American Journal of Engineering Research (AJER)
e-ISSN: 2320-0847 p-ISSN : 2320-0936
Volume-7, Issue-4, pp-68-77
www.ajer.org
Research Paper Open Access

Fast Motion and Mode Selection for Low Complexity Devices

[1]kamalu U.A., [2]deinbo-Briggs O.M.
Department of Electrical/Electronic Engineering, University of Port Harcourt.
Department of Electrical/Electronic Engineering, University of Port Harcourt
Corresponding author: kamalu U.A

ABSTRACT: *The motion estimation and compensation stage in video compression is a very computationally complex process, which can be really frustrating in low complexity devices and real – time applications such as video conferencing. The search for the best match in a video sequence can be obtained with a computational complexity of up to 17% of that of the full search by carrying out a combined search in both the motion and the mode domain, with little or no difference in the Peak Signal to Noise Ratio (PSNR) of the output sequence. The Three – Step Search and a fast mode algorithm based on the movement and detail characteristics of the video sequence are used in this research, resulting in getting the best match in the video sequence.*
Keywords: *Advanced Video Coding (AVC) ; H.264; Macroblock – 16 x 16 block of pixels; Peak Signal to Noise Ratio (PSNR); Sum of Absolute Errors (SAE); Three - Step Search (TSS) ; Mean Square Error (MSE)*

Date of Submission: 28-03-2018 Date of acceptance: 12-04-2018

I. INTRODUCTION

The compression of video sequences has been standardized by the ITU-T/ISO/IEC Joint Video Team and called the H.264/AVC standard [1]. The H.264/AVC has the advantage of producing videos of a lot better quality (higher PSNR) when coded at the same bitrates as its predecessors and lower bitrates with the same quality (the same PSNR) [2]. The processes involved in the standards available for the coding of a video sequence such as H.264/ AVC are usually time consuming, the process of motion estimation and compensation being the most time consuming of them all. A motion compensated frame usually has a smaller residual [3] than an uncompensated frame obtained from frame differencing. This work serves to investigate into the possibility of combining and optimizing the search in both the motion and mode domains to obtain better performance.

One previous investigation done in combining and optimizing the search in both domains has done so by implementing the Uneven Multi-Hexagon Search (UMHexagonS) as the fast motion search algorithm of choice in the H.264/AVC encoder reference software (JM 84) [4] and combining it with a developed fast mode search algorithm. This reduced the processing time of the motion estimation significantly without considering the use of multiple reference frames.

Another implemented the Predict Hexagon Search (PHS) and Edge Information Mode Decision (EIMD) [5]. This also showed a significant speed improvement of about 2-15 times that of a popular fast search algorithm.

This work implements the Three-Step Search (TSS) [6] algorithm as the fast search algorithm of choice because of its simplicity in terms of being easy to understand and implement and the quality of the output, which is relatively good and close to the quality of the full search. An advantage of the TSS is that the step size of the search never goes beyond the edge of the search window, which could happen in some other fast motion search algorithms. An illustration of the TSS is shown in figure 1.

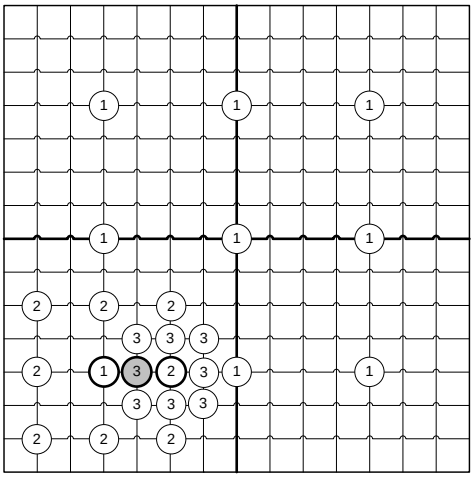


Figure 1: An Illustration of the Three – Step Search (TSS) [2]

The search window size for the TSS is +/- (2^3 – 1), which is +/- 7 and the step size S is 2^{3-1}, which is 4. In each step, eight locations +/- S pixels around and including the current best position are searched for a new best match. After each step, the value of S is divided by 2 for the next step until S = 1, which marks the last step. The position of the macroblock in the current frame is the origin with vector (0, 0).

The best match after the third step is the overall best match and is shown as the darkened circle 3 in the figure. This may or may not necessarily be the actual best match obtained from a full search and so the SAE may be higher than that of the full search but takes less time as only 27 blocks out of a total of 225 blocks are searched. This has a computational complexity of about 12% of the full search with a tradeoff of a larger compressed bit rate and slightly poorer picture quality.

Using MATLAB, the TSS was implemented and a fast mode search algorithm was developed and implemented as well. Then a combination of both the fast mode and the TSS was implemented. The PSNR, SAE and bitrate of each algorithm was compared to show the advantage of the combined search. The equation for the PSNR is shown in equation 1 [6, 7].

$$PSNR_{dB} = 10\log_{10}\left(\frac{\left(2^n - 1\right)^2}{MSE}\right) \qquad (1)$$

Where, n = number of bits representing a pixel (i.e. 8 bits); and from [8],

$$MSE = \frac{\sum_i \sum_j \left(Y_{ref}(i,j) - Y_{prc}(i,j)\right)^2}{N} \qquad (2)$$

The seven allowable block sizes in the H.264 standard [1, 6, 7] as shown in figure 2 are the 16 x 16, 16 x 8, 8 x 16, 8 x 8, where the 8 x 8 block size is chosen, it can further be divided into the 8 x 8, 8 x 4, 4 x 8 and 4 x 4 sub-block modes. These blocks can be used to get a lower residual while carrying out the motion estimation and compensation where there is a lot of detail and movement within the macroblock.

The search for the best match for all seven modes takes a lot of computational time and as such there is the need to reduce this time by implementing a fast mode search algorithm where only some specific modes would be searched for each macroblock, thereby reducing the computational time with little or no loss in the quality.

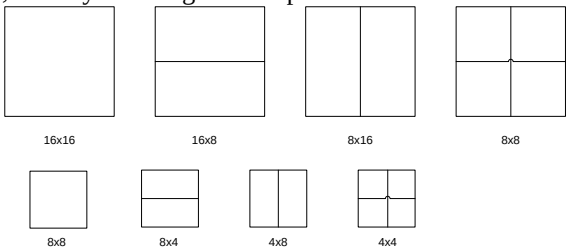


Figure 2: The Allowed Block Sizes in H.264/AVC [2, 6]

However, with a reasonably reduced computational complexity, there is a corresponding decrease in the quality (in terms of PSNR) and an increase in the compressed bit rate of the video sequence. Sometimes, a fast search algorithm can produce little or no compression of the video sequence. The resulting bit rate can even increase in some cases. Thus the Lagrangian Cost function [6] is used to determine if the result of the full or fast search is actually the best residual and motion vector and would result in a compression of the video sequence. From [2], the function is given as;

$J = D + \lambda R$ (3)

Where D is the distortion of the coded sequence
R is its corresponding bit rate, and
λ is a Lagrange multiplier.

The block that gives the smallest value of J in the search operation is taken as the best match and usually gives a compressed video sequence.

II. METHODOLOGY

The full search for the best match is an exhaustive search within the search window as long as the pixels in the search window around the current macroblock remain within the frame. The search is carried out for each mode and the best match is selected as the mode with the least combined residual and motion vector bit count. The Lagrange formula given in equation 3, where the value of the Lagrange multiplier is as given in equation 4, is used in the determination of the best match. The formula used for the Lagrange multiplier is as follows: -

$\lambda_{motion} = \sqrt{(\lambda_{mode})}$ (4)

Where $\lambda_{mode} = a\,(2^{(QP-12)/3})$ and a = 0.68, QP is an arbitrary value chosen.

The flow chart for the full search for the best match is shown in figure 3. The search is carried out in raster scan order within the search window.

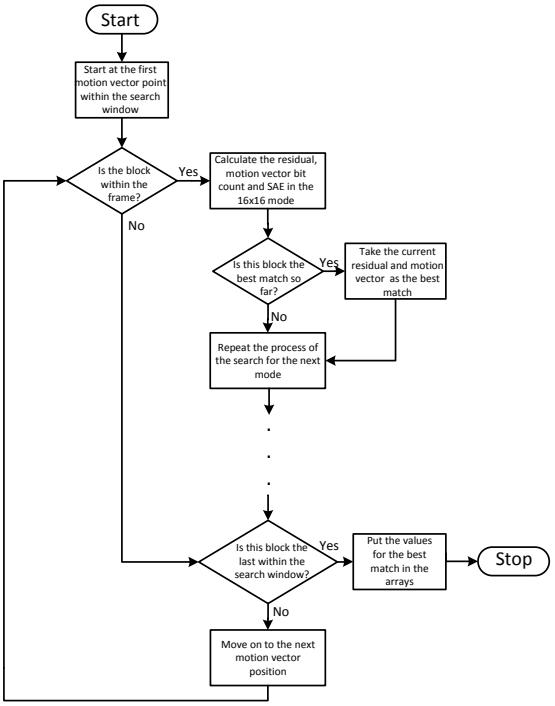


Figure 3: The Flow Chart for Finding the Best Match in all 4 Modes Using the Full Search

At the end of the search the total bit count and total SAE of the residual, the PSNR of the reconstructed frame and the computational time used for carrying out the whole process are calculated. The motion estimation and compensation process reduces the SAE and makes it considerably less than that of the uncompensated residual frame obtained by simple frame differencing.

The bit counts for both the motion vectors and the residuals are calculated after they have been coded using the Exp-Golomb code. The code is a variable length code, which assigns binary codewords to values from a look up table as shown in table 1.

Table 1: The Exp - Golomb Table of Codewords

Code Number	Signed Number	Codeword	Bit Count
0	0	0	1
1	1	010	3
2	-1	011	3
3	2	00100	5
4	-2	00101	5
5	3	00110	5
6	-3	00111	5

7	4	0001000	7
8	-4	0001001	7
…	…	…	…

The flow chart for the fast motion search algorithm is shown in figure 4. The best match is found for each mode one at a time and compared to the current best match and the mode which has the least cost is taken as the best match. At the end of the search for all modes, the best match is obtained and the mode chosen. The mode chosen for the fast search may or may not be the same as that obtained from the full search and so would not give as much compression as the full search and consequently a slightly lower quality.

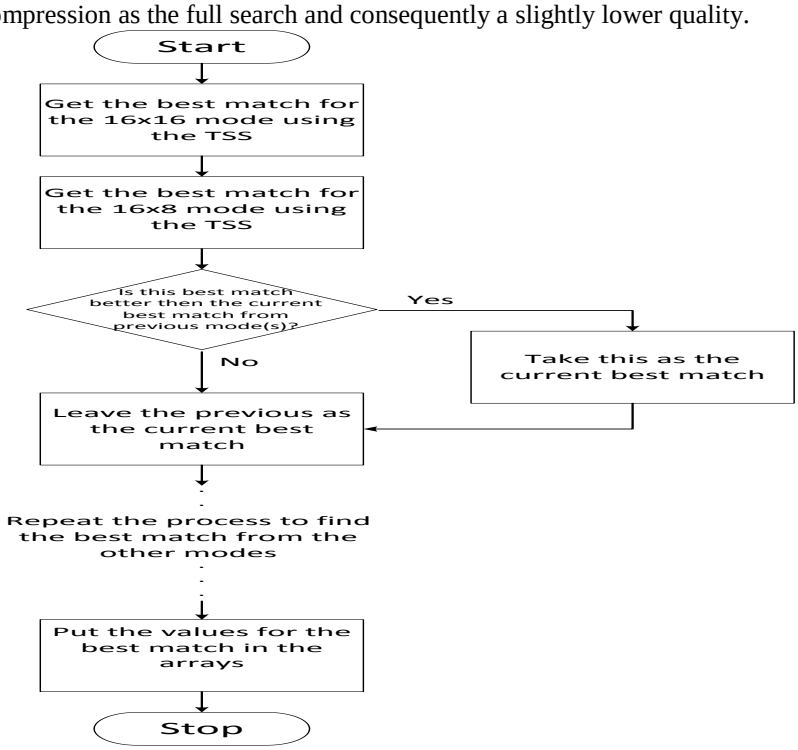
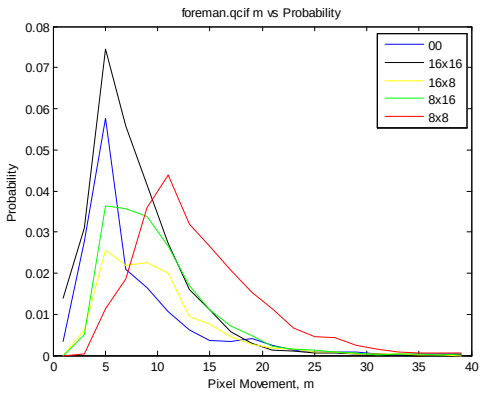
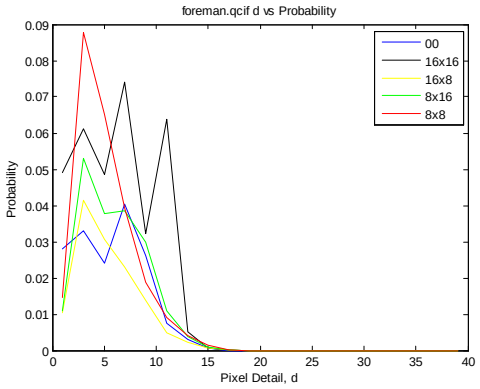


Figure 4: The Flow Chart for Finding the Best Match in all 4 Modes Using the Fast Motion Search

For the development of the fast mode algorithm, the movement and detail within each video sequence was the determining factor for what particular modes to search with to obtain the best match. The detail characteristics makes choice between the 16x16 mode for low detail and the 8x8 mode for high detail while the movement characteristics remains low. With a combination of low movement and low detail, the 16x16 mode chosen has a high probability of having motion vector combinations of (0, 0).The movement characteristics makes choice between the 16x16 mode for low movement and the 8x8 mode for high movement while the detail characteristics remains low. Figure 5 shows the choice of mode based on the movement and detail characteristics for the full search on foreman.



(a)



(b)

Figure 5: The Detail and Movement Characteristics for the Full Search

The foreman sequence was chosen out of the many available because it contains the most detail and movement. The foreman sequence has a panning movement as well as the movement of the man in the picture and there is sufficient detail as well. The thresholds were set based on these distributions. These distributions can generally be represented as shown in Figure 6.

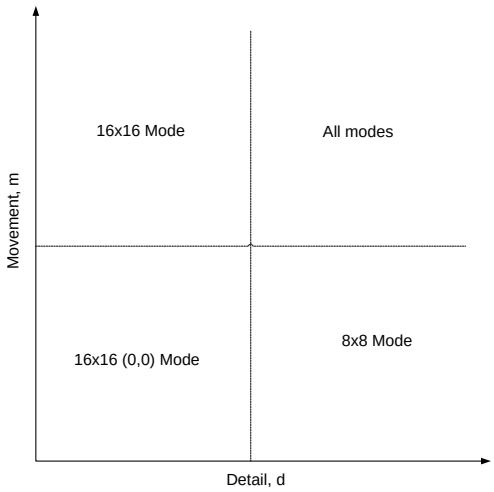


Figure 6: The Choice of Modes for Each Detail and Movement Characteristics Combination

The movement is measured simply by taking the average value of the difference between the current macroblock and the macroblock in its position in the reference frame. This also consists of additions and one division operation. The detail is calculated by the following steps.

- Break the macroblock into four 8x8 blocks.
- For each block, calculate the difference in pixel values in four directions – left to right, right to left, up to down and down to up – and take the average of the four arrays to get one array.
- Calculate the mean of the values within each of the four arrays. The order of the values is shown in figure 7.

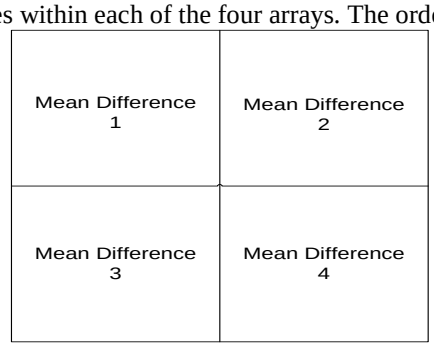


Figure 7: The Mean Difference Values within the Macroblock

- The following calculations are then carried out to obtain the detail characteristic value, d: -

Horizontal difference 1, h1 = |mean difference 1 – mean difference 2|
Horizontal difference 2, h2 = |mean difference 3 – mean difference 4|
Vertical difference 1, v1 = |mean difference 1 – mean difference 3|
Vertical difference 2, v2 = |mean difference 2 – mean difference 4|
Detail, d = (h1 + h2 + v1 + v2)/ 4
The calculations are purely additions and only one division as opposed to the square root and many square operations which make the variance more computationally complex than the full search.
With the thresholds set, the flow chart for the fast mode search is shown in Figure 8.

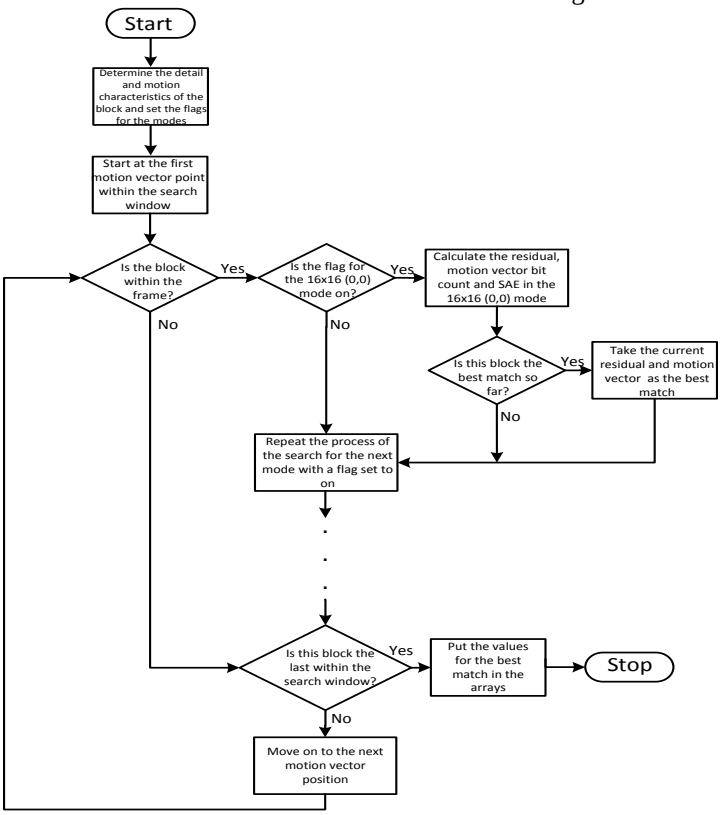


Figure 8: The Flow Chart for Finding the Best Match Using the Fast Mode Search

The combined fast motion and mode search is carried out by using the TSS to search for the best match for only specific block modes in each macroblock based on the thresholds set. These thresholds are based on the movement and detail characteristics for the TSS fast motion search for the best match on the foreman video sequence. With the thresholds set, the flow chart for fast motion and mode selection for the best match is shown in Figure 9.

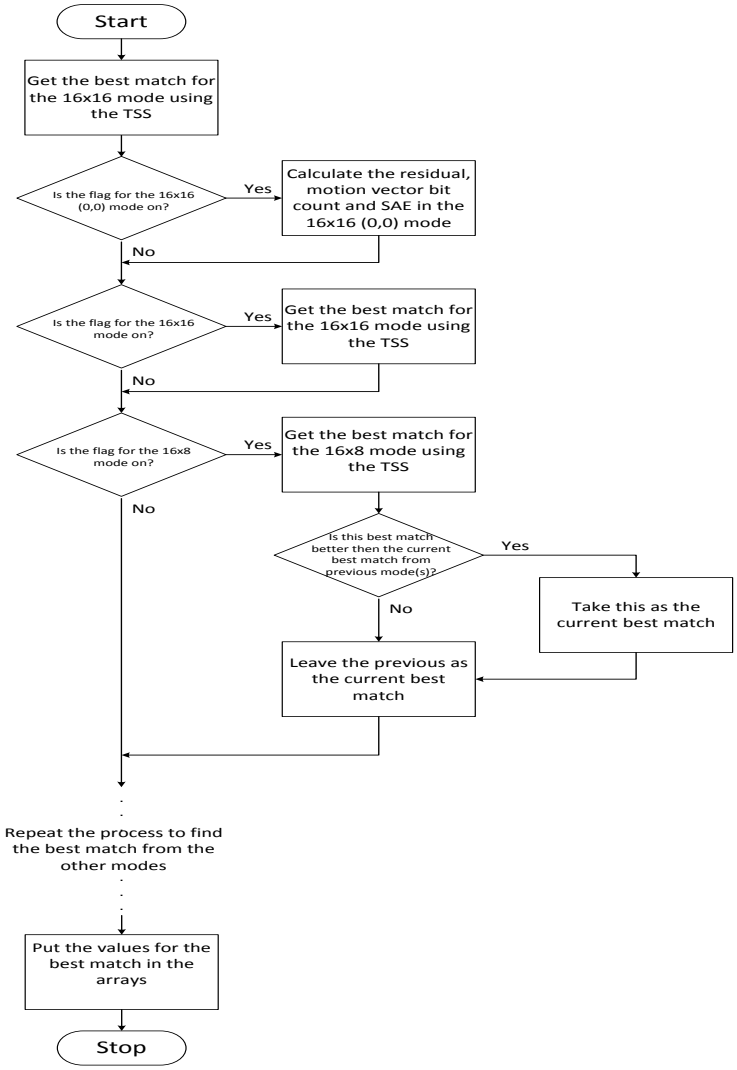


Figure 9: The Flow Chart for Finding the Best Match Using the Fast Motion and Mode Search

RESULTS AND ANALYSIS

It was noted that for a video sequence with a lot of detail and motion, the difference in the quality of the output using the TSS method may be considerably much. For sequences with little to moderate detail and motion, the quality of the output is very close to that of the output of the full search.

The reconstructed output frame from each of the four search algorithms are shown in Figure 10.

Figure 10: The Outputs of the Algorithms

The qualities of the outputs from the fast search algorithms are not so much different from that of the full search. It can be seen that with reduced computational complexities, the output shows little or no difference in PSNR, which is good. The rate – distortion performances for four different video sequences are shown in Figure 11.

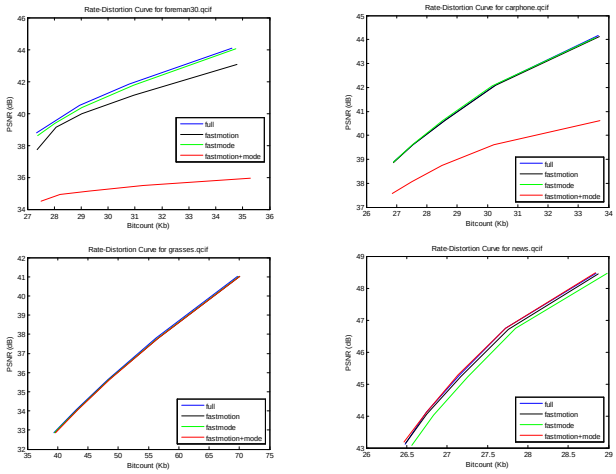


Figure 11: The Rate - Distortion Performances

The performance of the fast motion and mode search algorithm performs very well on the grasses and news sequence, which are sequences with high and low movement respectively. The poor performance on the foreman30 and carphone sequences is due the non-distinct distribution of the different modes as shown in figure 12.

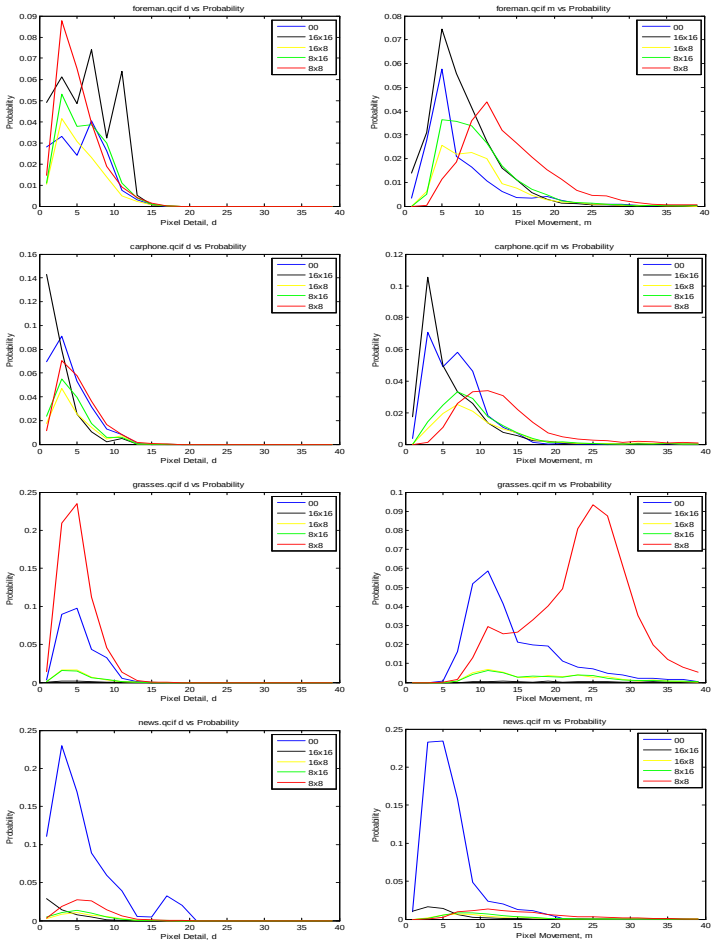


Figure 12: The Movement and Detail Characteristics of all Four Video Sequences

The time saved by the different algorithms varies and has a trade off of lower PSNR values and higher bit count values of the coded video sequence. The PSNR, SAE, bit count and computational time savings of the algorithms for frames 5 and 6 of the foreman30, carphone, grasses and news video sequences with a QP value of 24 are shown on tables 2 – 5.

Table 2: Results for foreman30.qcif (Uncompensated SAE = 98,785)

Algorithm	PSNR (dB)	ΔSAE (%)	Bit Count (Kb)	Complexity (%)
Full Search	38.797	-41.41	27.346	100
Fast Motion Search	-1.073	-39.28	+0.014	-78
Fast Mode Search	-0.179	-35.11	+0.034	-36
Fast Motion + Mode Search	-4.278	-31.25	+0.162	-82

Table 3: Results for carphone.qcif (Uncompensated SAE = 54,572)

Algorithm	PSNR (dB)	ΔSAE (%)	Bit Count (Kb)	Complexity (%)
Full Search	38.898	-9.92	26.920	100
Fast Motion Search	-0.032	-8.58	-0.034	-78
Fast Mode Search	-0.017	-6.51	-0.032	-34
Fast Motion + Mode Search	-1.335	-6.73	-0.062	-81

Table 4: Results for grasses.qcif (Uncompensated SAE = 232,695)

Algorithm	PSNR (dB)	ΔSAE (%)	Bit Count (Kb)	Complexity (%)
Full Search	32.8611	-9.06	39.356	100
Fast Motion Search	-0.0006	-7.97	+0.194	-78
Fast Mode Search	+0.0004	-8.53	+0.136	-47
Fast Motion + Mode Search	-0.0017	-5.45	+0.284	-83

Table 5: Results for news.qcif (Uncompensated SAE = 28,892)

Algorithm	PSNR (dB)	ΔSAE (%)	Bit Count (Kb)	Complexity (%)
Full Search	43.1144	-11.97	26.480	100
Fast Motion Search	+0.0266	-9.42	+0.004	-78
Fast Mode Search	-0.0353	-5.34	+0.084	-12
Fast Motion + Mode Search	+0.0689	-6.44	+0.006	-79

From the tables, it can be seen that the fast motion and mode search saves up to approximately 80% of computational time with very little changes in the PSNR and compressed rate of the grasses and news video sequences.

A comparison of the performance of the fast motion and mode search algorithm with the algorithms developed in [9, 10] is shown on table 6. The algorithm in [9] is represented as AC while that in [10] is represented by CLJ.

Table 6: The Comparison of the Proposed Algorithm with Other Algorithms

Video Sequence	Foreman			News		
Algorithm	ΔPSNR (dB)	ΔBit Rate (%)	Time Saving (%)	ΔPSNR (dB)	ΔBit Rate (%)	Time Saving (%)
AC	-0.0046	1.20	83.88	-0.0062	0.67	91.31
CLJ	-0.09	1.72	55.77	-0.02	0.31	70.33
Proposed	-4.278	0.59	82	+0.0689	0.02	79

The performance of the proposed algorithm when compared with the other algorithm is better in terms of the change in bit rate. It is also better in terms of the change in PSNR in the news sequence. In terms of the change in PSNR in the foreman sequence, the proposed algorithm has a fair performance. The time saved is better than that of CLJ but not as good as that of AC.

III. CONCLUSION

The coding of a video can be done using the combined fast motion and mode search with a computational complexity as low as 17% of the full search algorithm with a very little consequence to the quality performance of the compressed video sequence. The sequences with a clear distinction between the 8x8 and 16x16 modes, based on the detail and movement characteristics used, had a very good performance but those without a clear distinction did not have good enough performances.

REFERENCES

[1]. ITU-T. Recommendation H.264: Advanced Video Coding for Generic Audiovisual services. November 2007.
[2]. Obhuo O.M. Fast Motion and Mode Selection. [Unpublished Thesis]. Aberdeen: Robert Gordon University; 2008.
[3]. Kim H., Kamaci N., Altunbasak Y. Low-Complexity Rate-Distortion Optimal Macroblock Mode Selection and Motion Estimation for MPEG-Like Video Coders. IEEE Transactions on Circuits and System for Video Technology. 2005; 15(7). pp. 823-834.
[4]. Choi B.D. et al. Fast Motion Estimation and Intermode Selection for H.264. EURASIP Journal on Applied Signal Processing. 2006; pp. 122 – 129.
[5]. Pan Y.N., Tsai T.H. Fast Motion Estimatiom and Edge Information Inter-Mode Decision on H.264 Video Coding. San Antonio: IEEE International Conference in Image Processing (ICIP). 2007; pp. II 473 – II 476.
[6]. Richardson I.E.G. Video Codec Design: Developing Image and Video Compression Systems. Chichester: John Wiley; 2002.
[7]. Richardson I.E.G. H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia. Chichester: John Wiley; 2003.
[8]. Ghanbari M. Standard Codecs: Image Compression to Advanced Video Coding, 49. London: IEE; 2003.
[9]. Al Qaralleh E.A., Chang T.-S. Fast Variable Block Size Motion Estimation by Adaptive Early Termination. IEEE Transactions on Circuits and Systems for Video Technology. 2006; 16(8). pp. 1021 – 1026.
[10]. Choi I., Lee J., Jeon B. Fast Coding Mode Selection with Rate – Distortion Optimization for MPEG – 4 Part 10 AVC/ H.264. IEEE Transactions on Circuits and Systems for Video Technology. 2006; 16(12). pp. 1557 – 1561.

kamalu U.A." Fast Motion and Mode Selection for Low Complexity Devices”American Journal of Engineering Research (AJER), vol. 7, no. 4, 2018, pp.68-77.