A Fast Block Motion Estimation Algorithm Based on Motion Adaptive Partitioning

Mani V T, Anup Shah & G Chandrashekar Reddy
Tata Elxsi Ltd., Bangalore

Abstract

We present below a new efficient Block Matching Algorithm (BMA) for video coding applications. This method uses variable-sized blocks and motion based partitioning to predict the nature of the blocks and coalesce suitable blocks into larger blocks. These coalesced blocks are consequently motion estimated, whereby decreasing the computational complexity substantially.

Keywords*: Motion Estimation, Block Matching Algorithm and Block Distortion Method*

1 Introduction

Video compression is a necessity when it comes to efficient storage and transmission of digital video signals and thus has numerous applications in critical digital technologies such as High Definition Television (HDTV), teleconferencing and CD-ROM video data storage and retrieval. In particular, motion compensated interframe predictive coding has received considerable attention as it aims at removing the high temporal redundancy between successive frames. This has thus been adopted by many video coding standards such as H.261, H.262, H.263 [1], from ITU-T (Telecommunication standardization sector of International Telecommunication Union), MPEG-1 MPEG-2 [2] MPEG-4 [3] from ISO (International Standardization Organization).

Motion estimation obtains the motion compensated prediction by finding the motion field between the reference frame and the current frame. The most popular motion compensation method is the block-matching algorithm (BMA). The BMA partitions each image into a fixed number of small blocks and for each block a search is conducted with in a predefined window of the previous frame to find the best match. The acceptance criterion is usually based on minimizing the mean square error or mean absolute error between the two sets of pixels, and the relative displacement between the two blocks is taken to be the motion vector. The computational requirements depend on the search method, search window size and the acceptance criterion. Owing to their inherent simplicity, both in terms of hardware and software implementation, block-based methods have always been employed for motion estimation. Among the BMAs, the Full Search (FS) has a simple implementation but requires massive computation, which is often too high for real time implementation. To alleviate the high computational requirements of an FS, a number of fast BMAs such as 2D-Logarithmic search (2DLOG) [4], three-step search (3SS) [5], the new three-step search (N3SS) [6], the four-step search (4SS) [7], etc have been proposed. These rely on the assumption that Block Distortion Measure (BDM) increases monotonically as the checking point moves away from the global minimum BDM point. But these fast algorithms tend to get trapped in local minima en route of their search of a global minimum, owing to their sub-optimal search strategies.

The success of the above algorithms relies on the assumption that motion within each block is uniform. But motion in real life is rather composed of different sized blocks with different motions associated with them. In order to handle this, we propose a new algorithm using variable-sized blocks and motion based partitioning. In this approach, motion is classified into blocks of varying sizes based on an adaptation pattern. These variable-sized blocks are consequently motion estimated using different block matching algorithms depending on the type of motion that they represent.

The rest of the paper is organized as follows. Sec.2 discusses the algorithm. Sec.3 discusses the performance evaluation issues and the simulation results are presented in Sec. 4

2 Adaptive Search Algorithm

In general moving scenes, distribution of motion among spatially and temporally neighboring blocks have an implicit similarity. The clustering of motion in one frame can thus be predicted from the motion distribution of the blocks in the previous frame and the spatially adjacent blocks of the current frame. This fact is taken advantage of by the proposed algorithm, which adopts a two-stage partitioning strategy prior to motion estimation. The first stage is an initial partitioning stage, which is based on data available from the past. And the second stage is a refinement stage where the partitions formed from step one are refined with the data available currently based on the causal neighbors.

Step 1: *Determination of initial block sizes*

In this first stage, blocks are classified either as super macroblock (32X32) or macroblock (16X16) or block (8X8) depending on the distribution of motion in the block clusters. This distribution is obtained by analyzing the previous frame motion vectors for the co-located block. Empirically it was found that, if the magnitude of the full-pel motion vectors is <= 1 and the angular deviation is <= 45°, the region constituted slow or zero motion and the block cluster could be treated as a super macroblock (4MB). If magnitude is <= 2, it could be treated as a macroblock (MB) with medium scale motion. If the above conditions fail, it is treated as an individual Block. Thus in Figure 1 the first four blocks will be treated together as a macroblock while the remaining will be considered for block motion estimation.

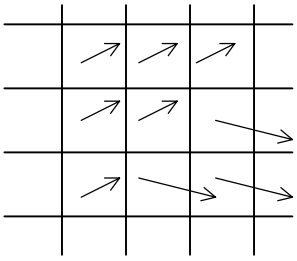

Figure 1 Motion Vector Deviations

Step 2: *Refinement of block sizes*

The second stage is to address the problem of motion adaptation and partitioning in case of a scene change. In this case, each 4MB or MB is checked with the state of the causal neighbors B1, B2 and B3 as shown below

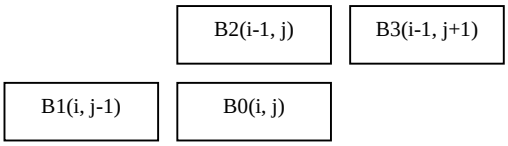

Figure 2 Geometry of Causal Neighbors

The causal neighbors give a good indication of the motion field and can be employed to refine the partition obtained from the first stage. Thus if the causal neighbors of a MB or 4MB have been classified as Blocks, motion field can be assumed to indicate high motion. It then becomes imperative to refine the classification from 4MB or MB to Block.

The algorithm is as follows

AdaptiveMotionEstimation
begin
  DecideBlockSize
  if ($block \equiv 4MB$)
    $4MBMotionEstimation$
    else if ($MinSAD_{4MB} \geq Threshold_{4MB}^{MB}$)
      $MBMotionEstimation$
    else if ($MinSAD_{4MB} \geq Threshold_{4MB}^{B}$)
      $BlockMotionEstimation$
    else
      $StoreMotionVector$ ($4MB$)
    endif
  else if ($block \equiv MB$)
    $MBMotionEstimation$
    if ($MinSAD_{MB} \geq Threshold_{MB}^{B}$)
      $BlockMotionEstimation$
    else
      $StoreMotionVector$ (MB)
    endif
  else
    $BlockMotionEstimation$
  endif
end

DecideBlockSize
begin
  if ($MVAnalysed \Rightarrow ZeroMotion$)
    $BlockType \Leftarrow 4MB$
  else if ($MVAnalysed \Rightarrow SlowMotion$)

$BlockType \Leftarrow MB$

else

$BlockType \Leftarrow B$

endif

if $(CausalNeighbours \Rightarrow HighMotion)$

$BlockType \Leftarrow B$

endif

end

The thresholds used in the above algorithm can be found heuristically. Figure 3 shows a graph of the distortion per block with the threshold per pixel. It can be seen that the distortion per block remains constant in the initial stages and slowly increases thereafter. Accordingly, a threshold value of 1.5 has been used for breaking out from the 4MB search or from MB search to perform a finer search in the regions where the motion field is more turbulent

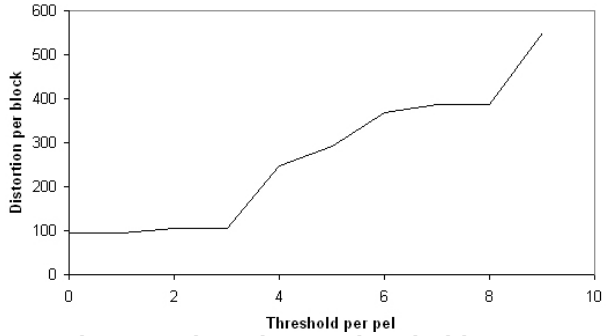

Figure 3 Distortion to Threshold Curve

3 Complexity Comparison

This algorithm has been compared with full search and log three-step search as can be seen in simulation results. The new algorithm has been implemented using log three-step search for the block and MB and a nine-point spiral search for the 4MB. The complexity factor $\Phi_{M,N}$, which can be defined by the number of points to be searched per block per frame, has been computed. This computation is performed for a search window of size 15 (-7 to +7) assuming the absence of unrestricted motion estimation. The reduction of search points at the fringes of the frame has been accommodated in the additional terms represented in the complexity factor. The starting search-point has been obtained by employing the Prediction Model Search [8].

3.1 Full search (FS):

The Full Search involves brute force searching of all the points in the search window to find the block with the minimum BDM. This BMA gives the best possible match for a block in the current frame, but is computationally very expensive. The computational requirements for a full search algorithm can be calculated using the equation

$$\Phi_{M,N} = \frac{[15(M-2)+16][15(N-2)+16]}{M*N}$$

where

M - Number of rows of blocks

N - Number of columns of blocks

3.2 Three-Step Search (3SS):

This fast BMA, proposed by Koga et al. [4] is based on a logarithmic descent in obtaining the best match. The computational requirements for this BMA can be found using the equation

$$\Phi_{M,N} = \frac{25[4(M-2)(N-2)+3(M-2)+3(N-2)+8]}{4*M*N}$$

where

M - Number of rows of blocks

N - Number of columns of blocks

3.3 Adaptive Search (AS):

This algorithm uses the partitioned information to motion estimate block clusters. In the case of super macroblock (4MB) a nine-point search is carried out around the co-located block. If the minimum calculated SAD exceeds $Threshold_{4MB}^{MB}$ or $Threshold_{4MB}^{B}$, the 4MB is broken down into MB or Blocks respectively and 3SS is performed. Similarly, the MB is broken down into Blocks depending on the $Threshold_{MB}^{B}$. This indicates that as the failure rate increases the complexity factor $\Phi_{M,N}$ also correspondingly increases. The overall computational requirements will be the product of the classification probability and the computational requirement of the strategy used. It can be thus put forward as follows

$$\Phi_{M,N} = \phi_B \sum_{i,j} p_{i,j}(B) + \phi_{MB} \sum_{i,j} p_{i,j}(MB) + \phi_{4MB} \sum_{i,j} p_{i,j}(4MB)$$

where

$$\phi_B = \frac{25[4(M-2)(N-2) + 3(M-2) + 3(N-2) + 8]}{4 * M * N}$$

$$\phi_{MB} = \frac{25[2(M-2)(N-2) + 3(M-4) + 3(N-4) + 16]}{8 * M * N}$$

$$\phi_{4MB} = \frac{[3(M-8) * 3(N-8)]}{16 * M * N}$$

with the constraint that

$$\sum_{i,j} [p_{i,j}(B) + p_{i,j}(MB) + p_{i,j}(4MB)] = 1$$

This probability distribution is highly sequence dependent and quantification is not viable. But in sequences with minimal motion variation $p_{i,j}(4MB)$ tends to supercede $p_{i,j}(MB)$ and $p_{i,j}(B)$ whereby the overall computational complexity is decreased. In case of the sequences with high motion variations, $p_{i,j}(B)$ causes the complexity to approach 3SS complexity. In addition to the above, the computational complexity is also influenced by the failure rate of the classification. In sequences with minimal motion variation, the classifications are maintained. This reduces the failure rate and consequently the complexity factor. In comparison, complex sequences tend to have an increased failure rate, which in turn leads to an increase in the computational complexity.

4 Simulation Results

The adaptive algorithm was simulated using the luminance components of sequences tabulated below. It was tested against Full Search and Three-Step Search using image sequences of both CIF (352 X 288) and QCIF (176 X 144) sizes and involving varied motion distributions. In the case of FS and 3SS, blocks of size 8X8 were used and in AS, 8X8, 16X16 and 32X32 sized blocks were employed using Sum of Absolute Differences (SAD) as the BDM. The performance comparison in terms of Mean Square Error (MSE) variation between the different algorithms are shown in Figure 4 (slow uniform motion - Akiyo), Figure 5 (medium motion - carphone) and Figure 6 (extreme motion - Stefan) for the first 40 frames. It may be observed that in the cases where motion is slow and uniform, the variation in MSE between Full search, 3SS and AS is negligible but as the motion field becomes more complex the variation between the algorithms become evident. It can also be observed that 3SS and AS have overlapping MSE pattern with minimal variations indicating the superiority of the adaptive search in terms of the computational complexity with imperceptible degradation of quality. Table I and Table 2 gives some statistical performance comparison of the new adaptive algorithm with respect to FS and 3SS. Table 1 shows the average search points per block required for getting the best match. Table 2 shows the average MSE using the three different algorithms. It may be seen that the average MSE of AS is very similar to that of 3SS. The computational advantage of AS can be observed in the sequences with slow uniform motion such as “akiyo” and “grandma”. Table 3 gives the probability distribution of the various classifications under the adaptive search and Table 4 provides the failure rate of the classification methodology. It is evident from the results, those in sequences where the motion is slow, the classification probability $p_{i,j}(4MB)$ is the factor, which influences the computational complexity, and due to the reduced failure rate in these sequences, a high computational advantage is obtained. Figure 7 shows the partitioned Blocks, MBs and 4MBs for the frame sequence “Tennis”.

5 Conclusion

A variable-sized BMA using motion based partitioning is described above. Computationally, this algorithm performs much better than the 3SS when the video involves a slow uniform varying sequence and has similar performance to the 3SS when the sequence involves a turbulent motion field. It is also evident that the variation of MSE between FS and AS is negligible when motion variation is slow and thus it can be successfully implemented in videophone applications where motion fields are usually uniform.

6 References

[1] ITU-T SG15, “Video coding for low bit rate communication”, in Draft ITU-T recommendation H.263, May 1996.

[2] ISO/IEC 13818-2 (MPEG-2 Video), “Information Technology - Generic coding of Moving Pictures and associated Audio Information: Video”, 1995.

[3] ISO/IEC 14496-2 (MPEG-4 Video), “Information Technology – Generic coding of audio-visual objects Part 2: visual”, Ver. 18 Dec. 1998.

[4] J.R.Jain and A.K.Jain, “Displacement measurement and its application in interframe image coding”, IEEE Trans. Commun., vol. COM-29, pp. 1799-1808, Dec. 1981

[5] T.Koga, K.Iinuma, A.Hirano, Y.Iijima and T.Ishiguro, “Motion compensated interframe coding for video conferencing”, in Proc. NTC81, pp. C.9.6.1-9.6.5, New Orleans, LA, Nov. 1981.

[6] R.Li, B.Zeng and M.L.Liou, “A new three-step search algorithm for block motion estimation”, IEEE Trans. on Circuits and systems for Video Technology, vol.4, no.4, pp.438-442, Aug. 1994.

[7] L.M.Po and W.C.Ma, “A novel four-step search algorithm for fast block motion estimation”, IEEE Trans. on Circuits and systems for Video Technology, vol. 6, no.3, pp.313-317, Jun.1996.

[8] Jie-Bin Xu, Lai-Man Po and Chok-Kwan Cheung, “A New Prediction Model Search Algorithm for Fast Block Motion Estimation”, Dept. of Electronic Engineering, City University of Hong Kong.

[9] G.R.Martin, R.A.Packwood and I.Rhee, "Variable size block matching motion estimation with minimal error", SPIE Proceedings, vol.2668, pp. 324-333, February 1996.

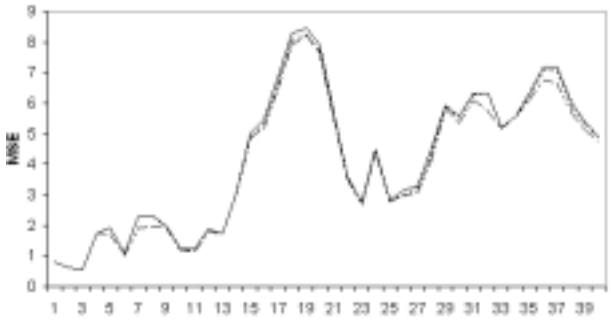

Figure 4 MSE Comparison for Akiyo

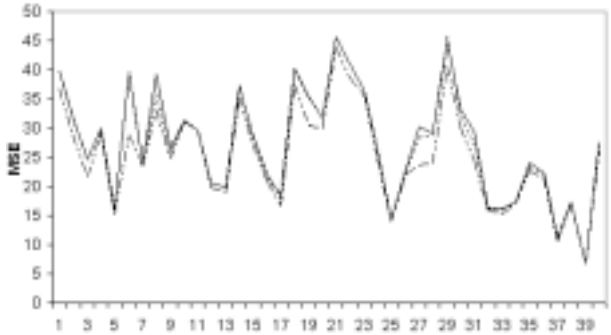

Figure 5 MSE Comparison for Carphone

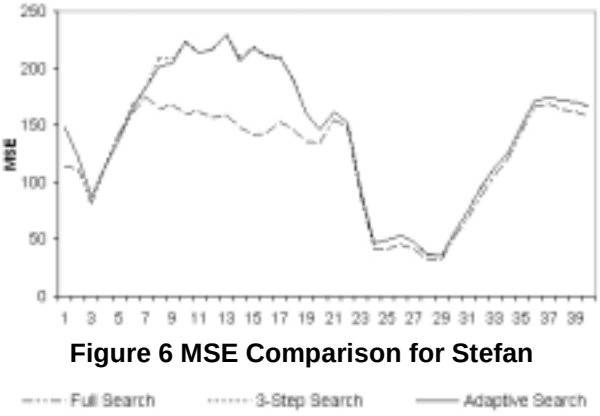

Figure 6 MSE Comparison for Stefan

Table 1 Average Search Points per Block

	FRMS	FS	3SS	AS
Akiyo	300	203.6	23.14	3.742
Car phone	300	204.1	23.54	19.08
Claire	300	203.6	23.17	4.545
Foreman	300	203.7	23.56	22.63
Ms. America	150	203.2	23.33	7.142
Salesman	400	203.8	23.17	5.153
Split Screen	150	203.1	23.20	11.52
Mother Child	300	214.2	24.33	11.28
Tennis	300	213.9	24.29	13.53
Stefan	300	214.4	24.68	20.70

Table 2 Average MSE per Block

	FRMS	FS	3SS	AS
Akiyo	300	2.982	2.991	3.012
Car phone	300	38.165	42.27	42.45
Claire	300	3.614	3.634	3.698
Foreman	300	32.48	37.65	37.887
Ms. America	150	4.762	4.847	5.008
Salesman	400	5.917	6.206	6.261

Split Screen	150	24.46	24.64	24.94
Mother Child	300	5.349	5.58	5.69
Tennis	300	32.83	39.103	39.18
Stefan	300	89.28	96.73	97.01

Table 3 Probability Function for AS in %

	FRMS	4MB	MB	B
Akiyo	300	75.024	18.35	6.621
Car phone	300	14.24	2.976	82.781
Claire	300	73.065	17.56	9.37
Foreman	300	9.081	3.277	87.642
Ms. America	150	64.619	14.602	20.778
Salesman	400	72.139	15.832	12.027
Split Screen	150	38.967	9.240	51.793
Mother Child	300	56.626	4.162	39.211
Tennis	300	46.34	3.745	49.911
Stefan	300	4.986	2.255	92.758

Table 4 Failure Rate for AS in %

	FRMS	4MB	MB
Akiyo	300	3.41	10.139
Car phone	300	21.019	59.494
Claire	300	3.901	12.314
Foreman	300	46.846	72.458
Ms. America	150	5.333	17.817
Salesman	400	4.382	14.703
Split Screen	150	9.708	31.952
Mother Child	300	5.638	44.703
Tennis	300	5.617	38.869
Stefan	300	17.195	30.042