its total block SAD that the partial SAD has a random value and is gradually increased to the total block SAD as the computation goes [8], [10]. First, we estimated a partial SAD $pSAD^n$ at the nth line by using the partial SAD $pSAD^k$ accumulated up to the kth line of original and candidate blocks as follows:

$$\begin{aligned} pSAD^n(p, P(mv)) \approx & pSAD^k(p, P(mv)) \\ & + \frac{\partial pSAD^k(p, P(mv))}{\partial k}(n - k) \end{aligned} \tag{5}$$

for $0 < k \leq n \leq N-1$, and an original and candidate blocks are located at p and $p + P(mv)$ in a current and a reference frames, respectively. It is easily seen that the proposed prediction depends on the gradient of the partial SAD. (5) indicates that the prediction has a high probability to reach the $SAD_{\min}$ faster and to save the SAD computations when the gradient of a $pSAD^k$ is big. However, a big SAD represents a big block matching error and may result in degrading the accuracy of the proposed scheme because the prediction scheme in (5) is not performed correctly with the erroneous information. Therefore, the second term on the right side of the above align needs to be adjusted adaptively according to the contents and complexity of its original block. In order to approximate the contents of a block even at the beginning of a prediction, we replace the block matching scheme in (3) with a sub-sampling based scheme as follows:

$$\begin{aligned} & pSAD^k(p, P(mv)) \\ & = \sum_{i,j=0}^{k=i+4j} \sum_{u,v=0}^{M} \left| f_t^p(i + 4u, j + 4v) - f_{t-1}^{p+P(mv)}(i + 4u, j + 4v) \right| \end{aligned} \tag{6}$$

where M is $N/4 - 1$, a block is separated into 16 sub-sampled blocks, and i and j indicate a starting point of each sub-sample block for $0 \leq i, j \leq M$. It is noted that any additional operations for sub-sampling based matching is not required since the locations of each pixel can be predetermined. For more robust estimation, we also reflect the neighboring SADs. Finally, the

content-reflected SAD estimation for a whole block p_bSAD is defined with a weighting factor w by

$$\begin{aligned} & p\_bSAD(p, P(mv)) \\ =& pSAD^{N-1}(p, P(mv)) \\ \approx& pSAD^{k}(p, P(mv)) + \frac{\partial pSAD^{k}(p, P(mv))}{\partial k}(N-1-k)w \end{aligned} \tag{7}$$

for $0 < k \leq N-1$. The weighting factor is employed for taking the neighboring blocks' complexities into account and it is computed by

$$w = g(\varpi),\ \varpi = \frac{1}{5}\sum_{i=0}^{4}\omega_i \tag{8}$$

where ϖ is an average value of neighboring $SAD_{\min}$s including an initial SAD ω_0 at the current location p. The neighborhood consists of available upper left ω_1, upper ω_2, upper right ω_3 and left block ω_4. And the weighting function $g(\chi)$ is defined by

$$g(\chi) = \begin{cases} 1, & \text{if } \chi < \tau_1 \\ \frac{-1}{\tau_2-\tau_1}(\chi-\tau_1)+1, & \text{for } \tau_1 \leq \chi < \tau_2 \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

Here, parameters τ_1 and τ_2 indicate the first and second thresholds of a fuzzy-like function, respectively. It is noted that the estimated SAD without considering neighboring blocks may be unstable especially around object boundaries.

Algorithm: We present the whole procedures of the proposed algorithm in Fig. 1.

1) We compute a SAD between original and candidate blocks at the location $P(mv) = [0\ 0]^T \in \boldsymbol{R}$ using (2), and set the SAD initially to the minimum SAD $SAD_{\min}$. Next, a weighting factor w in (8) is obtained from the initial SAD and four neighboring available SADs. Then, the algorithm goes to Step 2 for the next candidate block on a spiral scanning path by setting k to zero ($k = 0$).
2) We perform PDE for computing a partial SAD $pSAD^k$ of the block using (6) and compare the $pSAD^k$ with the $SAD_{\min}$ to find out the smallest matching error using (4). If the $pSAD^k$ is bigger than the $SAD_{\min}$, then the current candidate block is skipped and the algorithm repeats this step for the next candidate block, or it goes back to Step 1 for the next original block when it finishes the given search range. Otherwise, it goes to Step 3.
3) We predict a total block SAD p_bSAD using (7). Next, we compare the p_bSAD with the $SAD_{\min}$. If the p_bSAD is bigger, then the algorithm goes to Step 2 for the next candidate block. Otherwise it goes to Step 2 with $k = k+1$. When it reaches the end of the matching range, the $SAD_{\min}$ is replaced with the partial SAD $pSAD^k$ (actually, now it is the total SAD between the original and the candidate blocks), and the algorithm goes to Step 1 for the next available original block.

It is worth noting that each weighting factor w for each original block is computed once during an initial minimum SAD computation for the block (see more details for a practical example in [8]).

More computational improvement: The most critical part of PDE approaches is to find out earlier if a candidate block can be rejected or not. For a simple explanation, we describe how to save more computations with a conventional PDE because it can be easily extended to the proposed sub-sampled matching scheme. Basically, from (3), the matching error (SAD) computation is line-by-line basis. It means that we can not determine until the computation is completed for a line even though a $pSAD^k$ is larger than $SAD_{\min}$ during the computation. It is true that the $pSAD^k$ is getting increased as the computation goes. Therefore, we can modify (3) with the following pixel-based comparison

$$\begin{aligned} & pSAD^{k,k_1}(p, P(mv)) \\ &= \sum_{i=0}^{k}\sum_{j=0}^{k_1}\left| f_t^p(i,j) - f_{t-1}^{p+P(mv)}(i,j)\right| \end{aligned} \tag{10}$$

for $0 \leq k, k_1 \leq N-1$. Similarly, (6) is formulated as follows:

$$\begin{aligned} & pSAD^{k}_{k_1,k_2}(p, P(mv)) \\ &= \sum_{i,j=0}^{k=i+4j\,k_1,k_2}\ \sum_{u,v=0} |f_t^p(i+4u, j+4v) - f_{t-1}^{p+P(mv)}(i+4u, j+4v)| \end{aligned} \tag{11}$$

for $0 \leq k_1, k_2 \leq M$. However, a problem in the pixel-based comparison scheme is that this requires a comparison operation every pixel and gives a big computational burden to the algorithm. In order to solve the problem, we perform a selective pixel-based comparison only when the complexity measure ϖ of an original block is small enough (which is less than $\xi = \frac{1}{3}\tau_1$ in practice). This strategy results in achieving more efficiency as will be shown in Section IV for both the conventional PDE and the proposed algorithms. To apply this procedure, we change the matching error comparison scheme in Step 2 of the proposed algorithm described above as follows:

2)* If the block complexity measure ϖ of the given original block is bigger than the $\xi = \frac{1}{3}\tau_1$, we follow the procedures described in Step 2 of the sub-block-by-sub-block comparison. Otherwise, we perform PDE for computing a SAD of the block using (11) and compare the SAD at each pixel location of the block with the $SAD_{\min}$ to find out the smallest matching error using (4). If the SAD is bigger than the $SAD_{\min}$, then the current candidate block is skipped and the algorithm repeats this step for next candidate block, or it goes back to Step 1 for the next original block when it finishes the given search range. Otherwise, it goes to Step 3 when the current sub-sampled block is finished ($k_1, k_2 = M$), or repeats this step for the next pixel of the sub-sampled block.

Using this selective pixel-based comparison scheme in a sub-sampled block, we can save about 12% computations more that will be shown in experimental results with the current settings. It is obvious that we may save computations and process time more with the selective pixel-based matching error comparison as macro-block size and search range are getting bigger and bigger.

IV. EXPERIMENTS

This section performs a series of experiments to demonstrate the performance of the proposed algorithm and also compares the results with those generated by other baseline algorithms. In addition, the proposed scheme is applied to the conventional PDE algorithm, and we show that it can save the computational cost without significant changes.

The proposed algorithm is simulated with various video sequences—Akiyo, Container, Foreman, Mobile, News, Silent voice, Coast guard, and Mother&daughter—and they consist of 300 frames at 30 Hz in the format of QCIF. In these sequences as summarized in Table 1, Foreman, Mobile, and Coast guard have big motions compared with other sequences whereas akiyo and container are almost static sequences, and the remaining three sequences, News, Silent voice, and Mother&daughter, have intermediate motions. The used searching method is spiral scanning, the matching block size is 16×16 pixels, and the search range is ± 7. By default, the parameters τ_1 and τ_2 in computing weighting factor in (9) were empirically set to $300/(16 \times 16)$ and $900/(16 \times 16)$, respectively, for consistency over all of the tested sequences. In particular, SAD value was normalized by the size of a block, and only the luminance component was used in this experiment.

A. Existing Methods

In order to evaluate the performance of the proposed algorithm, three other existing PDE algorithms [4], [11], [12] and our previous method [8] are briefly described and are compared with the proposed approach.

The first algorithm is a conventional PDE algorithm [4] that can reject invalid candidate blocks earlier. SAD computation is performed by (3).

The second baseline system is a fast PDE algorithm [11] based on an adaptive matching scan by sorted sub-blocks. The algorithm assumes that localization of image complexity with small square sub-blocks can get a faster elimination of impossible candidates. The local complexity of the sub-block is defined as a spatial complexity of image data for each sub-block and measured with gradient magnitude.

A main difference between the sub-block in [11] and the sub-sampled blocks in the proposed algorithm is that the sub-block is a measure of the local spatial complexity while the sub-sampled block is that of the global contents of a total block.

The third baseline algorithm is another fast PDE algorithm [12] based on the reorder of pixels in a reference block according to some measures at each pixel location. In [12], authors proposed two algorithms which are called fast full search with sorting by distortion (FFSSD) and fast full search with sorting by gradient (FFSSG), respectively, in order to quickly discard invalid blocks. It is reported that FFSSG had better results in [12]. Therefore, we only evaluate the FFSSG scheme and report its results.

The common procedure used in [11] and [12] is that the two approaches reorder the matching scan order according to image complexities. The main differences between two algorithms are that: 1) The sorting is performed on the magnitude of the gradient of sub-blocks and on that of eight-neighboring pixels at each pixel, respectively, in an original block at p; and 2) the comparison between an original at p and a candidate at $p + P(mv)$ blocks is operated every 16 pixels (the size of a sub-block) and is operated every 8 pixels, respectively. It is noted that *counting sort* [13] is employed for a reordering the gradient magnitude as mentioned in [12].

In order to evaluate the fast PDE techniques, we first compare the result of the proposed algorithm with the baseline approaches including our previous scheme with respect to computational costs, speed, and peak signal-to-noise ratio (PSNR). Then, we show how much the baseline algorithms can be improved by the proposed sub-sampled block and selective pixel-based comparisons.

B. Experimental Results

In this experiment, 'PDE [4],' 'SBPDE [11],' 'FFSSG [12],' 'Prediction [8],' and 'Proposed' are indicating the results of the first, second, third baseline algorithms, our previous prediction-based approach, and the proposed algorithm, respectively. It is noted that the SBPDE and FFSSG algorithms require a gradient and a sorting computations of sub-blocks and neighboring pixels, respectively, for each original block. However, we considered each occurred operation such as a multiplication as an addition for simple comparison and expression. For an example, a gradient computation with four neighborhood requires four additions, four subtracts, and one division but we counted all of the operations as nine additions in this paper. And the proposed scheme requires one addition, one division, and two multiplications to estimate a total block matching error if necessary for the sub-sampled block based comparison and requires one more comparison operation per pixel for the selective pixel-based comparison scheme. Table 2 shows the required average number of computations per pixel for each baseline algorithm and the proposed algorithm. Four systems in the table show the line or sub-block (every 8 or 16 pixels) based SAD computational performances while 'Proposed' is the result of the proposed sub-sampling-based selective pixel comparison approach. It is noted that a parenthesis if any in each column for each sequence indicates a computational efficiency η to the conventional PDE algorithm. The efficiency of the compared approach versus the conventional PDE algorithm is defined by

$$\eta = \frac{\text{PDE} - \text{CPDE}}{\text{PDE}} \times 100 \tag{12}$$

where PDE and CPDE are the conventional PDE and the compared algorithms, respectively. For an example, the result value 16.461 of SBPDE for the sequence #1 is the required average operations per pixel to compute SAD while the value 4.230% in a parenthesis is an efficiency to the PDE algorithm. It is easily seen that the proposed algorithm can reduce the computational cost by about 44%, 32%, 17%, and 10% versus PDE, SBPDE, FFSSG, and prediction, respectively. It is interesting that sequences containing complicated contents affects the performance of sorting-based approaches more while the proposed scheme is effective regardless of the scene contents. It is because the proposed selective pixel comparison scheme is more efficient in the static sequences while the sub-sampled block based matching scheme is more effective in the complicated

Table 1. A set of test sequences.

Number	Sequence	Motion	Number	Sequence	Motion
#1	Akiyo	static	#5	News	intermediate
#2	Container	static	#6	Silent voice	intermediate
#3	Foreman	big	#7	Coast guard	big
#4	Mobile	big	#8	Mother&daughter	intermediate

Table 2. Result comparisons with respect to computational costs per pixel and efficiency versus the conventional PDE.

Sequence	Computations/pixel and efficiency (η)				
	PDE [4]	SBPDE [11]	FFSSG [12]	Prediction [8]	Proposed
#1	17.188	16.461 (4.230)	15.771 (8.244)	13.675 (20.439)	9.523 (44.596)
#2	31.817	26.184 (17.704)	23.378 (26.524)	15.538 (51.165)	14.381 (54.800)
#3	53.883	45.745 (15.103)	41.928 (22.187)	39.613 (26.484)	33.667 (37.518)
#4	46.677	39.870 (14.583)	35.005 (25.006)	30.458 (34.747)	24.291 (47.959)
#5	25.882	21.064 (18.615)	19.496 (24.674)	19.280 (25.510)	14.755 (42.989)
#6	27.750	27.233 (1.863)	26.353 (5.034)	21.187 (23.650)	20.847 (24.876)
#7	42.977	37.804 (12.037)	34.684 (19.296)	27.200 (36.711)	24.488 (43.020)
#8	41.347	34.184 (17.324)	30.494 (26.249)	19.607 (52.579)	17.005 (58.873)
Average	35.940	31.068 (12.682)	27.637 (19.652)	23.320 (33.911)	19.870 (44.329)

Table 3. Encoding time speed up versus the conventional PDE.

Sequence	Encoding time per picture [msec] and efficiency (η)				
	PDE [4]	SBPDE [11]	FFSSG [12]	Prediction [8]	Proposed
#1	1.465	1.361 (7.099)	3.763 (−156.860)	1.254 (14.403)	1.151 (21.433)
#2	2.508	1.987 (20.774)	5.538 (−120.813)	1.411 (43.740)	1.565 (37.600)
#3	4.027	3.344 (16.961)	8.987 (−123.169)	3.502 (13.037)	3.552 (11.795)
#4	3.552	2.926 (17.624)	7.893 (−122.213)	2.823 (20.524)	2.666 (24.944)
#5	2.040	1.672 (18.039)	4.756 (−133.137)	1.722 (15.588)	1.672 (18.039)
#6	2.144	2.090 (2.519)	6.060 (−182.649)	1.933 (9.841)	1.913 (10.774)
#7	3.291	2.823 (14.221)	7.682 (−133.424)	2.508 (23.792)	2.532 (23.063)
#8	3.187	2.562 (19.611)	6.900 (−116.505)	1.776 (44.274)	1.796 (43.646)
Average	2.777	2.346 (14.606)	6.447 (−136.096)	2.116 (23.150)	2.106 (23.912)

sequences. Therefore, in order to see how much the proposed schemes can affect the performance of the common PDE algorithms, we apply these two schemes without a prediction to the conventional PDE. The enhanced PDE, which is lossless as well, outperform the conventional PDE by about 12% computational efficiency.

In order to evaluate how much each algorithm can speed up the picture encoding time versus the conventional PDE algorithm, the elapsed time for encoding each sequence is shown in Table 3. It is seen that the proposed algorithm outperforms the PDE and the SBPDE algorithms by about 23% and 9%, respectively, in the picture coding time. The encoding time of FFSSG takes longer than those of PDE and SBPDE approaches even though it has less computations per pixel as in Table 2. It is realized that most of its encoding time is consumed to compute the gradients with eight neighbors at each pixel of a block and to sort them according to their contributions to SAD. In order to resolve these problems, authors in [12] tried to optimize those procedures by using bit-operations and hashing tables, respectively. However, we did not consider those optimizations in our implementation for the given comparison.

Now, a coding quality evaluation is performed on the results of pictures coded by the proposed algorithm, which is a lossy PDE approach even though the degradation is negligible as illustrated below. Table 4 shows a quality measure PSNR, which is the left half of the table, and the matched motion vector ratio to the PDE algorithm, which is in the right half of the table. In Table 4, 'Matched MV' indicates the percentage that matches the motion vector (MV) of the proposed algorithm to that of the conventional PDE approach. From the observation of the table, we can see that the proposed algorithm keeps almost same quality as the PDE algorithm even though there is ignorable degradation -0.0005 dB on average, and the motion vectors of the proposed algorithm are matched 99.47% average for the set of test sequences.

V. SUMMARY AND DISCUSSION

This paper proposed a scheme can reduce the computational costs while it can keep the image quality. In this paper, we estimated a total block SAD with using the partial SAD, determined if a candidate block can be eliminated earlier than the conventional PDE, and saved the computation cost considerably. For the estimation, we formulated the block matching process in a

Table 4. Comparisons with respect to PSNR and matched motion vectors to the PDEs.

Sequence	PSNR [dB]		Matched MV[%]
	PDE	The proposed	The proposed versus PDE
#1	44.296	44.296 (0.0000)	100.0000
#2	43.027	43.027 (−0.0001)	98.8109
#3	32.064	32.063 (−0.0009)	98.6082
#4	26.009	26.009 (0.0000)	100.0000
#5	36.235	36.234 (−0.0007)	99.9291
#6	35.003	35.002 (−0.0003)	99.9628
#7	32.483	32.483 (0.0000)	100.0000
#8	41.156	41.154 (−0.0023)	98.4157
Average	36.284	36.283 (−0.0005)	99.4658

block with Taylor series expansion, predicted a total block SAD from a partial block SAD. This paper introduced and combined two schemes based on a sub-sampled block and a selective pixel-to-pixel comparison schemes for the total block error prediction, and we could reduce the computational costs by 44% versus the conventional PDE algorithm for fast motion estimation at the cost of negligible image quality degradation −0.0005 dB. And we also verified that the proposed scheme can be applied to the conventional PDE without any significant changes and can improve the computational efficiency by about 12% on average. Therefore, we believe that the proposed algorithm can be an effective tool for the fast motion estimation through the experimental results.

REFERENCES

[1] C. Stiller, "Estimating motion in image sequences: A tutorial on modeling and computation of 2D motion," *IEEE Signal Process. Mag.*, pp. 70–91, July 1999.

[2] J. K. Aggarwal and N. Nandhakumar, "On the computation of motion from sequences of images—A review," *Proc. IEEE*, vol. 76, no. 8, Aug. 1998.

[3] F. Dufaux and F. Moscheni, "Motion estimation techniques for digital TV: A review and a new contribution," *Proc. IEEE*, vol. 83, no. 6, pp. 858–876, June 1998.

[4] S. Eckart and C. Fogg, "Iso/iec mpeg-2 software video codec," in *Proc. SPIE*, vol. 2419, 1995, pp. 100–118.

[5] K.-T. Choi, S. C. Chan, and T. S. Ng, "A new fast motion estimation algorithm using hexagonal subsampling pattern and multiple candidates search," in *Proc. Int. Conf. Image Process.*, Sept. 1996, pp. 497–500.

[6] A. C. K. Ng and B. Zeng, "A new fast motion estimation algorithm based on search window sub-sampling and object boundary pixel block matching," in *Proc. Int. Conf. Image Process.*, Oct. 1998, pp. 605–608.

[7] Y. Wang, Y. Wang, and H. Kuroda, "A globally adaptive pixel-decimation algorithm for block-motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 6, pp. 1006–1011, Sept. 2000.

[8] S.-I. Shin, S. Lee, and J.-S. Oh, "Fast partial difference elimination algorithm based on block matching error prediction," *Optical Engineering Lett.*, vol. 46, no. 4, pp. 1–5, Apr. 2007.

[9] *ITU-T Recommendation H.263 Software Implementation*. Digital Video Coding Group, Telenor R&D, 1995.

[10] J.-N. Kim and T.-S. Choi, "A fast full-search motion estimation algorithm using representative pixels and adaptive matching scan," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, Oct. 2000.

[11] J.-N. Kim, Y.-H. Kim, and B.-H. Ahn, "Fast full search motion estimation algorithm using early detection of impossible candidate vectors," *IEEE Trans. Signal Process.*, vol. 50, no. 9, pp. 2355–2365, Sept. 2002.

[12] B. Montrucchio and D. Quaglia, "New sorting-based lossless motion estimation algorithm and a partial distortion elimination performance analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, Feb. 2005.

[13] T. H. Musmann, P. Pirsch, and H. J. Grallert, *Introduction to Algorithms*. MA/New York: MIT Press/McGraw-Hill, 1990.

Sangkeun Lee received the B.S. and M.S. degrees in Electronic Engineering from Chung-Ang University, Seoul, Korea, in 1996 and 1999, respectively, and the Ph.D. degree in Electrical and Computer Engineering from Georgia Institute of Technology, Atlanta, GA, in 2003. He is an Assistant Professor in the Graduate School of Advanced Imaging Science, Multimedia & Film at Chung-Ang University, Seoul, Korea. From 2003 to 2008, he was a Staff Research Engineer with the Digital Media Solutions Lab, Samsung Information Systems America, Irvine, CA, where he was involved in the development of video processing and enhancement algorithms (DNIe) for Samsung's HDTV. His current research and development interests include digital video and image processing, especially video analysis/synthesis, denoising, compression for HDTV and multimedia applications, and content-based video representation and abstraction. He is a Member of the IEEE.

Eunjeong Park received the B.S. degree in Multimedia and B.F.A. degree in Department of Communication Design from Yeosu National University, Yeosu, Korea, in 2005. She is currently pursuing for her Master degree in Image Engineering-Technology Art at Chung-Ang University. Her current research interests include digital media design, public art, especially UX, network-based interaction, and real-time data visualization.