International Journal of Advances in Computer Networks and its Security

Architecture for the ARPS Algorithm

Sanipini Venkata Kiran, Kantipudi MVV Prasad, N.V.Lalitha, Dr.D.S.Rao
Department of Electronics and Communication Engineering
GIET, Rajahmundry
Email:prasad.rku@gmail.com

Abstract:-The Objective of my paper is to develop an efficient architecture for the ARPS algorithm that could reach the real time performance for HDTV applications. The architecture is targeted for FPGA. At present we have developed the architecture which could achieve half the real time performance of 15 frames per second. The architecture is based on six processing elements and is also extendable to any number of processing elements with little modifications. The proposed architecture could reach the real time performance if it could employ more number of processing elements. As we double the number of processing elements the frame rate doubles upto certain point.

Keywords: BMA's, PSNR, ASIC, HDTV

I. INTRODUCTION

Video compression is vital for efficient storage and transmission of digital signal. The hybrid video coding techniques based on predictive and transform coding are adopted by many video coding standards such as ISO MPEG-1/2 and ITU-T H.261/263, owing to its high compression efficiency. Motion compensation is a predictive technique for exploiting the temporal redundancy between successive frames of video sequence. Block matching techniques are widely used motion estimation methods to obtain the motion compensated prediction. By splitting each frame into macro blocks, motion vector of each macro block is obtained by using block matching algorithm (or motion estimation algorithm). In order to get motion vector of each macro block, the most obvious and simplistic method is full search algorithm. All possible displacements in the search window are evaluated using block-matching criteria (cost function). The advantage of full search is that we can find the absolute optimal solution. However, its high computational complexity makes it impossible for real-time implementation. Because the computational complexity of video compression, the compression efficiency and the compression quality is determined by the motion estimation algorithm, development of Fast Motion Estimation Algorithm for real-time application becomes compelling.

The computational complexity of a motion estimation technique can then be determined by three factors: 1. Search algorithm. 2. Cost function/Evaluate function. 3. Search range parameter *p*. actually, we can reduce the complexity of the motion estimation algorithms by reducing the complexity of the applied search algorithm and/or the complexity of the selected cost function. The full search algorithm evaluates all the weights in the search window and a more efficient, less complex search algorithm will decrease the search space.
Motion estimation is a technique to eliminate temporal redundancy of image sequences and it is a central part of the MPEGs (1/2/4) [1-3] and the video compression standards (H.261/H.263) [4]. However, motion estimation involves a lot of computational complexity in the video encoders. As motion estimation occupies no less than 70-75 percent of the entire processing time of a video coder, it is worthwhile to go for hardware architecture for motion estimation to achieve a fast encoder. Many motion estimation algorithms and architectures [6-13] have been proposed in the literature. Therefore, it usually requires hardware and becomes the bottleneck in the real-time applications, it seems that the motion estimation implementation will determine the hardware characteristics, and motion estimation was implemented by dedicated hardware, it has a limitation on flexibility. The requirement for applications differs significantly according to image encoding algorithms. Otherwise, the algorithms will not show the desired results. It is necessary to discuss trade-off for algorithms and architectures of motion estimation, applications, and take into account design issues, chip area, speed, I/O bandwidth, memory band width, power consumption, image quality, and some requirements for application. Many motion estimation algorithms and architectures have been proposed in the literature. One of the main research goals has been the reduction of computational complexity and the power consumption of the motion estimation while keeping quality of image.

There are two mainstream techniques of motion estimation: pel-recursive algorithm (PRA)[5] and block-matching algorithm (BMA). PRAs are iterative refining of motion estimation for individual pels by gradient methods. BMAs assume that all the pels within a block has the same motion activity. BMAs estimate motion on the basis of rectangular blocks and produce one motion vector for each block. PRAs involve more computational complexity and less regularity, so they are difficult to realize in hardware.

In general, BMAs are more suitable for a simple hardware realization because of their regularity and simplicity. The block-matching algorithms (BMA's) [6-9] for motion estimation can be realized by two different approaches:

programmable processors and dedicated hardware. Programmable general purpose processors are very flexible, but this advantage is not essential for BMA's because of their fixed computation types. Moreover, the very high computational complexity requires multiple high-performance (thus high-cost) video signal processors. In contrast, mapping the BMA to a dedicated architecture provides a less flexible yet much more efficient solution because it can be optimized for this special purpose. For this reason, dedicated motion estimation architectures have been applied to several video coding chip sets for low bit-rate applications. Furthermore, dedicated BMA chips or modules can be combined with programmable processors to provide flexible video encoders with adequate throughput rates.

Many BMA's have been developed to reduce the extremely high computational complexity of the optimal full search (FS) procedure. Among the BMA's, the Three Step Search algorithm (3SS) [7] became the most popular one and it is also recommended by CCITT RM8 [4] of H.261 and SM3 [1] of MPEG owing to its simplicity and effectiveness. However, the 3SS algorithm uses a uniformly allocated checking point pattern in its first step, which becomes inefficient for the estimation of small motions. The block motion field of real world image sequence is usually gentle, smooth, and varies slowly. It results in a center-biased global minimum motion vector distribution instead of a uniform distribution. Based on the characteristic of center-biased motion vector distribution, a New Three Step Search (N3SS) algorithm [9] for improving the performance of the 3SS algorithm on the estimation of small motions was proposed. The N3SS algorithm employed a center-biased checking point pattern in the first step that combined the original checking point used in 3SS and eight extra neighboring points of the search window center. The N3SS algorithm uses a halfway-stop technique to speed up the stationary or quasi-stationary blocks matching. The N3SS algorithm is much more robust, and produces smaller motion compensation errors as compared with the 3SS algorithm. For the maximum motion displacements of +/- 7, however, the N3SS algorithm in the worst case requires 33 block matches while 3SS needs only 25 block matches. For some image sequences with a lot of large motions, the computational requirement of N3SS algorithm may be higher than that of the 3SS algorithm. In addition, for the real-time or VLSI implementation of motion estimation, the worst case computational requirement should be considered instead of the average computation

II. Adaptive Rood Pattern Search Algorithm

In order to obtain the accurate MV prediction of the current block two factors need to be considered: 1) Choice of the Region Of Support (ROS) that consists of the neighboring blocks whose MVs are used to calculate the predicted MV, and 2) algorithm used to construct the predicted MV. In the temporal region the block in the reference frame at the same position as that of current block in the present frame is a straight forward choice as a temporal ROS candidate. However, the neighboring blocks from the same reference frame can also be used for prediction. However there would be a large requirement of memory if such a kind of operation is performed, as the MV information of the complete reference frame should be stored. So the choice of temporal prediction will be eliminated due to the huge memory requirement and computations. The other way possible is to go for the spatial prediction. Usage of the already calculated i.e the neighboring blocks MVs as a source for prediction will be a good option. It is the only possible way to have less memory requirement. The concept of Region of Support (ROS) is used for the prediction of current block MV. There are 4 kinds of ROS possible. They are as follows

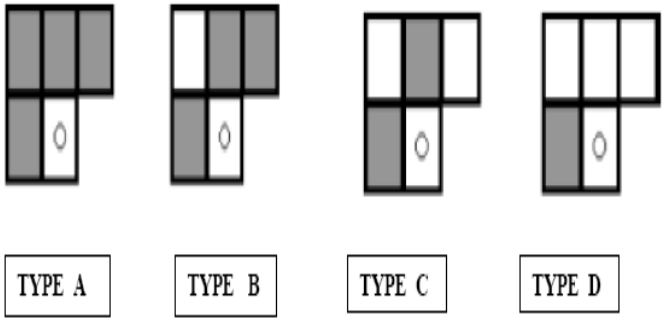


Fig 2.1:- Types Of Region Of Supports[16]

TYPE A ROS covers all the four neighboring blocks and TYPE B is the prediction ROS that is adopted in some international standards such as H.263 for the differential coding of the MVs. TYPE C composed of the two directly adjacent blocks, TYPE D consists of only one adjacent block that is left of the current MB. Experiments on various types of ROCs is being done and it was observed that they yield fairly similar results with a difference of less than 0.1 DB in PSNR and 5% in the number of search points. Hence it is wise to choose TYPE D kind of ROS hence it requires only one motion vector for prediction.

III. Selection of Search Patterns

Adaptive Rood Pattern:

For initial Search: The shape of the rood pattern is symmetrical that is shown in the figure 2. The main structure of ARP takes the rood shape; its size refers to the distance between center point and the any of the other vertex point. The shape of the rood pattern is determined on the basis of real world motion sequences. For most of the sequences it was observed that the motion vector distribution was mostly in horizontal and vertical direction than in other directions, since the camera movements are mostly in those directions. Since the rood pattern spreads in both the vertical and horizontal directions it can quickly detect the motion vectors and also can able to jump directly into the local region of the global minimum. Secondly, any MV can be decomposed into one vertical MV component and one horizontal MV component.

For a moving object which may introduce motion in any direction the rood shaped pattern can at least detect the major trend of the moving object which is the desired outcome of the initial search stage. Furthermore ARP's Symmetric shape is advantageous in terms of hardware implementation. As said in the previous section that, even though we use only single MV for prediction, the resulting performance is good when compared with the other kind of ROSs that covers more number of neighboring blocks. It shows that even though the predicted MV is not accurate then also the rood shaped pattern which spreads in the horizontal and vertical directions can still track the major direction and can follow up the refinement process.

In addition to the four search points it would be better to include the position of the predicted motion vector that aids in the termination in the initial search stage only if the predicted MV matches with the target MV. So in total there will be six search points in the initial search stage and then five search points for the further refinement process. The search pattern that will be used in the initial search stage is shown in the figure 2.2. In this method the Rood Arm Length (RAL) will be equal to the size of the predicted motion vector for the initial search stage, and the four arms are of equal length. Mathematically it can be expressed as follows. The size of the ARP, Ŕ is

Ŕ = Round (MV predicted)
= Round (MV2predicted(x) + MV2predicted(y))])

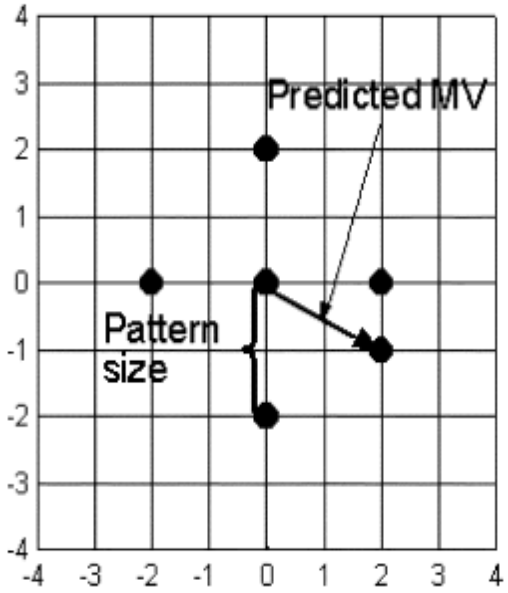


Figure 2.2:- Adaptive Rood Pattern[16]

Where the MVpredicted(x) is the x-component of the predicted motion vector and the MVpredicted(y) is the y-component of the predicted motion vector. Operator Round performs the rounding operation to the nearest possible integer since the displacement can be in terms of the integers.
The above equation involves quadratic computation, which is a time consuming process. In terms of hardware implementation it is suggested to not include square root kind of operations. In this concept of motion estimation we need to speed up the process as far as possible so that we can meet the desired frame rate. Instead of the square root operation, in order to maintain the hardware complexity less, the greater of the magnitudes of the x and y components taken as the size of the rood arm, that is

$$Ŕ = \text{Max}\ (|\text{MVpredicted(x)}|,|\ \text{MVpredicted(y)}|).$$

It should be kept in mind that the TYPE D kind of ROS cannot be used for all MBs, because there will be some MBs where there will be no left neighbor. So in that case the ARP with RAL of 2 (Ŕ = 2) is suggested, by taking the reference of LDSP, which has fairly a good amount of performance. Also larger MVs are not preferred as the boundary MVs mostly belong to the static background, which do not contribute larger MVs.

IV. Fixed Pattern

For Refined Local Search**:** In the initial search the adaptive rood pattern directly leads to the new search position which is somewhere around global minimum, which avoids the unnecessary search points in the intermediary search path. Since there is no chance of getting trapped into the local minimum we can use the fixed pattern for identifying the global minimum. The minimum error point in the first step is used to align as the centre of the fixed pattern in the second step. This process will be followed until the point of minimum error is the centre of the present iteration's search pattern. Two types of fixed patterns were proposed. The first one was the 3x3 square patterns as was proposed in the SDSP. The second pattern consists of a unit size rood arm pattern. The experimental results conducted by [16] showed that the 3x3 square pattern yields similar PSNR when compared to the Unit rood arm pattern but 40% to 80% more number of search points. This demonstrates the efficiency of the Unit Rood Arm Pattern. The proposed fixed patterns by [16] are shown in the figure 2.3.

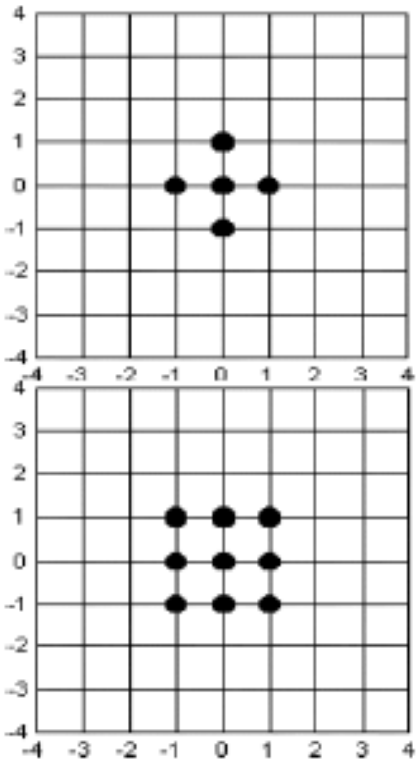


Fig 2.3:- Fixed size Patterns[16]

V. RESULTS

Algorithmic Verifications:

As the part of algorithmic verification, the ARPS algorithm was simulated in MATLAB 7.2 version and compared with the Diamond search, Gradient descent search, Full search. Comparison metrics used were PSNR and number of search points. The comparison is as follows.

Video Name	ARPS	DS	GDS	FS
Stock	29.39	30.55	28.9	30.33
Mob Cal	31.36	32.15	32.36	33.32
Shields	26.33	28.10	24.65	28.93
Park Run	19.11	21.11	19.12	21.11

Table 1: Comparison by PSNR (in db) as metric

Video Name	ARPS	DS	GDS	FS
Stock	11.33	12.63	16.32	256
Mob Cal	07.40	09.56	09.85	256
Shields	16.45	13.65	20.45	256
Park Run	11.55	11.96	16.12	256

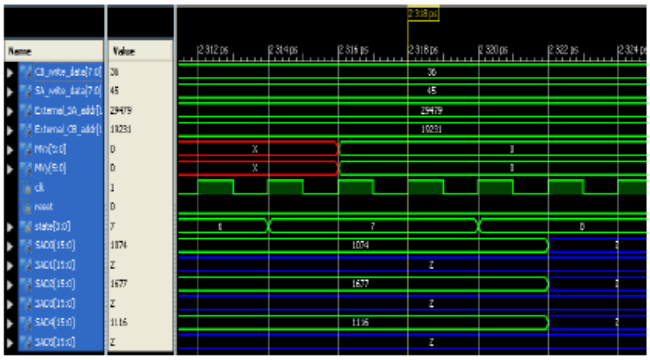
Table 2: Comparison by number of Search Points as Metric

Observations from algorithmic verification:

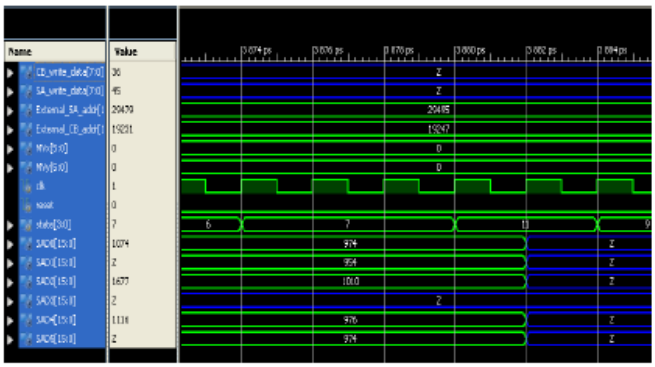
From the above comparison tables it can be inferred that the ARPS was faster than the other search techniques and was almost equal in performance when compared to the DS, FS and was better than GDS.

Hardware Simulation:

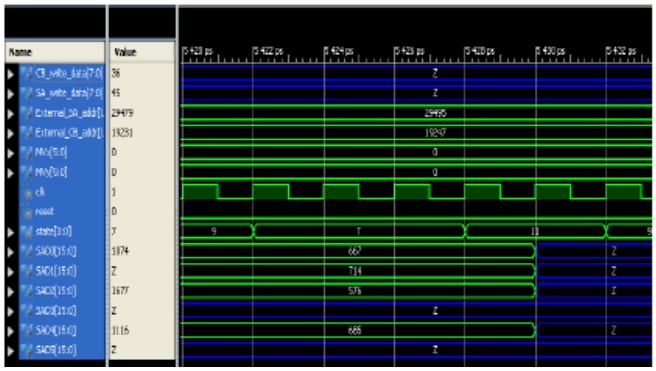
The HDL used for the proposed architecture is VERILOG. The Synthesis tool is Xilinx XST tool. The ISE tool is Xilinx ISE 11.1 version.



Output Fig: 1Waveform depicting the first motion vector output



Output Fig: 2Waveform depicting the motion estimation block entering into the second Iteration



Output Fig: 3 Waveform depicting the disabling of the SAD of the motion vector for the second iteration.

VI. SYNTHESIS RESULTS

Details of Critical path:
Timing constraint: Default period analysis for Clock 'clk'

Clock period: 9.254ns (frequency: 108.056MHz)

Total number of paths / destination ports: 28282962 / 4453

Delay: 9.254ns (Levels of Logic = 37)

Source: cbemi2/col_0 (FF)

Destination: cbemi2/External_CB_addr_19 (FF)

Source Clock: clk rising

Destination Clock: clk rising

Data Path: cbemi2/col_0 to cbemi2/External_CB_addr_19

VII. FUTURE WORK

The proposed architecture is to be modified to achieve the real time performance. Modification to the proposed architecture can be done by adding two port memories and increasing the number of processing elements which then can be used in the real time video encoder which is targeted for HDTV applications. The proposed architecture is targeted to FPGA; it can be extended to ASIC implementation also.

VIII. CONCLUSIONS

1) From the software simulation it was found that ARPS was better in performance when compared to GDS but was almost close to DS and FS.

2) ARPS was better than GDS, FS, DS in terms of number of search points, which was the main motivation for choosing the ARPS algorithm.

3) From synthesis reports it was observed that the critical path was in the current block external memory interface module.

4) Critical path can be reduced by developing the efficient 20 bit multiplier.

5) From the theoretical calculations it can be concluded that it is enough to reduce the number of clock cycles required for single iteration, to find the minimum position search point in order to achieve the real time performance.

6) Number of clock cycles can be reduced by using the multi port memories

REFERENCES

[1] MPEG, "IS0 CD11172-2; Coding of moving pictures and associated audio for digital storage
media at up to about 1.5 Mbits/s*,"* Nov. 1991.

[2] ISO/IEC 11 172-2 (MPEG-1 Video), "Information technology-Coding of moving pictures
and associated audio for digital storage media at up to about 1.5 Mbit/s: Video," 1993.

[3] ISO/IEC 13818-2 I ITU-T H.262 (MPEG-2 Video), "Information technology- Generic
coding of moving pictures and associated audio information:Video," 1995.

[4] CCITT SGXV, "Description of reference model 8 (RM8)," *Document 525, Working Party
XV/4, specialists Group on Coding for Visual Telephony*, Jun. 1989.

[5] A. N. Netravali and J. D. Robbins, "Motion compensated television coding: Part- I," *Bell
Syst. Tech. J*., vol. 58, pp. 631–670, Mar. 1979.

[6] Lai-Man Po and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block
Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol.6, no. 3, pp.313-317, June 1996
.

[7] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframecoding for video conferencing," in *Proc. Nat. Telecommun. Conf., New Orleans, LA*, Nov.29–Dec. 3 1981, pp. G5.3.1– G5.3.5.

[8] Jong-Nam Kim and Tae-Sun Choi, "A Fast Three Step Search Algorithm with Minimum
Checking Points," *Proc. of IEEE conference on Consumer Electronics*, pp.132-133, 2-4 June
1998.

[9] Renxiang Li, Bing Zeng, and Ming L. Liou, "A New Three-Step Search Algorithm for Block
Motion Estimation," *IEEE Transactions on Circuits and systems* for Video Technology,Vol.4, no.4, pp.438-442, August 1994.

[10] Angus Wu and Man F. So, "An Efficient VLSI Implementation of Four-Step Search
Algorithm", *IEEE International Conference on Electronics, Circuits and Systems*, Volume:
3, on page(s): 503-506 vol.3, 1998.

[11] Seth.K, Rangarajan.P, Srinivasan.S, Kamakoti.V, Bala Kuteshwar.V, "A Parallel Architectural Implementation of The New Three-Step Search Algorithm For Block Motion Estimation,*" Proc. of IEEE conference on VLSI Design*, pp.1071-1076, August 2004.

[12] Zhongli He, Ming L. Liou, Philip. C. H. Chan, and R. Li, "An Efficient VLSI Architecturefor New Three-Step search Algorithm", *38th Midwest Symposium on Circuits and Systems*, August 1996, pp. 1228-1231.

[13] B.K.N.S. Rao, S.K. Chatterjee, I. Chakrabarti, "Low Power VLSI Architecture for FTSSalgorithm", *International coferencec on RF and Signal processing systems*, Feb 2008, pp.286-291.

[14] Th. Zahariadis, D. Kalivas *"*A Spiral Search Algorithm for Fast Estimation of Block Motion Vectors," *Signal Processing VIII, theories and applications.Proceedings of the EUSIPCO 96. Eighth European Signal Processing Conference* p.3 vol. lxiii + 2144, 1079-82, vol. 2.

[15] Jaswant R. Jain and Anil K. Jain, *"*Displacement Measurement and Its Application inInterframe Image Coding," *IEEE Transactions on Communications*, VOL. COM-29, NO.12, December 1981.

[16] Y.Nie, K-k Ma, "Adaptive Rood Pattern Search for fast block matching algorithms,"*IEEE Trans. on Image Processing, vol 11 , no. 12*, pp. 1442-1449, DECEMBER 2002.

[17] Y.Qiu and W.Badawy, "The Hardware architecture of a novel motion estimator withadaptive crossed quarter polar search patterns for H.264 encoding," *Proc. of the 22ndCanadian Conference on Electrical and Computer Engineering, CCECE 2009, 3-6 May2009, Delta St. John's Hotel and Conference Centre, St. John's, Newfoundland, Canada.IEEE 2009*, pp.819 – 822.

[18] H.M. Jong, L.G. Chen, T.D. Chiueh, "Parallel Architectures for 3-Step Hierarchical Search Block Matching Algorithm," *IEEE Trans. on circuits and systems for video technology, vol4, no. 4, August 1994*, pp. 407-416.

[19] M.Porto, L.Agostini, S.Bampi, A.Susin, "A high throughput and low cost diamond search architecture for HDTV motion estimation," *IEEE International Conference on Multimedia and Expo 2008*, pp. 1033-1036.

[20] K.H. Lam and C.Y. Tsui, "An efficient VLSI architecture for Diamond Search Pattern based Motion Estimation algorithms," *Proc. of 2000 Asia Pacific Conference on Multimedia Technology and Applications (APCMTA2000)*, Kaohsiung, Taiwan, Dec, 2000.

[21] K.K. Ma, G. Qiu, "Adaptive Rood Pattern Search for Fast Block Matching Motion Estimation in JVT/H.26L," *IEEE International Conference on Image Processing 2003*, pp.ii-708 – ii-711.

[22] Peter Pirsch, "VLSI architectures for video compression - A Survey," *Proc. Of the IEEE, Vol 83, NO 2, February 1995*, pp. 220-246.

[23] V.S.K Reddy, S. Senguptha, Y.M.Latha,"New VLSI architecture for motion estimation Algorithm," *World academy of science, Engineering and technology 36 2007*.

[24] B.K.N.Srinivasa rao, I. Chakrabarti, "A parallel Architecture for Successive elimination Block Matching Algorithm," *Sixth Indian Conference on Computer Vision, Graphics &Image Processing, ICVGIP 2008, Bhubaneswar, India, 16-19 December 2008. IEEE 2008*, pp.226-231.

[25] C.H. Hsieh, "VLSI architecture for block –matching motion estimation algorithm," *IEEETrans. on circuits and systems for video technology. Vol 2. No 2, June 1992*, pp. 169-175.

[26] L.D.Vos, M.Stegherr, "Parameterizable VLSI architectures for the Full-Search block matching algorithm," *IEEE Trans. on circuits and systems, vol 36, No 10, October 1989*,pp. 1309-1316.

[27] K.M.yang, M.T.Sun, L.Wu, "A family of VLSI designs for the Motion compensation block matching algorithm," *IEEE Trans. on circuits and systems, vol 36, no 10, October 1989*. Pp.1317-1325.

[28] K.K.Ma, G.Qiu, "Unequal-Arm Adaptive Rood Pattern Search for fast block matching motion estimation in the JVT/H.26L," *International conference on Image Processing, 2003, vol.1*, pp. 901-904.