AN ADAPTIVE, FEATURE-BASED LOW POWER MOTION ESTIMATION ALGORITHM

Ajit Gupte[1], Amrutur Bharadwaj[2]

Texas Instruments India LTD[1], Indian Institute of Science (ECE Dept.) [1,2]

ABSTRACT

Motion Estimation is one of the most power hungry operations in video coding. While optimal search (eg. full search) methods give best quality, non optimal methods are often used in order to reduce cost and power. Various algorithms have been used in practice that trade off quality vs. complexity. Global elimination is an algorithm based on pixel averaging to reduce complexity of motion search while keeping performance close to that of full search. We propose an adaptive version of the global elimination algorithm that extracts individual macro-block features using Hadamard transform to optimize the search. Performance achieved is close to the full search method and global elimination. Operational complexity and hence power is reduced by 30% to 45% compared to global elimination method.

1. INTRODUCTION

Modern video coding methods use block matching technique for motion estimation. For each 16x16 pixel 'macro-block' in current frame a best match is found within a window (search window) in the reference frame. Matching criterion used is typically the sum of absolute difference in pixels or 'SAD'.

The complexity and power of motion estimation varies from 50% to 90% of total encoder complexity [1] and power dissipation. To reduce the complexity of motion estimation, various algorithms have been proposed. While lossless or optimal algorithms such as successive elimination[2] and integral projection[4] achieve the same quality as the full search algorithm (FS), they tend to be prohibitively expensive for real time processing in applications with high resolution video.

Three Step Search (TSS), New Three Step Search (NTSS)[4],[5],[6] etc. significantly reduce complexity by restricting the search range to very few points. Such algorithms assume 'Uni-modal Error Surface' in which the matching error would increase monotonically with the distance from the global minimum. In high resolution applications, such algorithms cause excessive quality loss compared to the full search method due to local minima in the search space. Some algorithms simplify the SAD calculation by pixel sub-sampling, exploiting spatial correlation of pixels. Eg. alternate pixel pattern, quincunx patterns etc. Partial Distortion Elimination(PDE) based on various approaches such as selecting high gradient pixels, pseudo-random patterns, uniform grid, etc [7],[8],[9] achieve results close to FS, with moderate complexity reduction in SAD computation compared to FS, and additional complexity of adaptively addressing only selected pixels in the reference blocks. Pixel truncation approach is used in [10]to reduce complexity.

Averaging, instead of sub-sampling, is used in the mean pyramid method[11]. A pyramid of reference frames is constructed by averaging over successively wider ranges of pixels. The motion search starts at the top of the pyramid and is refined at the lower levels. The global elimination (GE) method[12] also uses averaging approach but achieves comparable quality as the FS, thus providing a reasonable trade off between quality and complexity. It is less susceptible to local minima because it reduces complexity in spite of not eliminating any search points unlike the mean pyramid method.

We propose an adaptive version of global elimination. Macro-block features are extracted by calculating Hadamard transform coefficients, based on which, the GE search complexity is modulated, thereby optimizing the search effort. In the next section, we describe the global elimination method briefly. In section 3, the proposed algorithm is described in detail. In section 4 we analyze the complexity of the proposed method in comparison with the GE method and the FS method. In section 5 experimental results are provided.

2. OVERVIEW OF THE GLOBAL ELIMINATION (GE) ALGORITHM

In the first stage of global elimination method, each current macro-block (MB) is divided into sub-blocks and the sub-block averages are calculated. A reference macro-block at each search point is also identically divided into equal number of sub-blocks. The SAD in sub-block averages (SADA) between current macro-block and reference macro-block is used as the matching measure rather than a complete SAD. The complexity reduction is achieved both because 'SADA' is cheaper to calculate than 'SAD' and also because the averages at successive search points can be incrementally calculated from the previous search point's averages (Figure 5). Top 'M' candidates with least SADA values are selected for the 2nd stage search. During the 2nd stage, a full search (FS) is performed on all the M candidates and the block with minimum SAD value is selected as the best match.

3. PROPOSED ALGORITHM (AGE)

Figure 1 and Figure 2 represent error surfaces generated from two macro-blocks in a real video frame. Vertical axis is error measured as SAD between the current macro-block and a macro-block shifted by amount (x,y) in the same frame. The two horizontal axes define the displacement. The error is 0 at position of zero displacement ((x,y) = (33,33)) and increases with increasing distance from the current macro-block. Figure 1 is an error surface of an MB with a prominent edge, while Figure 2 is

that of an MB without any. The error increases much more rapidly with displacement in Figure 1 compared to Figure 2.

So, a featureless macro-block will typically have several candidate blocks in the reference frame which have a SAD value close to minimum, and hence it is difficult to find the best match for such a macro-block using a crude (simpler) search. On the other hand, a macro-block with one or more prominent features (edges) will have fewer candidates that match well and hence it is relatively easier to match.

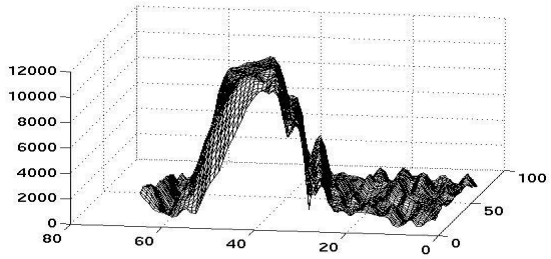

Figure 1 : Error surface of a macro-block containing a prominent edge

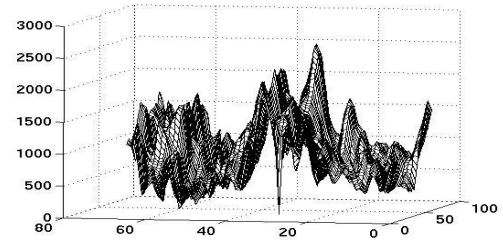

Figure 2 : Error surface of a macro-block containing no significant features.

We make use of this fact in the proposed algorithm which we call adaptive global elimination (AGE). Specifically, we modulate the number of sub-blocks in which a macro-block is divided in the global elimination algorithm according to the features in the current macro-block. Since a 'feature macro-block' is easy to match, we divide the macro-block in fewer number of sub-blocks, while a 'featureless macro-block' is divided into larger number of sub-blocks. Another important reason to why such a scheme would work, one can see that if a block has some significant visible features – they will still be distinguishable even after averaging over a large area that does not cross an edge. On the other hand, if a macro-block does not have too many distinguishing visible features, then one needs to avoid averaging over large areas so that the local small variations are not washed out – since they are essential to find a good match.

Hadamard transform[13] is used to extract features in current MB because of its relative simplicity. Also, it serves the purpose of extracting significant features or edges in the current macro-block very well. The current MB is partitioned into 4x4 sub-blocks and 4x4 Hadamard transform (equation 1) is performed on the sub-block averages. Let f(x,y) be the Hadamard transform coefficients.

$f = A^T MB_{4x4} A$, where MB_{4x4} is a 4x4 matrix of sub-block averages of current macro-block, and

$$A = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & -1 \end{bmatrix} \quad \text{-------- (1)}$$

Figure 3 illustrates some example macro-blocks with significant features and corresponding large-value Hadamard transform coefficients.

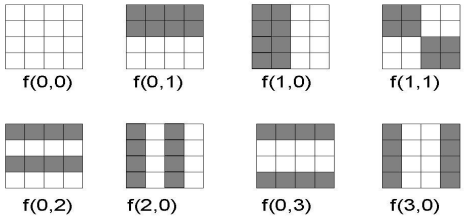

Figure 3 : Some 'featured' macro-blocks and corresponding large-value Hadamard coefficients

Based on the values of significant Hadamard transform coefficients, the sizes of partitions for first level AGE search are selected. Eg. 2x2 partitioning of an MB will preserve features corresponding to f(0,1), f(1,0) and f(1,1) but will not preserve any of the features in bottom row of Figure 3. Figure 4 pseudo-code illustrates AGE with 4 partitioning options.

```
If |f(0,1)| > thr2x2 or |f(1,0)| > thr2x2 or
|f(1,1)| > thr2x2
   Partition_option = 2x2
Else if (|f(0,2)| + |f(0,3)|> thrHI) and
(|f(2,0)| + |f(3,0)|< thrLO)
  Partition_option = 4x1
Else if  (|f(2,0)| + |f(3,0)| > thrHI) and
|f(0,2)| + |f(0,3)| < thrLO)
   Partition_option = 1x4
Else    Partition_option = 4x4
```

Figure 4 : Pseudo code for AGE with 4 partitioning options

More Hadamard transform coefficients can be observed to allow more possible partitioning options such as 4x2 or 2x4. Threshold values can be tuned to change number of MBs falling in each partition option to trade-off complexity vs quality.

4. COMPELXITY ANALYSIS

We analyze the complexity of first stage of GE with different partition sizes. Measures of complexity (and hence power) are number of add/sub/abs operations and number of on-chip memory accesses. Operations other than the first stage of GE are in-frequent and are ignored in the analysis. It is assumed that, mean values and sums of first column (SOFC) (or row – SOFR) of each partition of current MB are used for mean value calculations for successive MB in the current frame.

4.1. Number of computations

In case of 2x2 partitions shown in Figure 5, assume that m1, m2, m3, m4 values are known. For a macro-block shifted right (or down) by 1 pixel position, the new mean values m1'…m4' can be

calculated from m1..m4 by subtracting and adding sum of appropriate columns (or rows).

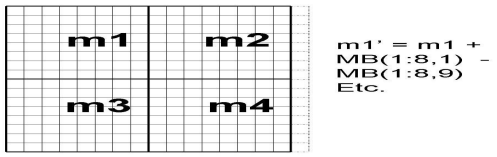

Figure 5 :dividing an MB into 2x2 partitions

4.2. Memory Accesses

In FS method, the motion estimation engine needs to access all the 256 pixels from memory (typically on-chip SRAM) for every search position, while in case of GE and AGE, only 16 new pixels for each consecutive reference position are accessed, along with the stored mean and SOFC/SOFR values. Table 1 summarizes the number of operations and memory accesses for SADA calculation with different partition sizes. When number of column (row) partitions is larger than number of row (column) partitions, the search order within the search window is made column (row)-wise. Hence, mxn and nxm partitions have equal complexity. Complexity increases with number of partitions.

Table 1 : Number of operations and memory accesses for SADA calculation for different partition sizes.

No. of Partitions	No. of Oper-ations	No. of ops relative to 4x4	No. of mem accesses	No. of mem accesses relative to 4x4
16x16 (F.S.)	767	8.429	256	3.2
4x4	91	1	80	1
4x2 or 2x4	51	0.56	48	0.6
4x1 or 1x4	31	0.341	32	0.4
2x2	33	0.363	32	0.4

5. EXPERIMENTAL RESULTS

In all experiments, the number of search points for the 2nd stage of GE is set to 10 i.e., the top 10 candidates are chosen from the first stage which are then examined using a full search in the second stage. The search window size is +/-32 in both horizontal and vertical directions ie. total 4225 search points. MPEG2 VLC encoding is used to determine bit rate. We begin with a simple demonstration the algorithm. In this experiment, only two partitioning choices are offered at the first stage of AGE (2x2 or 4x4). Table 2 compares resulting bit rate at the same quantization factor with five different algorithms applied to a single frame of 'coastguard' sequence. The FS algorithm gives the best performance closely followed by GE with 4x4 partitions. The performance of GE with 2x2 partitions is significantly poorer at much less ME complexity. The AGE provides a trade-off between motion estimation complexity and resulting quality. In 'inverted AGE' algorithm, we invert the decision to select 4x4 vs 2x2 partitions. This step is deliberately added to demonstrate the effectiveness of AGE algorithm. Even with smaller number of 4x4 partitions, the AGE results in smaller bit-rate compared to the inverted AGE, proving the usefulness of partition size selection based on macro-block features in AGE. In, Figure 6 the 'thr2x2' values are varied from low to high to sweep across entire range of number of 4x4 partitions possible. The AGE curve shows rapid fall in bit rate compared to inverted AGE with increase in number of 4x4 partitions.

Table 2 : Effectiveness of AGE on a 640x480 'coastguard' frame

Algo	#2x2 parti--tions	#4x4 parti--tions	Encoded Bits	#Ops	#mem ace-sses
FS	-	-	2,40,186	920400	307200
GE (all 4x4 parti--tions)	0	1200	2,43,476	109200	96000
GE (all 2x2 parti--tions)	1200	0	2,89,086	39600	38400
AGE	681	519	2,63,758	69702	63312
Inverted AGE	637	563	2,74,846	72254	65424

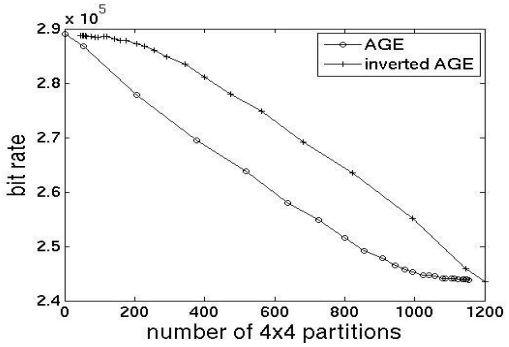

Figure 6 : Bit Rate Vs #4x4 partitions with correct and inverted thresholds

Table 3 : Experiment2 with 4 partitioning options on a 640x480 'coastguard' frame

Algo	#1x4 parts	#4x1 parts	#2x2 parts	#4x4 parts	Encoded Bits	#Ops
Case1	0	70	256	874	2,48,024	90152
Case2	70	0	256	874	2,52,019	90152

Case1:AGE; **Case2:**AGE with threshold interchanged for 4x1 and 1x4 selection.

Table 3 shows similar result with four possible partitioning options. Case1 and case2 have same complexity (number of operations and memory accesses) but case2 uses inverted threshold and results in poorer performance. (Number of memory accesses in both cases is identical (80352).)

We now demonstrate the results for longer video sequences. Table 4 shows PSNR degradation w.r.t. full search method at different bit rates for 58 frames of 'coastguard' (C.G.) strip. Six partition sizes (2x2,4x1,1x4,2x4,4x2,4x4) are allowed and many more Hadamard transform coefficients are used in selecting the appropriate partition size, again, the goal being to preserve

prominent features during averaging. The 'random match' experiment is designed to select identical number of partitions of various sizes as chosen by the AGE (found out by post processing AGE results), but in random fashion, so that it has same overall complexity as the AGE. However, it results in poorer performance compared to AGE.

Table 4 : 'Coastguard' video PSNR at different bit rates with GE, AGE & Random Match

Bits	30.2 Mb	22.4 Mb	17.7 Mb	14.4 Mb	12.0 Mb	8.6 Mb
Full Search	39.15	36.00	33.92	32.41	31.25	29.58
GE	39.02	35.87	33.76	32.18	30.98	29.08
AGE	38.85	35.70	33.57	31.99	30.76	28.80
Random Match	38.62	35.49	33.33	31.68	30.40	28.43

Figure 7 shows similar results for the 'mobile calendar' (M.C.) video strip graphically. Table 5 summarizes complexity savings for both video strips compared to GE.

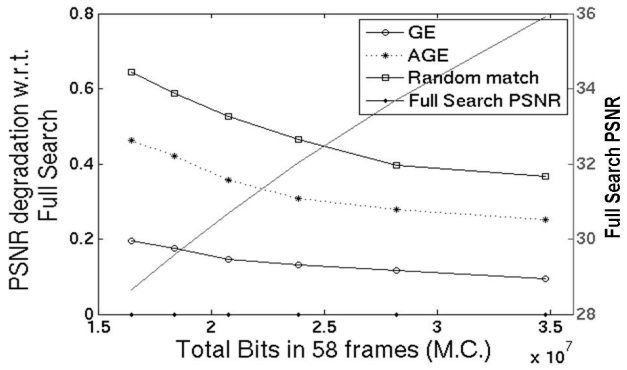

Figure 7 : 'Mobile-calendar' video PSNR degradation at different bit rates with GE, AGE and random match

Table 5 : Savings in #ops and #mem accesses wrt GE. The number of ops. and memory accesses are per MB averages.

Algo & video	Mem Acce-sses	Mem accesses Normalized to GE	Number of operations	#ops normalized to GE
FS	256	3.2	767	8.4
GE	80	1	91	1
AGE (M.C.)	45.13	0.56	48.3	0.53
AGE (C.G.)	56.88	0.71	62.36	0.69

6. CONCLUSION

It is demonstrated that motion search for blocks with prominent features is easier compared to blocks devoid of features. Further, search can be optimized by ensuring that key macro-block features are preserved while reducing complexity. An adaptive global elimination algorithm based on this idea is proposed that results in 30% to 45% power saving in motion estimation search without significant loss in performance compared to global elimination method. Hadamard transform is used to study the feature contents in a macro-block. This lends to optimized partition size selection for each macro-block. The effectiveness of AGE is proven with the help of random selection and inverted selection that show degradation in performance for same complexity compared to AGE.

7. REFERENCES

[1] Y.Murachi etal. "A 95mW MPEG2 MP@HL Motion Estimation Processor Core for Portable High Resolution Video Application", *2005 Symposium on VLSI Circuits Digest of Technical Papers.*

[2] W.Li, E. Salari "Successive Elimination Algorithm for Motion Estimation", *IEEE Transactions on Image Processing, Jan 1995*

[3] J.S. Kim, R.H. Park "A Fast Feature-based Block Matching Algorithm Using Integral Projections" *IEEE Journal on Selected Topics in Communications, June 1992, pp.968-971*

[4] J.N.Kim, T.S.Choi, "A Fast Three-Step Search Algorithm with Minimum Checking Points Using Unimodal Error Surface Assumption",*IEEE Transactions on Consumer Electronics, Aug 1998*

[5] L.M.Po, W.C.Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", *Circuits and Sytems for Video technology,* IEEE, June 1996, pp.313-317.

[6] R.Li, B.Zeng, M.L.Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", *IEEE Transactions on Circuits and Systems for Video Technology, Aug 1995*

[7] B. Tao, M.T. Orchard, "Gradient-Based Residual Variance Modeling and Its Applications to Motion-Compensated Video Coding", *IEEE Trans. on Image Processing, Jan 2001*

[8] B.Montrucchio, D.Quaglia, "New Sorting Based Lossless Motion Estimation Algorithms and a Partial Distortion Elimination performance Analysis", *IEEE Transactions on Circuits and Systems for Video Technology, Feb 2005*

[9] Y.L.Chan, W.C.Siu, "New Adaptive Pixel Decimation for Block Motion Vector Estimation", *IEEE Transactions on Circuits and Systems for Video Technology, Feb 1996*

[10] Z.L.He,C.Y.Tsui,K.K.Chan,M.K.Liou,"Low Power VLSI Design for Motion Estimation Using Adaptive Pixel Truncation" *Ciruits and Systems for Video Technology*, IEEE, August, 2000, pp.669-678.

[11] K.W.Nam, J.S. Kim, R.H. Park, Y.S.Shim, "A Fast Hierarchical Motion Estimation Algorithm Using Mean Pyramid", *Ciruits and Systems for Video Technology*, IEEE, Aug 2005, pp. 344-355.

[12] Y.W.Huang, S.Y.Chien, B.Y.Hsieh, L.G. Chen, "Global Elimination Algorithm and Architecture Design for Fast Block Motion Estimation.", *Ciruits and Systems for Video Technology*, IEEE, June, 2004, pp. 898-907.

[13] M.S.Porto etal., "Design Space Exploration On the H.264 4x4 Hadamard Transform.", 2004 IEEE International Conference on Multimedia and Expo (ICME).