International Journal of u-and e-Service, Science and Technology
Vol.7, No.4 (2014), pp.339-348
http://dx.doi.org/10.14257/ijunnesst.2014.7.4.31

A Two-Step Methodology for Minimization of Computational Overhead on Full Search Block Motion Estimation.

Linganagoud Kulkarni[1], Manu T. M.[2] and Basavaraj. S. Anami[3]

[1]B.V.B College of Engineering and Technology, Hubli-30, Karanataka, India
[2,3]K.L.E Institute of Technology, Hubli-30, Karnataka, India
[1]linganagouda@yahoo.co.uk, [2]manutmece@yahoo.com, [3]anami_basu@hotmail.com

Abstract

In Video coding, Motion Estimation(ME) is the major step to remove the temporal redundancy between adjacent frames. Several algorithms have been developed for acceleration of ME by reducing the number of computations. In this paper, we propose, two-step motion estimation technique to minimize the computational overhead of general full search block matching motion estimation algorithm. In our proposed methodology, we first find the matching block using the trace and upon match, the Sum of Absolute Difference (SAD) is calculated for the entire block in order to ensure the best match motion prediction. In trace matching, the number of computations observed are O(n) rather than O(n^2)in case of full search. Hence, the two- step methodology reduces the computational overhead and also ensures compression quality comparable with the existing fast motion search techniques.

Keywords: *Video Frames, Motion Vector, Temporal Prediction, Trace*

1. Introduction

Motion estimation is the heart of all video coding standards such as MPEG and H.26X series. Over the past two decades, hundreds of methods have been proposed to design efficient motion estimation algorithms and hardware architectures for real-time applications. In the latest video coding standard, H.264/AVC, motion estimation is characterized with three new tools including Variable Block Size, quarter-pixel accurate motion vector (MV), and Multi Reference Frame. These tools significantly improve the coding performance of H.264/AVC. However, they require huge computational complexity and memory bandwidth. Therefore, there is a need for novel and efficient designs that support new features of motion estimation in H.264/AVC.

Motion estimation is used as a powerful technique to remove temporal redundancy between successive frames. Motion estimation is a computationally intensive process, which typically consumes more than 85% of encoding time of video coder. Among different motion estimation algorithms, full search (*i.e.*, exhaustive search) block matching motion estimation algorithm is widely used due to its better coding performance and regularity compared with fast search algorithms. Most of the available algorithms exhibit a tradeoff between quality and speed. Since motion estimation is scene dependent, not single technique is fully reliable to generate good visual quality for all kinds of video scenes. Instead, it demands combination of variety of techniques, such as motion starting point ,motion search patterns, adaptive control to curb the search and avoidance of search of stationary regions etc., that makes an ME algorithm robust. Hence there is a requirement of best optimized algorithm and hardware architecture to cope with the increased complexity.

Rest of the paper is organized as follows: Section 2 literature survey is given and in Section 3. Full search block motion estimation algorithm is explained. In Section. 4. A new two-step methodology for minimization of computational overhead using trace based method is discussed. A temporal reduced search detail is given in Section 5. In Section 6. Experimental results and discussion are made. Finally conclusions are drawn in Section 7.

2. Literature Survey

To understand and to identify a state of art problem, a rigorous literature survey is carried out. Considerable amount of research has been taken place to develop a fast block matching algorithms that finds a suitable match by using only few calculations, like three step search[1], novel four step search[2],the cross search [3],the diamond search algorithm[4],the hexagon based search algorithm[5],block-based gradient descent search algorithm[6], *etc.,* In all these fast algorithms the search test is done only some of the candidate blocks within the search area, the match is chosen from the subset of blocks and the search is done only in some fixed point position, results error in motion prediction. Any error in motion prediction may lead to wrong motion vectors, resulting in high Mean Square Error which degrades PSNR.

To achieve best tradeoff between the computational complexity of FSBM and degraded PSNR of motion compensated frame, recently some researchers have proposed the reduction of computational complexity of FSBM, without causing the significant reduction in video quality. Moshnyaga[14] presented a modified FSBM, where the number of computations is four times less than those of FSBM. This works on a data driven threshold, which updates according to the picture variations. Another computation reduction algorithm where the motion vector can be determined based on the constant and reference thresholds [11]. A novel method is presented in[15], which lower bounds of the matching criteria for subdivided block are evaluated in order to reduce the number search positions. A predictive line search proposed is based on the mean value of motion vectors of the neighbor macro blocks [16]. A new predictive search area approach presented in paper[12], in which predicted search area can be obtained from sub area of the neighboring blocks. It is also possible to reduce further by employing a predicted region full search. The total full search range is divided into overlapping regions [17]. The overlapping regions ensure that the motion vectors close to the boundary of the two regions can be determined from either of the two regions.

In all these fast algorithm, search is mainly carried out based on the assumption that the matching error decreases monotonically when approaching towards the global best points and error may drops down to local minimum points. These algorithms are having lack of regular dataflow and the predictable terminations. Hence, these algorithms suffer large quality degradation when the motion field is large and complex.

We propose a new two-step methodology for minimization of computational overhead in motion estimation for the conventional FSBME algorithm. In this method, instead of exploiting the motion activity by calculating SAD between adjacent frames directly [22], in the first step, we investigate the motion activity by comparing the trace of the current block and the candidate block. If the trace of the source and target block is exactly equal or highly similar according to the predefined or specified threshold value, then two blocks are considered to be matching [5]. If the trace of the source and target blocks is unequal or far different according to the specified threshold value, then the two blocks are considered to be dissimilar. On matching, the best matching block is found by calculating the SAD in the second step. Thus, the numbers of computations are reduced.

3. Full Search Block Motion Estimation Algorithm

Motion estimation in video encoding standards such as H.263 and MPEG-4 employs block-based motion estimation to reduce temporal redundancy between frames. The simplest and most effective method of motion estimation is Full Search Block Matching Motion Estimation (FSBME). In which exhaustive comparison of each N x N macro block of the current frame with all the candidate blocks of the same size in the reference frame is done[3]. The best matching block is found by a cost function, the lowest distortion measure of the cost function specifies the best matching position. Full search block matching process with a search range p has a search window of size (2p+N) x (2p+N) pixels and a total of $(2p+1)^2$ candidate blocks in the reference frame for each macroblock of the current frame as shown in Figure (1).

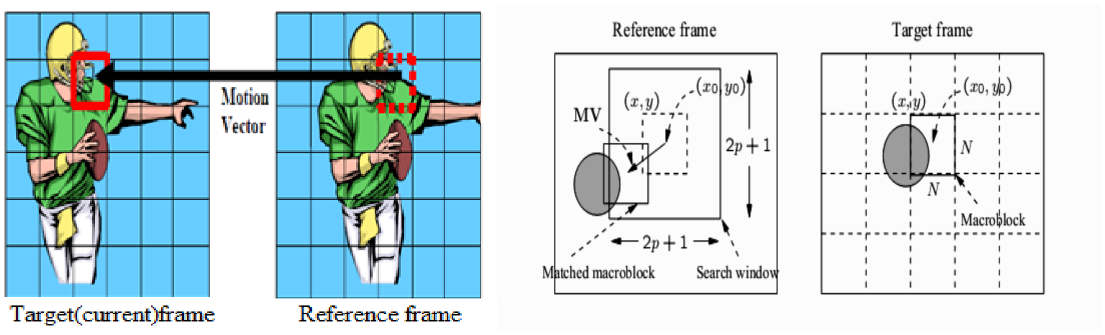


Figure 1. Typical Frames with Block Movement and Working Mechanism of Motion Estimation

The distortion values are computed for a macroblock centered at each of the positions within the search window is compared with the macroblock in the target frame pixel by pixel and their respective distortion value is calculated by the equation(1). The popular distortion measure is Sum of Absolute Difference for its simplicity. The vector, which offers the least distortion or the minimum SAD is designated as Motion Vector MV (u, v) for the current macroblock searched from set of (2p+1)2 candidate blocks in the reference frame is calculated by using equation (2).

$$SAD(i,j) = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} |\mathrm{C}(\mathrm{x}+\mathrm{k}, \mathrm{y}+\mathrm{l}) - R(x+i+k, y+j+l)| \ldots \ (1)$$

where N is the size of the macroblock, k and l are indices for pixels of the macroblock, i and j are horizontal and vertical displacements. C(x+k,y+l)-pixels in macroblock in target frame and R(x+i+k,y+j+l) are the pixels in candidate block in reference frame.The comparison process is depicted in Figure (1).

$$(u,v) = \big[(i,j) \big| SAD(i,j) is\ minimum, i \in [-p,p], j \in [-p,p]\big] \ldots (2)$$

The vector(i, j)that offers the least SAD is designated as the MV(u, v) for the macroblock in the current frame. In FSBM, for each block of size of 16X16, there are 256 subtractions and 255 additions are to be carried out in calculating the Sum of Absolute Difference (SAD) related to each block. Hence, the computing complexity is O(n^2), where n is the number rows or columns in NXN square macro block. To calculate a motion vector for one current macro block, 255 SAD calculations and 255 SAD comparisons in the search range p=7 and for the entire frame of size 176X144 there will be 25245 SAD calculations. For higher resolutions considerably large computations and huge amount of time is required to compute. This most computationally expensive and resource hungry process gives a scope for a researcher to find a methodology to reduce the number of computation as well as hardware resources.

4. Proposed Methodology

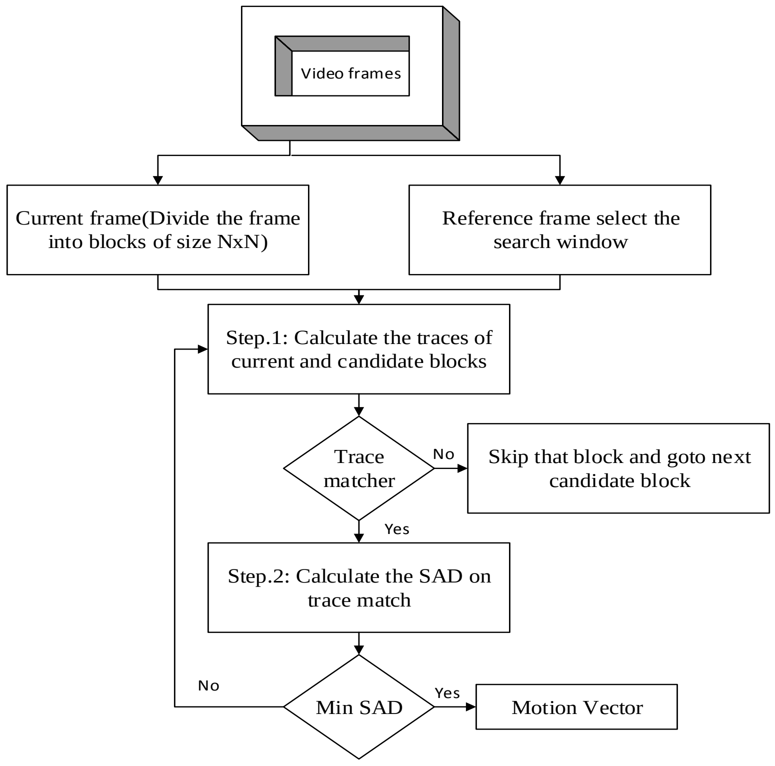

Video frames
Current frame(Divide the frame into blocks of size NxN)
Reference frame select the search window
Step.1: Calculate the traces of current and candidate blocks
Trace matcher
No
Skip that block and goto next candidate block
Yes
Step.2: Calculate the SAD on trace match
No
Min SAD
Yes
Motion Vector

Figure 2. Block Diagram of Proposed Trace Match based FSBME

4.1. Trace Match for Block Motion Estimation

4.1.1. Trace of a Block

In matrix algebra, the Trace of a square matrix is the sum of all diagonal elements, defined by the equation (3). A square block with diagonal elements used for Trace calculation,

$$Trace = \sum_{x=0}^{N-1} |C(x\,,x)| \ldots (3)$$

The trace calculation has the following characteristics: It contains, one value from each row and column of the entire block, and hence no row or column is omitted in the block matching. Also for the trace calucalation dioganal elements are considered which are position wise linear, and so the motion through the pixels are uniform. The computing time for trace is O(n), and for full search it is O(n^2). These said features are incorporated into full search block motion estimation method in order to minimize the computations.

4.1.2 Trace and off diagonal Sum Match in BME

The proposed method has two steps over block matching and successive elimination of dissimilar blocks as described by [5]. In our proposed method the elimination of blocks are based on trace calculation and trace match. Hence, equation (2) is redefined and modified SAD calculation based on trace is given by equation (4).

$$\mathbf{SAD_{Tr}(i,j)} = \sum_{\mathbf{i=0}}^{\mathbf{N-1}} \sum_{\mathbf{j=0}}^{\mathbf{M-1}} |\mathbf{C(i\,,i)} - \mathbf{R(i+k,i+l)}| \ldots \mathbf{(4)}$$

The trace of absolute difference of diagonal pixels of matching blocks of Reference frame and Current frame is calculated by (4) and also the sum of off diagonal elements is calculated for further reduction in computations of SAD, this value is used as a deciding factor to eliminate a block or to redo the full SAD for best match in the second stage by using equation (1).

Trace of current block 213+212+214+163=802=TR

Sum of the off diagonal elements (ODS)= 213+212+216+211=852

213	216	216	213
212	212	212	214
218	216	214	213
211	195	178	163

Figure 3. Current

216	216	213	211
212	212	214	216
216	214	213	204
195	178	163	151

Figure 4. Candidate Block at P(0,0)

856	851	837
802	792	778
746	738	731

Figure 5. TR of Candidate Blocks

861	855	850
852	834	814
777	752	723

Figure 6.ODC of Candidate Blocks

Note: TR=Trace of Current Block, TRC=Trace of Candidate Block, ODS=Sum of off diagonal Elements of Current Block, ODC=Sum of off diagonal elements of Candidate Block

The steps of our proposed methodology is illustrated in the Figures(3,4,5,6) by considering a typical macroblock of size 4X4 with the gray values of standard video of foreman as shown in Figure 3. The candidate block is selected from search window of reference frame. In Figure 5 and Figure 6 trace values and sum of the off diagonal elements of candidate are listed. The first candidate block is selected at the position P (0,0), is the centre of the search window. In the first step, trace of current block (TR) and candidate block is calculated. And sum of the off diagonal elements is also calculated to check the effect of reduction in number of SAD calculations *i.e.*, TR=802, TRC=792 and ODS=852, ODC=834.TR and TRC are compared if they are not matched then ODS and ODC are compared. If both are matched, only one value SAD calculated represents the best match. If any of them is matched then a couple of SAD values are to be calculated to predict best match, otherwise SAD calculation is skipped.

MV(-1,0)

117	104	69
0	72	176
280	355	416

Figure 7. SAD Values

In the second step SAD values are calculated by using equation (1) after the match. In Figure 7. SAD values for the entire candidate blocks are shown. But, according to our methodology only one SAD value is calculated, because for that candidate block the trace and Sum of off diagonal of current block are exactly matched. The SAD value at position P(-1,0) is 0, indicates the true motion and the best match of that candidate block. Hence further calculations of SAD are not necessary. In our analysis the execution on different standard videos shows that on an average 80% of computations of SAD are reduced when compared to FSBME.

4.2. Algorithm of Two-Step Trace match and SAD Computations

```
        Input: Video file, frame( i) and reference frame (i-1),
        Output=Motion Vectors.
        Descriptions: Specification: Frame size-:376X288, Search window=16X16, 40X40;
Start,
         Step 1: Read the video frames; convert the color frames into gray scale frames.
         Step 2:  Divide the current frame into macro blocks of size typically 16X16 or 8X8,
                  and select the search window of size p=7, in the reference frame.
         Step 3:  Calculate the trace and SOD of current block and candidate block.
         Step 4: Perform trace match,
              Step 4.1 If (trace of the two blocks $\leq$ specified threshold value) then
                     go to  Step 5,
                   else
                     go to Step 3.
         Step 5:  Compute the full SAD.
         Step 6:  Find the minimum SAD.
                  Step 6.1 If (minimum SAD obtained)
                go to Step 7
              else
                go to Step 3 and repeat for the next block.
         Step 7:  Calculate the Motion Vector.
Stop
```

Exhaustive experimentation is done with different block sizes and search window sizes to evaluate the performance of our method. In Table 1, number of SAD calculations required when Trace or sum of off diagonal elements and both are listed.

Table 1. Number of SAD Calculation

Current block Size(NXN)	No of candidate blocks in search window p=7,$[2p+1]^2$	No. of SAD in FS	No. of SAD with TR	No. of SAD with ODS	Both TR +ODS	% of saving in computations
16X16	225	225	03	03	01	85
8X8	225	225	07	05	02	83

In Table 2, Percentage of savings in computations for entire frame of size 352X288 are listed for standard video sequences. It is observed that on an average of 85% computations are reduced and the PSNR is on par with exhaustive search. Hence the video quality is maintained as that of the original.

Table 2. Number of SAD Computations for Video Sequences

Video	Block size	No. of SAD Computations		% of Savings in Computations	PSNR	
		FS	Our method		FS	Our method
Foreman	8X8	403920	3168	78.43	31.07	29.08

	16X16	200980	1192	84	29.48	28-16
Football	8X8	403920	4392	85	32.76	31.40
	16X16	200980	1231	82.03	31.42	30.08
Tennis	8X8	403920	3296	81.45	29.46	28.76
	16X16	200980	1276	87.12	27.98	26.50

5. Inter Block Search Reduction.

To further reduce the number of computations to speed up the calculation prediction of motion vectors is done. Motion of objects often covers many small blocks in a general moving video scene, such that the motion field of the spatial neighbor block is very similar. The motion of the current block is tracked from the neighbor blocks in spatial direction, Therefore, the motion information of the block C(i, j) is approximated by the neighboring blocks in the same frame. The horizontal and vertical components of the MV are predicted from the motion values of the horizontal and vertical components, respectively of MV1,MV2,MV3,MV4 from the previous, above and above and right as shown in the Figure 9.

MV1(u-1,v-1)	MV2(u,v-1)	MV3(u+1,v)
MV4(u-1,v)	MV(u, v)	

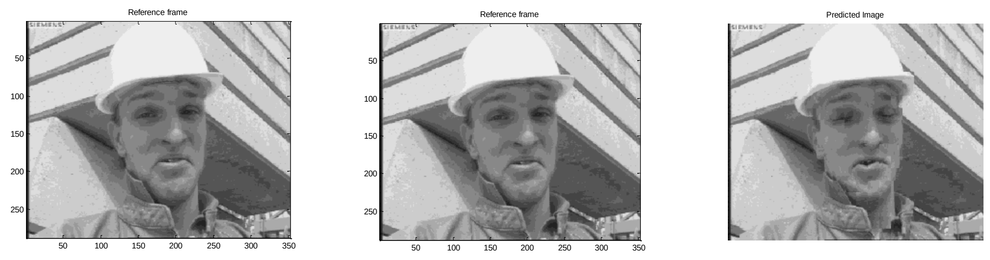
Figure 8. Spatial Motion Prediction

For the macro block with MV(u,v) is calculated, u=median(u1,u2,u3,u4),v=median(v1,v2,v3,v4).

The past information which is available at the encoder and decoder side are used to predict the motion vector, Hence, the SAD calculations are skipped for the entire current block, which in turn reduces the computation overhead. Analysis of this reduced search method which shows that on an average 75 % of computation are saved and the speed up three to four times faster than the convention full search algorithm. The detailed result analysis is discussed below.

6. Results and Discussions

The proposed methodology is implemented using MATLAB 7.1. The speed up of the proposed method is analyzed and a comparison is made with four existing algorithms to correlate the improvement of the proposed methodology namely, Full Search (FS), Three Step Search(3SS), New 3SS and Diagonal search (DS).The parameters such as execution time (CPU time only) and Peak to Signal Noise ratio are considered for comparative study. Figure 10 shows the result using our methodology, (a) current frame, (b) reference frame,(c) reconstructed frame by using motion vectors.



a) b) c)

Figure 10. Video Frames of Foreman

6.1. Speed Up

Speed up ratio is calculated to compare the time required for FSBME and FS with our methodology. The execution time reduces as the reference block size increases *i.e.*, speed up ratio increases. Different current block size and reference block size are chosen from the search window. The CPU speed has been taken for time calculations.

Table 3. Execution Time for Different Size Current Block and Reference Block

Current Block	Reference Block Size	Execution time(Sec)		Speed Up Ratio
		FS	FS+TR+ODS	
8X8	24X24	37.52	12.5	3
8X8	32X32	49.53	15.77	3.14
16X16	24X24	35.14	11.2	3.13
16X16	32X32	44.28	12.82	3.45

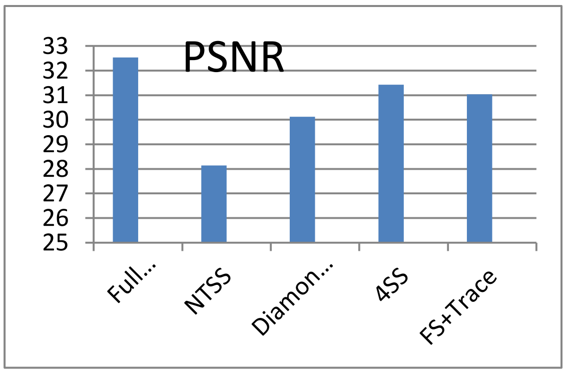
The huge number of computations is reduced compared to the conventional full search. Speed up is increased by three times. In table 4. The speed up ratio for different video with varying current block sizes is shown. The videos are selected for experimentation in such way that they have more objects and large movement within a frame as well as between the frames

Table 3. Execution Time for Different Videos with Speed Up Ratio

Videos	Current Block	Search Window Size	Execution Time		Speed Up Ratio
			FS	FS+TR+ODS	
Foreman	16 X 16	24 X 24	237.52	51.24	4.42
		32 X 32	249.64	52.12	4.57
Football	16 X 16	24 X 24	284.22	64.12	4.43
		32 X 32	292.24	66.21	4.41
Mobile	16 X 16	24 X 24	262.34	61.38	4.27
		32 X 32	270.25	64.48	4.19

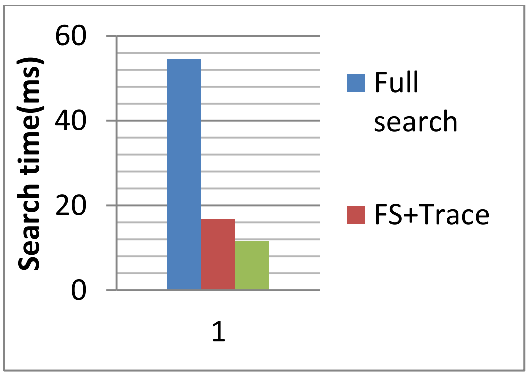
6.2. Peak Signal to Noise Ratio (PSNR)

To analyze the quality of proposed method, the Peak Signal to Noise Ratio (PSNR) is calculated for the compensated frame. PSNR for FSBME and some the fast algorithms such as New TSS,4SS and Diamond search are compared and it is found that PSNR values are nearly same and hence the quality of the video is not degraded.



(13)

Figure 13. PSNR Values of Search Algorithms



(14)

Figure 14. Search Time Vs Proposed Methods

The graph in Figure 14 gives further reduction in computations using prediction with reduced search. The computations are reduced by three to four times compared to conventional full search block estimation method. Hence, overall speed up ratio is increased by six to seven times depending upon the video content. Thus, the proposed methodology is more suitable for real time applications as well as for mobile applications.

7. Conclusion

Full search (FS) block matching motion search is computationally expensive. The two-step methodology for minimization of computational overhead in motion estimation using trace is proposed in this paper, it is implemented using FPGA. In this method the number of computation as well as the number of arithmetic operations involved in SAD calculation are reduced considerably because of trace matching. Experimental results show that the proposed methodology is superior in terms of speed-up than the conventional FS block matching. A FPGA implementation of trace and SAD calculations are carried out. It is concluded that the reduction in computations and the frequency (420 MHz) at which it works makes the methodology more suitable for mobile applications. Since, no change in PSNR assures video quality.

References

[1] R. Li, B. Zeng and M. L. Liou, “A new three step search algorithm for block motion estimation”, IEEE Trans. On CSVT, vol. 4. (**1994**) October, pp. 432-442.
[2] L. M. Po and W. C. Ma, “A novel four-step search algorithm for fast block motion estimation”, IEEE Trans. Circuits Syst. Video Technology, vol. 6, (**1996**), pp. 313-317.
[3] M. Ghanbari,” The cross search algorithm for motion estimation”IEEE Transaction on communication, vol. 38, (**1990**) July, pp. 950-953.
[4] Y. Li and K. Sayood, “Lossless Video Sequence Compression Using Adaptive Prediction”, IEEE Trans. Image Processing, vol. 16, no. 4, (**2007**) April.
[5] Y. Wu and G. Megson, “Two-pass hexagonal algorithm with improved hash table structure for motion estimation”, in Proc. IEEE Conference on Advanced Video and Signal Based Surveillance, (**2005**), pp. 564 - 569.
[6] S. Zhu and K. K. Ma, “A new diamond search algorithm for fast block matching motion estimation”, IEEE Trans. Image Processing, vol. 9, (**2000**) February, pp. 287-290.
[7] C. Zhu, X. Lin and L. P. Chau, “Hexagon-based search pattern for fast block motion estimation”, IEEE Trans. Circuits Syst. Video Technology, vol. 12, (**2002**) May, pp. 349-355.
[8] L. K. Liu and E. Eig, “A block based gradient descent search algorithm for fast block motion video coding “IEEE Transaction on CSVT”, vol. 6, (**1996**), pp. 419-422.
[9] H. Loukil, F. Ghozza and A. Samet, *et al.* “Hardware implementation of block matching algorithm with FPGA technology,” 6th International Conference On Microelectronics, Proceedings, (**2004**), pp. 542–546.
[10] D. Brunello, D. Calvagno, G. A. Mian and R. Rina ldo, “Lossless compression of video using temporal information”, IEEE Trans. Image Process., (**2003**), pp. 132–139.
[11] Y. Chan and W. Siu, “An efficient search strategy for block motion estimation using image features”, IEEE Trans. Image Process., vol. 10, (**2001**), pp. 1223–1238.
[12] S. Wong, S. Vassiliadis and S. Cotofana, “A Sum of Absolute Differences Implementation in FPGA Hardware,” 28th Euromicro Conference (EUROMICRO’02), Dortmund, Germany, (**2002**), pp. 183–188.
[13] X. Q. Gao, C. J. Duanmu and C. R. Zou, “A multilevel successive elimination algorithm for block matching motion estimation”, IEEE Trans. Image Processing, vol. 9, (**2000**) March, pp. 501-504.
[14] A. Puri, H. M. Hang and Schilling, “An efficient block matching algorithm for motion compensated coding”, in Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, (**1997**), pp. 1063–1066.
[15] B. Baumgarrtner, "An Inequality for the trace of matrix products", using absolute values.arXiv:1106.6189v2.sep 2011.
[16] K. Liang and S.-T. Ma,” Predictive Line search: An Efficient Motion Estimation Algorithm for MPEG-4Encoding Systems on Multimedia processors, IEEE Transaction on CSVT, vol. 13, no. 1, (**2003**) January.
[17] K.-L. Chung and L. C. Chang, “A new Predictive Search Area Approach for Fast Block Motion estimation”, IEEE Transaction on Image processing, vol. 12, no. 6, (**2003**) June.
[18] V. G. Moshnyaga, “A New Computationally adaptive Formulation of Block Matching Motion estimation”, IEEE Transaction on CSVT, vol. 11, no. 1, (**2001**) January.
[19] M. Bruing and W. Niehen, “Fast Full Search Block Matching”, IEEE Transaction on CSVT, vol. 11, no. 2, (**2001**) February.

[20] H. C. Huang and Y. P. Hung, "Adaptive Early Jump-out Technique for Fast Motion Estimation In Video Coding", Graphical Method and Image Processing, vol. 59, no. 6, (**1997**) November, pp. 388-394.
[21] S. Senagupta and V. S. K Reddy, "A Fast and Efficient Predictive Block Matching Motion estimation", IJCNSN, vol. 7, no.12, (**2007**) December.
[22] S. Sundaravadivelu and S. Jeyakumar "An Efficient Motion Estimation Algorithm using Trace Match for Fast Video Compression", European Journal of Scientific Research ISSN 1450-216X, vol. 53, no. 4, (**2011**), pp. 546-554.