International Journal of Engineering Research (ISSN:2319-6890)(online),2347-5013(print)
Volume No.3, Issue No.2, pp : 121-125 01 Feb. 2014

A New Survey on Block Matching Algorithms in Video Coding

L.C.Manikandan, Dr. R. K. Selvakumar

Asst. Professor, School of CSE, Mar Ephraem College of Engineering& Technology, MarthandamTamilnadu, India

Professor, Dept. of IT,Agni College of Technology, Chennai, Tamilnadu, India
E-mail: lcmanikandan@gmail.com, rkselvam@rediffmail.com

Abstract-Block matching motion estimation is the essence of video coding systems. This paper is a survey of the existing block matching algorithms used for motion estimation in video coding. The algorithms that are surveyed in this paper are widely accepted by the video coding community and have been used in implementing various standards, ranging from MPEG1/H.261 to MPEG10/H.264. The aim of this review is to provide the succeeding researchers with some constructive information in design of the fast motion estimation (ME) algorithms.

Keywords-Blockmatching, Motion Estimation, Video Coding, Absolute Mean Difference, MPEG, H.261, H.264.

I. Introduction

Video Coding is a necessary function of Recording Video and TV signals onto a Computer Hard Drive. Because raw Video footage requires lots of space, without Video Coding, Video files would quickly eat up gigabytes of hard drive space, which would result in only short amounts of Video or TV recorded onto the Computer's Hard Drive. With Video Coding, smaller Video files can be stored on your PCs Hard Drive, resulting in much more space for Video files. In other words, Video Coding lets you store much more TV and video on your Computer than if the Video files were not compressed. There are several kinds of video and audio coding formats, also known as codecs. Familiar coding formats are MPEG-1, MPEG-2, MPEG-4, H.261, H.263 and H.264/AVC.Through this study we reviewed the Block Matching Algorithms(BMA), Cross-Search Algorithm[1], Full Search Motion Estimation[3],Three Step Search Algorithm[5], New Three-Step Search Algorithm[5], Four-Step Search Algorithm[7], Diamond Search Algorithm[8], Cross Diamond Search Algorithm[9], Hexagonal Search[10], Spiral Search[6,18], Adaptive Rood Pattern Search[11].

II. Block Matching Algorithms (BMA)

Block Matching Algorithm (BMA) is the most popular motion estimation algorithm. The block-matching algorithms eliminate the temporal redundancy, which is found predominantly in any video sequence. The idea behind block matching is to divides frames into equal sized non-overlapping blocks and calculates the displacement of the best-matched block from the previous frame as the motion vector of the block in the current frame within the search window. During block matching, each target block of the current frame is compared with a previous frame in order to find the best matching block. Block-matching algorithms calculate the best match using Mean Absolute Difference (MAD). BMA algorithm is illustrated in Figure 2.1.

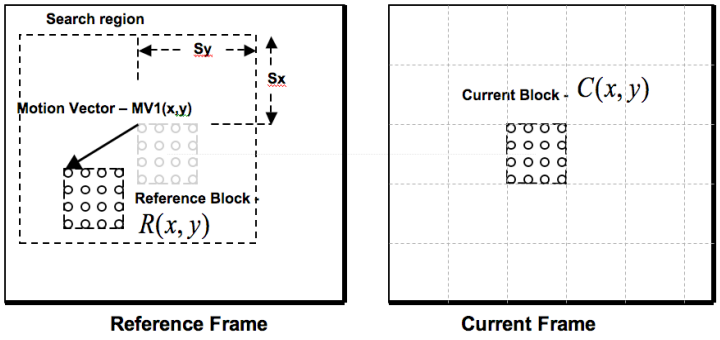

Figure 2.1 Block matching algorithm

2.1 Cross-Search Algorithm

The cross search algorithm (CSA) has been proposed by Ghanbari in 1990. It is also a logarithmic step search algorithm using a (X) cross searching pattern in each step. The basic idea is still a logarithmic step search where in each search step only 4 locations are tested. The cross search algorithm can then be described as follows:

Step1:The current block and the block at (0,0), are compared and if the value of the distortion function is less than a predefined threshold T then the current block is classified as a nonmoving block and the search stops. Otherwise go to Step 2.

Step 2:Initialize the minimum position (m,n) at m = 0, n = 0 and set the search step size p equal to half of the maximum motion displacement w, i.e., p = w/2.

Step 3:Move the coordinates (i, j) to the minimum position (m,n), that is i = m and j = n.

Step4:Find the minimum position (m, n) of the coordinates(i,j) , (i - p , j - p) , (i - p , j + p) , (i + p , j - p) and (i + p , j +P).

Step 5:If p = 1 go to Step 6, otherwise halve the step size p, and then go to Step 3.

Step 6:If the final minimum position (m, n) is either (i, j), (i - 1, j - 1) or (i + 1, j + 1) go to Step 7, otherwise go to Step 8.

Step 7:Search for the minimum position at (m,n), (m - 1, n), (m,n - l), (m + 1, n) and (m,n + 1). Here the end points of a Greek cross (+) are searched.

Step 8: Search for the minimum position at (m,n), (m- 1, n - l), (m - 1, n + l), (m + 1, n - 1) and (rn + 1, n + 1). In this case the end points of a St. Andrew's cross (X) are searched.

CSA for a maximum motion displacement of w = 8pels/frame is shown in Figure 2.2.

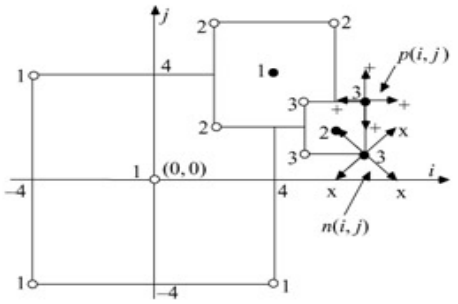

Figure 2.2An example of the CSA search for w=8 pels/frame

In CSA almost 25% (one in every four) of the pels at the boundaries are not searched.

2.2 Full Search Motion Estimation (FSA)

In order to get the best match block in the reference frame, it is necessary to compare the current block with all the candidate blocks of the reference frames. Full search motion estimation calculates the sum absolute difference (SAD) value at each possible location in the search window. Full search computed the all candidate blocks intensive for the large search window.Full search algorithm is illustrated in Figure 2.3.

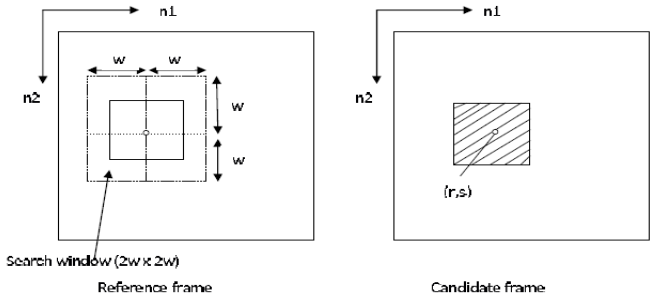

Figure 2.3 Full search motion estimation

2.3 Three Step Search Algorithm (3SS)

3SS algorithm was proposed by Koga et al. [2], this is a fine-coarse search mechanism. The 3SS algorithm can then be described as follows:

Step1:It involves search based on 4-pixel/4-line resolution at nine locations i.e. 9x9 search window, with the center point corresponding to zero MV.

Step 2: It involves search based on 2-pixel/2-line resolution i.e. 5x5 search window around the location determined by the first step.

Step 3: This is repeated in the third step with 1-pixel/1-line resolution and a search window of 3x3. Step4: The last step yields the MV.

The search pattern of TSS is shown in Figure 2.4.

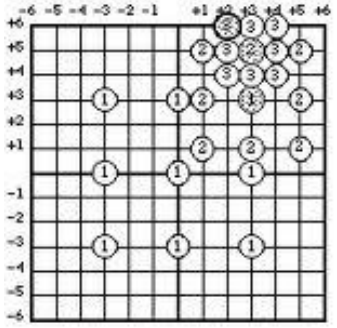

Figure 2.4Search pattern of three step search algorithm

2.4 New Three-Step Search Algorithm (N3SS)

N3SS [4] improves on 3SS results by providing a center biased searching scheme and having provisions for half way stop to reduce computational cost. It was one of the first widely accepted fast algorithms and frequently used for implementing earlier standards like MPEG 1 and H.261. The N3SS algorithm is described as follows.

Step 1:Totally points are checked including the central nine points on the 3 x 3 grid and the eight neighboring points on the 9 x 9 grid. If the minimum BDM point is the search window center, the search will be terminated; otherwise go to Step2.

Step 2: If one of the central eight neighboring points on the 3 x 3 grid is found to be the minimum in the first step, go to Step 3; otherwise go to Step 4.

Step 3: Move the small 3 x 3 search window so that the window center is the winning point found in Step1. Search additional five or three points according to the location of the previous winning point, then the search will stop.

Step 4:Reduce the large 9 x 9 search window size by half and move the center to the minimum BDM point in Step 1, follow the searching process of Step 2 and Step 3 in 3SS.

Figure2.5 shows two different search paths for finding motion vector within 5 x 5 area.

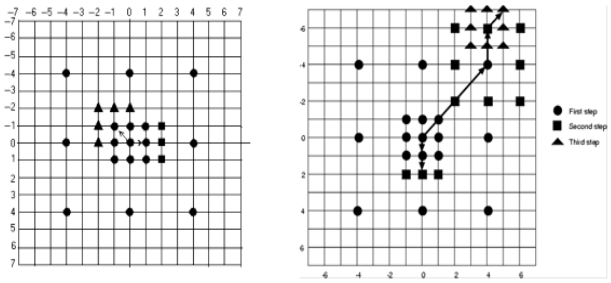

Figure 2.5 Two different search paths for finding MV within 5 x 5 area in N3SS.

2.5 Four-Step Search Algorithm (4SS)

The 4SS algorithm is summarized as follows:

Step 1:A minimum BDM point is found from a nine-checking point's pattern on a 5 x 5 window located at the center of the 15 x 15 searching area as shown in Figure 2.6(a). If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 2.

Step 2:The search window size is maintained in 5 x 5. However, the search pattern will depend on the position of the previous minimum BDM point.

a) If the previous minimum BDM point is located at the corner of the previous search window, five additional checking points as shown in Figure 2.6(b) are used.

b) If the previous minimum BDM point is located at the middle of horizontal or vertical axis of the previous search window, three additional checking points as shown in Figure 2.6(c) are used. If the minimum BDM point is found at the center of the search window, go to Step 4; otherwise go to Step 3.

Step 3: The searching pattern strategy is the same as Step 2, but finally it will go to Step 4.

Step 4:The search window is reduced to 3 x 3 as shown in Figure 2.6(d) and the direction of the overall motion

vector is considered as the minimum BDM point among these nine searching points.

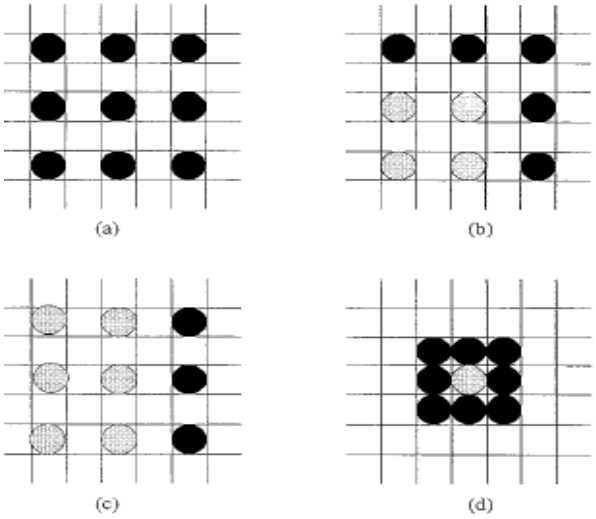

Figure 2.6 Search Patterns of 4SS. (a) First Step, (b) second/third step, (c) second/third step, (d) fourth step

Two examples of 4SS are shown in Figure 2.7with different search paths.

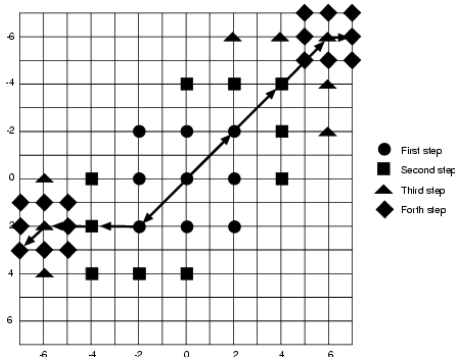

Figure 2.7 Two different search paths of 4SS

2.6 Diamond Search Algorithm (DS)

DS algorithm employs two search patterns as illustrated in Figure 2.9, which are derived from the crosses (×) in Figure 2.8. 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).

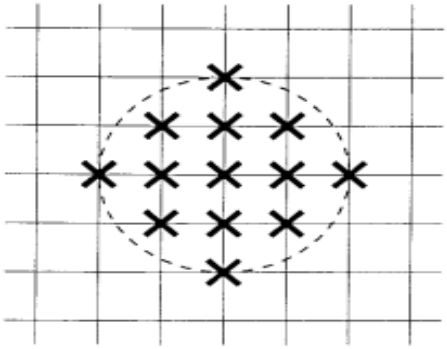

Figure 2.8An appropriate search pattern support—circular area with a radium of 2 pels.

The 13 crosses show all possible checking points within the circle.

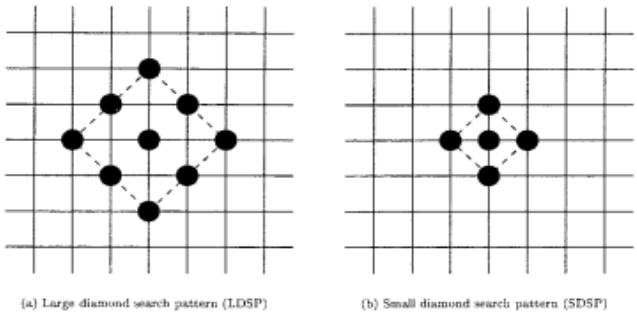

Figure 2.9 Two search patterns derived from Figure 2.8

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.

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.

2.7 Cross Diamond Search Algorithm(CDS)

The DS algorithm uses a large diamond-shaped pattern (LDSP) and small diamond-shaped pattern (SDSP), as depicted in Figure 2.10.

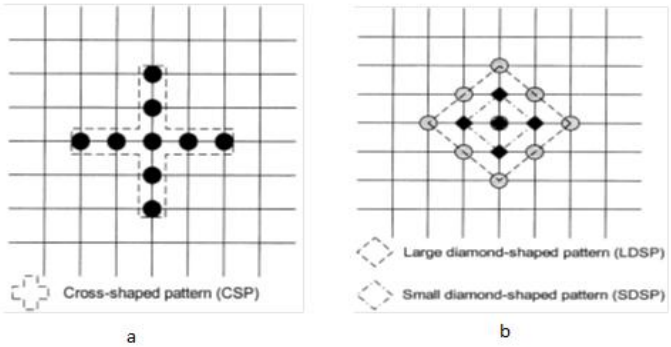

Figure 2.10 Search patterns used in the CDS algorithm.(a) CSP. (b) LDSP andSDSP.

Below summarizes the CDS algorithm:

Step 1: Starting: A minimum BDM is found from the nine search points of the CSP located at the center of search window. If the minimum BDM point occurs at the center of the CSP, the search stops. Otherwise, go to Step (2).

Step 2:Half-diamond Searching: Two additional search points of the central LDSP closest to the current minimum of the central CSP are checked, i.e., two of the four candidate points located at (±1, ±1). If the minimum BDM found in previous step located at the middle wing of the CSP, i.e., (±1, 0) or (0, ±1) and the new

minimum BDM found in this step still coincides with this point, the search stops. Otherwise, go to Step (3).

Step 3: Searching: A new LDSP is formed by repositioning the minimum BDM found in previous step as the center of the LDSP. If the new minimum BDM point is still at the center of the newly formed LDSP, then go to Step (4) (Ending); otherwise, this step is repeated again.

Step 4:Ending: With the minimum BDM point in the previous step as the center, a new SDSP is formed. Identify the new minimum BDM point from the four new candidate points, which is the final solution for the motion vector.

2.8 Hexagonal Search (HEXBS)

The HEXBS algorithm is summarized as follows:

Step 1: The large hexagon with seven checking points is centered at, the center of a predefined search window in the motion field. If the MBD point is found to be at the center of the hexagon, proceed to Step (3) (Ending); otherwise, proceed to Step (2) (Searching).

Step 2: With the MBD point in the previous search step as the center, a new large hexagon is formed. Three new candidate points are checked, and the MBD point is again identified. If the MBD point is still the center point of the newly formed hexagon, then go to Step (3) (Ending); otherwise, repeat this step continuously.

Step 3:Switch the search pattern from the large to the small size of the hexagon. The four points covered by the small hexagon are evaluated to compare with the current MBD point. The new MBD point is the final solution of the motion vector.

A hexagon-based search pattern is depicted in Figure 2.11.

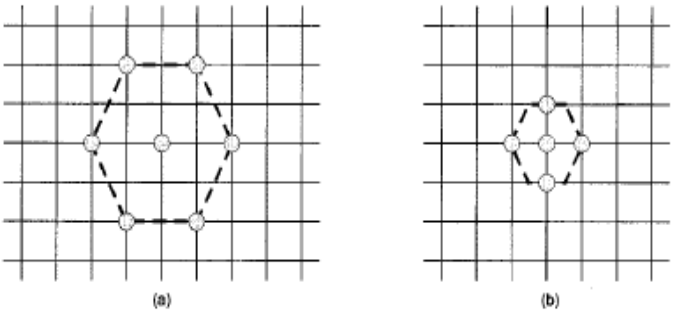

Figure 2.11(a) Large Hexagonal Block Search (HEXBS), (b) Small Hexagonal Block Search (HEXBS)

2.9 Spiral Search (SSA)

The spiral search algorithm was proposed by Zahariadis and Kalivas in 1995. It seeks to combine the ideas of the Three Step Search and the Binary search. By doing so, it tends to not only speed up the computation, but also removes the problem of the Binary search, where there is a zone of pixels that is never evaluated. The SSAalgorithm is summarized as follows:

Step1:The step size is picked to half the maximum displacement in the search window. The point of minimum distortion is found from among the nine points picked in the following manner. Five points are picked in the shape of a "+" around the centre of the search window (at a distance of step size in the vertical and horizontal directions). The remaining four points are picked at the corners of the search window.

Step 2: The step size is reduced and a search is performed around the point with the smallest distortion. This is repeated till the step size falls to 1.

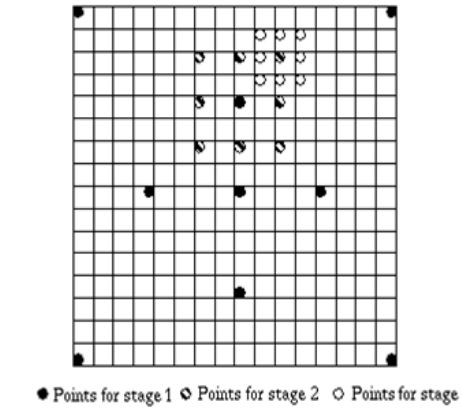

Figure 2.12Example path for convergence of Spiral Search

2.10Adaptive Rood Pattern Search (ARPS)

The ARPS algorithm is summarized as follows:

Step 1: Compute the matching error (SAD centre) between the current block and the block at the same location in the reference frame.

Step 2: Align the center of ARP with the center point of the search window and check it's four search points plus the position of the predicted MV to find out the current minimum matching error (MME) point.

Step 3: Set the center point of the unit-size rood pattern (URP) at the MME point found in the previous step and check its points. If the new MME point is not incurred at the center of the current URP, repeat this step; otherwise, the MV is found, corresponding to the MME point identified in this step.

ARPS algorithm search pattern is as shown in Figure 2.13.

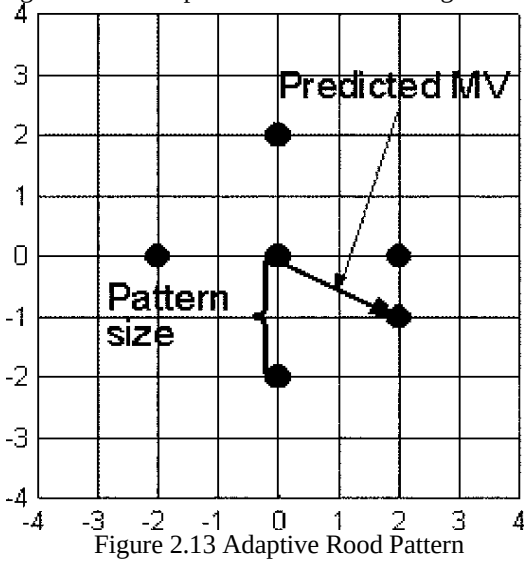

Figure 2.13 Adaptive Rood Pattern

III. Comparative study

Here we are categorizing the algorithms into nonlinear method and linear method based on maximum searching points. Most of the algorithms follow nonlinear approach. In our survey CDS algorithm only follow linear search approach.The Table 3.1 shows the computational complexity of the searching points.

Table 3.1 Comparison Table – Maximum Searching Points.

Method	Algorithm	Maximum Searching Time	Maximum number of search points – Window Size W = 4, 8, 16		
			4	8	16
Non - Linear Model	CSA	$5 + 4 \log_2 w$	13	17	21
	FS	$(2w + 1)^2$	81	269	1089
	3SS	$1 + 8 \log_2 w$	17	25	33
	N3SS	$[1+8 \log_2 w] + 8$	25	33	41
	4SS	$18(\log2(w/4))+9$	81	289	1089
	DS	$9+msx\{5,4,3\}*\log_2 w$	19	24	29
	HEXBS	$7+3*\log_2 w+4$	17	20	23
	SSA	$1 + 8 \log_2 w - 1$	17	25	33
	ARPS	$1+4*\log_2 w$	9	13	17
Linear Model	CDS	$3 + 2w$	11	19	35

IV. Conclusion

The past decades have seenthe growth of wide acceptance of multimedia. Video coding plays an important role in archival of entertainment based video (CD/DVD) as well as real-time reconnaissance / video conferencing applications. Block matching techniques are the most popular and efficient of the various motion estimation techniques. In this paper we presented an overview of Block matching motion estimation algorithms in video coding, with their comparative study at the end.

Acknowledgment

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

References

i. *M. Ghanbari, "The Cross-Search Algorithm for Motion Estimation", Vol. 38, No. I, pp 950-953, IEEE Transactions on Communications, July 1990.*

ii. *J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," IEEE Trans. on Communications, Vol. COM-29, No. 12, pp. 1799-1808, Dec. 1981*

iii. *.S. Immanuel Alex Pandian et al., "A Study on Block Matching Algorithms for Motion Estimation", Vol.3, No.1, pp 34-44, International Journal on Computer Science and Engineering, Jan 2011.*

iv. *T. Koga, K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro, "Motion compensated interframe coding for video conferencing," pp. G5.3.1-5.3.5, Pro. Nat. Telecommun. Conf., New Orleans, Nov. 1981.*

v. *Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", IEEE Trans. Circuits And Systems For Video Technology, Vol 4., No. 4, pp. 438-442, August 1994.*

vi. *Deepak Turaga, Mohamed Alkanhal, "Search Algorithms for Block-Matching in Motion Estimation" Spring, 1998.*

vii. *Lai-Man Po Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", vol. 6, No. 3, pp. 313-317, IEEE Transactions on Circuits Syst. Video Technol., June 1996.*

viii. *Shan Zhu and Kai-Kuang Ma, "A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", Vol. 9, No. 2, pp 287-290, I EEE Transactions on Image Processing, FEBRUARY 2000.*

ix. *Chun-Ho Cheung and Lai-Man Po, "A Novel Cross-Diamond Search Algorithm for Fast Block Motion Estimation", Vol. 12,No. 12, pp 1168-1177, IEEE Transactions on Circuits and Systems for Video Technology, December 2002.*

x. *Ce Zhu, Xiao Lin, and Lap-Pui Chau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation", Vol. 12, No. 5, pp 349-355, IEEE Transactions on Circuits and Systems for Video Technology, May 2002.*

xi. *Yao Nie, and Kai-Kuang Ma, "Adaptive Rood Pattern Search for Fast Block Matching Motion Estimation", Vol 11, no. 12, pp. 1442-1448, IEEE Transactions on Image Processing, December 2002.*

xii. *Aroh Barjatya, "Block Matching Algorithms For Motion Estimation", Student Member, IEEE, DIP 6620 Spring 2004.*

xiii. *Lurng-Kuo Liu and Ephraim Feig, "A block based gradient descent search algorithm for block motion estimation in video coding," Vol. 6, No. 4, pp. 419-422, IEEE Trans. on Circuits and Systems for Video Technology, Aug. 1996*

xiv. *Chandra Sekhar. CH, J.V.K. Ratnam, "Comparison of Fast Block Matching Algorithms for Motion Estimation", pp 1609-1618, International Journal of Electronics and Computer Science Engineering.*

xv. *M. Ezhilarasan and P. Thambidurai, "Simplified Block Matching Algorithm for Fast Motion Estimation in Video Compression " vol. 4, pp. 282-289, Journal of Computer Science, 2008.*

xvi. *A. Barjatya, "Block Matching Algorithms for Motion Estimation," DIP 6620, Spring 2004.*

xvii. *Kwon Moon Nam, Joon-Seek Kim, Rae-Hong Park "A Fast Hierarchical Motion Vector Estimation Algorithm Using Mean Pyramid" Vol.5, No.4, pp 344-351, IEEE Transactions on Circuits and Systems for Video technology, August 1995.*

xviii. *Th.Zahariadis and D.Kalivas, "A Spiral search algorithm for fast estimation of block motion vectors, 1995.*

xix. *Kyoung Won Lim, Jong Boem Ra "Improved Hierarchical Search Block Matching Algorithm byUsing Multiple Motion Vector Candidates" vol. 33, no.21, pp 1771- 1772, Electroonic Letters, October, 1997.*

xx. *MPEG-4 Video Verification Model (Version 14.0), ISO/IEC JTC1/SC29/WG11 N2932, Oct. 1999.*

xxi. *R. Srinivasan and K. R. Rao, "Predictive coding based on efficient motion estimation," vol. COM-33, pp. 888–896, IEEE Trans. Commun., Aug. 1985.*