A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation

Shan Zhu and Kai-Kuang Ma

***Abstract*—Based on the study of motion vector distribution from several commonly used test image sequences, a new *diamond search* (DS) algorithm for fast block-matching motion estimation (BMME) is proposed in this paper. Simulation results demonstrate that the proposed DS algorithm greatly outperforms the well-known three-step search (TSS) algorithm. Compared with the new three-step search (NTSS) algorithm, the DS algorithm achieves close performance but requires less computation by up to 22% on average. Experimental results also show that the DS algorithm is better than recently proposed four-step search (4SS) and block-based gradient descent search (BBGDS), in terms of mean-square error performance and required number of search points.**

***Index Terms*—Block-matching algorithm, diamond search, H.261, H.263, motion estimation, MPEG, video coding, video compression.**

I. Introduction

Due to limited channel bandwidth and stringent requirements of real-time video playback, video coding is an indispensable process for many visual communication applications and always requires a very high compression ratio. The large amount of temporal correlation, or so-called *temporal redundancy* from the compression viewpoint, between adjacent frames in a video sequence, requires to be properly identified and eliminated to achieve this objective. An effective and popular method to reduce the temporal redundancy, called *block-matching motion estimation* (BMME), has been widely adopted in various video coding standards, such as CCITT (now ITU-T) H.261, H.263, MPEG-1, MPEG-2, and MPEG-4 [1], and in any motion-compensated video coding technique. Therefore, fast and accurate block-based search technique is highly desirable to assure much reduced processing delay while maintaining good reconstructed image quality.

By exhaustively testing all the candidate blocks within the search window, *full search* (FS) algorithm gives the global optimum solution (i.e., the miimum matching error point over the search window) to the motion estimation, while a substantial amount of computational load is demanded. To overcome this drawback, many fast *block-matching algorithms* (BMA's) have been developed, for example, *2-D logarithmic search* (LOGS) [2], *three-step search* (TSS) [3], *conjugate direction search* (CDS) [4], *cross search* (CS) [5], *new three-step search* (NTSS) [6], *four-step search* (4SS) [7], *block-based gradient descent search* (BBGDS) [8], etc. These fast BMA's exploit different search patterns and search strategies for finding the optimum motion vector with drastically reduced number of search points as compared with the FS algorithm.

In this paper, we propose a simple, robust and efficient fast BMME algorithm, called *diamond search* (DS) [9], [10], after observing the following facts.

Manuscript received November 11, 1997; revised February 26, 1999. This paper was presented in part at the 1st IEEE International Conference on Information, Communications, and Signal Processing (ICICS), Singapore, Sept. 9–12, 1997. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Steven D. Blostein.

S. Zhu is with V-Bits Video Engineering Dept., Cisco Systems, Inc., San Jose, CA, 95134 USA.

K.-K. Ma is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore (e-mail: ekkma@ntu.edu.sg).

Publisher Item Identifier S 1057-7149(00)01253-7.

II. Observations

Fundamentally speaking, the search pattern's shape and size exploited in the fast algorithm jointly determine not only its search speed but also resulted performance. Block distortions (or block-matching errors) form an error surface over the search window, and the global minimum point corresponds to the motion vector where the best matching (or the least error) incurs. Since the error surface is usually not monotonic, multiple local minimum points generally exist in the search window especially for those image sequences with large motion content. Therefore, searching with a small search pattern, such as the one used in BBGDS [8] with size of 3×3, is quite likely to be trapped into a local minimum for those video sequences with large motion content. On the other hand, a large search pattern with size of 9×9 and sparse checking points as exploited in the first step of TSS is most likely to mislead the search path to a wrong direction and hence misses the optimum point.

Since the distribution of the global minimum points in real-world video sequences is centered at the position of zero motion (i.e., search window center) [6], the center-biased NTSS algorithm, which is an improved version of TSS, tends to achieve much superior performance with fewer number of search points on average. However, NTSS loses the regularity and simplicity of TSS to some extent. Using a moderate search pattern with fixed size of 5×5, 4SS [7] obtains similar performance compared to that of NTSS. Note that both NTSS and 4SS utilize the overlapping of checking points between adjacent search steps to reduce the computational complexity further. However, 4SS still requires to test 17 checking points for a stationary block, which is much more than the nine checking points used by BBGDS in the same case. It is worth to mention that except BBGDS, the other three fast BMA's, i.e., TSS, NTSS and 4SS, commonly refrain the search window size to be 15×15 as their searching frameworks require.

Table II documents the motion vector distribution probabilities within certain distances from the search window center by exploiting the FS algorithm to six commonly used test image sequences (refer to Table I) based on the mean-square error (MSE) matching criterion. The predicted frame is contiguous to the previous original frame in the experiments. As indicated in Table II, about 52.76% to 98.70% of the motion vectors are enclosed in a circular support with a radium of 2 pels and centered on the position of zero motion. Second, the block displacement of real-world video sequences could be in any direction, but mainly in horizontal and vertical directions (e.g., camera panning).

Based on these two crucial observations, the search points (marked by "×" in Fig. 1) incurred within the circle with aradium of 2 pels (the dotted line in Fig. 1) are the most appropriate ones to be chosen to compose the search pattern, and a new *diamond search* (DS) algorithm is developed as described in the following section.

III. Diamond Search Algorithm

The proposed DS algorithm employs two search patterns as illustrated in Fig. 2, which are derived from the crosses (×) in Fig. 1. The first pattern, called *large diamond search pattern* (LDSP), comprises nine checking points from which eight points surround the center one to compose a diamond shape (◇). The second pattern consisting of five checking points forms a smaller diamond shape, called *small diamond search pattern* (SDSP).

In the searching procedure of the DS algorithm, LDSP is repeatedly used until the step in which the minimum block distortion (MBD) occurs at the center point. The search pattern is then switched from LDSP to SDSP as reaching to the final search stage. Among the five checking

TABLE I
IMAGE SEQUENCES USED FOR SIMULATIONS

Image sequence	Frame size	Length	Image sequence	Frame size	Length
Tennis	240×352	89	Susie	480×720	150
Football	240×352	90	Salesman	288×352	70
Caltrain	400×512	33	Claire	272×304	128

TABLE II
MOTION VECTOR DISTRIBUTION PROBABILITIES AGGREGATELY MEASURED AT VARIOUS MOTION DISTANCES (IN PEL) WITH REGARD TO THE CENTER POSITION USING THE FULL SEARCH (FS) ALGORITHM BASED ON MSE MATCHING CRITERION

Radium (pel)	Tennis	Football	Caltrain	Susie	Salesman	Claire
0	0.2622	0.6196	0.0416	0.0938	0.6562	0.9076
1	0.3751	0.7297	0.5373	0.3592	0.9452	0.9702
2	0.5276	0.7983	0.8523	0.5950	0.9609	0.9870
3	0.7178	0.8641	0.9168	0.7622	0.9741	0.9932
4	0.8402	0.9042	0.9380	0.8225	0.9795	0.9950
5	0.8930	0.9329	0.9561	0.8779	0.9853	0.9957
6	0.9200	0.9483	0.9720	0.9038	0.9957	0.9964
7	0.9599	0.9658	0.9894	0.9365	0.9975	0.9973

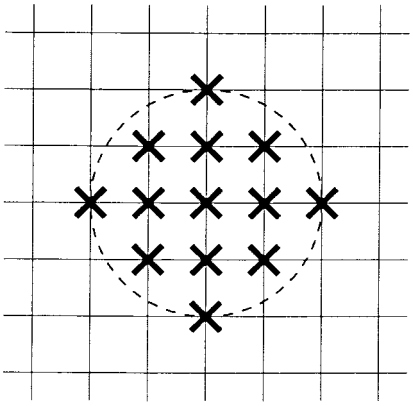


Fig. 1. An appropiate search pattern support—circular area with a radium of 2 pels. The 13 crosses show all possible checking points within the circle.

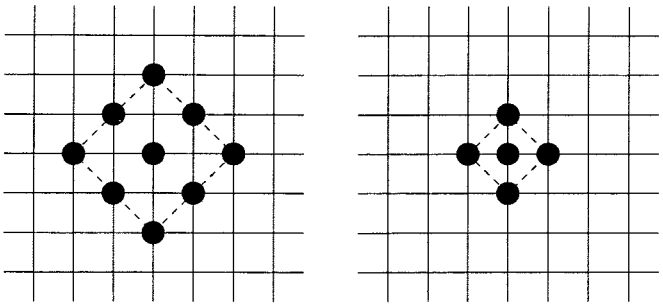
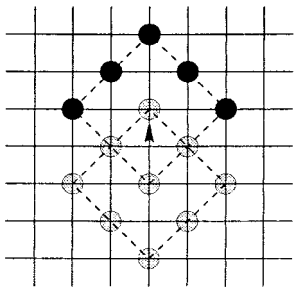


Fig. 2. Two search patterns derieved from Fig. 1 are employed in the proposed DS algorithm.

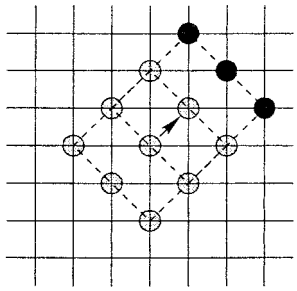
points in SDSP, the position yielding the MBD provides the motion vector of the best matching block.

The DS algorithm is summarized as follows.

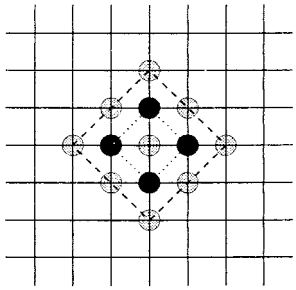
Step 1) The initial LDSP is centered at the origin of the search window, and the 9 checking points of LDSP are tested. If the MBD point calculated is located at the center position, go to **Step 3**; otherwise, go to **Step 2**.



(a) Case 1: the corner point. LDSP → LDSP



(b) Case 2: the edge point. LDSP → LDSP



(c) Case 3: the center point. LDSP → SDSP

Fig. 3. Three cases of checking-point overlapping in LDSP when the MBD point found in the previous search step (shaded dots) is located at (a) one of the corner points, (b) one of the edge points, and (c) the center point. The solid black dots are the new checking points where the computation of block-distortion measurement is required for the current search step.

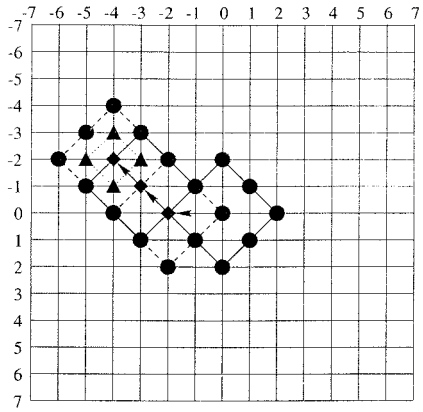
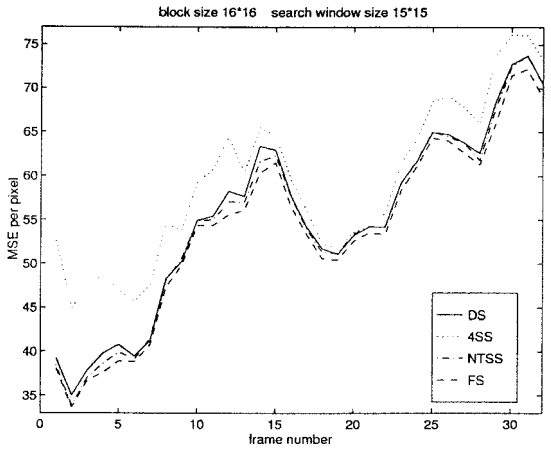
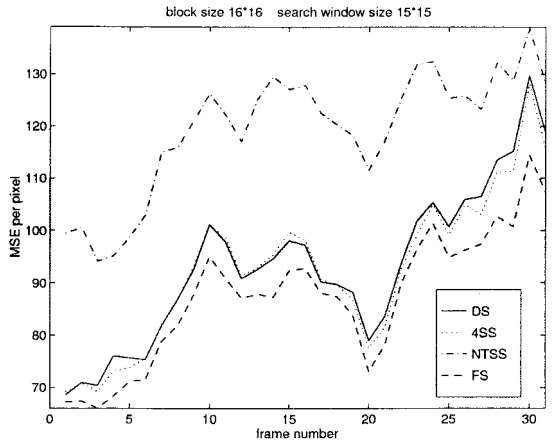


Fig. 4. Search path example which leads to the motion vector (–4, –2) in five search steps—four times of LDSP and one time SDSP at the final step. There are 24 search points in total—taking nine, five, three, three, and four search points at each step, sequentially.

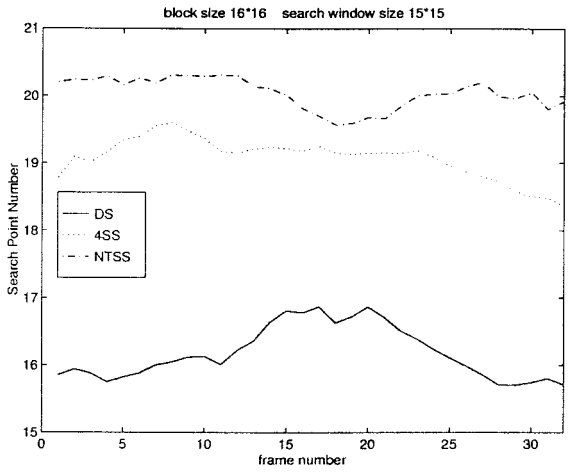


(a) Frame Distance = 1

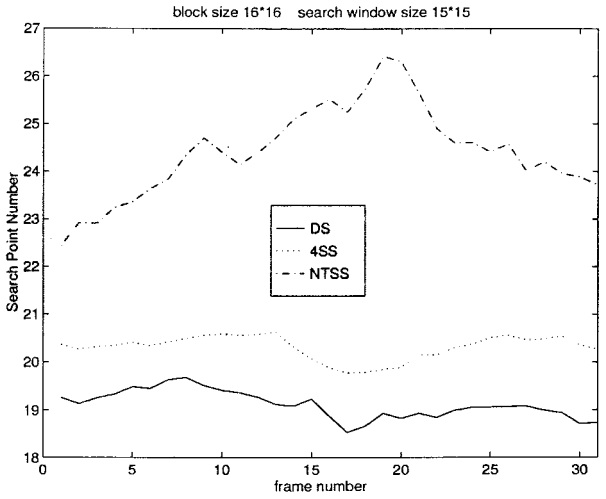


(b) Frame Distance = 2

Fig. 5. MSE comparison of DS, 4SS, NTSS, and FS for "Caltrain" sequence when (a) frame distance = 1 and (b) frame distance = 2.



(a) Frame Distance = 1



(b) Frame Distance = 2

Fig. 6. Comparison of the average number of search points applying DS, 4SS, and NTSS to "Caltrain" sequence individually when (a) frame distance = 1 and (b) frame distance = 2.

Step 2) The MBD point found in the previous search step is re-positioned as the center point to form a new LDSP. If the new MBD point obtained is located at the center position, go to **Step 3**; otherwise, recursively repeat this step.

Step 3) Switch the search pattern from LDSP to SDSP. The MBD point found in this step is the final solution of the motion vector which points to the best matching block.

Some insightful remarks on implementing the DS algorithm are provided as follows.

IV. Comments on DS Algorithm Implementation

First, when the search pattern (LDSP or SDSP) is near to or at the search window boundary, the checking points outside the search window are truncated. That is, the search is confined within the search window boundary. Note that this is a necessary constraint in MPEG standards since the variable-length codebook size for encoding motion vectors is limited. It is straightforward to employ the proposed DS algorithm to any larger search window, if required.

Second, DS algorithm doesn't restrict the number of search steps essentially. The MBD point found in each step has less or equal matching error compared with all the other checking points in the search pattern, which includes the MBD point found in the previous step, thus the MBD values found along the search path are in a nonincreasing order. Note that within each search iteration using LDSP, theoretically speaking, it is possible to have tie MBD values, which might even further cause search looping situation. But, practically speaking, it is extremely unlikely to happen as we have not encountered throughout our extensive simulation experiments so far. Any simple tie-break policy can be easily incorporated to avoid the above-mentioned situation so that the convergence of DS algorithm is always guaranteed.

Third, the checking points are partially overlapped between adjacent steps; especially, when LDSP is repeatedly used. For illustration, three cases of checking-point overlapping are presented in Fig. 3. When the previous MBD point is located at one of the corners or edge points of LDSP, only five or three new checking points are required to be tested as shown in Fig. 3(a) and (b), respectively. If the center point of LDSP produces the MBD, the search pattern is changed from LDSP to SDSP in the final search. In this case, only four new points are required to be tested, as shown in Fig. 3(c). An example of possible search path using our DS algorithm within a 15×15 search window is illustrated in Fig. 4 to demonstrate the checking- point overlapping along the search path.

TABLE III
AVERAGE MSE PER PIXEL

Algorithm	Frame Distance = 1			Frame Distance = 2		
	Tennis	Caltrain	Claire	Tennis	Caltrain	Claire
DS	161.5	55.11	5.671	347.7	93.32	10.73
4SS	171.8	59.3	5.727	327.3	92.51	10.88
BBGDS	176.1	54.88	5.66	426.2	130.5	10.69
NTSS	177.1	54.74	5.673	286.8	119.4	10.78
TSS	213.1	68.8	5.748	301.2	134.3	11.00
FS	139.9	53.9	5.652	212.6	87.27	10.60

TABLE IV
AVERAGE NUMBER OF SEARCH POINTS PER MOTION VECTOR ESTIMATION. NOTE THAT THE SEARCH-POINT NUMBER OF TSS AND FS ARE FIXED, 25 AND 255, RESPECTIVELY

Algorithm	Frame Distance = 1			Frame Distance = 2		
	Tennis	Caltrain	Claire	Tennis	Caltrain	Claire
DS	16.52	16.18	12.31	18.64	19.11	12.79
4SS	18.89	19.08	15.8	20.06	20.31	16.11
BBGDS	14.1	12.69	8.676	16.33	16.86	9.447
NTSS	21.02	20.04	16.0	24.15	24.43	16.78

V. SIMULATION RESULTS

In our simulation experiments, the block size is fixed at 16×16. To make a consistent comparison, block matching is conducted within a 15×15 search window (i.e., ± 7 pels displacement in horizontal and vertical directions), although our DS search algorithm has no restriction on search window size. In addition, the *full*-pixel grid is set for all BMA's concerned in this paper. Frame distance between the predicted frame and the original frame is set to be either 1 or 2 separately. Mean absolute distance (MAD), rather than MSE, is used as the matching criterion to reduce the block-matching computation in practice.

For BMME, computational complexity could be measured by average number of search points required for each motion vector estimation. Figs. 5 and 6 illustrate the frame-by-frame comparison of MSE and the average search-point numbers after applying DS, 4SS, NTSS, and FS algorithms to "Caltrain" sequence under different frame distances, respectively. The average MSE values and search-point numbers of "Caltrain" and other test sequences are presented in Tables III and IV.

For the image sequence with small-motion content, such as "talking-head" sequences (e.g., "Claire"), DS, 4SS, BBGDS and NTSS algorithms achieve close MSE performance as expected. For moderate to large motion image sequences, DS, 4SS and NTSS maintain close performance while the BBGDS degrades distinctly. Therefore, the BBGDS' MSE performance is not stable and highly depends on the motion content, although BBGDS constantly demands the smallest number of search points.

Since typically encountered image sequence has wide range of motion content, DS, 4SS, and NTSS are more appropriate to use. Among them, our DS algorithm requires the smallest average number of search points. Finally, the TSS algorithm clearly achieves the worst performance among all the fast BMA's experimented.

Why does the proposed DS algorithm work so well? The search-step length of our DS algorithm has two pels in horizontal and vertical directions and one pel in each diagonal direction. Therefore, for large motion blocks, the DS algorithm is not so easy to be trapped into a local minimum point as BBGDS would do and can find the global minimum point using relatively few search points. For quasistationary or stationary blocks, the search points of the DS algorithm will be fewer than that of the 4SS. In addition, the compact shape of the search patterns used in the DS algorithm increases the possibility of finding the global minimum point located inside the search pattern. Therefore, the DS algorithm tends to produce smaller or at least similar motion estimation error compared with other fast BMA's.

VI. CONCLUSION

In this paper, search patterns and search strategies of certain existing fast BMA's are analyzed. The distribution of motion vectors based on several commonly experimented test image sequences are also studied. Based on these analyses and observations, a new *diamond search* (DS) algorithm for fast block-matching motion estimation is developed.

Unlike TSS, NTSS and 4SS, the search window size is not restricted by the searching strategy in our DS algorithm. Simulation experiments conducted clearly demonstrate that the proposed DS algorithm greatly outperforms the well-known TSS algorithm and achieves close MSE performance compared to NTSS while reducing its computation by up to 22% approximately. Compared with other recently proposed BMA's such as 4SS and BBGDS, our DS algorithm also works better on average in terms of MSE values, reconstructed image quality, and average number of search points.

The DS is implemented in the MPEG-4 video-encoding environment and its efficacy is demonstrated through core experimental results [11]. Based on these results, it is adopted and incorporated in MPEG-4 verification model [12].

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their comments and suggestions that helped improve the quality of this paper.

REFERENCES

[1] K. R. Rao and J. J Hwang, *Techniques and Standards for Image, Video and Audio Coding*. Englewood Cliffs, NJ: Prentice Hall, 1996.
[2] J. Jain and A. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COMM-29, pp. 1799–1808, Dec. 1981.
[3] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. Nat. Telecommun. Conf.*, New Orleans, LA, Nov. 29–Dec. 3 1981, pp. G5.3.1–5.3.5.
[4] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COMM-33, pp. 888–896, Aug. 1985.
[5] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, pp. 950–953, July 1990.
[6] 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.
[7] 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.
[8] L. K. Liu and E. Feig, "A block-based gradient descent search algorithm for block motion estimation in video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, pp. 419–423, Aug. 1996.
[9] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block matching motion estimation," in *Proc. Int. Conf. Inform., Commun., Signal Process.*, Singapore, Sept. 9–12, 1997, pp. 292–296.
[10] S. Zhu, "Fast motion estimation algorithms for video coding," M.S. thesis, School Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, 1998. (supervised by K.-K. Ma).
[11] K. K. Ma and P. I. Hosur, "Core experimental results of fast motion estimation based on new test conditions (Q4a)," ISO/IEC JTC1/SC29/WG11/M4934, Vancouver, B.C., Canada, July 1999.
[12] "MPEG-4 video verification model, ver. 14.0," ISO/IEC JTC1/SC29/WG11/N2932, Oct. 1999.