Complexity Reduction of Fast Block Matching Algorithm

P. Muralidhar, A. Vishnupriya, C. B. RamaRao

Abstract: ***This paper presents a new block matching motion estimation algorithm using the macro block features to reduce the computational complexity of motion estimation in video encode applications. Motion estimation block is the computationally intensive block in video encoders. To reduce computational cost various motion estimation algorithms have been proposed. Global Elimination is an algorithm based on pixel averaging to reduce the complexity of motion search while keeping performance close to that of full search. Here adaptive version of Global elimination is proposed that uses macro block features like variance and Hadamard transform to further reduce the computational complexity of motion estimation. Performance achieved is close to the full search method and global elimination. Operational complexity is reduced compared to global elimination method.***

Keywords: Block Matching Motion Estimation Algorithm, Global Elimination, Matching complexity reduction, Feature based partitioning.

I. INTRODUCTION

Digital video is to be compressed for efficient storage or transmission due to its massive amount of data. To achieve a high compression ratio, a Block Matching Motion Estimation (BMME) algorithm is widely used to remove temporal redundancy in video coding standards such as H.263**,** H.264, and MPEG-4. For each macroblock (MB) of size NxNin the current frame, the algorithm searches for the best matching block within a search region around the current block position in the previous frame. Sum of absolute differences (SAD) is the commonly used matching criterion due to its simple operations. Speed of motion estimation depends on the computational complexity of the motion estimation algorithm used.

Three Step Search(TSS)[2], Four Step Search(FSS)[3], Hexagon Search(HS)[6] are some of the algorithms that reduce computational complexity by reducing the no. of search points in search window. These algorithms assume 'Uni-modal Error Surface' in which the matching error would increase with the distance from the global minimum, thus results in quality loss compared to full search in high resolution applications because of local minimum.Apart from the search point reduction, there are many algorithms that optimize the motion estimation task by reducing matching complexity at each search position. Some algorithms simplify the SAD calculation by pixel sub-sampling, exploiting spatial correlation of pixels. E.g., alternate pixel pattern[4], quincunx patterns etc. since only a subset of pixels are used for SAD calculation in sub-sampling based algorithms, the quality loss can be large for large sub-sampling ratio.

Manuscript received on August 25, 2012

P. Muralidhar, Department of Electronica and communication Engineering, N.I.T. Warangal, India,

A. Vishnupriya, Department of Electronica and communication Engineering, N.I.T. Warangal, India.

C. B. amaRao, Department of Electronica and communication Engineering, N.I.T. Warangal, India.

Averaging of pixels is also quite commonly used to reduce matching complexity. This prevents elimination of large number of pixels from distortion measurement. Successive Elimination Algorithm(SEA)[1], Multilevel successive Elimination Algorithm(MSEA)[7],GlobalElimination Algorithm(GEA)[5] uses averaging of pixels and achieve quality comparable with the full search algorithm.

Successive Elimination Algorithm works on the principle that the absolute difference of sum-of-pixels cannot be larger than the SAD of pixels themselves. A large number of search points can be eliminated early if the Absolute Difference of sum-of-pixels at these points is larger than the best SAD till that point. Thus complete SAD calculation can be avoided at many of the search points by using this method. In this at each search point, SEA (sum of current pixels – sum of reference pixels) is computed to know whether the search point is to be eliminated or not.

Multilevel SEA algorithms help in eliminating more candidates quickly by creating successively tighter bounds and hence avoiding full SAD calculation at most of the points. In MSEA, a series of such bounds is constructed by using the fact that sum of absolute difference of larger block sums would be smaller than sum of absolute difference of smaller block-sums within a macro-block.In the above two algorithms, computational complexity is reduced compared to traditional full search algorithm but they are not suitable for hardware implementation due toirregular data flow of the algorithm and these algorithms requires initial threshold. Choosing the threshold value will affect the speed of motion estimation. Hence to overcome these problems Global Elimination algorithm is proposed.

II. GLOBAL ELIMINATION ALGORITHM

Global Elimination is a lossy two stage motion estimation algorithm. During the first stage, matching operation is performed using SEA or MSEA criterion. Best 'N' candidate pixels are chosen from the 1st stage are passed onto the 2nd stage where full SAD calculation is performed on these 'N' candidates and the best candidate is selected.

In the first stage, a rectangular search window in a reference frame is scanned in raster scan order, and a matching operation is performed at each reference position. At the end of the first stage, a small number of candidate points (e.g, 'N'= 7) are selected, on which a full SAD calculations are performed in the 2nd stage. The computational complexity of GE algorithm is thus heavily dominated by its 1st stage. Flow chart for GE is as shown below,

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

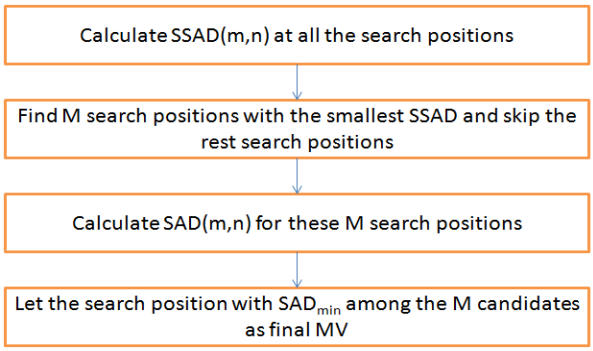

Fig. 1: flow chart for Global Elimination Algorithm

III. PROPOSED GLOBAL ELIMINATION ALGORITHMS

When we use GE with level=3, each Macro Block is divided in to fixed size 16 no. of sub blocks. But for Mb's with less variation in pixel values, division in to less no. of sub blocks can also achieve similar quality. Thus partitioning of MB can be done adaptively using MB features.

A. Partitioning using Macro Block Variance

Many video sequences contain spatial regions which are nearly plain with nearly identical pixel values. If we look at the SAD values for plain MBs with different motion vectors, we would get a gradual error surface with many motion vector candidates resulting in very small and similar SAD values.

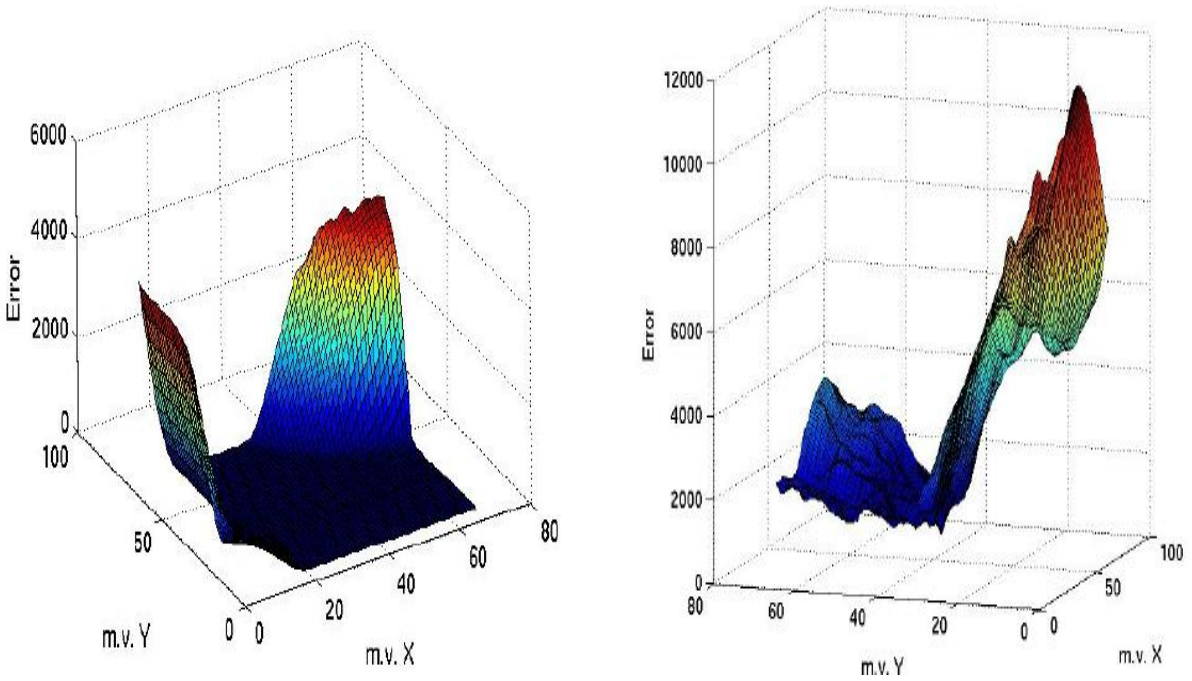

Fig. 2: Error surface of MB with less and large variance

The resulting bit rate due to these MBs will also be very small due to absence of high frequencies and small error in motion vector estimation will not result in large increase in bit rate. If we plot error surface of a plain macro block and a macro block with large variance, The plain macro-block has an error surface with a large flat area with small and nearly equal error (SAD) values in the center, while the macro-block with large variance has uneven surface with large number of local minima.

This observation is used to adapt partitioning of a macro block. In this algorithm, the pixel variance within current MB is computed. If the variance exceeds a certain threshold, then the MB is partitioned into 4x4=16 sub blocks, each containing 16 pixels. If the variance is smaller than the threshold value, then the MB is partitioned into 2x2=4 sub-blocks of 64 pixels each. The Variance measure that is used is sum of absolute difference between pixel values and mean value of the Macro Block.

B. Partitioning based on Variance and Hadamard coefficients

Hadamard transform is also used to find the features such as spatial frequency components within the MB and partitioning options are chosen depending on the dominant transform coefficients.

In this algorithm, MB is divided into four 8x8 blocks. Each block of 64 pixels is analyzed using the sub-block variance as well as 2x2 Hadamard transform of four mean values of 4x4 pixels within the sub-block. Based on this analysis, the block can be further refined into 1,2,3, or 4 sub-blocks as dictated by the algorithm.Hadamard transform applied is,

$H=A^T MA$

Where, M is mean matrix, H is transformed matrix and A is transformation matrix, $A = \begin{matrix} 1 & 1 \\ 1 & -1 \end{matrix}$

If the sub-block has no significant frequency coefficient and also if its variance is small, then it is not partitioned further. On the other hand if transform coefficient (1,0)

is strong, while other coefficients are negligible, then the 8x8 sub block is partitioned into two 4x8 sub blocks. Similarly, we decide partitioning in favor of two 8x4 sub-blocks if (0,1) frequency component is large and all other frequency components are small. Further, in each of these two cases, variance of pixels in each sub-partition is separately analyzed. For example, if two 4x8 sub-partitions were chosen, and the variance within the top sub-partition is larger than certain threshold, then that sub-partition is further divided into two partitions of 4x4 pixels.

For each 8x 8 sub-blocks in a macro block:

If({HmdTr[1][0], HmdTr[0][1], HmdTr[1][1]} <Frq_Thr) && (Var8x8blk < 2*Var_Thr)
{
No Further Partitioning of 8x8 sub block }
Else If(HmdTr[1][0] >Frq_Thr) && ({HmdTr[1][1], HmdTr[0][1]} <Frq_Thr) {
Divide the 8x8 sub-block into two 4x8 sub-blocks

If(top sub-block variance >Var_Thr) {
Divide the top sub-block into two 4x4 partitions }
If(bottom sub-block variance >Var_Thr) {
Divide the bottom sub-block into two 4x4 partitions
} }
Else If(HmdTr[0][1] >Frq_Thr) && ({HmdTr[1][1], HmdTr[1][0]} <Frq_Thr) {
Partition in to 8x4 sub blocks and further partition in to 4x4 sub blocks
Based on variance criteria same as above 4x8 case }
Else{
Divide the 8x8 sub block into four 4x4 sub blocks.
}

Fig.3:pseudo code for proposed adaptive GE

If both (0,1) and (1,0) frequency coefficients are large or the coefficient (1,1) is large, it leads to partitioning the block into 4 blocks of 4x4 pixels each. Also, a large sub-block variance at 8x8 level in absence of a single significant frequency component also leads to four partitions of 4x4 pixels. Flow chart for proposed adaptive GE is as shown in figure 3.

IV. EXPERIMENTAL RESULTS

GEA without partitioning(GE with SEA), GE4x4 (GE with MSEA i.e., GE with fixed size sub blocks, for level=3), GE with partitioning using variance of MB and GE with partitioning using variance and Hadamard coefficients have been implemented using matlab and the PSNR variations for four different input sequences have been observed.

Caltrain, missa, trevor and salesman are the four different input sequences used. Average PSNR for 10 input frames is computed.

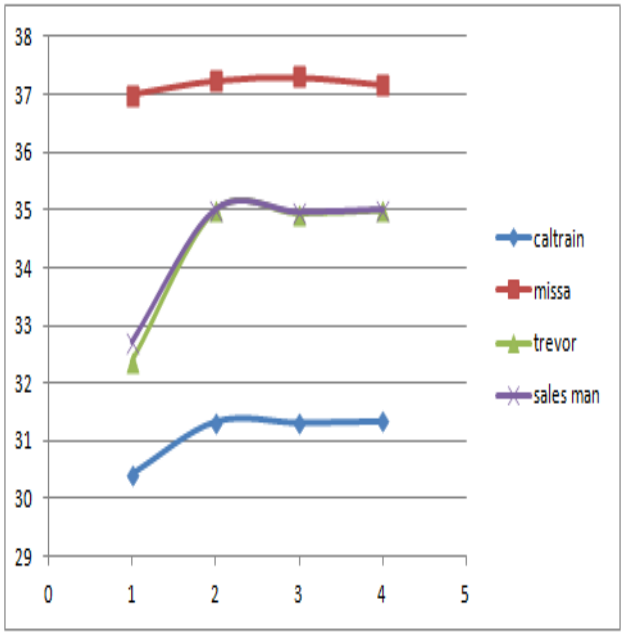

Fig. 4: PSNR variation for different algorithms(matlab)

X-axis
1 represents GEA without sub blocks
2represents GE4x4(GE with level=3)
3 represents GEA with variance based partitioning (Variance threshold=6000)
4 represents GEA with variance and hadamardtransform based partitioning (variance threshold=100, frequency threshold=10).
Y-axis represents Average PSNR values in dB

Table1 represents the effect on PSNR due to adaptive partitioning. For GE with variance based partitioning, Variance threshold1 used is, 6000 and variance threshold2 is, 600.For GE with variance and Hadamard transform based partitioning, variance threshold is 100 and frequency threshold is 10.

From table 1, GE with variance based partitioning results in slight reduction in PSNR values but the no. of computations are also reduced. Computational complexity is heavily reduced for slight reduction in PSNR values. For threshold2, PSNR values are increased but also the no. of sub blocks because of reducing the variance threshold.

From table 1, for Caltrain input sequence proposed adaptive GE will result in increase in average PSNR compared to GE4x4 and GE with variance based partitioning for chosen frequency and variance threshold values. Due to adaptive partitioning no. of sub blocks and the computations will be reduced compared to GE4x4.

JM12.4 platform is also used to implement MSEA, GEA without sub blocks, GE4x4 and GE with variance based partitioning. Six different input sequences are used. the average PSNR and bit rate variations have been observed. Figure 5 shows the PSNR variation of different sequences for different algorithms.

In this, On X-axis,

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

1represents GEA with SSAD similar to Sea,
2 represents GEA with SSAD similar to MSEA and 3 represents GEA with variance based partitioning.
Y- axis represents PSNR values in dB.

Table 2 represents the variation in PSNR for different input sequences. Variance threshold used is, 7000. There is only minimal variation in the PSNR values but no. of computations will reduce due to the high threshold used.

Table 1 PSNR variation for different input sequences

S. No.	Input Sequence	GE4x4	GE with variance based partitioning Threshold1	GE with variance based partitioning Threshold2	GE with variance and Hadamard transform based partitioning
1	Caltrain	31.3301	31.3128	31.32794	31.3376
2	Missa	37.42	37.3003	37.1022	37.1595
3	Trevor	34.9785	34.9237	34.97848	34.9779
4	Sales man	35.0042	34.958	35.00356	35.0041

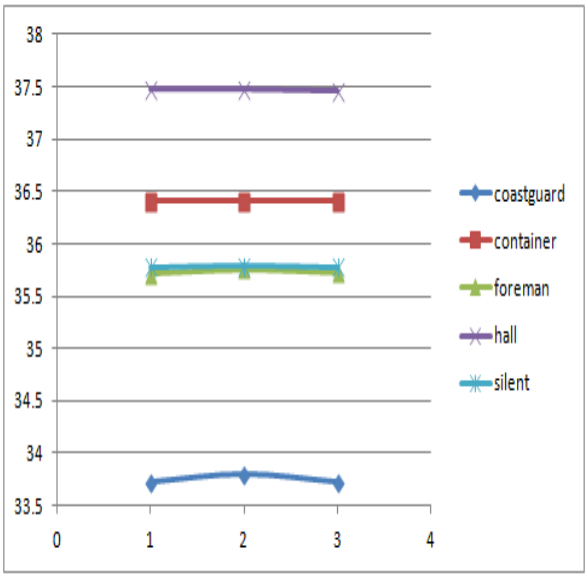

Fig. 5: PSNR variation for different algorithms(JM)

Below figure shows the Bit rate variation of different sequences for different algorithms. In this X-axis represents different algorithms as mentioned above and Y-axis represents Bit-rate in Kbps.

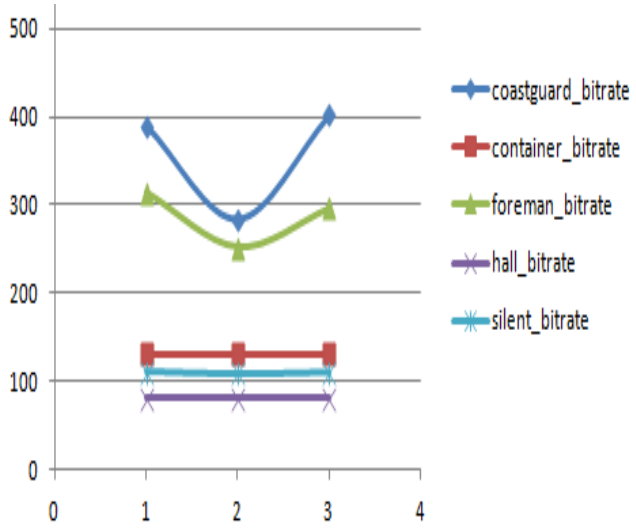

Fig. 6: Bit rate variation for different algorithms(JM)

Because of adaptive GE there is no variation in PSNR bur bit rate increases slightly because of reduction in computational complexity.

V. CONCLUSION

Proposed Adaptive Global Elimination algorithms resulted in less no. of computations and PSNR values achieved are similar to that of full search. Reduction in computational complexity resulted in slight increase in bit rate. Variance based partitioning is simple to implement compared to variance and hadamard based partitioning but uses only fixed size sub blocks and hence computations will be high compared to variance and hadamard based partitioning.

Table 2:PSNR variation for different inputs(JM)

S.No.	Input Sequence	GE4x4	GE with variance based partitioning
1	Coast guard	33.8	33.72
2	Container	36.41	36.41
3	Foreman	35.76	35.72
4	Hall	37.48	37.46
5	Mobile	32.83	32.83
6	Silent	35.79	35.78

REFERENCES

1. Li, W.; Salari, E., "Successive elimination algorithm for motion estimation," *Image processing, IEEE Transactions on* , vol.4, no.1, pp.105-107, (Jan 1995)
2. Jong-Nam Kim; Tae-Sun Choi, "A fast three-step search algorithm with minimum checking points using unimodal error surface assumption", *Consumer Electronics, IEEE Transactions on* , vol.44, no.3, pp.638-648, (Aug 1998)
3. Lai-Man Po; Wing-Chung Ma, "A novel four-step search algorithm for fast block motion estimation ," *Circuits and Systems for Video Technology, IEEE Transactions on* ,vol.6, no.3, pp.313-317, (Jun 1996)

4. Y.L.Chan, W.C.Siu, "New Adaptive Pixel Decimation for Block Motion Vector Estimation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, No. 1, pp.113-118, (Feb 1996)
5. Yu-Wen Huang; Shao-Yi Chien; Bing-Yu Hsieh; Liang-Gee Chen, "Global elimination algorithm and architecture design for fast block matching motion estimation," *Circuits and Systems for Video Technology, IEEE Transactions on* , vol.14, no.6, pp. 898-907, (June 2004)
6. C.Z., Xiao Lin, Lap-PuiChau, "Hexagon-Based Search Pattern for Fast Block Motion Estimation", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 12, No. 5, pp349-355 (May 2002)
7. X. Q. Gao, C. J. Duanmu, and C. R. Zou, "A Multilevel Successive Elimination Algorithm for Block Matching Motion Estimation", *IEEE Transactions On Image Processing*, Vol. 9, No. 3, pp.501-504, (March 2000)