# A Genetic Rhombus Pattern Search for Block Motion Estimation

Jang-Jer Tsai and Hsueh-Ming Hang
Department of Electronics Engineering,
National Chiao-Tung University,
Hsinchu, Taiwan, R.O.C
jjtsai@aa.nctu.edu.tw, hmhang@mail.nctu.eud.tw

*Abstract*—**Pattern-based block motion estimation (PBME) is one of the most effective yet computational intensive tools in digital video coding standards. Many PBME algorithms have been proposed but few papers have investigated in depth the fundamental characteristics of various PBME algorithms. Why one search algorithm outperforms the others? In this paper, we propose a systematic approach to examine this problem. The minimal numbers of search points achievable by a PBME algorithm form a discrete function in the search area. We analyze this so-called weighting function and suggest an ideal target. Then, we design a genetic rhombus pattern search (GRPS) to match this ideal weighting function. Simulations show that, comparing to the other popular search algorithms, GRPS reduces the average search points for more than 20% while it maintains a similar level of coded image PSNR quality.**

## I. Introduction

Block-based motion estimation (BME) has been widely adopted by modern video coding standards [1] such as H.26X series and MPEG-1/2/4. According to [2], fast BME algorithms can be classified into two categories, i.e., reduction of the number of checking (search) points and reducing computational complexity in calculating the block-matching cost for each search point. This paper focuses on the algorithms in the first category.

For reducing search points, BME typically uses three techniques: 1) an operative threshold for terminating search process [3][4], 2) the selection of starting (initial) points [5], and 3) an effective set of search patterns [4][6][7][8]. Combining all these techniques, the latest BME algorithms achieve a dramatic speed-up in finding the near-optimal motion vectors while maintaining the desired level of quality. The first and second speed-up techniques make use of the data correlation inside one frame or between nearby frames. And the third technique (search pattern) is effective when the matching-cost surface is nearly monotonic. Among these techniques, the search pattern plays a key role in deciding the performance of a search algorithm especially when the data correlation is low. Consequently, we like to explore further on the search patterns.

Recent research works on ME often collect the statistics of motion vectors and design good search patterns accordingly. In designing a search algorithm, a search algorithm shall have small number of search points at the locations where the probabilities of best motion vectors ($P_{BMV}$) are high. Although almost all popular search algorithms are devised based on the statistical fact that $P_{BMV}$ generally exhibit a radiating nature, however, the performance of a typical search pattern strongly depends on the nature of video sequences. Among the existing algorithms, the rhombus patterns are quite effective for low motion sequences [4], and the hexagonal patterns are very powerful for high motion sequences [6]. Combining these two sets of search patterns, [9] uses rhombus patterns for initial searches and switches to the hexagonal search for the succeeding searches. Is this the best search algorithm? What determines the search speed of a search algorithm?

In this paper, we first analyze the search points of several representative search algorithms and formulate the minimal number of search points, namely, *weighting function*, in section II. We find that the weighting function is highly correlated with the performance of a search algorithm. In section III, a genetic rhombus pattern search (GRPS) is proposed. Section IV shows the experimental results of the proposed algorithm in comparing with several popular search algorithms. Finally, conclusions are given in section V.

## II. Analysis on the number of Search Points

Search patterns are generally designed based on the assumption that the matching cost surface is monotonic. Under this assumption, we define the number of search points as the minimal number of search points in all possible paths leading to the best-matched point from the starting point. Thus, the number of search points is a function of location and is called *weighting function*. By examining the search process of a PBME, we can construct its associated weighting function.

Four representative pattern-based search methods, Four Step Search (FSS) [7], Diamond Search (DS) [8], Enhanced

1-4244-0921-7/07 $25.00 © 2007 IEEE.

Hexagonal Search (EHS) [6], and Easy Rhombus Pattern Search (ERPS), are chosen to illustrate the construction of weight functions. These pattern-based search algorithms are selected because of their well-recognized performance. EHS performs rather well particularly for high motion sequences, and ERPS is more suitable for low motion sequences. ERPS is a simplified version of the adaptive rood pattern search (ARPS [4]). It is ARPS without initial rood patterns and it uses only one predicted motion vector (PMV) as the starting point, which is the medium of motion vectors of neighboring blocks.

$$WF_{FSS}(x,y) = 9 + M_{FSS}(x,y) \times n_{FSS}(x,y) + 8\,. \qquad (1)$$

$$WF_{DS}(x,y) = 9 + M_{DS}(x,y) \times n_{DS}(x,y) + 4\,. \qquad (2)$$

$$WF_{EHS}(x,y) = 7 + 3 \times n_{EHS}(x,y) + K_{EHS}(x,y)\,. \qquad (3)$$

$$WF_{ERPS}(x,y) = 1 + M_{ERPS}(x,y) \times n_{ERPS}(x,y) + 4\,. \qquad (4)$$

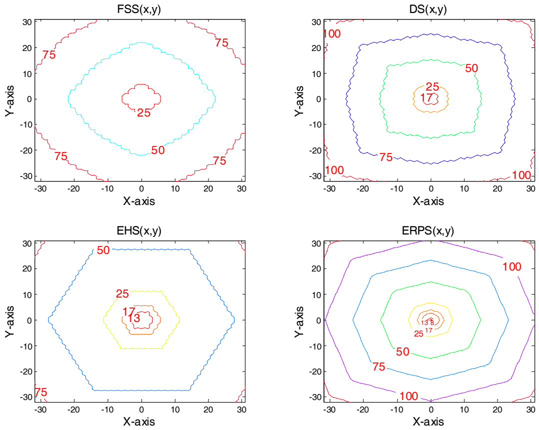

Figure 1. Contour plots of the weighting functions of FSS, DS, EHS, and ERPS, respectively.

According to [7], [8], [6], and [4], the weighting function of FSS, DS, EHS, and ERPS can be respectively expressed as (1), (2), (3), and (4) by analyzing the search processes of the algorithms. In Eqs. (1) to (4), $M_{FSS}(x,y)$ is either 5 or 3, $M_{DS}(x,y)$ is either 5 or 3, $K_{EHS}(x,y)$ is either 3 or 2, $M_{ERPS}(x,y)$ is either 3 or 2, all depending on the search direction. And the values of $n_{FSS}(x,y)$, $n_{DS}(x,y)$, $n_{EHS}(x,y)$ and $n_{ERPS}(x,y)$ are the number of movements.

Setting the number of movements in Eqs. (1)-(4) to zero, we find that, ERPS checks only 5 points in the best cases, while FSS checks 17 points, DS checks 13 points and EHS checks 9 points. Thus, in the best cases ERPS has the smallest number of search points among the four algorithms. The best cases refer to the situations that the best-matched motion vector is located at the starting point.

Figure 1. shows the contour plots of the weighting functions of FSS, DS, EHS, and ERPS, respectively. The value on a contour represents the minimal number of search points for a search algorithm to move from the origin to a point (location) with integer coordinates on the contour. The weighting function is a discrete function, thus data points only exists on the integer coordinates. For the ease of visualization, the data points are connected to form the continuous contour curve using a simple interpolation method. Thus, there are ripples on the contours due to interpolation.

Because EHS moves faster than any other algorithm, EHS surpasses the other algorithms at distant locations. Its weighting function $WF_{EHS}(x,y)$ has smaller values at the outer contours. Consequently, it exceeds other algorithms for fast motion situations. As another example, because $WF_{ERPS}(x,y)$ has the smallest values around the starting point, it has advantages for slow motion situations. Therefore, looking into the weighting function of a search algorithm, we understand why it works better for a particular situation (fast motion or slow motion).

## III. Algorithm Design

The analysis in the previous section gives us a clue that the weighting function is strongly related to the performance of a search algorithm, and, therefore, we like to design a fast search algorithm that has the smallest possible weighting function value at all locations. We do this in two steps. We first construct a target weighting function that, based on the prior knowledge, has the smallest possible values at all locations. Then, we design a search pattern that hopefully achieves the desired weighting function.

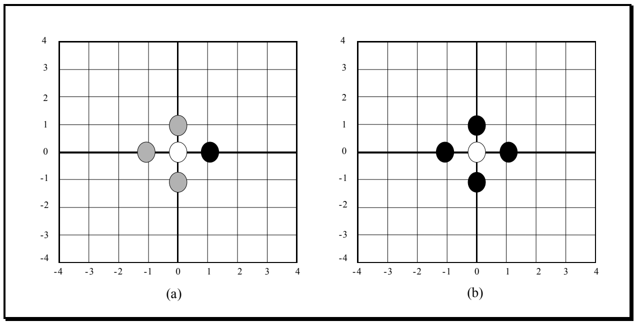

Figure 2. Search patterns used in GRPS

Most pattern-based search algorithms typically consist of two stages: 1) coarse initial search stage and 2) fine ending search stage. Generally, the coarse search stage focuses on finding a rough location of the optimal motion vector, and the fine ending search stage locates its precise location. We thus devise one search pattern for each stage. In the coarse search stage, because the shortest path between two points on a plane is the strait line, the fastest search path for a search algorithm is the strait line from the starting point directly to the best motion vector. Based on previous experiments, let us assume that a doable search method moves at most one unit distance horizontally or vertically per step, as shown in Figure 2. (a). Then, the minimal number of search points for reaching motion vector (x,y) is 'abs(x)+abs(y)+1'. At the ending stage, to decide the best motion vector generally requires examining at least the 4 neighboring points and the center point itself, as shown in Figure 2. (b). Consequently, the minimal number of search points for motion vector (x,y) can be expressed by (5) and its contour plot is illustrated by Figure 3. To our knowledge, $WF_{GRPS}(x,y)$ in (5) has the smallest value as compared to all the known search algorithms.

$$WF_{GRPS}(x, y) = \mathrm{Max}(5, 4 + \mathrm{abs}(x) + \mathrm{abs}(y)) \qquad (5)$$

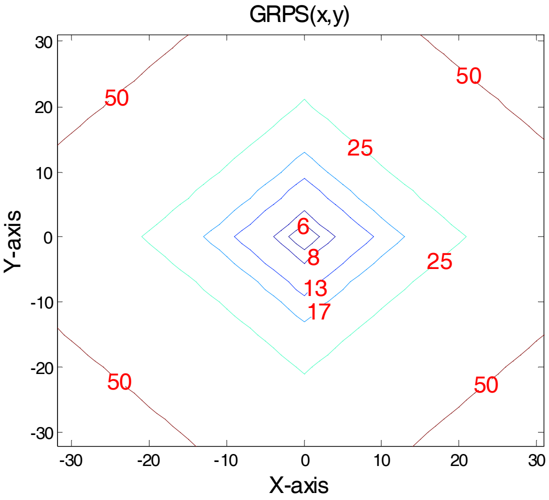

Figure 3. Weighting function of GRPS

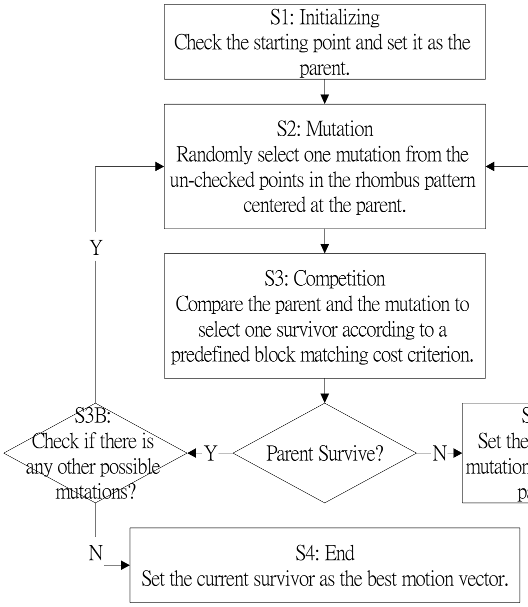
S1: Initializing
Check the starting point and set it as the parent.
S2: Mutation
Randomly select one mutation from the un-checked points in the rhombus pattern centered at the parent.
S3: Competition
Compare the parent and the mutation to select one survivor according to a predefined block matching cost criterion.
Parent Survive?
Y
N
S3A:
Set the surviving mutation as the next parent.
S3B:
Check if there is any other possible mutations?
Y
N
S4: End
Set the current survivor as the best motion vector.

Figure 4. Flow chart of the GRPS

The second step is choosing proper search pattern(s), which could match the target weighting function. Inspired by the genetic search algorithms in [10] and [11] and the rhombus search pattern, we propose a "genetic rhombus pattern search" (GRPS) to accomplish (5).

Flow chart of the proposed search algorithm is shown in Figure 4. Mainly, it consists of 4 stages: initializing stage, mutation, competition and end stage. Herein, in step 1 (S1), the starting point is PMV. In step 2, it uses a one-bit array to record if a point (location) is checked or not. And, a mutation is one of the unchecked points in Figure 2. (a). In step 3 (S3), the predefined block matching cost is defined as the sum of absolute difference (SAD) between the current image block and the reference image block in our simulations. And the point with smaller block matching cost survives. In step 3B (S3B), it inspects if all the locations in Figure 2. (b) are checked.

Figure 5. shows two examples of the GRPS process. The points with the same number denote the mutations from the same parent. Figure 5. (a) is the simplest search process. The initial search examines 1 points and the ending search examines 4 points. Figure 5. (b) shows a typical GRPS search process. In addition to the sole initial search point and 3 ending search points, GRPS may check 1, 2, 3 or 4 new points for a successful mutation. And if all the possible mutations in the rhombus pattern have been checked, and the current parent survives, then, the current parent is the final motion vector.

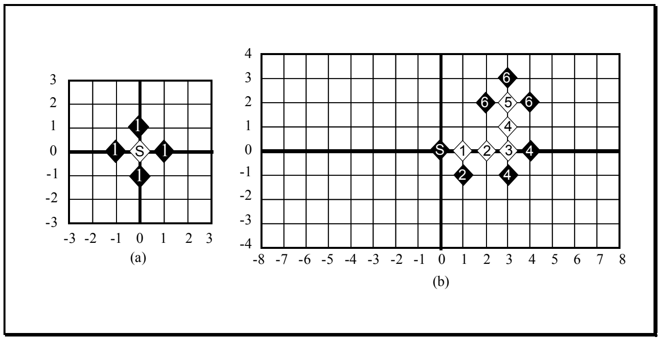

Figure 5. Examples of the GRPS search process

Moreover, the overhead of the proposed genetic search algorithm almost does not exist when it is compared with the other popular non-genetic search algorithms. In designing the proposed algorithm, we only adopt the concept but not the complete structure of genetic algorithm (for example, the reproduction stage is merged into the mutation stage) because the exact genetic algorithm is too complicated. The only extra burden is a random location generator in step 2 (S2), which can be simulated by a light-weighted pseudo one.

## IV. Experimental Results

We select a few representative sequences at various bit rates under the settings given in TABLE I. to test the proposed GRPS. The selected sequences are coded by an MPEG-4 SP@L3 encoder. All the sequences are in CIF (352X288) format. Only the first frame is coded as I frame, and all the remaining frames are coded as P frames. No frame skip is allowed. The motion vector search range is set to 16, the initial quantizers are set to 15, and the block size is

16x16. And, the encoder may vary the quantizers to achieve the desired target bit rate.

The average number of search points (ASP) and the peak signal noise ratio (PSNR) for various sequences and search algorithms are shown in TABLE II. and TABLE III. , respectively. And a pair-wise performance comparison is given in TABLE IV. In TABLE IV. , the computing gain (CG) is defined as the ratio of ASP minus one, and the quality gain (QG) is defined as the PSNR difference. In summary, the ASP of GRPS on the average is 26% faster than that of ERPS, 51% faster than EHS, 125% faster than DS, 160% faster than FSS, and 136 times faster than the full search (FS). And the PSNR of GRPS on the average is the same or better than all the other search algorithms (0~+0.18 dB). It is clear that GRPS outperforms all the other search algorithms in terms of ASP for all sequences, while its quality on the average is comparable to all the other algorithms.

TABLE I. SELECTED SEQUENCES AND THEIR SETTINGS.

| **Abbreviation** | ***Sequence*** | ***Bitrate (K bps)*** | ***Frame rate (fps)*** | ***Number of frames*** |
|---|---|---|---|---|
| ct40 | container | 40 | 7.5 | 300 |
| md96 | mother and daughter | 96 | 10.0 | 300 |
| cg112 | coastguard | 112 | 30.0 | 300 |
| fm512 | foreman | 512 | 30.0 | 300 |
| fb1024 | football | 1,024 | 30.0 | 90 |
| st1024 | steven | 1,024 | 30.0 | 300 |

TABLE II. ASP (AVERAGE NUMBER OF SEARCH POINTS).

| **ASP** | ***GRPS*** | ***ERPS*** | ***EHS*** | ***DS*** | ***FSS*** | ***FS*** |
|---|---|---|---|---|---|---|
| ct40 | 5.98 | 7.04 | 10.42 | 15.03 | 18.38 | 1024 |
| md96 | 5.98 | 6.83 | 10.32 | 14.85 | 18.37 | 1024 |
| cg112 | 6.08 | 7.63 | 10.31 | 15.09 | 18.25 | 1024 |
| fm512 | 7.13 | 8.65 | 10.76 | 16.17 | 19.03 | 1024 |
| fb1024 | 11.89 | 16.36 | 14.29 | 22.36 | 22.70 | 1024 |
| st1024 | 7.65 | 9.95 | 11.48 | 16.96 | 19.47 | 1024 |
| Average | 7.45 | 9.41 | 11.26 | 16.74 | 19.37 | 1024 |

TABLE III. PSNR (PEAK SINGNAL NOISE RATIO)

| **PSNR** | ***GRPS*** | ***ERPS*** | ***EHS*** | ***DS*** | ***FSS*** | ***FS*** |
|---|---|---|---|---|---|---|
| ct40 | 32.21 | 32.08 | 31.46 | 31.92 | 31.69 | 32.04 |
| md96 | 40.08 | 40.09 | 39.87 | 39.99 | 39.93 | 39.80 |
| cg112 | 29.14 | 29.16 | 29.07 | 29.14 | 29.13 | 29.08 |
| fm512 | 34.05 | 34.10 | 33.94 | 34.06 | 34.02 | 34.06 |
| fb1024 | 34.87 | 34.88 | 34.86 | 34.93 | 34.94 | 35.28 |
| st1024 | 29.39 | 29.31 | 29.47 | 29.44 | 29.35 | 29.48 |
| Average | 33.29 | 33.27 | 33.11 | 33.25 | 33.18 | 33.29 |

## V. CONCLUSIONS

A systematic approach is taken in this paper for designing a new search algorithm for pattern-based block motion estimation (PBME). We first propose a weighting function model that describes the minimal search points of a search algorithm. Based on the prior knowledge, we suggest a target weighting function that has (nearly) the smallest possible values. Then, we design a new genetic rhombus pattern search algorithm that accomplishes the desired weighting function. The simulations show that, indeed, this new algorithm saves more than 20% of computing power in terms of average search point when it is compared to the popular existing PBME algorithms. Being tested on the MPEG-4 SP codec, it has about the same level of PSNR performance.

TABLE IV. PERFORMANCE COMPARISON

| | ***GRPS over ERPS*** | | ***GRPS over EHS*** | | ***GRPS over DS*** | | ***GRPS over FSS*** | | ***GRPS over FS*** | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Gain** | ***CG*** | ***QG*** | ***CG*** | ***QG*** | ***CG*** | ***QG*** | ***CG*** | ***QG*** | ***CG*** | ***QG*** |
| ct40 | 0.18 | 0.13 | 0.74 | 0.74 | 1.51 | 0.28 | 2.07 | 0.51 | 170.24 | 0.16 |
| md96 | 0.14 | -0.02 | 0.73 | 0.20 | 1.48 | 0.08 | 2.07 | 0.15 | 170.24 | 0.27 |
| cg112 | 0.25 | -0.02 | 0.70 | 0.07 | 1.48 | 0.00 | 2.00 | 0.01 | 167.42 | 0.06 |
| fm512 | 0.21 | -0.05 | 0.51 | 0.12 | 1.27 | -0.01 | 1.67 | 0.03 | 142.62 | -0.00 |
| fb1024 | 0.38 | -0.01 | 0.20 | 0.01 | 0.88 | -0.06 | 0.91 | -0.06 | 85.12 | -0.41 |
| st1024 | 0.30 | 0.07 | 0.50 | -0.08 | 1.22 | -0.06 | 1.55 | 0.04 | 132.86 | -0.09 |
| Average | 0.26 | 0.02 | 0.51 | 0.18 | 1.25 | 0.04 | 1.60 | 0.11 | 136.42 | -0.00 |

## REFERENCES

[1] A. H. Sadka, *Compressed Video Communications*, John Wiley and Sons Ltd, 2002

[2] C. Zhu, W. S. Qi and W. Ser, "Predictive fine granularity successive elimination for fast optimal block-matching motion estimation", I*EEE Trans. Image Processing*, vol 14, no 2, Feb 2005.

[3] P. I. Hosur and K.-K. Ma, "Motion vector field adaptive fast motion estimation," presented at the 2nd I*nt. Conf. Information, Communications, and Signal Processing* (ICICS), Singapore, Dec. 1999, CDROM.

[4] Y. Nie and K.-K. Ma, "Adaptive rood pattern search for fast block-matching motion estimation", *IEEE Trans, Image Processing*, vol. 11, No. 12, Dec 2002.

[5] A.M. Tourapis, O.C. Au, and M.L. Liou, "New results on zonal based motion estimation algorithms-advanced predictive diamond zonal search", in 2001 *Proc. IEEE Int. Symp. Circuits and Systems* (ISCAS), vol 5, May 2001.

[6] C. Zhu, X. Lin, L. P. Chau, and L. M. Po, "Enhanced hexagonal search for fast block motion estimation," *IEEE Trans. Circuits Systems for Video Technology*, vol. 14, no. 10, pp. 1210-1214, Oct. 2004.

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

[8] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," in 1997 *Proc. Int. Conf. Information, Communications and Signal Processing* (ICICS), vol. 1, pp. 292-296, Sept. 9-12, 1997.

[9] C. H. Cheung and L.-M. Po, "Novel cross-diamond-hexagonal search algorithms for fast block motion estimation," *IEEE Trans. Multimedia*, vol 7, no 1, p.p.16-22, Feb. 2005

[10] C.-H. Lin and J.-L. Wu, "A lightweight genetic block-matching algorithm for video coding", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8, no 4, pp.386-392, Aug. 1998

[11] M. F. So and A. Wu, "Four-step genetic search for block motion estimation," in 1998 *Proc. IEEE Int. Conf. Acoustic, Speech, and Signal Processing* (ICASSP), vol. 3, p.p. 1393-1396, May 1998