Transactions Letters

Normalized Partial Distortion Search Algorithm for Block Motion Estimation

Chok-Kwan Cheung and Lai-Man Po

***Abstract*—Many fast block-matching algorithms reduce computations by limiting the number of checking points. They can achieve high computation reduction, but often result in relatively higher matching error compared with the full-search algorithm. In this letter, a novel fast block-matching algorithm named normalized partial distortion search is proposed. The proposed algorithm reduces computations by using a halfway-stop technique in the calculation of block distortion measure. In order to increase the probability of early rejection of non-possible candidate motion vectors, the proposed algorithm normalized the accumulated partial distortion and the current minimum distortion before comparison. Experimental results show that the proposed algorithm can maintain its mean square error performance very close to the full search algorithm while achieving an average computation reduction of 12–13 times, with respect to the full-search algorithm.**

I. Introduction

Motion compensation is a vital component of many video-coding standards (e.g., ISO MPEG-1/2 [1], [2] and ITU-T H.261/262/263 [2]–[4]) due to its high efficiency in reducing temporal redundancy between successive frames. Block-based motion estimation is the most popular method to obtain motion-compensated prediction. By dividing each frame into rectangular blocks of equal size, the motion estimator obtains a motion vector for each of the blocks within a search window in the reference frame using the block-matching algorithm (BMA). The full-search algorithm (FS) is the most straightforward BMA, which provides an optimal solution by matching all the candidate blocks inside a search window. However, the computational complexity of FS is always too high for real-time implementation. A number of fast algorithms are developed to reduce the computational complexity of motion estimation. Some of the famous examples include the three-step search (3SS) [5], cross search algorithm (CSA) [6], orthogonal search algorithm (OSA) [7], 2-D-logarithmic search (2DLOG) [8], new three-step search (N3SS) [9], and four-step search (4SS) [10] algorithms. These algorithms greatly reduce motion-estimation complexity by matching only some of the checking points inside the search window. They are based on the assumption that the block distortion measure (BDM) increases as the checking point moves away from the global minimum point. However, this assumption does not always hold in the real-world video sequence [11]. These algorithms are easily trapped into local minimum points, and thus, often produce higher matching error compared with the FS algorithm. Recently, Liu and Zaccarin proposed an alternating subsampling search algorithm (ASSA) [12] which reduces the number of pixels used in each BDM instead of reducing the number of checking point. This algorithm uses alternating subsampling patterns in calculating different locations' BDM's. Experimental results show that the ASSA can achieve four-times computation reduction with its mean square error (MSE) performance very close to that of FS.

Apart from using this subsampling technique, halfway-stop techniques can also be used to reduce computational complexity in the BDM calculation. One of the examples is the partial distance search algorithm (PDS) [13] used in the vector quantization (VQ) encoding process. The basic idea of the PDS algorithm is as follows. Suppose the input vector consists of k components. The total distortion is obtained by adding these k partial distortions. If the ith accumulated partial distortion is greater than the current minimum distortion, the coder can just reject this vector and does not calculate the remaining partial distortions. This algorithm can greatly reduce computation of the distortion calculation if the computational complexity of comparison operation is relatively lower than that of multiplication operation. However, the efficiency of this algorithm is limited if it is directly applied to the BDM calculation in motion estimation. Paricularly if sum absolute error (SAE) is chosen as the matching criterion, the computational complexity of comparison becomes significant since it is comparable with that of addition.

In this letter, a new fast BMA named normalized partial distortion search (NPDS) is proposed. The proposed algorithm reduces computation by using a halfway-stop technique in the BDM calculation similar to the PDS algorithm. The major difference from the PDS algorithm is that it normalizes the accumulated partial distortion and the current minimum distortion before comparison. The probability of early rejection of non-possible candidate motion vectors (CMV) is thus increased. Experimental results show that the NPDS algorithm can achieve higher computation reduction than the ASSA algorithm while maintaining its MSE performance very close to that of FS. The rest of this letter is organized as follows. The formation of partial distortions is described in Section II. The normalized partial distortion search algorithm is given in Section III. Section IV gives the experimental results and conclusions are given in Section V.

Manuscript received September 3, 1998; revised August 11, 1999. This paper was recommended by Associate Editor S.-U. Lee.

The authors are with City University Image Processing Lab, Department of Electronic Engineering, City University of Hong Kong, Kowloon, Hong Kong (e-mail: ckcheung@image.cityu.edu.hk; eelmpo@cityu.edu.hk).

Publisher Item Identifier S 1051-8215(00)02799-3.

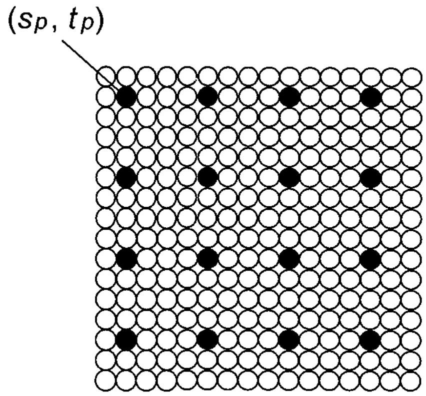


Fig. 1. The group of pixel locations for the calculation of the partial distortion $d_p(u, v)$. s_p, t_p: the offsets of the upper left corner point of the partial distortion from the upper left corner point of the block.

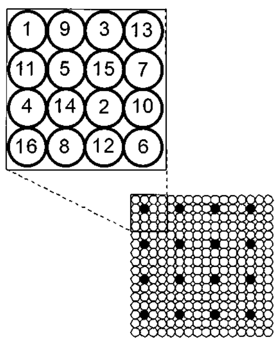


Fig. 2. Order of calculation of the partial distortions.

TABLE I
OFFSETS (s_p, t_p) OF THE 16 PARTIAL DISTORTIONS

p	(s_p, t_p)	p	(s_p, t_p)
1	(0, 0)	9	(1, 0)
2	(2, 2)	10	(3, 2)
3	(2, 0)	11	(0, 1)
4	(0, 2)	12	(2, 3)
5	(1, 1)	13	(3, 0)
6	(3, 3)	14	(1, 2)
7	(3, 1)	15	(2, 1)
8	(1, 3)	16	(0, 3)

II. FORMATION OF PARTIAL DISTORTIONS

SAE is chosen as the matching criterion in this letter due to it has lower computational complexity than that of MSE but has similar performance. In addition, 16 × 16 block size is used, as it is the most commonly used size in video coding. 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 16 × 16 block. The SAE between the block (k, l) of frame n and the block $(k+u, l+v)$ of frame $n-1$ is given by

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

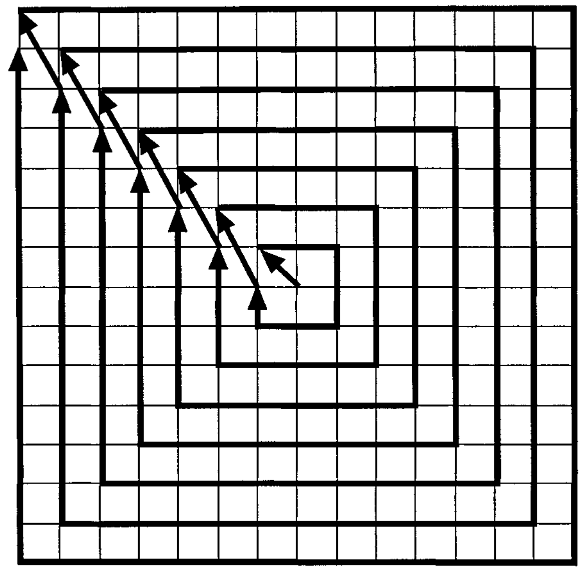


Fig. 3. Spiral scanning path of NPDS.

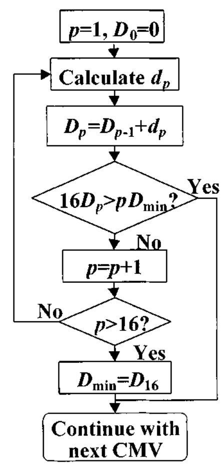


Fig. 4. Flowchart of a BDM calculation in the NPDS algorithm. d_p: pth partial distortion, D_p: pth accumulated partial distortion, $D_{\min}$: current minimum, and D_{16}: BDM of the CMV.

To reduce the number of comparison operations, the partial distortion is defined as a group of pixels' distortion instead of a single pixel's distortion used in the PDS algorithm. Thus, the $D(k, l;\ u, v)$ is divided into 16 partial distortions (d_p), where each partial distortion consists of 16 points spaced equally between adjacent points, as shown in Fig. 1. This grouping method is to ensure that each partial distortion does not localized in a particular region on the block. The pth partial distortion is defined as

$$d_p(k, l;\ u, v) = \sum_{i=0}^{3}\sum_{j=0}^{3} |I_n(k+4i+s_p, l+4j+t_p) - I_{n-1}(k+4i+s_p+u, l+4j+t_p+v)|. \quad (2)$$

The values s_p and t_p are the horizontal and vertical offsets of the upper left corner point of the pth partial distortion from the upper left corner point of the block, respectively.

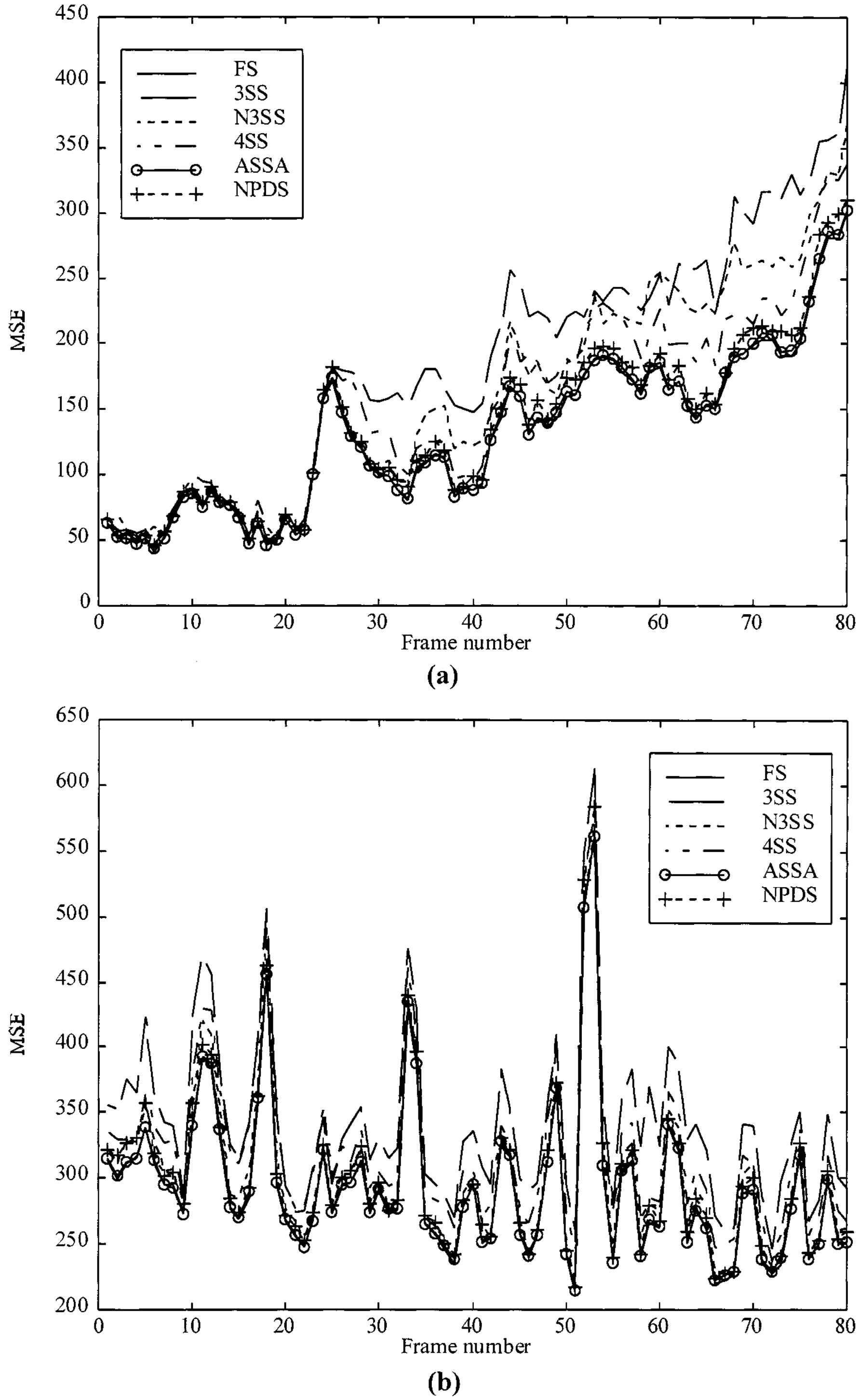
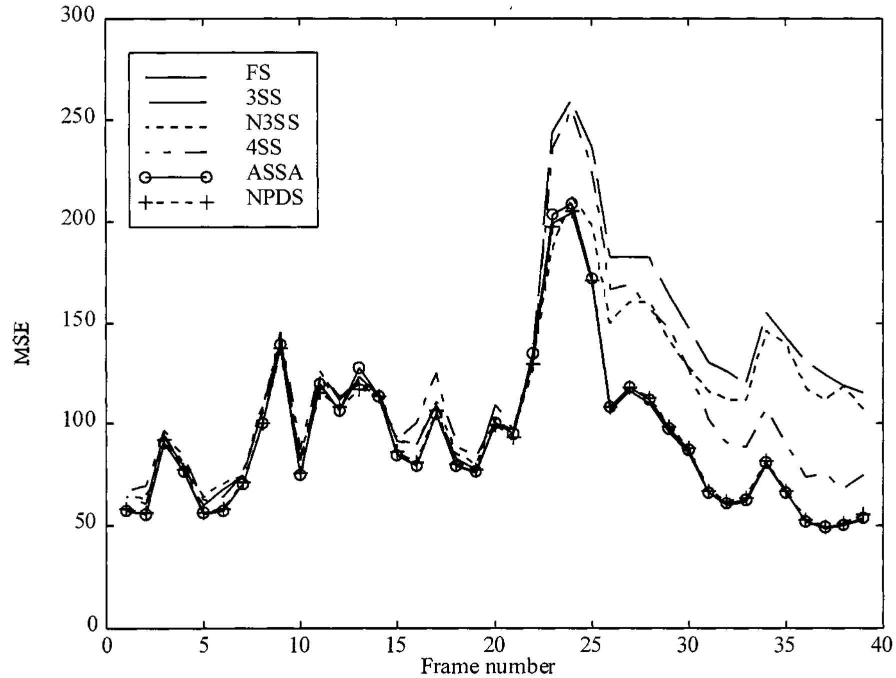


Fig. 5. MSE performance comparisons of SIF sequences. (a) Tennis. (b) Garden.

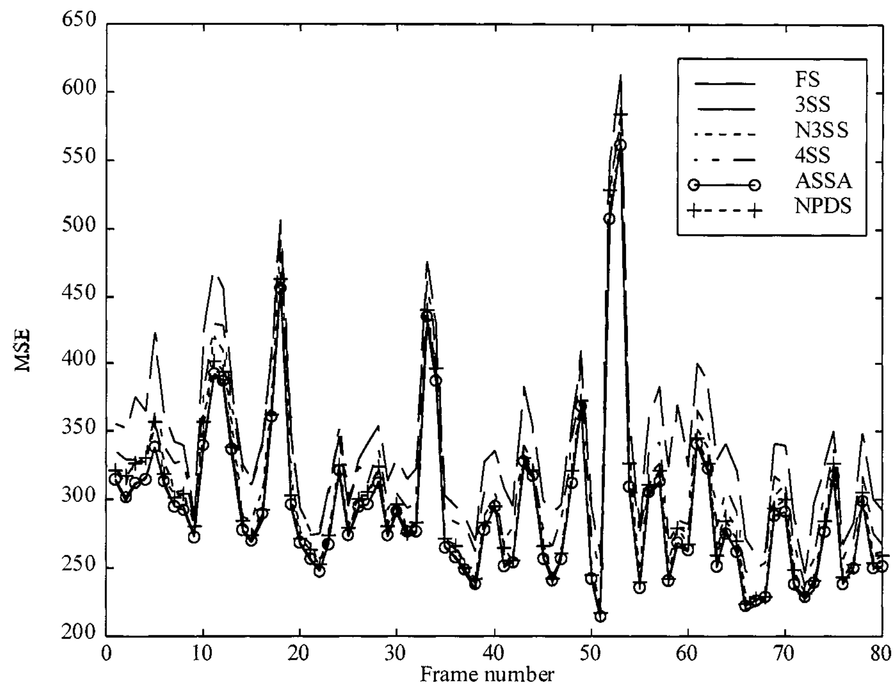
The order of calculation of the partial distortions d_p, where $p = 1, 2, \cdots, 16$, is depicted in Fig. 2. Taking the upper left corner points of the partial distortions as references, the numbers in Fig. 2 indicate the calculation order of the 16 partial distortions. Such calculation order is to ensure that for each of the accumulated partial distortion, the pixels considered for the calculation are evenly distributed on the block. The corresponding (s_p, t_p) values of (2) for the 16 partial distortions are listed in Table I. 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 accumulated partial distortion is used for the distortion comparison with the current minimum distortion. The details about the distortion comparison method are described in Section III.



(a)



(b)

Fig. 6. MSE performance comparison of CCIR 601 sequences. (a) Tennis. (b) Garden.

III. NPDS

The NPDS algorithm matches all the checking points inside the search window as the FS algorithm. The search begins at the origin checking point and then moves outwards with a spiral scanning path, as shown in Fig. 3. This order of searching is to exploit the center-biased motion-vector distribution characteristics of the real-world video sequence [9], [10]. During each block matching, the NPDS algorithm compares each accumulated partial distortion D_p with the normalized minimum distortion $(pD_{\min}/16)$ instead of the minimum distortion $D_{\min}$; it is because such comparison will increase the probability of early rejection of non-possible CMV's. However, the comparison between D_p and $pD_{\min}/16$ is most likely a floating comparison operation, which will increase the implementation complexity. The proposed NPDS algorithm, therefore, implements the normalized comparison by an integer comparison between M_p and N_p defined as follows:

$$M_p = pD_{\min} \quad (4)$$

$$N_p = 16D_p(k, l; u, v) \quad (5)$$

Basically, the variables M_p and N_p are the normalized versions of $D_{\min}$ and D_p, respectively. The comparison of M_p and N_p is equivalent to the normalized comparison of D_p with $pD_{\min}/16$. The procedures of a BDM calculation in the NPDS algorithm is summarized in the flow-chat as shown in Fig. 4. 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 normalized current minimum distortion. At the end of comparison (i.e., $p = 16$), if N_{16} is smaller than M_{16}, then this CMV becomes the new current minimum point.

It is noted that there is a multiplication operation in both (4) and (5). It can be easily translated to combinations of "left-shift" ($\ll$) and "addition" ($\pm$) operations. After the translation, there are at most two "addition" operations and two "left-shift" operations for the implementation of (4) and just one "left-shift" operation for (5). Thus, the overhead of the normalized comparison is negligible compared with the computation required by each partial distortion calculation.

IV. Experimental Results

The proposed algorithm is simulated using the luminance component of two famous video sequences "tennis" and "garden." The "tennis" sequence consists of various kinds of motions, including translation, zooming, and panning. The "garden" sequence consists of high portions of fast panning motions. These two sequences have two different frame sizes CCIR 601 (720 $\times$ 480 pixels) and SIF (360 $\times$ 240 pixels). All the simulated sequences are uniformly quantized to 8-bit per pixel. The block size for motion estimation is 16 $\times$ 16 pixels. The search windows for the SIF and CCIR 601 sequences are $\pm$ 7 and $\pm$ 15, respectively, in both horizontal and vertical directions. For the CCIR 601 sequences, half-pel accurate motion estimation is used for all simulated BMA's. The proposed NPDS algorithm is simulated with the FS, 3SS, N3SS, 4SS, and ASSA algorithms. For fare comparison, the BDM calculation of the 3SS, 4SS, and N3SS algorithms are also implemented with the use of halfway-stop technique that divides the BDM into 16 partial distortions. Their MSE performance and computational complexity are compared. For the computational complexity, it is compared in terms of four types of operations: absolution, addition, comparison, and left-shift, while the computation reduction is based on the total operations of these four types of operations.

Fig. 5(a) and (b) compares the MSE performance of different BMA's for the SIF sequences. It is noted that the ASSA algorithm performs very close to the FS algorithm for both video sequences. The proposed NPDS algorithm always performs better than the other fast BMA's and is close to the ASSA and FS algorithms. For the CCIR 601 sequences, the MSE performance of different BMA's is shown in Fig. 6(a) and (b). Since the frame sizes of these sequences are four-times greater than that of the SIF sequences, their motion displacements in pixel are also greater. It is clearly observed that the NPDS and ASSA algorithms can still maintain their MSE performances very close to the FS algorithm, while the other fast BMA's produce much higher average MSE's. These results indicate that the NPDS algorithm is more robust than the other compared fast BMA's that limit the number of checking points for reducing computation.

Table II shows the average operations required by different BMA's for the SIF sequences. The "absolution" and "addition" are the dominant components of the total operations for all the simulated BMA's. The average "comparison" and "left-shift" operations are much fewer than the other two types of operations. Especially, the "left-shift" operations are less than 0.5% of the total computations of NPDS. The NPDS algorithm has an average computation reduction of 12 times compared with the FS algorithm, and this reduction is slightly lower than the other compared fast BMA's. The ASSA algorithm, which also achieved similar MSE performance as the FS algorithm, has a computation reduction of 4 times only. Table III shows the cases for the CCIR 601 sequences. The computation required by calculating the half-pel BDM's is also counted. The 4SS algorithm got the highest average computation reduction among the compared fast BMA's and there is an average reduction of 44 times

TABLE II
Average Operations per Block for SIF Sequences.
(Top: Tennis; Bottom: Garden)

BMA	Abs.	Add.	Comp.	L-shift	Total
FS	52870.21	105533.89	205.52	-	158609.62
3SS	4022.07	8020.61	257.92	-	12300.60
N3SS	3795.03	7568.85	241.41	-	11605.29
4SS	3228.03	6437.03	203.79	-	9868.85
ASSA	14241.55	28272.58	210.52	-	42724.65
NPDS	4233.99	8292.73	454.15	56.26	13037.13

BMA	Abs.	Add.	Comp.	L-shift	Total
FS	51724.41	103246.78	201.05	-	155172.24
3SS	4022.49	8021.81	257.58	-	12301.88
N3SS	3764.46	7507.43	239.76	-	11511.65
4SS	3175.35	6331.97	200.20	-	9707.52
ASSA	13955.10	27704.16	206.05	-	41865.31
NPDS	4138.01	8105.78	443.67	55.30	12742.76

TABLE III
Average Operations per Block for the CCIR 601 Sequences.
(Top: Tennis; Bottom: Garden)

BMA	Abs.	Add.	Comp.	L-shift	Total
FS	236561.07	472198.07	923.07	-	709682.20
3SS	5882.73	11733.14	254.99	-	17870.86
N3SS	5907.10	11784.05	254.35	-	17945.50
4SS	5320.26	10613.15	214.89	-	16148.31
ASSA	60676.27	120428.47	924.07	-	182028.80
NPDS	17812.47	34732.89	1892.35	66.72	54504.43

BMA	Abs.	Add.	Comp.	L-shift	Total
FS	236561.07	472198.07	923.07	-	709682.20
3SS	6243.74	12455.08	277.64	-	18976.46
N3SS	7131.49	14227.62	336.08	-	21695.18
4SS	5809.12	11588.80	247.52	-	17645.44
ASSA	60676.27	120428.47	924.07	-	182028.80
NPDS	18447.35	36038.41	1932.03	122.70	56540.49

for the "tennis" sequence. The average computation reduction of NPDS is lower than the other compared fast BMA's except the ASSA algorithm, but it still has an average reduction of 13 times. The computation reduction of ASSA is 4 times, which is the same as its reduction for the SIF sequences.

V. CONCLUSION

This letter proposes the NPDS algorithm, which uses a halfway-stop technique in the calculation of BDM to reduce the computational complexity of block-motion estimation. By normalizing the accumulated partial distortion and the current minimum BDM, the probability of early rejection of non-possible CMV's is increased. Experimental results show that the proposed algorithm has similar MSE performance as the FS and ASSA algorithms for various image sequences, while it has an average computation reduction of 12–13 times. The NPDS algorithm is suitable for real-time implementation of high quality digital video applications in powerful PC's or workstations.

REFERENCES

[1] *Information Technology—Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 Mbit/s—Part 2: Video*, ISO/IEC 11172-2 (MPEG-1 Video), 1993.

[2] *Information Technology—Generic Coding of Moving Pictures and Associated Audio Information: Video*, ISO/IEC 13818-2 (MPEG-2 Video)/ITU-T H. 262, 1996.

[3] *Video Codec for Audiovisual Services at* $p \times 64$ *kbit/s*, ITU-T recommendation H.261 Version 3, Mar. 1993.

[4] *Video Coding for Low Bitrate Communication*, Draft ITU-T recommendation H.263, May 1996.

[5] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. NTC81*, Nov. 1981, pp. G5.3.1–G5.3.5.

[6] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, pp. 950–953, July 1990.

[7] A. Puri, H. M. Hang, and D. L. Schilling, "An efficient block matching algorithm for motion-compensated coding," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, 1987, pp. 1063–1066.

[8] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, pp. 1799–1806, Dec. 1981.

[9] 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.

[10] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 313–317, June 1996.

[11] K. H. K. Chow and M. L. Liou, "Generic motion search algorithm for video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, pp. 440–445, Dec. 1993.

[12] 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.

[13] C. D. Bei and R. M. Gray, "An improvement of the minimum distortion encoding algorithm for vector quantization," *IEEE Trans. Commun.*, vol. COM-33, pp. 1132–1133, Oct. 1985.