International Conference on
Information, Communications and Signal Processing
ICICS '97
Singapore, 9-12 September 1997



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

Shan Zhu † and Kai-Kuang Ma ‡
School of Electrical and Electronic Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798
Republic of Singapore
† Email: saint_jose@bigfoot.com ‡ Email: ekkma@ntu.edu.sg

## Abstract

Based on the analysis of certain existing fast block-matching algorithms (BMAs) and study of motion vector distributions of real-world image sequences, a new *diamond search* (DS) algorithm for fast block-matching motion estimation 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 similar performance but requires approximate 20%-25% less computation. Compared with some recently proposed fast BMAs, such as the four-step search (4SS) and the block-based gradient descent search (BBGDS), our DS algorithm also shows its superiority.

## 1 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 high temporal correlation, or so-called *redundancy* from the compression viewpoint, between adjacent frames in an image sequence, requires to be properly identified and eliminated to achieve this objective. An effective and popular technique to reduce the temporal redundancy, called *block-matching motion estimation*, has been widely adopted in various video coding standards, such as CCITT (now ITU-T) H.261, H.263, MPEG-1 and MPEG-2 [1]. In fact, block-matching motion estimation is instrumental to any motion-compensated video coding technique. Therefore, fast algorithms for block matching are highly desirable.

By exhaustively testing all the candidate blocks within the search window, *full search* (FS) algorithm gives the global minimum block distortion (MBD) position which corresponds to the best matching block. However, a substantial computational load is demanded. To overcome this drawback, many fast BMAs have been developed, for examples, *2-D logarithmic search* (LOGS) [2], *three-step search* (TSS) [3], *conjugate direction search* (CDS) [4], *cross search* (CS) [5], *dynamic search-window adjustment* (DSWA) search [6], *new three-step search* (NTSS) [7], *four-step search* (4SS) [8], *block-based gradient descent search* (BBGDS) [9], etc.

Based on the analysis of the above-mentioned fast BMAs and study of motion vector distribution from several commonly used real-world test image sequences, a new *diamond search* (DS) algorithm for fast block-matching motion estimation is proposed in this paper. It will be shown that our DS algorithm greatly outperforms the well-known TSS algorithm and works better than NTSS, 4SS and BBGDS on average, in terms of error performance and computational complexity.

## 2 Observations

The main objective of fast BMAs [2]–[9] is to search for an optimum or near-optimum solution with drastically reduced number of search points which is constantly required in the FS algorithm. The shape and size of search patterns exploited by the fast BMAs jointly determines not only the error performance but also the computation complexity. Due to the multiple local MBD points located in the search window, searching with a small search pattern (for example, the pattern exploited in BBGDS [9] with size of $3 \times 3$) is quite likely to be trapped into a local minimum and hence deteriorates the performance especially for those image sequences with large motion content.

On the other hand, a large search pattern exploited by TSS [3] with size of $9 \times 9$ and nine sparse checking points is quite likely to mislead the search path to a wrong direction and hence miss the optimum point. Since the distribution of the global minimum point in real-world video is centered at zero [7], a center-biased TSS algorithm, called NTSS [7], tends to achieve better performance because it has a higher possibility to catch the global optimum point. The average number of search points in NTSS is fewer than that of TSS, however NTSS loses the regularity and simplicity of TSS to

0-7803-3676-3/97/$10.00 © 1997 IEEE

some extent.

Using a moderate search pattern with fixed size of $5 \times 5$, 4SS obtains a performance which is similar to NTSS and outperforms TSS. However, 4SS still requires to test 17 checking points for a stationary block, which is much more than the 9 checking points used by BBGDS in the same case.

Table 1 documents the motion vector distribution probabilities within certain distances from the search window center by exploiting the FS algorithm to four commonly used test image sequences, "Tennis," "Football," "Susie" and "Salesman," based on the mean-square error (MSE) matching criterion. As indicated in Table 1, about 52.76% to 96.09% of the motion vectors are enclosed in a circular support with a radium of 2 pels and centered on the zero-motion position.

Secondly, it is fairly reasonable to assume that the block displacement at any direction is equally distributed for real-world image sequences. Based on these two crucial observations, the search-point positions (the "$\times$" in Fig. 1) incurred within the circle with a radium of 2 pels (the dotted line in Fig. 1) are the most appropriate ones to be chosen to compose the search pattern. Consequently, a new *diamond search* (DS) algorithm is proposed in the following.

# 3 Diamond Search Algorithm

The proposed DS algorithm employs two search patterns as illustrated in Fig. 2, which are derived from the crosses ($\times$) in Fig. 1. The first pattern, called *large diamond search pattern* (LDSP), comprises of 9 checking points from which eight points surround the center one to compose a diamond shape ($\diamond$). The second pattern consisting of 5 checking points forms a smaller diamond shape, called *small diamond search pattern* (SDSP).

First, LDSP is repeatedly used until the MBD occurs at the center point in any step. The search pattern is then switched from LDSP to SDSP as reaching to the final search stage. Among the five checking points in SDSP, the position yielding the MBD provides the motion vector of the best matching block.

Note that checking points are partially overlapped between adjacent steps especially when LDSP is continuously being applied. For illustration, three cases of the checking-point overlapping are presented in Fig. 3. When the previous minimum 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 Fig. 3(b), respectively. If the center point of LDSP produces the MBD, search pattern is changed from LDSP to SDSP, and 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 $15 \times 15$ search window is illustrated in Fig. 4 to demonstrate the checking-point overlapping along the search path.

Unlike other fast BMAs which might only work well for the image sequences with certain degree of motion, our DS algorithm can deal with both small and large motion image sequences with smaller motion estimation error. It also requires smaller number of search points compared with NTSS, 4SS, and TSS.

# 4 Simulation Results

In our simulation experiments, the block size is fixed at $16 \times 16$. Block matching is conducted within a $15 \times 15$ search window on full-pixel basis, i.e., the maximum block displacement in horizontal or vertical direction is $\pm 7$ pels. Mean absolute distance (MAD), rather than MSE, is used as the matching criterion to reduce the block-matching computation in practice for reduced computation. Four test image sequences with large, moderate or small motion are exploited in our experiments.

Statistical performance comparisons of DS, 4SS, BBGDS, NTSS, TSS and FS algorithms using all the image sequences are summarized in Tables 2 through 5. For block-matching motion estimation, computational complexity is mainly dependent on the average search-point numbers required for each motion vector estimation. The average MSE values and search-point numbers are presented in Table 2 and Table 3, respectively. The accuracy of finding the best matching blocks can be evaluated by measuring the average distance from the resulting motion vectors to the ones obtained by using the FS algorithm (see Table 4). Another kind of accuracy measurement is the probability of finding the same motion vector as obtained by applying the FS method (refer to Table 5).

Based on the simulation results documented in Tables 2 through 5, the TSS algorithm is obviously worse than all the other fast BMAs. Notice that although BBGDS constantly demands the smallest number of search points, its MSE performance is quite unstable and highly depends on the image sequence content. For small motion image sequence, such as "Salesman", DS, 4SS, BBGDS and NTSS algorithms achieve similar MSE performance. For moderate to large motion image sequences, "Football" particularly, DS, 4SS and NTSS maintains similar performance while the BBGDS degrades distinctly. Compared with 4SS and NTSS, our DS algorithm provides similar MSE with the smallest number of search points on average. Compared with NTSS, the DS method can save the search points by approximate 20%–25% on average. On the other hand, compared with 4SS, our DS algorithm almost always produces slightly smaller MSE with smaller number of search points.

To illustrate further, the original 63-rd frame of "Tennis" sequence and the estimated 63-rd frames using FS, DS, NTSS, 4SS and BBGDS algorithms are shown in Fig. 5. One can see that the imaging quality achieved by the DS algorithm is clearly better than those using

other fast BMAs, especially along the top edge of the net and the poster's characters on the wall. For the latter, it is even much better than that of using the FS algorithm.

# 5 Conclusion

In this paper, strategies and search patterns of certain existing fast BMAs are analyzed and benchmarked. The distribution of motion vector based on several commonly experimented 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 other fast BMAs, our proposed DS algorithm consistently performs well for the image sequence with wide range of motion content.

Experiment results show that the proposed DS algorithm greatly outperforms the well-known TSS algorithm and achieves close performance compared to NTSS while reducing computation by 20%-25% approximately. Compared with other recently proposed BMAs such as 4SS and BBGDS, our DS algorithm also works better on average based on several meaningful measurements as documented in this paper.

# References


[1] K. R. Rao and J. J. Hwang, *Techniques and Standards for Image, Video and Audio Coding*, Prentice Hall PTR, 1996.

[2] J. Jain and A. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Commun.*, vol. COM-29, no. 12, pp. 1799-1808, Dec. 1981.

[3] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," *Proc. Nat. Telecommun. Conf.*, New Orleans, LA, pp. G5.3.1-5.3.5, Nov. 29 - Dec. 3, 1981.

[4] R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," *IEEE Trans. Commun.*, vol. COM-33, no. 8, Aug. 1985.

[5] M. Ghanbari, "The cross-search algorithm for motion estimation," *IEEE Trans. Commun.*, vol. 38, no. 7, pp. 950-953, Jul. 1990.

[6] L. W. Lee, J. F. Wang, J. Y. Lee, and J. D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 1, Feb. 1993.

[7] 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, no. 4, Aug. 1994.

[8] 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, no. 3, pp. 313-317, Jun. 1996.

[9] 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, no. 4, Aug. 1996.


| radium (pel) | Tennis | Football | Susie | Salesman |
|---|---|---|---|---|
| 0 | 0.2622 | 0.6196 | 0.0938 | 0.6562 |
| 1 | 0.3751 | 0.7297 | 0.3592 | 0.9452 |
| 2 | 0.5276 | 0.7983 | 0.5950 | 0.9609 |
| 3 | 0.7178 | 0.8641 | 0.7622 | 0.9741 |
| 4 | 0.8402 | 0.9042 | 0.8225 | 0.9795 |
| 5 | 0.8930 | 0.9329 | 0.8779 | 0.9853 |
| 6 | 0.9200 | 0.9483 | 0.9038 | 0.9957 |
| 7 | 0.9599 | 0.9658 | 0.9365 | 0.9975 |

Table 1: Motion vector distribution 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.

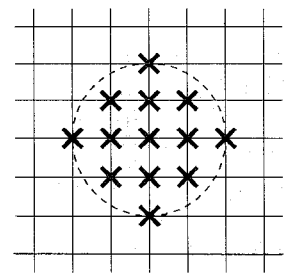
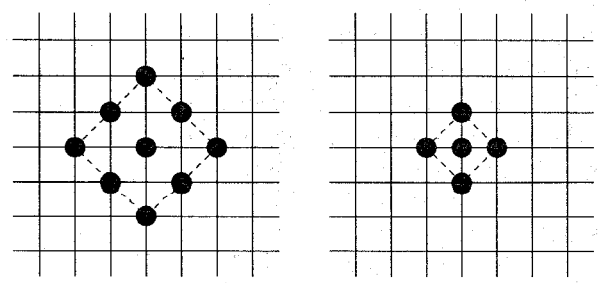


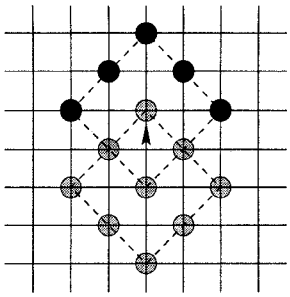
Figure 1: An appropriate search pattern support — circular area with a radium of 2 pels. The thirteen crosses show all possible checking-point positions within the circle.



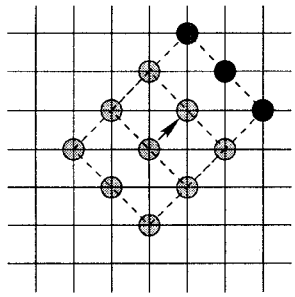
(a) Large diamond search pattern (LDSP)

(b) Small diamond search pattern (SDSP)

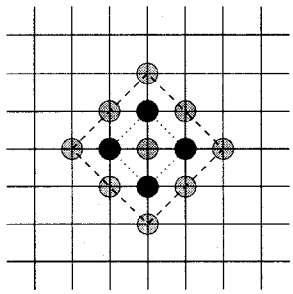
Figure 2: Two search patterns derived from Fig. 1 are employed in the proposed Diamond Search (DS) algorithm.



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



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



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

Figure 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 for the current search step is required.

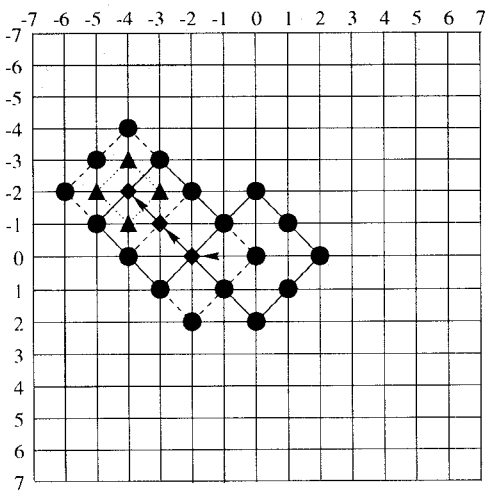



Figure 4: A search path example which leads to the motion vector (-4, -2) in five search steps. There are 24 search points in total — taking 9, 5, 3, 3 and 4 search points in each step, sequentially.

| Algorithm | Tennis | Football | Susie | Salesman |
|---|---|---|---|---|
| DS | 161.5 | 188.5 | 22.42 | 20.34 |
| 4SS | 171.8 | 188.4 | 23.1 | 20.44 |
| BBGDS | 176.1 | 194.2 | 27.61 | 20.09 |
| NTSS | 177.1 | 183.6 | 22.69 | 20.15 |
| TSS | 213.1 | 202.2 | 24.4 | 20.67 |
| FS | 139.9 | 171.6 | 20.71 | 19.88 |

Table 2: Average mean-square error (MSE) per pixel.

| Algorithm | Tennis | Football | Susie | Salesman |
|---|---|---|---|---|
| DS | 16.52 | 14.46 | 17.76 | 12.87 |
| 4SS | 18.89 | 17.17 | 19.76 | 16.2 |
| BBGDS | 14.1 | 11.74 | 14.41 | 9.603 |
| NTSS | 21.02 | 18.58 | 21.83 | 16.94 |

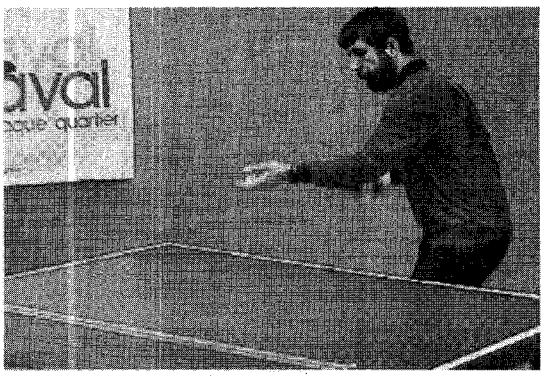
Table 3: Average number of search points per motion vector estimation. Note that the search-point number of TSS and FS are fixed, 25 and 255, respectively.

| Algorithm | Tennis | Football | Susie | Salesman |
|---|---|---|---|---|
| DS | 0.7644 | 0.2788 | 0.9417 | 0.1314 |
| 4SS | 1.017 | 0.295 | 1.059 | 0.1267 |
| BBGDS | 0.9171 | 0.31 | 1.176 | 0.0881 |
| NTSS | 0.9481 | 0.2379 | 0.942 | 0.08968 |
| TSS | 1.766 | 0.3384 | 1.299 | 0.1597 |

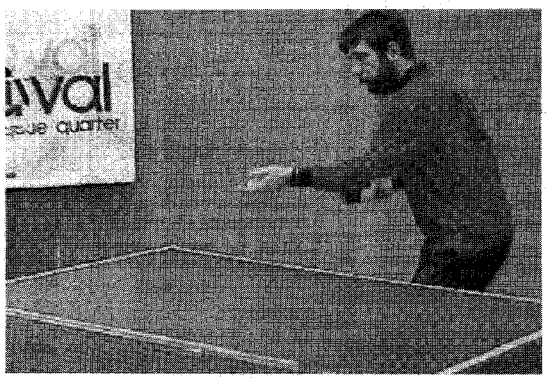
Table 4: Average distance measured from the motion vector found by exploiting various BMAs to the one obtained by applying the FS method.

| Algorithm | Tennis | Football | Susie | Salesman |
|---|---|---|---|---|
| DS | 0.8587 | 0.9463 | 0.7492 | 0.9676 |
| 4SS | 0.7957 | 0.9292 | 0.7063 | 0.9691 |
| BBGDS | 0.8203 | 0.9447 | 0.6863 | 0.9833 |
| NTSS | 0.7836 | 0.9386 | 0.7347 | 0.9805 |
| TSS | 0.6706 | 0.9144 | 0.6444 | 0.9621 |

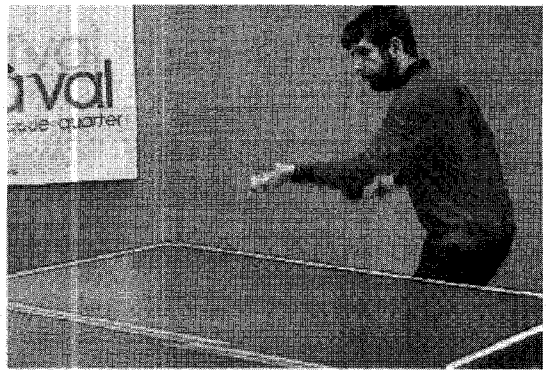
Table 5: Probability of finding the same motion vector as obtained in the FS method.



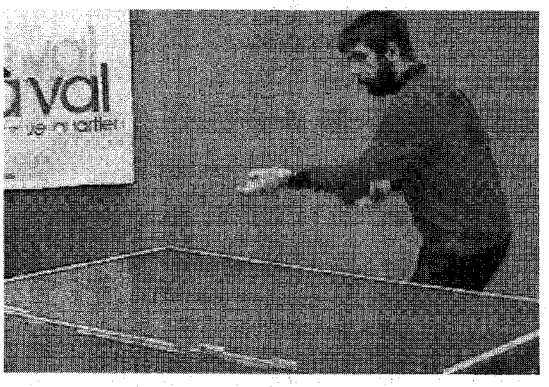
(a) The original image frame.



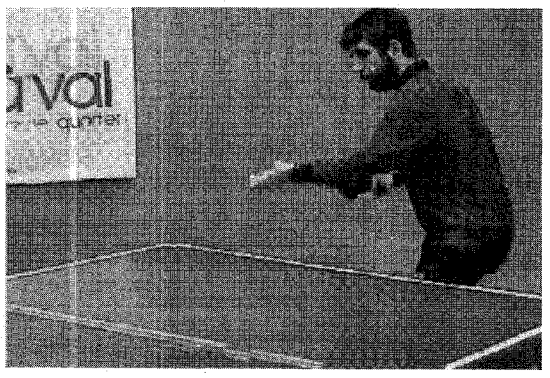
(b) FS algorithm.



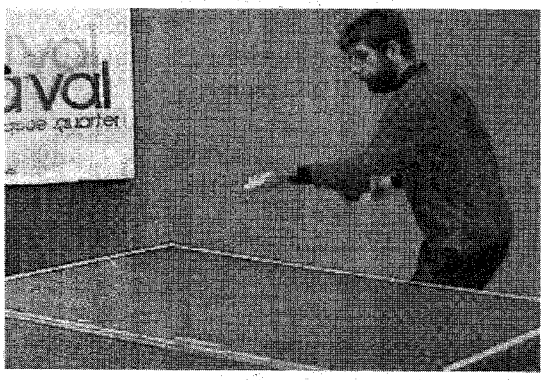
(c) DS algorithm.



(d) NTSS algorithm.



(e) 4SS algorithm.



(f) BBGDS algorithm.

Figure 5: The 63-rd frame of "Tennis" sequence: (a) the original image and the estimated image frames using (b) FS, (c) DS, (d) NTSS, (e) 4SS, and (f) BBGDS.