Fast Implementation of Block Motion Estimation Algorithms in Video Encoders

N.R. Koduri¹, M.E. Dlodlo¹, G. DeJager¹ and K.L. Ferguson²
Department of Electrical Engineering,
University of Cape Town¹, Private Bag X3, Rondebosch,
South Africa
and CSIR², Meraka Institute, P.O Box 395, Pretoria 0001,
South Africa
Email: {rohini.koduri, Mqhele.Dlodlo, Gerhard.dejager} @uct.ac.za¹; kferguson@csir.co.za²

Abstract — Block matching algorithms (BMA) are central to optimal frame prediction for motion estimation in video compression. This paper focuses on the efficiency of Hierarchical Search (HS) algorithms. The research proposed two new combinations of fast algorithms like Small Diamond-shaped Search Pattern (SDSP) and Square-Shaped Search Pattern (SSSP) with a three-level Hierarchical algorithm at different levels of hierarchy. The computational complexity and efficiency for each combination algorithm were of interest. Simulation results show that the hierarchical combination algorithms are around 10% faster than the classic hierarchical algorithm with slight improvement or no significant change in video quality when compared to general HS algorithm.

Keywords — Block motion estimation algorithms, Fast algorithms, Multi-resolution algorithms, Combination algorithms

I. INTRODUCTION

A Video sequence can be interpreted as a series of pictures referred to as ‘frames’, assuming non-interlaced sources. High correlation between the successive frames can be expected with high frame rate (measured in frames/second). Video compression adopts an approach where the current frame is predicted from past or near-future frames. Motion estimation involves comparing the macroblocs of the current frame with the reference frame(s) and storing the displacement through motion vector(s) for each macroblock. Thus the current frame is constructed and eventually, the difference between the predicted and the original frame is transmitted instead of the original frame itself. Thus, motion estimation plays a vital part in video compression process. The technique for prediction is provided by Block Matching Algorithm [1].

The full search algorithm (FSA) [2], [3] is employed for optimum prediction as it compares each block of the current frame with every possible block position in the search region. But due to its number of computation per frame, FSA is unacceptably complex in software-based real-time applications. A fast search algorithm attempts to reduce the number of computations involved in the algorithm. The key idea in fast algorithms is to sub-sample the search points to avoid searching all possible locations. A number of fast algorithms such as Cross Search (CS) [4], Three-step-Search (TSS) [2], Efficient Three-Step-Search (E3SS) [5], Diamond Search (DS) [6], HS algorithm [3] etc. have been proposed over time. The main principle of the fast algorithms is to search few sampled points unlike all the points in the FSA. Search patterns like SDSP [7] and SSSP [7] have also been proposed in where the search pattern shape (here a small diamond or square) gets converged around the best match in each step. This paper proposed new algorithms which are modified versions of HS algorithms.

The comparison criteria used in all the algorithms involved in the research is Mean Absolute Difference (MAE). The MAE for NXN sample block can be calculated as follows

\[ MSE = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}| \]  (1)

Where \( C_{ij} \) is a sample of current is block and \( R_{ij} \) is from the reference frame [2].

II. HIERARCHICAL SEARCH ALGORITHM

The Hierarchical Search is a fast algorithm which starts the search with a coarsely sub-sampled version of an image and then successively searches the higher resolution versions until the original resolution of the image is reached. In HS both the current and reference frames are sub-sampled and are compared at different levels of resolution. The versions of the image can be considered as ‘levels’ with the full resolution version being at level 0.

The procedure for the Hierarchical Search is as follows:
Step1: Subsample level 0 by a factor of 2 to produce level 1.
Step2: Subsample level 1 by a factor of 2 to produce level 2. Repeat the sub-sampling to produce further levels (typically three to four levels are sufficient).
Step3: Search the highest level to find the best match to have an initial coarse vector.
Step4: Search the lower level around the coarse motion vector and find the best match.
Step5: Repeat until a best match at level 0 is found.
Step6: Search the half pixel points around the level 0 match [3].

This work is supported by the Council for Scientific and Industrial Research (CSIR) through the DST-Innovation Fund on project no. T70026: Bandwidth Adaptive real-Time Video Broadcasting over Internet.
A 16x16 block at level 0 would imply 8x8 block at level 1 and 4x4 block at level 2 [3]. The computations are made at highest level and with relatively smaller blocks. There might be similar values at several places at next the level. To have maximum refinement for the coarse vector, full search in a smaller search window is done at level 1 with the best match at the search centre.

Many modifications have been experimented on HS to refine the motion vector through the levels. The HS which is considered in this research has a fixed search window for all the three levels. FS is adopted at all the levels of the hierarchy to find the winning positions.

III. Hierarchical Search with SDSP at Top Level (HS+SDSP)

This is the first proposed fast algorithm. This algorithm takes into account the motion range of the motion vector at level 2. The search points are confined to the motion range at level 2. In addition to considering the motion range, it also incorporates SDSP in the highest level of the hierarchy. This attempt is to reduce the computations at this level as the HS would use a full search over a large search area. The figure below explains the algorithm.

The HS with SDSP can be detailed as:
Step1: Subsample level 0 by a factor of 2 to produce level1.
Step2: Subsample level 1 by 2 to produce level2.
Step3: Search the 4 points of SDSP (centre excluded) at the highest level. The diamond points here are placed at the borders of the motion range. Once the best match is found, reduce the width and height of the diamond by half. Continue the search with the best match as the new centre. Repeat this process until a best match is found at level 2. This generates an initial coarse vector.

Step4: Set the best match from level 2 as the centre at level1. Apply FS around the best match. The search must be confined to the search window. The best match found at level1 becomes the centre at level0.
Step5: At level 0 FS is applied until a best match at level 0 is found.
Step6: Continue the search to refine the best point at a half pixel domain.

IV. Hierarchical Search with SDSP and SSSP at Different Levels

This is the second fast search proposed in this research. This algorithm accounts motion range in level 2 and level 1. It merges two search patterns, the SDSP and the SSSP with HS. The attempt is to reduce the computations at two levels.

The SDSP is carried out at the top level which is level 2, searching the 4 points of the diamond at each stage and SSSP is applied at level 1. Figure below shows the process.

Figure 1: Level wise illustration of HS + SDSP algorithm
The procedure of the search is as follows:

Step 1: Subsample the frame to produce a level 1 and level 2 as described in the first two steps of the first proposed algorithm.

Step 2: Calculate the motion ranges at level 2 and level 1. Multiply the motion range at level 2 to get motion range at level 1.

Step 3: Search the 4 points of small diamond at the highest level. Start the search at the borders of motion range and converge the 4 point small diamond around the best match in each stage of level 2. Thus, generate the initial coarse vector.

Step 4: The best match from level 2 forms the centre for level 1. Apply SSSP to search 8 points around the centre. Initially the 8 points of the square are at the edge of the motion range and eventually the square converged around the best match in each stage. Now this is the refined motion vector in the designated motion range at level 1.

Step 5: Carrying out FS at level 0 in the search window around the winning position from level 1.

Step 6: Continue search around the full pixel winner at level 0 at half pixel domain. This is the end result.

V. DISCUSSION ON THE COUNT OF SEARCH POINTS IN EACH ALGORITHM

The speed of a BMA is dependent on various factors such as detail of the video sequence, movement in the video sequence and the computations in the algorithm itself. The lesser the search points faster is the algorithm.

The search window in HS is 14X14 pixels at every level. To allow calculation at the edges, an extended boundary is used in all the levels. FS is employed which searches 196(4X4) blocks at level 2, 196 (8X8) blocks at level 1 and 196(16X16) blocks at level 0. The half pixel search is done only at level 0. 8 half pixel points in a square pattern are searched around the best match from level 0.

The first proposed algorithm which is HS+SDSP uses motion range instead of a constant search window at level 2. The motion range in the level 2 is 8. SDSP at level 2 searches 4 points at each stage with the width (from the centre to the diamond point) of the diamond being 8, 4, 2 and 1. At level 1 the search is again FS over 14X14 search window and so is at level 0. The search proceeds to the half pixel search where it searches 8 points around the best match from the level 0.

The second proposed algorithm strives to reduce the computations at level 1 in addition to level 2. The motion range is considered at both level 2 and level 1. The motion range at level 2 being 8, the motion range at level 1 is 16. The search procedure at level 2 is similar to the previous proposed algorithm, searching 16 points at level 2. At level 1 however, a SSSP is used. SSSP searches 8 points in each stage with the width (from the centre to the square point) being 16, 8, 4, 2 and 1. Thus, searching 40 points at level 1. At level 0 a FS is applied in the search window of 14X14 and proceeds for half pixel refinement.

The table below specifies the maximum number of computations at each level for the classic HS, HS with SDSP and HS with SDSP and SSSP.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Level 0</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Half Pixel level</th>
<th>Total Computations</th>
</tr>
</thead>
<tbody>
<tr>
<td>HS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>596</td>
</tr>
<tr>
<td>HS+SD</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>416</td>
</tr>
<tr>
<td>HS+SD+SS</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>260</td>
</tr>
</tbody>
</table>

The table infers that the computation complexity in HS+SDSP has been reduced by 30% when compared to the HS. For HS+SDSP+SSSP the computations are reduced by 56% when compared to HS.

VI. SIMULATION RESULTS

The experiments were conducted with luminous components of two standard video sequences. The video sequences include “Foreman” (352X288, 30 frames/second (fps)), “Soccer” (352X288, 30fps) and “Mobile” (352X288, 30fps).

A. Results for Speed Comparisons

Generation of results considered the quality of the video at different quality factors (qf). The qf is considered as the measure of loss in the compression process. The higher the Q factor lower is the video quality. The figure below shows a video frame at different qfs.

![Figure 3: A frame at different values of qf](image)

The tables below show the speed comparisons of the proposed algorithms. The readings are with respect to HS algorithm speed at qf =4 for three video sequences.
TABLE II SPEED COMPARISON FOR “FOREMAN” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.0</td>
<td>0.9231</td>
<td>0.9208</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.9460</td>
<td>0.8626</td>
<td>0.8402</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.9217</td>
<td>0.8186</td>
<td>0.8089</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>0.9261</td>
<td>0.8154</td>
<td>0.8037</td>
<td></td>
</tr>
</tbody>
</table>

TABLE III SPEED COMPARISON FOR “SOCCER” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.0</td>
<td>0.9656</td>
<td>0.9544</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.9375</td>
<td>0.8918</td>
<td>0.8730</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.9218</td>
<td>0.8544</td>
<td>0.8472</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>0.9283</td>
<td>0.8367</td>
<td>0.8182</td>
<td></td>
</tr>
</tbody>
</table>

TABLE IV SPEED COMPARISON FOR “MOBILE” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1.0</td>
<td>0.9107</td>
<td>0.9072</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.8626</td>
<td>0.7853</td>
<td>0.7801</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.8036</td>
<td>0.7237</td>
<td>0.7083</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>0.7786</td>
<td>0.6920</td>
<td>0.6854</td>
<td></td>
</tr>
</tbody>
</table>

The “Foreman” sequence has relatively less motion activity but includes switching of frames. The “Soccer” sequence has high motion activity with consistent motion along the frames. The “Mobile” sequence has complex relative movements in the background. All the video sequences used for simulations were of 300 frames.

The crude measurements were taken in mill-seconds (ms). The total encoding and decoding times for “Foreman”, “Soccer” and “Mobile” sequences were 2, 1216ms, 22,355ms and 24,714ms respectively.

B. Results for video quality comparisons

The PSNR value has been used as a metric to analyze the quality of the video. The PSNR is measured on logarithmic scale and depends on Mean Squared Error(MSE) and the square of the highest possible signal value in the image which is $(2^n-1)^2$, where n being the number of bits per image sample. Thus PSNR can be calculated as,

$$PSNR_{db} = 10 \log_{10} \frac{(2^n-1)^2}{MSE}$$

Where MSE for a NXN block sized compensation block can be calculated as,

$$MSE = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (C_{ij} - R_{ij})^2$$

$C_{ij}$ and $R_{ij}$ being the current and reference samples respectively[2]. Similar to speed comparisons, the PSNR readings were also taken at different values of qf. Each value in the table is the average PSNR of 300 frames.

TABLE V PSNR COMPARISON FOR “FOREMAN” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>41.3441</td>
<td>43.2403</td>
<td>43.2499</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>37.5444</td>
<td>36.0435</td>
<td>36.0605</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>33.3650</td>
<td>32.8412</td>
<td>32.8218</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>14.1571</td>
<td>14.1675</td>
<td>14.1641</td>
<td></td>
</tr>
</tbody>
</table>

TABLE VI PSNR COMPARISON FOR “SOCCER” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>40.1436</td>
<td>40.0852</td>
<td>40.1095</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>36.1248</td>
<td>36.0435</td>
<td>36.0535</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>32.9873</td>
<td>32.8412</td>
<td>32.8528</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>17.2025</td>
<td>17.2361</td>
<td>17.2351</td>
<td></td>
</tr>
</tbody>
</table>

TABLE VII PSNR COMPARISON FOR “MOBILE” SEQUENCE

<table>
<thead>
<tr>
<th>Quality Factor (qf)</th>
<th>BMA</th>
<th>HS</th>
<th>HS+SD</th>
<th>HS+SD+SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>38.3783</td>
<td>38.3828</td>
<td>38.3841</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>33.2760</td>
<td>32.2843</td>
<td>33.2842</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>28.7160</td>
<td>28.7281</td>
<td>28.7276</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>13.5130</td>
<td>13.5115</td>
<td>13.5148</td>
<td></td>
</tr>
</tbody>
</table>

VII. ANALYSIS OF THE RESULTS

A. Discussion on speed comparisons

Simulation results show that proposed algorithms are considerably fast for all the three test video sequences. The “Foreman” series is the fastest as it has less movement followed by “Soccer” series. The “Mobile” sequence is relatively slow due to the nature of the video with high relative motions and movements in the background. The process of the compression gets faster with the increasing values of qf in all the test videos as the low resolution of the video primarily encourages fast compression.

B. Discussion on PSNR comparisions

It must be noted that when the fast searches were introduced in HS, the search window size was increased at respective levels. This potentially increases the possibility of finding a better match when the video has relative motion or switching frames. This might explain the scenario of HS+SD+SS algorithm having the highest PSNR for “Foreman” and “Mobile” sequences. The “Soccer” sequence which has less relative motion behaves expectedly with the PSNRs of the proposed algorithms being lower than HS.

C. Discussion on speed and PSNR

The simulation results make it evident that speed is directly proportional to qf whereas, PSNR is inversely proportional for any video. Generally, PSNR is inversely proportional to the speed [2], [3]. However, results show very slight increase in PSNR for HS+DS+SS algorithm for “Foreman” and “Mobile” sequences. The possible reason for this behaviour can be increased search window size at level0 and level1 which might increase the PSNR but at the same time quicker than HS and HS+SD for searching lesser points.

VIII. CONCLUSION

This paper proposes improvised versions of the HS algorithm by incorporating fast search patterns into its hierarchical levels. Results have shown that by reducing the number of computations at each level, the HS can be implemented faster. It has been proved that HS+SDSP+SSSP are the fastest as it reduces the computation at two levels. The experiments show that HS+SDSP and HS+SDSP+SSSP are faster than HS with no significant degradation in video quality. The proposed algorithms were proved to be more productive for videos.
with sudden frame changes or with high background movements. Hence the proposed algorithms can be used for real-time videos to perform faster motion estimation process thus speeding up the video compression process.

REFERENCES


http://www.di.unipi.it/~lonetti/PhDCourse/ME/CrossSearch.pdf [online resource]

http://www.contrib.andrew.cmu.edu/~nrp3.pdf [online resource]

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=821744 [online resource]

http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=821744 [online resource]

BIOGRAPHY

NagaRohini Koduri obtained her B. Tech (Electronics and Communication Eng.) Degree from Periyar University, India in 2004. She is currently in her second year of her MSc (Elec Eng) degree from University of Cape Town. Her research is sponsored by the Council for Scientific and Industrial Research (CSIR).