ISSN 2250-0987
Faizul Hadi Jamil et al, UNIASCIT, Vol 2 (3), 2012, 286-291

PERFORMANCE OF A VARIABLE MB AND SEARCH RANGE SIZE IN ADAPTIVE ROOD PATTERN SEARCH (ARPS)

Faizul Hadi Jamil[1], Ali Chekima[2], Rosalyn R. Porle[3]

School of Engineering and Information Technology, Universiti Malaysia Sabah (UMS)
Kota Kinabalu, Malaysia

***Abstract*— The BMA or Block Matching Algorithm has played an important role in video coding technique. One of the famous BMA techniques is ARPS (Adaptive Rood Pattern Search) because it proves that it can achieve low computational complexity and higher quality compared to the other BMA techniques such as Exhaustive Search (ES), Three Step Search (TSS), Simple and Efficient Three Step Search (SETSS), 4-Step Search (4SS) and Diamond Search (DS). Further improvement in ARPS technique is created depending on type of motion whether it is fast, medium or slow motion. This Hybrid Model will determine the video motion type and it will choose the macroblock (MB) and search range (*p*) size depending on video motion type. Slow motion video contains more redundant data among its frame compared to fast motion video. By using this fact, it is important to use a hybrid program that run based on motion type. By using the proposed hybrid model, it can gain an acceptable video quality and lower computational complexity at the same time.**

***Keywords*— Block Matching Algorithm (BMA), Adaptive Rood Pattern Search (ARPS), video coding, motion vector, motion estimation.**

I. INTRODUCTION

Over the last two decades, various types of Block Matching Algorithm or BMA have been proposed to achieve one or more of the three objectives that are to reduce the computational complexity, to gain higher video quality, and also to gain higher compression ratio [1]. One of the well known BMA techniques is called Exhaustive Search (ES) where it exhaustively search all possible blocks over the determined search window to find the best matched macroblock (MB). It is the most straight forward technique and it can generate the most accurate motion compared to other BMA techniques [2]. However, this technique has too high computational cost since it exhaustively search all of the possible MB. Too high computational cost require more time to complete the encode/decode process and it is not practical for the 'power-limited' device. Several improved algorithm were proposed to improve the lack of the Exhaustive Search technique regarding to its high computational complexity. Some of the famous BMA techniques are Three Step Search (TSS), Simple and Efficient Three Step Search [3], New Three Step Search [4], Four Step Search [5], Diamond Search [6] and Adaptive Rood Pattern Search [7].

The BMA will produce a motion vector (MV) that represents the direction of the current MB for the next frame. In term of compression, this technique is very efficient because it stores the direction of MV in *x* and *y* direction of a MB instead of storing 16x16 pixels. The MV accuracy basically depends on its MB size, its search range *p* value and also the BMA type. Theoretically, the lower MB size and bigger *p* value produces accurate motion vector value. However, it yields to higher computational complexity during the motion estimation process [8].

II. BMA ANALYSIS

To choose the best BMA technique, an analysis has been made among the seven famous BMA techniques that are the Exhaustive Search (ES), Three Step Search (TSS), Simple and Efficient Three Step Search [3], New Three Step Search [4], Four Step Search [5], Diamond Search [6] and Adaptive Rood Pattern Search [7]. Three raw video with QCIF format (176x144 pixels) with different movement motion types is used in the experimentation. "football_qcif.yuv" "coastguard_qcif.yuv" and "akiyo_qcif.yuv" are used for high motion video, medium motion video and low motion video respectively. Only the Y component (luma) is required to be encoded and all of the video samples were encoded frame by frame with a distance of two frames between the current frame and reference frame.

The BMA performance can be varied based on the performance of the PSNR (Peak-Signal-to-Noise-Ratio) comparison and also its computational complexity [10]. The main function of PSNR is to measure the quality between the generated video and the original video. High PSNR value means high picture quality and vice versa. For the computational complexity, the longer computation time requires for the searching of the best matched block means that it has high computational complexity. A good BMA technique should have high PSNR value and low computation complexity. The formula of PSNR computation is given in equation (1) and (2) where *N* is the MB size, C_{ij} is the current block, R_{ij} is the reference MB and *n* is the number of bits per image sample.

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

$$PSNR = 10\log_{10}\left[\frac{(2^n - 1)^2}{MSE}\right] \qquad (2)$$

Figures 1 to 6 show the PSNR performance and the computation complexity of search points per MB for the encoded video with different BMA techniques. In these Figures, the MB size used is 16×16 and search range p is 7. From the experimental result, all of the encoded video using the Exhaustive-search (ES) have slightly higher or equal PSNR value with the other six BMA techniques as shown in Figure 1 (football video), Figure 2 (coastguard video) and Figure 3 (akiyo video). For the Four-step-search (4SS), Diamond-search (DS) and Adaptive-rood-pattern-search (ARPS), its PSNR performances are very close to the PSNR of ES especially for the high motion video (football) and normal motion video (coastguard). For the low motion video (akiyo), all of the seven BMA techniques give almost the same value for its PSNR performance as illustrated in Figure 3.

Although ES PSNR performance is slightly higher than the other BMA techniques, its computational complexity is the highest compared to the other BMA techniques. It takes on an average around 184 searches per MB for all types of motion video. The other six BMA techniques take less than 30 searches per MB as illustrated in Figure 4, 5, and 6. It shows that the higher video motion will take longer computation time compared to the low video motion. Among the six BMA techniques, the ARPS performance on computational complexity is the best for this experiment since it take the lowest number of search point per MB and its PSNR value is high and very close to the PSNR of ES. Hence, the ARPS technique has been chosen for further analysis in video coding technique by applying different MB size and p value.

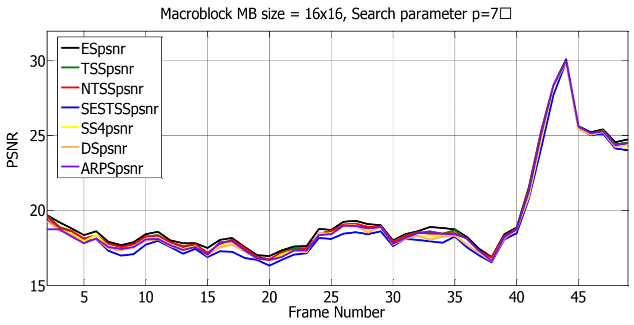

Fig. 1 PSNR performance of the encoded 'football_qcif.yuv' video with different BMA techniques.

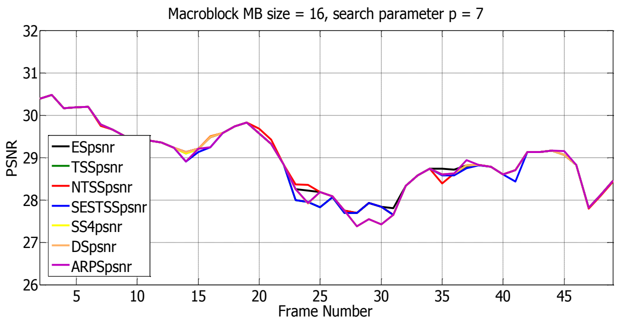

Fig. 2 PSNR performance of the encoded 'coastguard_qcif.yuv' video with different BMA techniques.

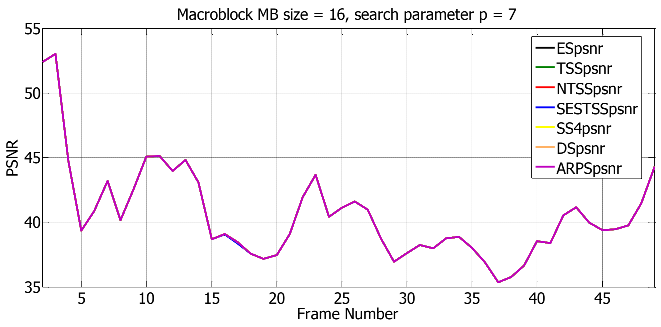

Fig. 3 PSNR performance of the encoded 'akiyo_qcif.yuv' video with different BMA techniques.

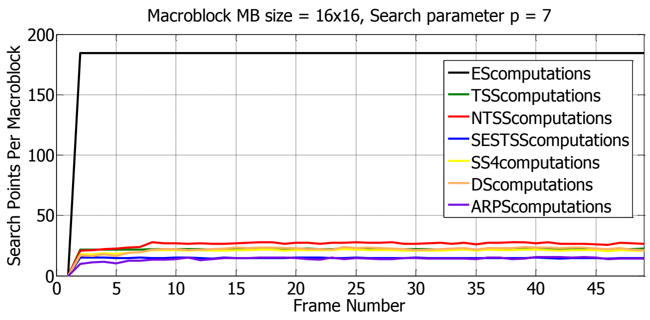

Fig. 4 Number of search points per MB for the encoded 'football_qcif.yuv' video with different BMA techniques.

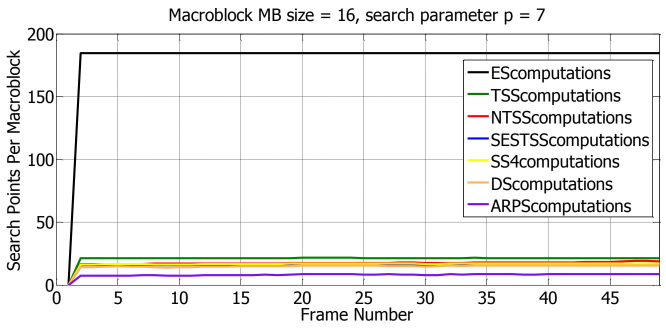

Fig. 5 Number of search points per MB for the encoded 'coastguard_qcif.yuv' video with different BMA techniques.

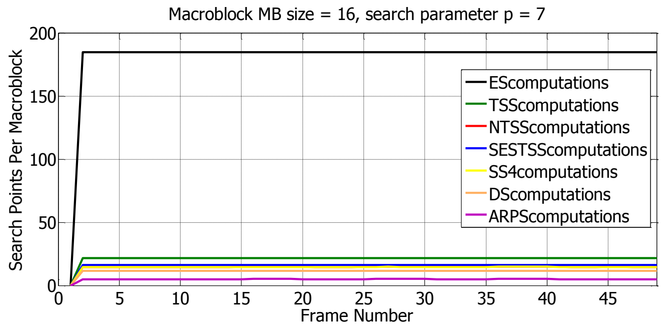

Fig. 6 Number of search points per MB for the encoded 'akiyo_qcif.yuv' video with different BMA techniques.

A. The ARPS Performance

In term of combination of computational complexity and PSNR, it proves that the ARPS technique gives the best result among the other six BMA techniques. Theoretically, smaller MB size or bigger p value will produce higher PSNR performance vice versa. By using ARPS technique, this experiment examined the low, medium and high motion video by applying different MB size that are 4×4, 8×8 and 16×16 and also different search parameter p that are 5, 7, 9, 11, 13, and 15. The results are illustrated in Figures 7 to 12 and it proves the theoretical statement. For the high motion video, it seems that its PSNR value and its number of computations complexity change dramatically with the changes of MB and *p* value as shown in Figures 7 and 8. For the medium and slow motion videos, the PSNR value and its number of computations complexity also change dramatically with the changes of MB and very slightly change (or no changes) with the changes of *p* value as illustrated in Figures 9 to 12 for coastguard and akiyo video. It also can be concluded that to gain high video quality for high motion video, the encoder is needed to use smaller MB size and bigger *p* value while for the medium and slow motion video, bigger MB and smaller *p* can be applied to save the computation complexity and hence, faster encoding time.

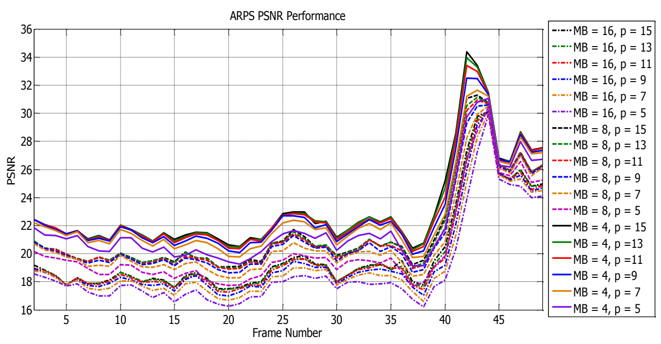

Fig. 7 ARPS PSNR performance of the encoded 'football_qcif.yuv' video with different MB size and different search parameter (*p*).

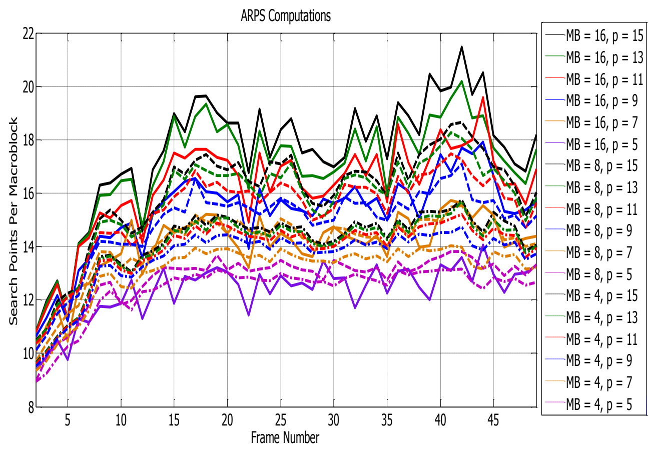

Fig. 8 Number of search points per MB for the encoded 'football_qcif.yuv' video with different MB size and different search parameter (*p*) by using ARPS technique.

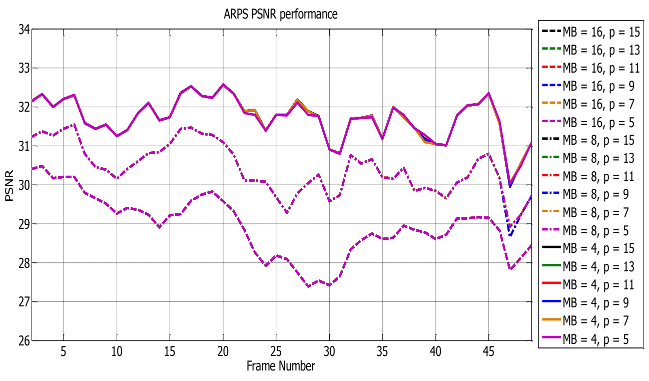

Fig. 9 ARPS PSNR performance of the encoded 'coastguard_qcif.yuv' video with different MB size and different search parameter (*p*).

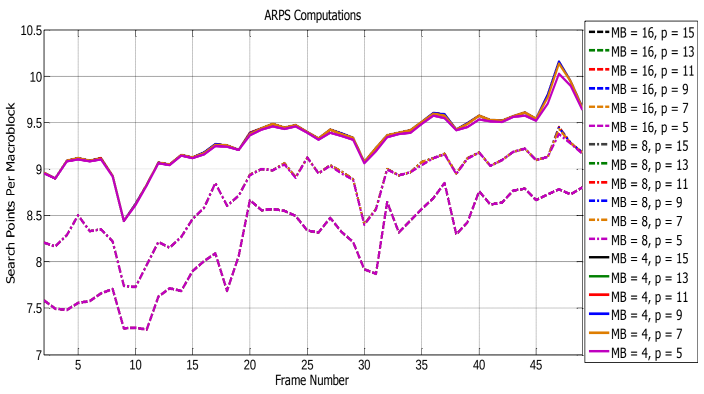

Fig. 10 Number of search points per MB for the encoded 'coastguard_qcif.yuv' video with different MB size and different search parameter (*p*) by using ARPS technique.

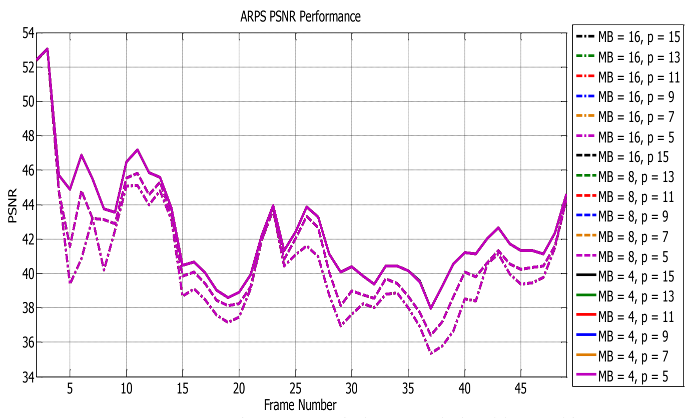

Fig. 11 ARPS PSNR performance of the encoded 'akiyo_qcif.yuv' video with different MB size and different search parameter (*p*).

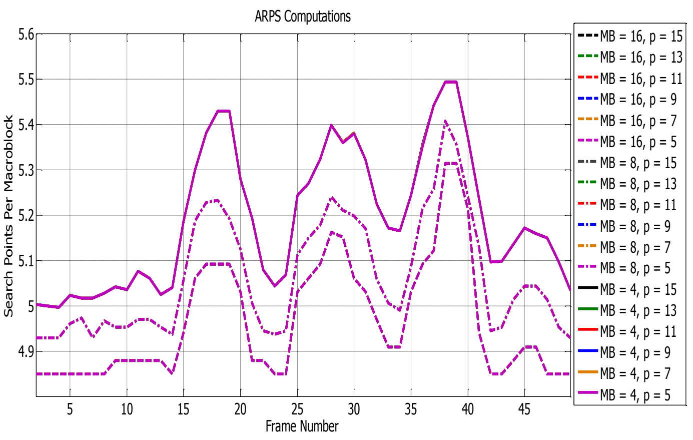

Fig. 12 Number of search points per MB for the encoded 'akiyo_qcif.yuv' video with different MB size and different search parameter (*p*) by using ARPS technique.

III. The Hybrid ARPS

Based on the previous works, it is shown that the performance of high motion video depends on the MB size and *p* value while the low and medium motion are not varying too much with the changes of MB and *p* value. By using this fact, a hybrid program has been developed so that the encoded high motion video can have high video quality and the slow and medium motion encoded video can have an acceptable video quality and reduces its computational complexity. The proposed Hybrid ARPS encoder consists of two main programs that are *I* and *P* encoder. Firstly, this hybrid program encodes the decided *I* frame with the *I* encoder using DCT technique. For *P* frame, it will be encoded using the ARPS technique. In this work, different parameter of *I* and *P* mode sequence have been used, they are *IP*, *IPP*, *IPPP*, *IPPPP* and *IPPPPP* mode. It will encode the *I* frame first, and then the *P* encoder will be used for encoding the *P* frame until it found the *I* mode status.

This hybrid program starts by determining the frame different between the current and previous frame. If the difference exceeded the maximum threshold value, the program will decide that it is a high motion video. If the frame difference value is between the maximum and minimum threshold value, it is a medium motion video and finally, if the frame difference value is lower than the minimum threshold value, it is a slow motion video. For the high motion video, the MB size is chosen to be 4x4 and *p* value is 15. For medium motion video, the chosen MB size is 8x8 with *p* value is 7 and finally, for the low motion video, it will be encoded by using 16x16 as its MB size and 5 for its *p* value. The flowchart of the hybrid video coding program is shown in Figure 13.

A. *I Encoder*

The encoder main function is to encode the raw original video into the compressed form. In this experiment, firstly, the raw original video is divided into a frame sequence and then, each frame will be further divided into several fixed blocks or macro blocks. Next, the system will identify whether the current frame or block has *I* mode or *P* mode as illustrated in Figure 12. The *I* encoder encodes the *I* frame by transforming the original frame using the Discrete Cosines Transform (DCT) technique [9].

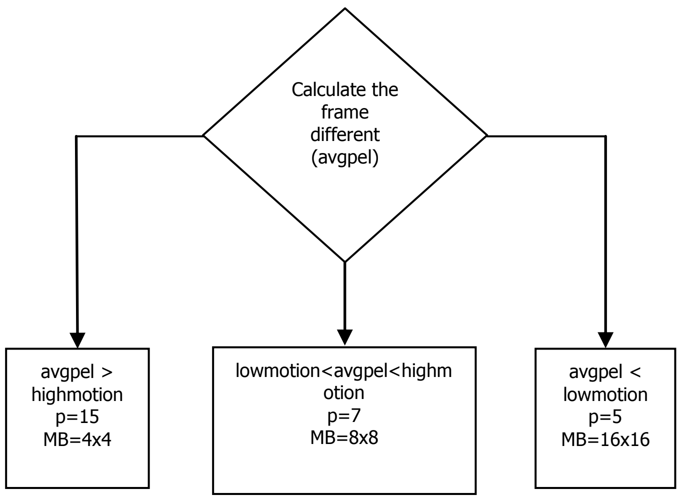
Calculate the frame different (avgpel)
avgpel > highmotion p=15 MB=4x4
lowmotion<avgpel<highmotion p=7 MB=8x8
avgpel < lowmotion p=5 MB=16x16

Fig. 13 The Flow diagram of variable MB and *p* size implementation.

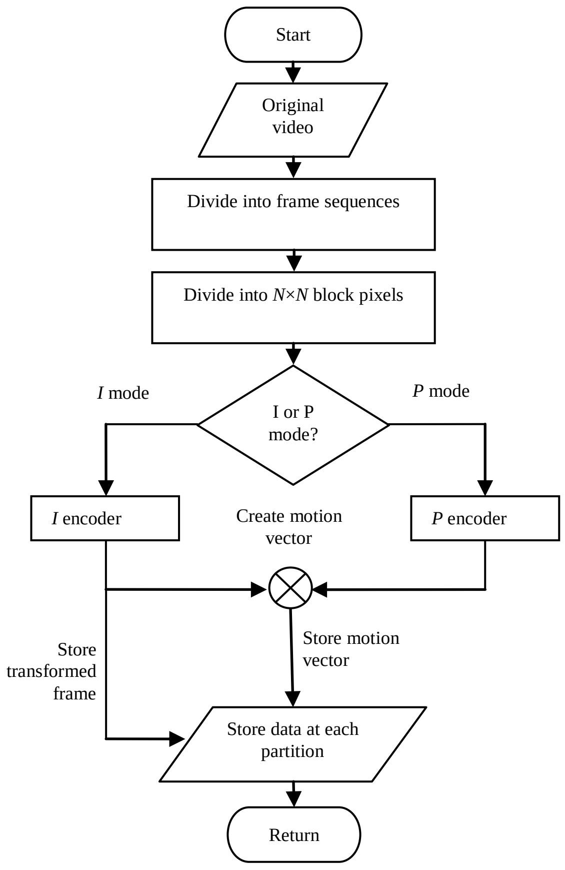

Fig. 14 The Encoder part flowchart

B. *P Encoder*

For *P* frame, it will be encoded using the *P* encoder. This encoder block is based on the BMA which will compare the previous and current block within a frame and creates the motion vector (MV) from it. One MB has one direction with two dimensional function defined as *dx* for vertical direction and *dy* for horizontal direction. The best matched block is defined from the cost function of MAD which is described in equation (3). After the entire motion vectors is produced, its *dx* and *dy* value will be stored into the memory.

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|C_{ij} - R_{ij}\right| \qquad (3)$$

C. *Decoder*

Like encoder, the decoder part in this experiment also has two parts, *I* and *P* decoder. The *I* decoder will inverse the transformed DCT data from the memory. In *P* decoder part, the previous decoded I frame will be merged with the stored motion vector (created from the *P* encoder) to construct a new frame. Then, that reconstructed frame will be used as a reference frame and it will be merged again with the motion vector to create the next frame and so on until the decoder found the *I* mode status. This process is iterated until it reaches the last frame. At last, it will produce a video with the same frame number with the original video, but slightly different data information with the original video.

D. Result and Discussion

From the experimental result, it is shown that for the fast moving motion video like 'football' video in Figure 15, the decoded video contains errors which range from 17 to 30 for its PSNR value if it is encoded using the P encoder. However, it is still a better result since this hybrid program chooses a small MB size and high *p* value. For the medium motion coastguard video, it gives acceptable PSNR values as illustrated in Figure 16 which range from 25 to 36 for its encoded and decoded *P* frames. Finally, for the low motion akiyo video, the hybrid program has chosen bigger MB size (16x16) and low *p* value (7) and it produced a better PSNR result compared to the coastguard and football video. Its PSNR values for the encoded and decoded P frames are ranging from 30 to 52.

In term of *I* and *P* frame sequence, it is shown that the more *P* frame defined in video, the lower PSNR value which bring to lower picture quality. More mismatched motion vector is produced between the *P* frames. The highest peak is achieved when the frame is encoded using the *I* encoder. The *I* encoder main function is to recover the mismatched motion vector produced from the previous frame. Note that the *I* encoder do not depend on any previous frame while the *P* encoder depends with its previous frame. It can be concluded that, if the previous reconstructed frame contains small number of errors, then, there is high possibility that the next reconstructed frame has small number of errors. If the previous reconstructed frame has too many errors, the next reconstructed frame will have more errors and it will become worse for its next reconstructed frame. To reduce the number of errors, some frames are needed to be encoded by using *I* encoder especially for the high motion video. More *P* frames yield to more mismatch motion vector but it can exploit the compression efficiency since it only encode the motion vector of a MB instead of every pixel in a MB.

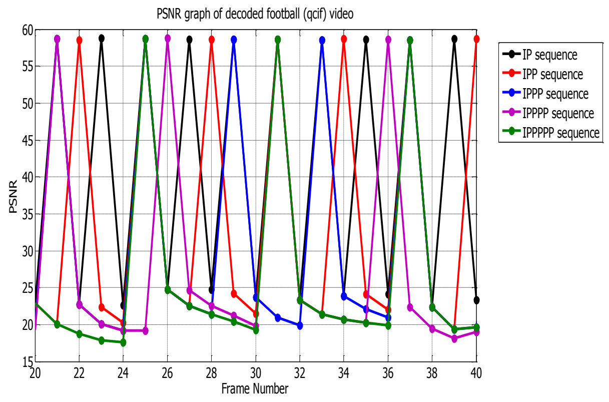

Fig. 15 PSNR performance of football_qcif video with different I and P frame sequences (MB = 4x4, *p* = 15).

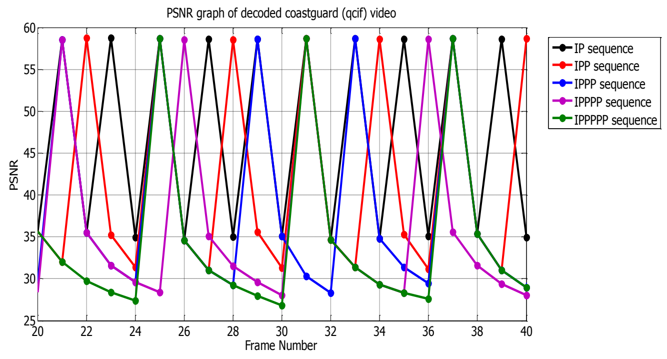

Fig. 16 PSNR performance of coastguard_qcif video with different I and P frame sequences (MB = 8x8, *p* = 7)

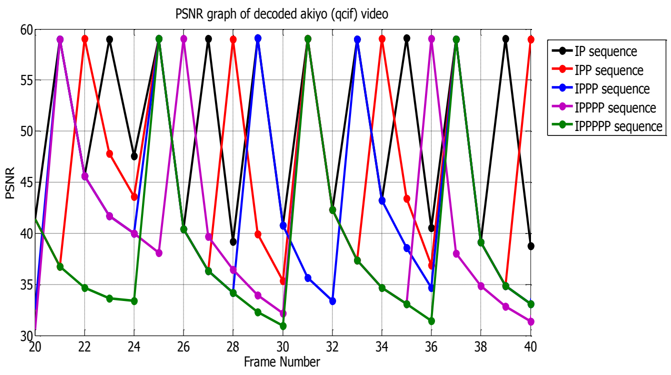

Fig. 17 PSNR performance of akiyo_qcif video with different I and P frame sequences (MB = 16x16, *p* = 5)

IV. Conclusions

Block Matching Algorithm (BMA) techniques are the most popular and efficient of various motion estimation techniques and have been used in many video coding applications. This paper investigates three different MB size (16×16, 8×8 and 4×4) and several search range *p* value for motion estimation with different *I* and *P* frame sequences. The latest video coding standard, H.264 (MPEG-4 part 10) has seven partitions of block that are 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, and 4×4 block size in order to improve the video quality and its coding efficiency [11]. The proposed hybrid ARPS program has improve the BMA efficiency based on the motion types whether it is a fast, medium and slow motion video. The difference between the frames is computed and examined whether it exceeded the threshold value or not. It will be better if the encoder is encoded by comparing the difference among the MBs and the system will decide what MB size and search range value is suitable for the encoder.

Acknowledgment

I would like to thank my main supervisor, Prof. Dr. Ali Chekima, my co-supervisor, Dr. Rosalyn R. Porle, for their constructive advices, and my friends for their valuable help.

References

[1] M. Manikandan, P. Vijayakumar, N. Ramadass, "Motion Estimation Method for Video Compression – An Overview", *IFIP International Conference on Wireless and Optical Communications Networks*, August 2006.

[2] Deepak J. Jayaswal and Mukesh A. Zaveri, "Probability Based Search Motion Estimation Algorithm", *2009 First International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 357-362, 2009.

[3] Jianhua Lu and Ming L. Liou, "A Simple and Efficient Search Algorithm for Block-Matching Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 7, no. 2, pp. 429-433, April 1997.

[4] Renxiang Li, Bing Zeng and Ming L. Liou, "A New Three-Step Search Algorithm for Block Motion Estimation", *IEEE Trans. Circuits and Systems for Video Technology*, vol. 4, no. 4, pp. 438-442, August 1994.

[5] Lai-Man Po, and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", *IEEE Trans. Circuits And Systems For Video Technology*, vol 6, no. 3, pp. 313-317, June 1996.

[6] Shan Zhu, and Kai-Kuang Ma, " A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation", *IEEE Trans. Image Processing*, vol 9, no. 2, pp. 287-290, February 2000.

[7] Yao Nie, and Kai-Kuang Ma, "Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation", *IEEE Trans. Image Processing*, vol 11, no. 12, pp. 1442-1448, December 2002.

[8] ThomasWiegand, Gary J. Sullivan, *Senior Member, IEEE*, Gisle Bjøntegaard, and Ajay Luthra, *Senior Member, IEEE, "*Overview of the H.264/AVC Video Coding Standard", *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560-576, July 2003.

[9] Rafael C. Gonzalez, Digital image processing (3rd ed), Pearson Prentice Hall, 2008

[10] Iain E.G. Richardson, H.264 and MPEG-4 Video Compression, John Wiley and Sons Ltd, 2003

[11] Gary J. Sullivan, Pankaj Topiwala and Ajay Luthra, "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions", *SPIE Conference on Applications of Digital Image Processing XXVII, Speacial session on Advances in the New Emerging Standards: H.264/AVC*, August 2004.