A Survey on Block Matching Algorithms for Video Coding

Adapa Venkata Paramkusam, Vuyyuru Arun
Department of Electronics and Communication Engineering, MLR Institute of Technology, Hyderabad, India

Article Info

Article history:

Received Oct 7, 2016
Revised Dec 30, 2016
Accepted Jan 15, 2017

Keyword:

Block matching algorithm
Motion estimation
Motion vector
Search pattern

ABSTRACT

Block matching algorithm (BMA) for motion estimation (ME) is the heart to many motion-compensated video-coding techniques/standards, such as ISO MPEG-1/2/4 and ITU-T H.261/262/263/264/265, to reduce the temporal redundancy between different frames. During the last three decades, hundreds of fast block matching algorithms have been proposed. The shape and size of search patterns in motion estimation will influence more on the searching speed and quality of performance. This article provides an overview of the famous block matching algorithms and compares their computational complexity and motion prediction quality.

Corresponding Author:

Adapa Venkata Paramkusam,
Department of Electronics and Communication Engineering,
MLR Institute of Technology,
Hyderabad, India.
Email: adapa74@gmail.com

1. INTRODUCTION

The main objective of block matching algorithms is to determine the direction and magnitude of motion (motion vector) for a macroblock in the current frame relative to the best matched candidate block in the reference frame. The search for the best matching candidate block may be carried out by comparing the macroblock in the current frame with some or all the possible candidate blocks in a search window. Full-Search (FS) is an exhaustive search algorithm and it is the simplest method to find the optimal motion vectors for each macroblock. It determines the best matched candidate block through calculating the Sum of Absolute Difference (SAD) for all candidate blocks in the search window. Although Full-Search can obtain the global optimal result, it has very intensive computations.

To reduce the complexity of Full-Search, several fast search motion estimation algorithms have been proposed at the price of slightly impaired Peak Signal-to-Noise Ratio (PSNR) performance. A possible classification of these fast search motion estimation algorithms into the following categories: reduction in search positions [1-9], predictive search [10-14], search pattern switching [15], [16], multi-resolution search [17-20] and Fractional-Pixel Interpolation [21-22]. Existing fast search motion estimation algorithms use one or a combination of these categories. A popular example is the three-step search (TSS) algorithm [1]. However, its uniformly spaced search pattern is not well matched to most real-world video sequences in which the motion vector distribution is non-uniformly biased toward the zero vector. Such an observation inspired the new three-step search (NTSS) which has a centre-biased search pattern and supports a halfway-stop technique [2]. The centre-biased NTSS algorithm is an improved version of TSS, tends to achieve much superior performance with fewer number of search points on average. The search pattern has an important influence on speed and distortion performance in block motion estimation.

In TSS and NTSS algorithms square-shaped search patterns of different sizes are employed. The Diamond search (DS) algorithm adopts a diamond-shaped search pattern [3], which is more efficient than square-shaped search patterns TSS and NTSS. The search patterns in DS algorithm are derived from the checking points within circle of radium of 2 pels. The hexagon based search algorithm [4] was developed by investigating why the DS pattern can yield speed improvement over some square-shaped search patterns and what the mechanism behind is.

Hexagon-based search (HS) algorithm can achieve substantial speed improvement over the DS algorithm with similar distortion performance. The enhanced versions of hexagonal search algorithm [6-8] have been proposed for further reduction of the search points over Hexagonal Search algorithm. These algorithms mainly concentrate on the fast inner search technique to improve the inner search speed. The main purpose of this paper is to make a comprehensive studyof block matching algorithms. Comparisons in many directions are made for system designers to determine the best tradeoff. The rest of this paper is organized as follows. In section II, The well known block matching algorithms such as three step search, New three step search, Diamond search, Hexagon based search and enhanced versions of hexagonal search algorithm have been discussed. Section III is devoted to the discussions of the experimental results for various sequences. Finally, section IV concludes this paper.

2. OVERVIEW

2.1. Three Step Search (TSS) Block Matching Algorithm

Three Step Search (TSS) is one of the first non full search algorithms and was introduced by Koga et al [1] in 1981. It searches for the best motion vectors in three steps. The search pattern of TSS is shown in Figure 1. In the first step 8 blocks, at a distance of step size equal to or slightly larger than half of the maximum search range, from the centre point corresponding to zero motion vector are selected for comparison. In the second step the step size is halved, the centre point is moved to the point with the minimum distortion. Step-1 and step-2 are repeated till the step size becomes smaller than one. It is mainly used for real time video compression with low bit rate video application such as video conferencing and videophone.

The TSS is one of the most popular BMAs and is also recommended by RM8 of H.261 and SM3 of MPEG owing to its simplicity and effectiveness. For a maximum displacement window of 7 i.e. d=7, the number of checking points required is (9+8+8) =25. For a maximum displacement window of‘d’, the number of checking points required equals to [1+8{log2(d+1)}].

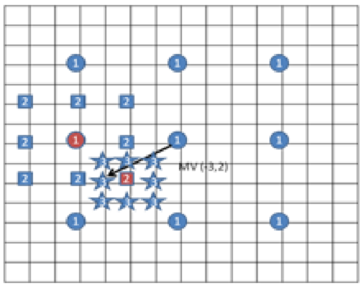


Figure 1. Example of Three Step Search Path to Locate a Motion Vector at (-3, 2)

2.2. New Three Step Search (NTSS) Block Matching Algorithm

This algorithm was proposed by Renxiang Li, Bing Zeng and Ming L.Liou [2]. It is a modified version of the three step search algorithm for searching small motion video sequences. For these video sequences, the motion vector distribution is highly centre biased. Therefore, in addition to the original checking points used in TSS, 8 extra neighbouring checking points of search window centre are searched in the first step of NTSS (total 17) as shown Figure 2. There are three cases where the minimum BDM point occurs.

a. For stationary block the minimum BDM point occurs at the search window centre then searching will stop (This is called first step stop).
b. For quasi stationary block (small motion video sequence) the minimum BDM point occurs at any one of the eight neighbours of the search window centre, the search in the second step will be performed only for eight neighbouring points of the minimum BDM point and then stop the search (This is called second step stop). Depending on position of this minimum BDM point, only five or three new checking points are

required to be tested. The number of checking points required is then either (17+3) =20 or (17+5) =22.

c. If the minimum BDM point occurs at any one of the remaining eight checking points then complete three step search will be executed. In the worst case (i.e. there is no single stationary block) the NTSS requires 33 block matches as compared to 25 matches in TSS. Eight block matches will be saved once a first step-stop occurs. Depending on the position of the minimum BDM point (in the first step) on the 8 neighbours of the window centre, five or three block matches will be saved once a second step-stop occurs:(1) if the minimum is one of the four neighbouring positions along the horizontal or vertical directions, five block matches will be saved;(2) if the minimum is one of the four neighbouring positions along the two diagonal directions, three block matches will be saved.

The number of block matches needed in NTSS for estimating a block motion vector can be estimated by 17P1 + 20P2 + 22P2' + 33 (1-P1-P2- P2'), where P1 is the probability of occurring a first step-stop while P2 and P2' are respectively probabilities of occurring a second step-stop in the two cases mentioned above. These probabilities are dependent on how many stationary or quasi-stationary blocks a video frame contains.

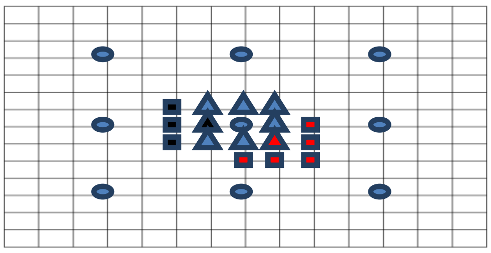


Figure 2. Circles are the checking points in the first step of TSS, triangles are the 8 extra points added in the first step of NTSS, and squares are the new checking points (3 or 5) in the second step depending on minimum BDM point, in the first step, on the 8 neighbors of the window centre

2.3. Diamond Search (DS) Block Matching Algorithm

The diamond search algorithm (DS) is proposed by S. Zhu and K. K. Ma [3].It is based on MV distribution of real world video sequences. It is an outstanding algorithm adopted by MPEG-4 verification model (VM) due to its superiority to other methods in the class of fixed search pattern algorithms. It employs two search patterns as shown in Figure 3, which are derived from the checking points within circle of radium of 2 pels. The first pattern, called large diamond search pattern (LDSP) comprises nine checking points and form a diamond shape. The second pattern consisting of five checking points forms a smaller diamond shape, called Small diamond search pattern (SDSP).The search starts with the LDSP which is centered at the origin of the search window, and the 9 checking points of LDSP are tested. If the minimum block distortion (MBD) calculated is not located at the centre position, new LDSP is formed which is centered at the MBD point found inthe search. The search procedure is repeated until MBD occurs at the centre point. At any stage if MBD occurs at centre of LDSP, the search pattern is switched 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.

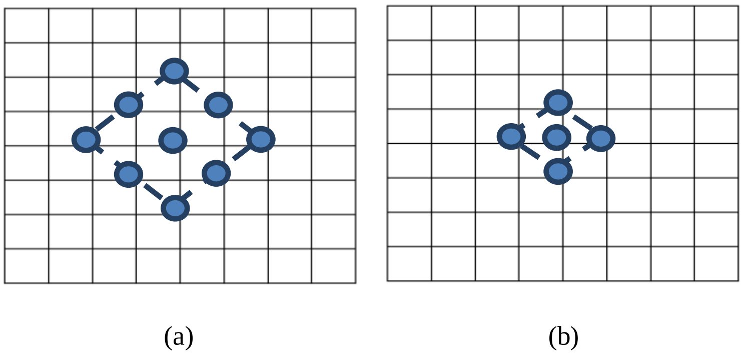


Figure 3. Search Patterns of DS (a) LDSP with 9 Search Points (b) SDSP with 5 Search Points

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 Figure 4. 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 Figure 4(a) and (b), respectively. If the centre 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 Figure 4(c).

2.4. Hexagon Based Search (HS) Block Matching Algorithm

Based on an in-depth examination of the influence of search pattern on speed performance in block motion estimation, Ce Zhu, Xiao Lin, and Lap-Pui Chau [4]. Propose an algorithm using a hexagon-based search pattern to achieve substantial speed improvement over the DS algorithm with similar distortion performance. The diamond shaped pattern is more efficient than square shaped search patterns, but the diamond shape is not approximate enough to a circle, which is just 90 rotation of a square. The 8 checking points have different distances from the centre, so that each search point cannot be equally utilized with maximum efficiency in DS algorithm.

A hexagon-based search pattern is depicted in Figure 5(a), which consists of seven checking points with the centre surrounded by six endpoints of the hexagon with the two edge points (up and down) being excluded. Of the six endpoints in the hexagon, two horizontal points are away from the centre with distance 2 and the remaining four points have a distance of from the centre point, respectively. From the Figure 5(a), we can see the six endpoints are approximately uniformly distributed around the centre, which is highly desirable.

This algorithm uses two search patterns large HS and small HS as shown in Figure 5(a) and 5(b). In the first step of search, the large hexagonal pattern with seven checking points is used for search. In the second step if the MBD is found at the centre, switch to use the small hexagonal pattern, including four checking points for the focused inner search. Otherwise, the search continues around the point with minimum block distortion (MBD) using the same large hexagonal pattern. Note that while the large hexagonal pattern moves along the direction of decreasing distortion, only three new non overlapped checking points will be evaluated as candidates each time. The total number of search points per block will be

$$NHS\ (mx, my) = 7\ X\ (3\ X\ n) + 4. \quad (1)$$

Where (mx, my) is the final motion vector found, and n is the number of executions of Step 2. In Equation 1 the digit 7 indicates number of checking points used in step 1, the digit 3 indicates number of checking points used in each execution ofstep 2 and the digit 4 indicates number of checking points used with small HS as a final stage.

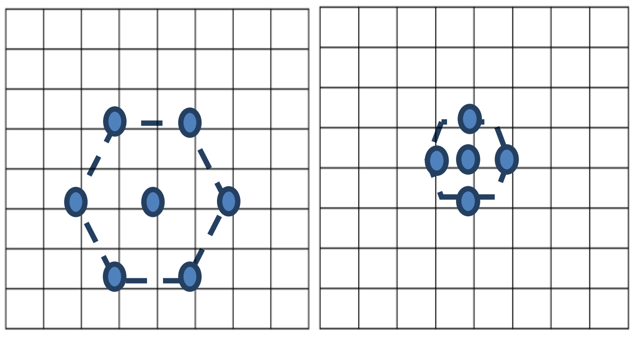


Figure 5. Search patterns ofHS (a) Large HS (b) Small HS

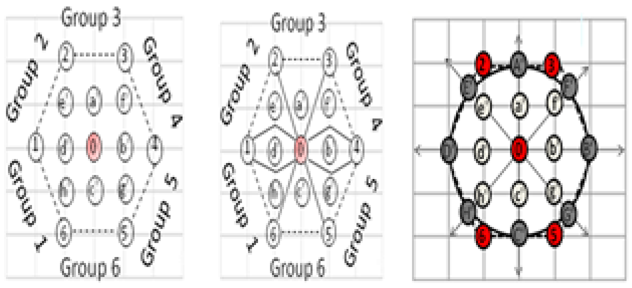
2.5. Enhanced Versions of Hexagonal Search Algorithm

An Enhanced Hexagonal Search (EHS) algorithm [6] is proposed for further reduction of the search points over Hexagonal Search algorithm. This EHS uses the 6-side-based fast inner search technique to improve the inner search speed against the Hexagonal Search. In Hexagonal Search algorithm, all the search points inside the large hexagon need to be evaluated, this is computationally ineffective.

The EHS algorithm only checks the most promising inner search points by exploiting the group-sum distortion information of the six checked endpoints of the large hexagon. At first EHS starts coarse search with the large hexagonal pattern (Figure 5(b)) to trace a small area of optimal motion vector. After locating a small area in the coarse search, EHS groups the six checked endpoints of the large hexagon as shown in Figure 6(a). At each group, the group-sum distortion is calculated and then focused inner checking points nearest to the smallest distortion group will be evaluated to obtain the minimum distortion point. Three inner searching points nearby the smallest distortion group will be calculated in the focused inner search if the smallest distortion group is 3 or 6. Otherwise, two inner searching points nearest to the smallest distortion group will be used in the focused inner search.

An Enhanced Hexagonal-based Search using Point-Oriented Inner Search (EHS-POIS) is presented in [7] to reduce the search points further. The EHS-POIS uses an efficient grouping method for the large hexagon which is based on minimizing Mean Internal Distance (MID) for each inner point, as shown in Figure 6(b). The eight inner points enclosed by large hexagon are divided into two sets by the MID value. First set contains a′, c′, e′, f′, g′ and h′ points and second set includes b′ and d′ points. These points are surrounded by two or three nearest checked large hexagon points.

The Normalized Group Distortions (NGDs) of the hexagon are calculated and the minimum NGDs in each set are found. The two inner points related to minimum NGDs in two sets are finally evaluated to find the final motion vector. The fine search of EHS requires 2 or 3 search points depending on the portion of inner points with smallest group distortion whereas the fine search of EHS-POIS requires only 2 search points constantly.



(a) (b) (c)

Figure 6. Strategies used to predict portion of inner searches in EHS, EHS-POIS and EHS-DOIS, (a) Group oriented prediction of EHS, (b) Point-oriented prediction of EHS-POIS, (c) Direction-oriented prediction of EHS-DOIS

An Enhanced Hexagonal-based Search using Direction-Oriented Inner Search (EHS-DOIS) [8] has shown the considerable improvement over EHS-POIS by increasing the inner search speed double. The EHS-DOIS forms pseudo-points prediction pattern {A′, B′, C′, D′, E′, F′, G′, H′} as shown in Figure 6(c). The group distortions of these eight pseudo-points are calculated through six checked points of large hexagon. Select one point in {a′, b′, c′, d′, e′, f′, g′, h′} that would be on the arrow from the center of large hexagon to the pseudo-point with minimum distortion among eightpseudo-points. The SAD at this selected point is finally evaluated to find the final motion vector.

3. COMPARISION

A comprehensive set of experiments have been conducted using the luminance components of the first 100 frames of six video sequences to assess the performance and computational complexity of state-of-the-art fast search motion estimation algorithms. These video sequences consist of different types of motion characteristics and have various formats including QCIF, CIF, and HD. The search range is set to ±63 for HD video sequences (Kirsten-Sara and Rocket launch sequences) and ±15 for the remaining (QCIF and CIF) video sequences.

The experimental results are shown in terms of two testing criteria: speed and motion prediction quality. The speed performance is shown in the Average Number of total Operations per Block (ANOB). For the latter, the average Peak Signal to Noise Ratio (PSNR) per frame is calculated. The size of a macroblock is set to 16 × 16 pixels in all the fast block matching algorithms.

One problem that occurs with the TSS is that it uses a uniformly allocated checking point pattern in the first step, which is not very efficient to search small motions appearing in stationary or quasi-stationary blocks. To remedy this problem the NTSS employs a centre-biased checking point pattern in the first step and halfway-stop technique. As compared to TSS, NTSS is much more robust, produces smaller motion compensation errors, and has a very compatible computational complexity.

Although NTSS uses more checking points in its first step as compared to TSS, the first step-stop and second step-stop can reduce computation effectively. Eight block matches will be saved once a first step-stop occurs and five or three block matches will be saved once a second step-stop occurs. From the experimental results conducted on Salesman and Miss America test sequences, TSS and NTSS are compared in terms of speed-up ratios with respect to Full search, Probabilities of catching true motions, and Average distances, as shown in Table I. For a search window of size 15x15, w=7. Full search will check (2w+1)2=225, while TSS check (1+8 log2(w+1)) = 25, thus leading to a speed-up ratio of 9. The speed-up ratios of NTSS with respect to full search are given in Table 1. Comparing with the results of TSS the NTSS basically possess rather comparable computational complexity. The probability that the true motion vector is found by NTSS and TSS are also presented in Table 1, from which it is seen that NTSS provides higher probabilities than those of TSS. The average distances calculated in NTSS are small as compared with TSS, this is because of centre-biased pattern is used in NTSS where as TSS employs uniformly distributed pattern.

In TSS and NTSS algorithms, square-shaped search patterns are employed, whereas the DS algorithm adopts a diamond-shaped search pattern, which has faster processing with similar distortion in comparison with TSS and NTSS. Compared with TSS, the DS pattern can find large motion blocks with fewer search points and also reduce its susceptibility to getting stuck in local optima due to its relatively large step size in horizontal and vertical directions.

Table 1. Comparison between TSS and NTSS (In terms of speed-up-ratios w.r.t.Full search, Probabilities of catching True motions, Average distances)

	Speed-up ratios		Probabilities		Average distances	
	Salesman	Miss America	Salesman	Miss America	Salesman	Miss America
TSS	9.0	9.0	0.951	0.535	0.369	1.193
NTSS	10.94	7.94	0.990	0.722	0.044	0.687

The compact shape of the DS pattern around the centre also yields fewer search points than NTSS for finding stationary or small motion vectors. The diamond pattern (large one) is so compact in terms of distance between neighboring points that there may exist some redundancy among the search points, especially in the beginning of lower resolution search. Consequently, such distribution of search points in DS pattern is inefficient in finding possible candidates in the next step. The reason for this disadvantage is that the diamond shape is not approximate enough to a circle, which is just 90 rotation of a square. The Hexagon based search pattern is more approximated to a circle with a uniform distribution of a minimum number of search points and each search point is equally utilized with maximum efficiency, where the redundancy among search points is removed maximally. This HS algorithm can find a same motion vector with fewer search points than the DS algorithm. Generally speaking, the larger the motion vector, the more search points the HS algorithm can save. The average number of search points per block with respect to different algorithms and different video sequences are shown in Table 2.

Table 2. The average number of search points per block with respect to block matching different algorithms and different video sequence

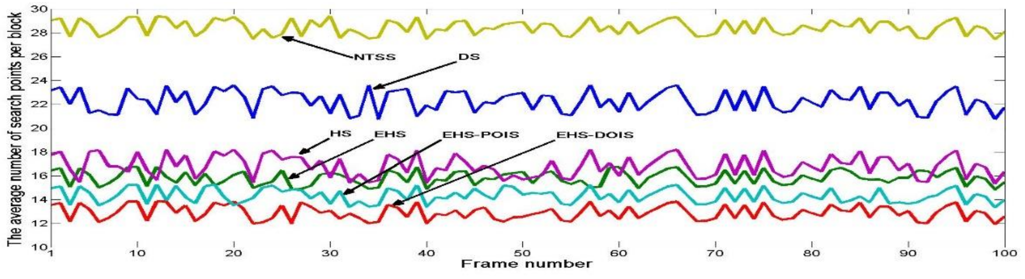
	Salesman	Miss America	Tennis	Mobile	Kirsten-Sara	Rocket launch
NTSS	18.0	21.3	22.7	28.4	23.8	24.9
DS	13.0	16.3	16.9	22.7	19.0	19.8
HS	10.7	12.8	13.0	16.3	13.9	14.8
EHS	10.3	12.2	12.8	15.8	13.3	14.1
EHS-POIS	10.1	12.0	12.3	15.3	13.0	12.8
EHS-DOIS	09.5	11.2	11.6	11.6	12.6	12.1

The search point number per block for the FS and TSS are fixed as 255 and 25 respectively for all video sequences. Compared with TSS, NTSS and DS search patterns theHS takes less number of search points in an average per block. The EHS-DOIS is the fastest among the compared algorithms, andits performance is in the most cases comparable with DSand HS.The average PSNR values per frame in all the fast block matching algorithms are furnished in Table 3. It can be observed from Table 3 that the average PSNRs obtained by DS are better than those of HSand enhanced versions of HSi.e., EHS, EHS-POIS and EHS-DOIS algorithms in all the cases. The total average PSNR of the DS is better by 1.11dB when compared to that of EHS-DOIS algorithm.

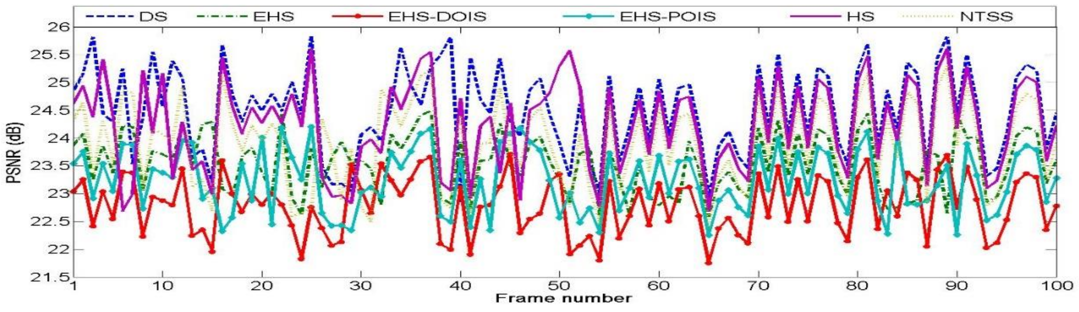
Table 3. The average PSNRs in dB for all the fastalgorithms

	Salesman	Miss America	Tennis	Mobile	Kirsten-Sara	Rocket launch
NTSS	26.52	32.32	24.22	23.32	44.13	37.43
DS	27.45	33.85	24.82	23.85	44.21	37.69
HS	26.83	32.69	24.45	23.63	44.04	37.12
EHS	26.64	32.55	24.56	23.54	43.68	36.86
EHS-POIS	26.31	32.23	24.33	23.21	43.34	36.54
EHS-DOIS	26.91	32.91	23.81	22.71	42.45	36.20

To comprehend the efficiency of all the fast block matching algorithms more vividly, speed and motion prediction quality of all the fast block matching algorithms using Mobile and Rocket launch video sequences have been plotted in Figure 7 and Figure 8 respectively. Figure 7(a) and Figure 8(a) plot a frame-by-frame comparison of the average number of search points per block for all the fast block matching algorithms applied to the Mobile and Rocket launch video sequences respectively. Figure 7(b) and Figure 8(b) plot a frame-by-frame comparison of average PSNR for all the fast block matching algorithms applied to the Mobile and Rocket launch video sequences respectively. Figure 7(a) and Figure 8(a) clearly manifest the superiority of the EHS-DOIS against other algorithms in terms of average number of search points, while Figure 7(b) and 8(b) show that the PSNR performance of the DS are better than that of EHS-DOIS.

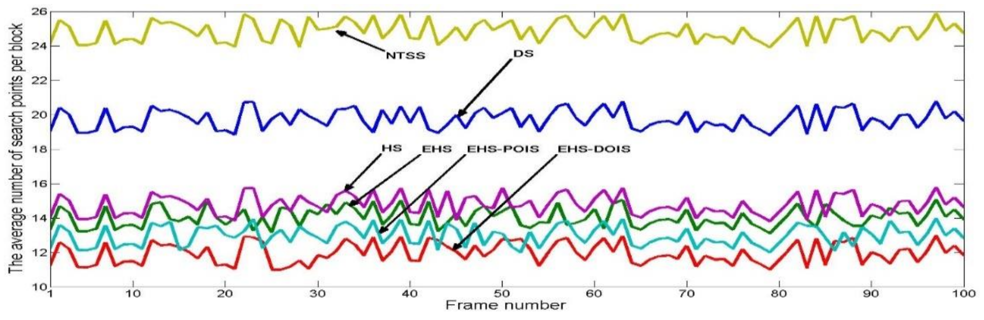


(a)

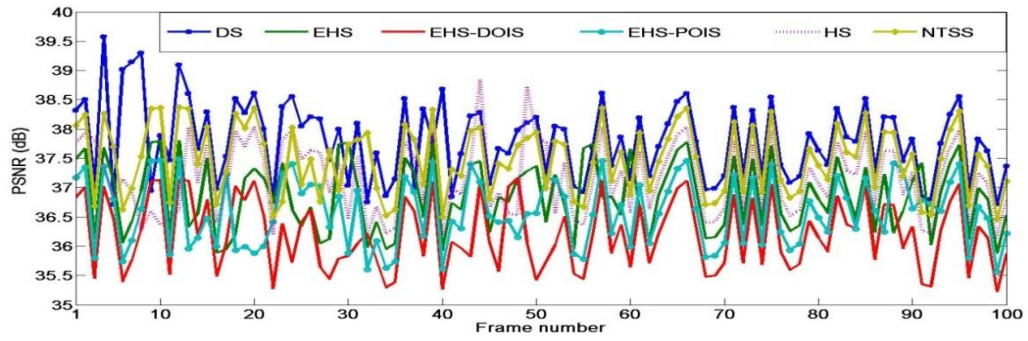


(b)

Figure 7. Performance comparison of the fast block matching algorithms for “Mobile” sequence in terms of: (a) theaverage number of search points per block and (b) average PSNR per frame



(a)



(b)

Figure 8. Performance comparison of the fast block matching algorithms for "Rocket launch" sequence in terms of: (a) theaverage number of search points per block and (b) average PSNR per frame

4. CONCLUSION

Motion estimation generally consumes the most computation in a video coding. In this paper, many fast BMA algorithmsbelonging to different search patterns and search strategies are analyzed. Performance and computational complexity of selected algorithms is compared to facilitate the choice of an appropriate algorithm to a specific application.

REFERENCES

[1] T. Koga, *et al.*, "Motion compensated interframe coding for video conferencing," in *Proc. Nat. Telecommun. Conf.*, pp. C9.6.1–C9.6.5, 1981.
[2] R. Li, *et al.*, "A New Three-step Search Algorithm for Block Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol/issue: 4(4), pp. 438–442, 1994.
[3] S. Zhu and K. K. Ma, "A New Diamond Search Algorithm for Fast Block-matching Motion Estimation," *IEEE Trans. Image Processing*, vol/issue: 9(2), pp. 287–290, 2000.
[4] C. Zhu, *et al.*, "Hexagon-based Search Pattern for Fast Block Motion Estimation," *IEEE Trans. on Circuits and Systems for Video Technology*, vol/issue: 12(5), pp. 349-355, 2002.
[5] C. H. Cheung and L. M. Po, "Novel Cross-Diamond-Hexagonal Search Algorithms for Fast Block Motion Estimation," *IEEE Trans.on Multimedia*, vol/issue: 7(1), pp. 16–22, 2005.
[6] C. Zhu, *et al.*, "Enhanced Hexagonal Search for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol/issue: 14(10), pp. 1210–1214, 2004.
[7] L. M. Po, *et al.*, "Novel point oriented inner searches for fast block motion estimation," *IEEE Trans.on Multimedia*, vol/issue: 9(1), pp. 9–15, 2007.
[8] B. J. Zou, *et al.*, "Enhanced Hexagonal-Based Search Using Direction-Oriented Inner Search for Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol/issue: 20(1), pp. 156–160, 2010.
[9] O. Ndili and T. Ogunfunmi, "Algorithm and Architecture Co-Design of Hardware-Oriented, Modified Diamond Search for Fast Motion Estimation in H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol/issue: 21(9), pp. 1214–1227, 2011.
[10] Z. Shi, *et al.*, "A motion estimation algorithm based on Predictive Intensive Direction Search for H.264/AVC," *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pp. 667, 672, 19-23 July 2010.

[11] Y. H. Ko, *et al.*, "Adaptive search range motion estimation using neighboring motion vector differences," *Consumer Electronics, IEEE Transactions on*, vol/issue: 57(2), pp. 726, 730, 2011.

[12] L. Hsieh, *et al.*, "Motion estimation using two-stage predictive search algorithms based on joint spatio-temporal correlation information," *Expert Systems with Applications*, vol/issue: 38(9), pp. 11608-11623, 2011.

[13] H. Nisar, *et al.*, "Content adaptive fast motion estimation based on spatio-temporal homogeneity analysis and motion classification," *Pattern Recognition Letters*, vol/issue: 33(1), pp. 52-61, 2012.

[14] Y. H. Ko, *et al.*, "Fast motion estimation algorithm combining search point sampling technique with adaptive search range algorithm," *Circuits and Systems (MWSCAS), 2012 IEEE 55th International Midwest Symposium on*, pp. 988, 991, 5-8 Aug 2012.

[15] K. H. Ng, *et al.*, "A Search Patterns Switching Algorithm for Block Motion Estimation," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol/issue: 19(5), pp. 753, 759, 2009.

[16] J. J. Tsai and H. M. Hang, "On the Design of Pattern-Based Block Motion Estimation Algorithms," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol/issue: 20(1), pp. 136, 143, 2010.

[17] F. Varray and H. Liebgott, "Multi-resolution transverse oscillation in ultrasound imaging for motion estimation," *Ultrasonics, Ferroelectrics, and Frequency Control, IEEE Transactions on*, vol/issue: 60(7), pp. 1333, 1342, 2013.

[18] J. Stuckler and S. Behnke, "Multi-resolution surfel maps for efficient dense 3D modeling and tracking," *Journal of Visual Communication and Image Representation*, vol/issue: 25(1), pp. 137-147, 2014.

[19] M. Nieuwenhuisen and S. Behnke, "Hierarchical planning with 3d local multiresolution obstacle avoidance for micro aerial vehicles," *Proceedings of the Joint Int. Symposium on Robotics (ISR) and the German Conference on Robotics (ROBOTIK)*, 2014.

[20] D. Droeschel, *et al.*, "Local multi-resolution representation for 6D motion estimation and mapping with a continuously rotating 3D laser scanner.In: Robotics and Automation (ICRA)," *IEEE International Conference on*, 2014.

[21] Y. Lin and Y. C. Wang, "Improved parabolic prediction-based fractional search for H.264/AVC video coding," *Image Process. IET*, vol/issue: 3(5), pp. 261–271, 2009.

[22] S. Dikbas, *et al.*, "Fast motion estimation with interpolation-free sub-sample accuracy," *IEEE Trans. Circuits Syst. Video Technol.*, vol/issue: 20(7), pp. 1047–1051, 2010.