International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:3, No:2, 2009

Adaptive Motion Estimator Based on Variable Block Size Scheme

S. Dhahri, A. Zitouni, H. Chaouch, and R. Tourki

***Abstract*—**This paper presents an adaptive motion estimator that can be dynamically reconfigured by the best algorithm depending on the variation of the video nature during the lifetime of an application under running. The 4 Step Search (4SS) and the Gradient Search (GS) algorithms are integrated in the estimator in order to be used in the case of rapid and slow video sequences respectively. The Full Search Block Matching (FSBM) algorithm has been also integrated in order to be used in the case of the video sequences which are not real time oriented.

In order to efficiently reduce the computational cost while achieving better visual quality with low cost power, the proposed motion estimator is based on a Variable Block Size (VBS) scheme that uses only the 16x16, 16x8, 8x16 and 8x8 modes.

Experimental results show that the adaptive motion estimator allows better results in term of Peak Signal to Noise Ratio (PSNR), computational cost, FPGA occupied area, and dissipated power relatively to the most popular variable block size schemes presented in the literature.

***Keywords*—**H264, Configurable Motion Estimator, Variable Block Size, PSNR, Dissipated power.

I. Introduction

VIDEO communication is a rapidly evolving field for several applications which include video telephony, videoconference, remote surveillance, remote working and learning, etc. It is also a key feature for the upcoming information and communication technologies based on residential digital lines (VDSL, ADSL and ISDN) and the 3rd generation of mobile telephony system (UMTS). In this scenario, video image compression plays a fundamental role in reducing the enormous bit-rate for transmission and storage (approximately hundreds of Mbits/s for main image formats). To this objective, the ISO and the ITU-T committees have worked on several compression standards such as JPEG, MPEG (versions 1, 2, 4), H.261, H.263 and H.26L [1, 2].

Motion estimation is regarded as one of the most effective technique to reduce the flow required by a video codec. This stage is the most expensive in computing times and power consumption for the standard H264. Indeed, the method of estimation influences enormously the quality of the image and the handled video. The choice of the research strategy to be used in the stage of motion estimation is decisive in the total execution and on the video quality. One of the principal strategies to obtain an effective compression is to use a technique which exploits the space and temporal redundancy. Two main techniques are used for motion estimation: pixel recursive algorithms and block matching algorithm (BMA). The first one estimates the motion between successive frames on a pixel by pixel base, whereas the BMA estimates motion on a block by block basis. In general, due to its simple hardware realization, the block-matching approach is the most suitable.

Authors are with Electronics and Micro-Electronics Laboratory (LAB-IT06), Faculty of Sciences of Monastir, Monastir, 5000, Tunisia.

The computational cost of a motion estimation algorithm is greatly affected by the number of candidate blocks to be computed. In order to compute the Somme of Absolute Difference (SAD), the FBMA evaluates every possible motion vector inside the search area, find the best matched block and give the highest PSNR. Other strategies are based on the so called fast block motion estimation algorithms group which includes the Three-Step Search algorithm (TSS), the Modified Log Search algorithm (LS), the Four Steep Search (FSS) and the Alternative Pixel-Decimation Search algorithm (APDS) [3, 4, 5, 6, 7, 8, 9]. These algorithms take into account a reduced group of available candidate blocks at the expense of a reduction in term of PSNR.

Experimental results performed in [10] show that the full search algorithm is very greedy in computing times, but always gives the best results in compression ratio and PSNR (Fig. 1). This work shows also that in the case of fast sequences (ex. Forman) the 4SS algorithm gives better PSNR and compression rates relatively to other algorithms. In the case of slow sequences (ex. Miss America), [10] shows that the GS algorithm is more advantageous since the best block are always very close to the fetched block. By studying the sequences that contain the two types of motion (fast and slow) (ex. News and Carphone), [10] shows that the results are very close and speed can be improved by considering methods of research adapted for each type of block.

The motion estimation for each 16x16 macro-block in the emerging H.264/AVC can be divided into 16x16, 16x8, 8x16, and 8x8 modes. If the 8x8 mode is chosen, each 8x8 block can be independently coded into 8x4, 4x8, and even 4x4 blocks. Consequently, many VBS motion estimation algorithms has been proposed in literature, such as bottom-up merge and top-down split [11], prediction by merge and split [12], and low complexity motion estimation [13], etc.

In general, since, a video application can contain many sequences that are different in terms of motion nature (slow, stationary, fast), an adaptive motion estimation scheme is very suitable to improve performances of a video Codec (Fig. 1).

This paper presents the proposed adaptive motion estimator that can be dynamically configured by three algorithms (FSBM, 4SS, GS) depending on the variation of the video nature during the lifetime of a running application. This configuration is based on a recognition step that collects motion information of each block from its same block in the previous images and their neighbor's blocks in the current image. Contrary to many solutions proposed in the literature, the proposed approach is based on an efficient technique for decomposing the motion block. In fact, a new VBS motion estimation algorithm based on 16x16, 16x8, 8x16 and 8x8 only has been proposed, which can efficiently reduce the computational cost while achieving similar or better visual quality with low cost power on different architectures.

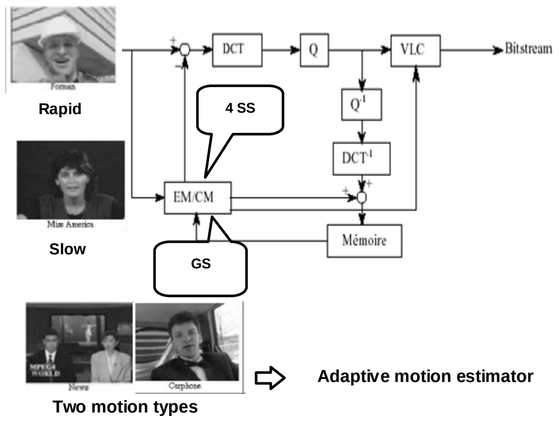


Fig. 1 Principal of the adaptive motion estimator

This paper is organized as follow: Section 2 presents brief overview of the Block Matching Algorithms (BMA) which are integrated in the adaptive motion estimator (4SS, GS, FS). In section 3, the architecture of the proposed motion estimator is presented. The proposed VBS scheme is described in section 4. Section 5 presents some experimental results and Section 6 concludes the paper.

II. Block Matching Algorithms

The FSBM algorithm is the most straightforward motion estimation operation. The process of block-matching is found in a search window of previous frames of the macro blocks most similar to the macro blocks in the current frame. The accuracy of ME depends on the matching criteria and one of the most popular criteria is the sum of absolute difference (SAD) given by:

$$SAD(k,l) = \sum_{i=1}^{16} \sum_{j=1}^{16} \left| P_t(i,j) - P_{t-1}(i+k, j+l) \right|$$

Where (k, l) is the location in the search window, Pt(i, j) is a pixel at (i, j) in the current frame, and Pt−1(i, j) is a pixel in the previous frame. When the value SAD(k,l) is minimum, (k,l) is the motion vector of the macro block.

The advantages of implementing FSBM as the motion estimation algorithm include both the guaranteed optimality of the solution and the regularity of a hardware implementation. This regularity stems from the fact that consecutive motion vectors share many of the pixel values in calculation.

A. Four Step Search

The Four-Step Search (4SS) is an algorithm that builds upon the TSS, while being more center-based. The 4SS differs in that it maintains a more regular search pattern with half-stop techniques employed in the algorithm [6]. The 4SS offers a similar best-case scenario for computational complexity when compared to the TSS (17 searches). The worst-case situation for the 4SS is 27 search points compared to 33 for the TSS, which is an improvement. Experimental studies performed on multitude standard video sequences show that this algorithm is well adapted to the fast sequences by giving the better PSNR and compression rates relatively to other algorithms.

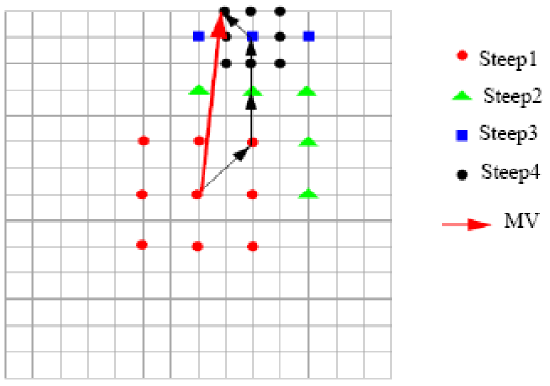


Fig. 2 Four Step Search paths

B. Gradient Search

The block-based gradient descent search (BBGDS) algorithm has been proposed by literature [5]. Based on the observation that global minimum distribution is centralized in real video sequence, this algorithm uses a very center-based search pattern of nine checking points in each step with a step size of one. It does not restrict the number of searching steps but it stops when the minimum checking point of the current step is the central one or it reaches the search window boundary. This algorithm has advantages: very low complexity in terms of candidate to evaluate, good regularity in terms of motion vector generation and only portion of search area memory can be accessed. Experimental studies performed on multitude standard video sequences show that this algorithm is well adapted to the slow sequences by giving the better PSNR and compression rates relatively to other algorithms

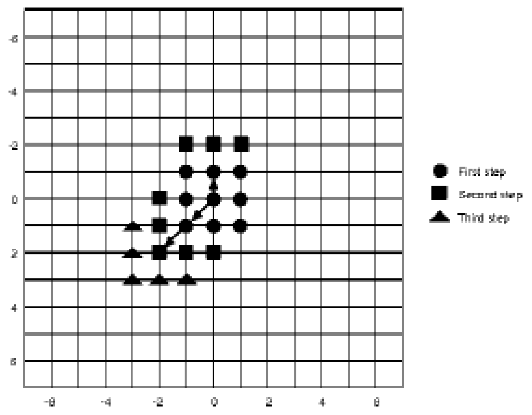


Fig.3 Gradient Search paths

C. Full Search

This algorithm defines a rectangular geometry for the image areas, consequently, the research window is limited to [- p, p] which is the area of research around the origin of the block in the reference image (Fig. 4). For an image with a size of (2p+1) × (2p+1) and for a window with a size of M*M and a block with a size of N ×N, the number of operations that are treated by the FSBM algorithm for each block is Nb = (2p+1)2×N2. The number of operations for each window is Nf = (2p+1)2×M2. Since this algorithm is the slowest in term of computing times, it is generally used for the applications with high image quality. It allows the sweeping of all the blocks of the research window. This makes it possible to reach the adequate motion vector in this window.

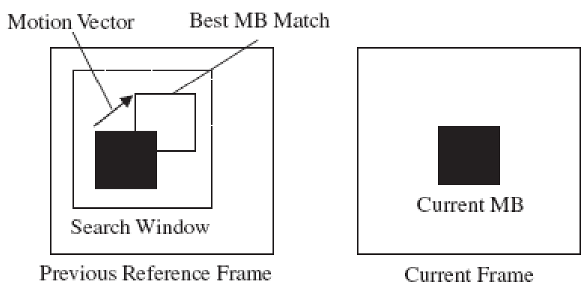


Fig. 4 Method of search for the full search algorithm

This algorithm selects the minimum among the possible vectors inside the research window (the motion vector belongs to the area [- p, p]). This approach is inadaptable by the majority of real time applications because of the high number of operations that is performed. Obviously, this algorithm always finds the optimal motion vector for the given research window. It should also be obvious, that this algorithm suffers from greatest data-processing complexity. For example, considering the research area of [- 7, +7], there are 226 calculations steps of the SAD which must be carried out for each macro block in a window.

III. Adaptive Motion Estimator Modeling

The motion estimation suggested integrates three algorithms: FSBM, 4SS and GS. The 4 modes (16x16, 16x8, 8x16 and 8x8) defined by the proposed VBS technique are integrated in this estimator. Knowing that all the motion estimators use memory to store the pixels of the block under running as well as block of reference and carry out the SAD calculation, these modules were defined only once inside the proposed architecture. Thus, each technique of motion estimation is defined only by its controller. Once configured, the selected controller communicates with the common SAD module inside architecture by using the memory common resources and ensures the estimation according to the algorithm of estimation in question. Such a strategy allows the estimator to be configured according to the nature of the application under running. Generally, the new Codec use motion estimation which integrates only one algorithm for each application [14, 15, 16, 17, 18]. For an application which handles video sequences of variable types, several estimators are normally necessary in order to guarantee an acceptable video quality for each type of sequence.

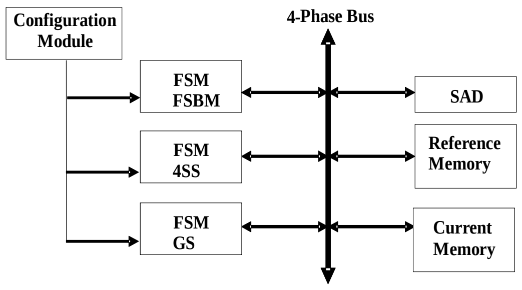


Fig. 5 Optimal architecture of the estimator

In our realization, each algorithm is specified by its control part that is described as a Finite State Machine (FSM). The architecture is optimized in order to make the redundant blocks reusable by the three algorithms (Fig. 5). According to the algorithm to be implemented, the corresponding FSM communicates with the other blocks via a 4-Phase handshaking bus. According to the application to be treated, the estimator is configured by the suitable algorithm. This allows the use of a single estimator for various applications, which makes it possible to guarantee an effective video compression. The estimator is composed of a set of modules which are configured with the proposed VBS scheme. The calculated SAD is stored in a memory. If the scanning of the search window is finished, then the memorized SAD is compared and the minimal SAD will be detected. Then a new research can be started. The optimal SAD with its position is sent towards a buffer memory.

The proposed motion estimator has been designed by Register Transfer Level (RTL) level by using the VHDL language. After being calculated by the chosen algorithm, the calculated SAD values are memorized in a memory in order to define for each block the starting point in the research window. All the calculated SAD values of the window are stored in a memory buffer and the minimal value is determined by trying. Then the output of the corresponding algorithm search block is the value of the minimal SAD and their positions. In other terms, the outputs are the motion vectors which will be used for the reconstruction of the image by the coder.

```
For (k = -p; k ≤ (p- bloc_col_v+1); k++)
      For l = -p; l≤ (p- bloc_line_v +1); l++)
        req <= '1';
        For (r = 0; r≤ length_v-1; r++)
            data_cour <= var_cour(r);
            data_ref <= var_ref(r);
         End;
          For v = 0; v≤length_v-1; v++)
              var_cour (v) <= "00000000";
          End;
             wait until ack ='1';
             motion_vec (z) <= sad_in;
               motion_vec2 (z, 0) <= k;
               motion_vec2 (z, 1) <= l;
               Req <= '0';
               z: = z+1;
             wait until ack ='0';
    For (m = k; m≤k+ (bloc_col_v-1) ; m++)
     For (n = l; n ≤ l+ (bloc_line_v-1); n++)
            var_cour (v) <= memory (m, n);
             v: =v+1;
           End;
          End;
        End;
    End;
      For (h = 0; h≤ nsad_v -1; h++)
        If (motion_vec (h) <= sad_min) then
          sad_min:= motion_vec (h);
          posit_x:= motion_vec2 (h, 0);
          posit_y:= motion_vec2 (h, 1);
        Else
          sad_min:= sad_min;
         End if;
         sum_sort0 <= sad_min;
         x_sort0   <= posit_x;
         y_sort0   <= posit_y;
      End;
```

Fig. 6 Algorithmic description of the Full Search algorithm

In the proposed architecture, the presence of a new window is indicated by a command signal (cmd_fen). This signal allows the initialization of the memory. After initialization, the algorithm begins the treatment by a signal called wr_ena (Fig. 7). The setting to '1' of the signal wr_ena makes it possible to charge a new window (current and reference). The setting to '0' of this signal allows the beginning of the sweeping of the window. For a new window the same operation will be repeated. An algorithmic description of the Full Search algorithm is represented by Fig. 6.

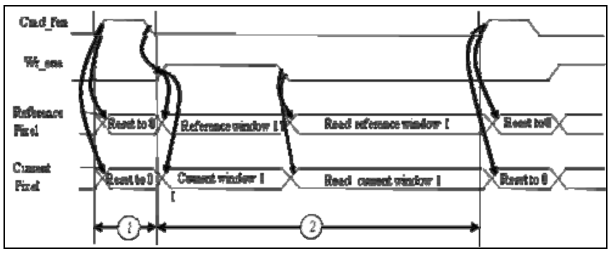


Fig. 7 The window read/write protocol

IV. Variable Block Size (VBS) Schemes

A. Variable Block Size 1 (VBS1)

Unlike previous standards, H.264/AVC adopts a tree-based decomposition to partition the macro-block (MB) into smaller sub-blocks of specified sizes. The quad-tree structure enables the possibility of a MB being coded in four different modes illustrated in Fig. 8, with partitions sizes of 16x16, 16x8, 8x16 and 8x8, while in the 8x8 partition mode, each 8x8 partition can be further split into 8x4, 4x8, and 4x4 sub-partitions. The availability of smaller ME blocks improves prediction in general, and in particular, the small blocks improve the ability of the model to handle fine motion detail and result in better subjective viewing quality because they do not produce large blocking artifacts.

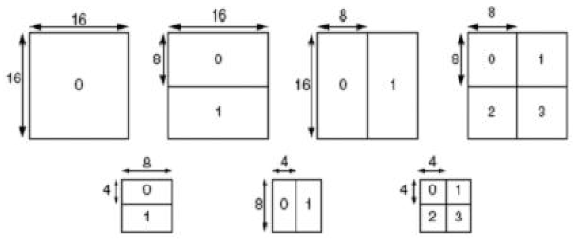


Fig. 8 Multiple motion estimation modes defined in H.264/AVC

B. Variable Block Size 2 (VBS2)

Serious experiments on the test video sequences used in JVT Test Model Ad Hoc Group [11] show that there is an average of 35 per cent homogeneous area in a typical video frame, and these areas are suitable for larger size inter mode coding. Therefore, several cost calculation of small size modes can be saved. Based on this consideration, several mode decision algorithms were proposed to reduce the number of candidate modes [11, 12]. In [19], many Fast VBS Motion Estimation algorithms were tested before extracting the Zoom Motion Estimation (ZME) algorithm [19] based on 3 partitions sizes 16x16, 8x8 and 4x4 arranged as shown in Fig. 9.

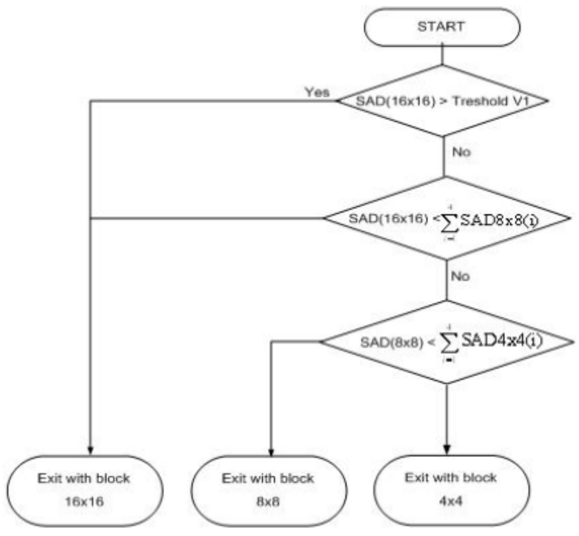


Fig. 9 Multiple motion estimation modes defined in H.264/AVC

C. Variable Block Size 3 (VBS3)

The proposal of the new typical based VBS scheme is to extract, using the SAD criteria, the block-size introducing the minimum computational complexity with minimum memory requirement. In this algorithm, only 16x16, 16x8, 8x16, and 8x8 block-sizes for motion estimation are considered. This approach will reduce extensively the computational complexity since the 3 other modes are eliminated, without affecting the overall visual video quality. For better understanding, the following flowchart explains in detail our proposed algorithm steps Fig. 10.

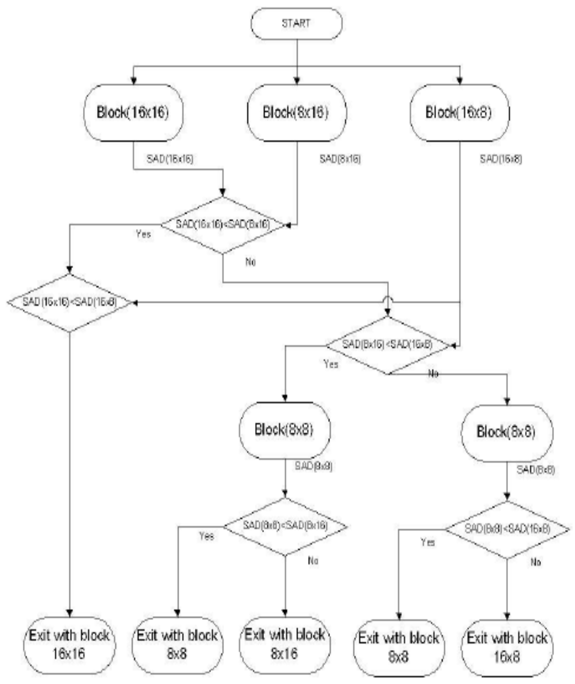


Fig. 10 Fast block size selection algorithm FBSA

V. Experimental Results

To evaluate the performances of the proposed adaptive motion estimator, both software and hardware implementations have been performed. The impact of the new VBS3 motion scheme on the encoding process, along with several analyses on three CIF (352x288) sequences test with different characteristics is proposed. At the beginning, the performances evaluation of the VBS3 algorithm compared to the VBS1 and VBS2 on a SW DSP (TMS320C64) design platform are presented. The used parameters consists on a QP =38 and a search window (Horizontal: [-15, 15] and Vertical:[-15, 15]). Finally, the logic synthesis and the power estimation results of the proposed estimator with the VBS3 scheme are presented and compared with the VBS1 and VBS2 schemes. These designs target the Altera FPGA technology and performed by the ISE and the X-Power tools.

A. Subjective, Objective and Complexity Analysis

Table I and Table II show that the elimination of 4x8, 8x4 and 4x4 block size (performed by our VBS3 scheme) from the mode selection does not affect the video quality (PSNR and SSIM [20]). Thus, the VBS3 algorithm provides better subjective and objective quality compared to the VBS2 scheme with complexity reduction in terms of clock cycles up to 44 per cent (Table III).

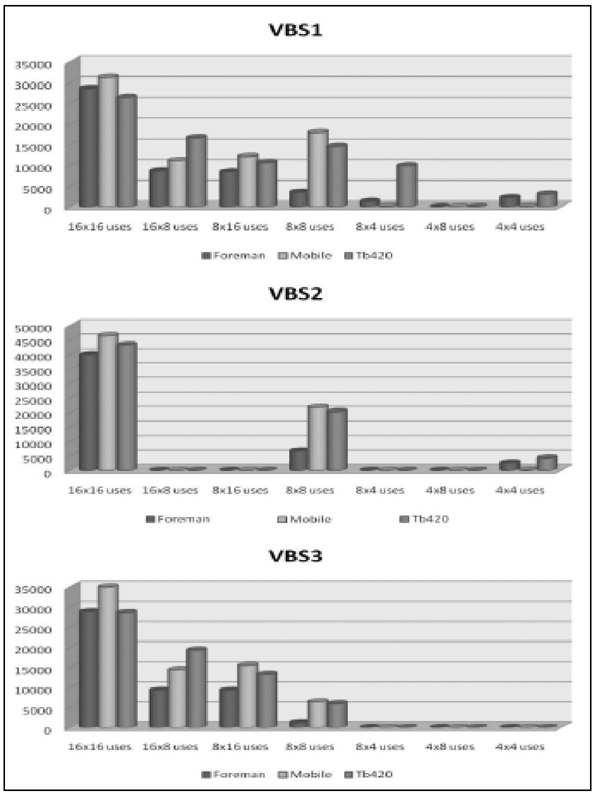
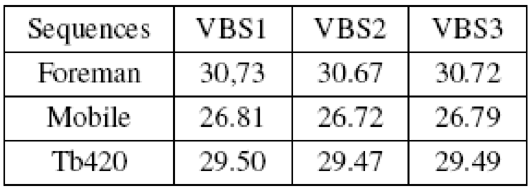


Fig. 11 Number of use modes for different variable block decision algorithm

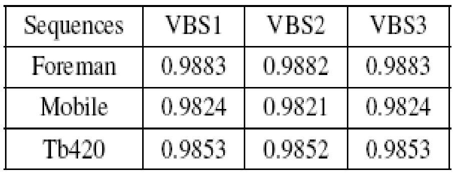
These results demonstrate the fact that from one hand, VBS3 outperforms the other variable block size techniques used in this study, such as VBS1 and VBS2 in terms of MB sizes (bit) distortion with a major loss in complexity. On the other hand, VBS3 is very similar to the other VBS predicting algorithms in terms of subjective quality with a major gain in the computational complexity and a better image quality depending on the motion characteristic of the video. In fact, the visual image quality with a major reduction in the computational effort complexity has been objectively improved. However, the penalized is performed only for the low motion sequence since the extra search will not be necessary (Fig. 11).

TABLE I
OBJECTIVE QUALITY PERFORMANCE (PSNR) OF BLOCK MODE SELECTION SCHEMES



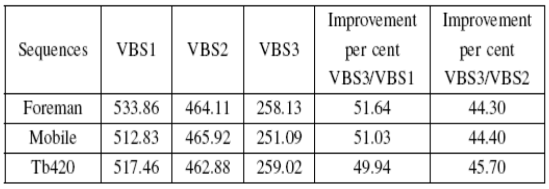
Sequences	VBS1	VBS2	VBS3
Foreman	30,73	30.67	30.72
Mobile	26.81	26.72	26.79
Tb420	29.50	29.47	29.49

TABLE II
SUBJECTIVE QUALITY PERFORMANCE (SSIM) OF BLOCK MODE SELECTION SCHEMES



Sequences	VBS1	VBS2	VBS3
Foreman	0.9883	0.9882	0.9883
Mobile	0.9824	0.9821	0.9824
Tb420	0.9853	0.9852	0.9853

TABLE III
SPEED PERFORMANCE (MILLIONS CYCLES) OF BLOCK MODE SELECTION SCHEMES



Sequences	VBS1	VBS2	VBS3	Improvement per cent VBS3/VBS1	Improvement per cent VBS3/VBS2
Foreman	533.86	464.11	258.13	51.64	44.30
Mobile	512.83	465.92	251.09	51.03	44.40
Tb420	517.46	462.88	259.02	49.94	45.70

B. *RTL Design Results*

In order to evaluate the occupied area and the dissipated power of the proposed motion estimator, the ISE and the X-Power FPGA design tools have been used. Table IV presents the logic synthesis and the power estimation results of the adaptive motion estimator configured with the FSBM algorithm and integrating the VBS1, VBS2, and the VBS3 schemes targeting the Virtex2pro FPGA technology.

Experimental results show that the FPGA equivalent area and the dissipated power obtained by our VBS scheme are slightly inferior to those obtained by the VBS1 and the VBS2 schemes. But since the motion estimator is responsible for nearly 70% of the power dissipated in certain video encoders, any reduction in power in the motion estimator would be critical for inclusion in a portable or other power-conscious device.

The gain in terms of the FPGA equivalent area designed with our VBS and the VBS2 schemes is about +4,35% relatively to the FPGA equivalent area designed by the standard VBS1. Also the gain in terms of the power dissipated by our VBS scheme is about +5,17% and +1,79% relatively to the power dissipated by the VBS1 and the VBS2 schemes (Table V).

TABLE IV
FPGA EQUIVALENT AREA AND POWER ESTIMATION RESULTS

Used VBS	**Equivalent area**	**Dissipated power (mW)**
VBS1	23%	58
VBS2	22%	56
VBS3	22%	55

TABLE V
COMPARISON OF THE VBS SCHEMES IN TERMS OF FPGA EQUIVALENT AREA AND DISSIPATED POWER

Equivalent area: VBS2 vs. VBS1	$(100-\frac{\mathbf{VBS2}}{\mathbf{VBS1}}\times 100)\%$	+4,35%
Equivalent area: VBS3. vs. VBS1	$(100-\frac{\mathbf{VBS3}}{\mathbf{VBS1}}\times 100)\%$	+4,35%
Dissipated power: VBS2 vs. VBS1	$(100-\frac{\mathbf{VBS2}}{\mathbf{VBS1}}\times 100)\%$	+3,45%
Dissipated power: VBS3. vs. VBS1	$(100-\frac{\mathbf{VBS3}}{\mathbf{VBS1}}\times 100)\%$	+5,17%
Dissipated power: VBS3. vs. VBS2	$(100-\frac{\mathbf{VBS3}}{\mathbf{VBS2}}\times 100)\%$	+1,79%

VI. CONCLUSION

In this paper a configurable motion estimator for the codec H264 has been presented. This estimator integrates three algorithms (FSBM, 4SS and GS) and implements a VBS scheme. Depending on the nature of the application under running this algorithm will be reconfigured dynamically by the adequate algorithm. Since, the motion estimator is responsible for nearly 70% of the power dissipated and computation cost in certain video encoders, any amelioration of the motion estimator performances would be critical for inclusion in a portable or other power-conscious device. Experimental results lead us to conclude that the proposed estimator integrating our VBS scheme allows better image quality with less computing time, less FPGA occupied area and less dissipated power relatively to the other VBS schemes. As a perspective, further design targeting ASIC technology for proposed adaptive motion estimator show that substantial improvements, in terms of encoding speed and dissipated power, can be obtained through optimizing heavily. This research work is ongoing, and the results will be presented in future publications.

REFERENCES

[1] G. Cote, and L. Winger, Recent progress in the field of video compression, IEEE Canadian Review- Spring 2002.
[2] ITU-T SG15, Video Codec for Audiovisual Service at Px64 Kbits/s, In ITU-T recommendation H.261 Version 3, Mars 1993.
[3] R. Srinivasan, and K. R. Rao, “Predictive coding based on efficient motion estimation”, IEEE Trans. on Circuits and Syst. on Video Technol., vol. 33, no. 8, August 1985, pp. 888-896.
[4] M. Ghanbari, “The cross-search algorithm for motion estimation,” IEEE Trans. on Communications, vol. 38, no. 7, July 1990, pp. 950-953.
[5] L.-K. Liu et E. Feig, 'A block-based gradient descent search algorithm for block motion estimation in video coding', IEEE Trans. on Circuits and Sys. for Video Technol., Vol. 6, No 4, pp. 419-422,1996.
[6] Lai-Man Po, and Wing-Chung Ma, “A novel four-step search algorithm for fast block motion estimation,” IEEE Trans. on Circuits and Syst. for Video Technol., Vol. 6, no. 3, June 1996, pp.313 – 317.
[7] C. Zhu, X. Lin, L-P. Chau, K-P. Lim, H-A. Ang and C-Y. Ong, “A novel hexagon-based search algorithm for fast block motion estimation,” IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001. Vol. 3, pp.1593 – 1596.
[8] C. Chun-Ho, and P. Lai-Man, “A novel cross-diamond search algorithm for fast block motion estimation,” IEEE Trans. on Circuits and Syst. for Video Technol., vol. 12, no. 12, December 2002, pp.1168-1177.
[9] Jae Hun Lee, and Al, “Variable block size motion estimation algorithm and its hardware architecture for H.264 /AVC”, IEEE Inter. Symp. on Circuits and Sys., May 2004.
[10] A. Djeffal, and Z. Baarir, “Video coding adaptive block matching “, 8th African Conference on Research in Computer Science, November 2006, pp.1-8.
[11] Y.K. Tu, J.F. Yang and M.T. Sun, Fast Variable-size Block Motion Estimation Using Merging Procedure with an Adaptive Threshold, IEEE International Conference on Multimedia and Expo. Baltimore, July 2003, p.II-789-792.
[12] Z. Zhou, M.T. Sun and Y.F. Hsu, Fast variable block-size motion estimation algorithms based on merge and split procedures for H.264/MPEG-4 AVC, IEEE International Symposium on Circuits and Syst., ISCAS. Vancouver, British Columbia, Canada, May 23-26, 2004.II-789-792.
[13] Y. Jiang,S. Li and S. Goto, A Low Complexity Variable Block Size Motion Estimation Algorithm for Video Telephony Communication, 47th IEEE International Midwest Symposium on Circuits and Syst.. July 2004, p.II-465 - II-468.II-789-792.
[14] M. G. Xavier, “Optimizing performance of an encoder following the standard Advanced Video Coding for a vector machine,” Master Memory, Faculty of Sciences libre University of Bruxcelle, 2006.
[15] S. Yalcin, H.F. Ates, I.Hamzaoglu, “A high performance hardware architecture for an SAD reuse based hierarchical motion estimation algorithm for H.264 video coding”, Proc. Inter. Conf. on Field Prog. Logic and App., Tampere, Finland, Aug 2005.
[16] P. Brault, “Motion Estimation and image segmentation”, Thesis Memory, Faculty of Sciences of Orsay, November 2005.
[17] LIU Hao, ZHANG Wen-jun, CAI Jun, “A fast block-matching algorithm based on variable shape search”, Journal of Zhejiang University Science A, Mars 2005.
[18] Tiago Miguel Braga da Silva Dias, “High-Performance VLSI Motion Estimation Processors: Data Reuse and Sub-Pixel Accuracy”, Master Memoir, Univ. de Tec. de Lisboa, Inst. Sup. Tec., Sep. 2004.
[19] M.A. Ben Ayed,A. Samet and N. Masmoudi, Toward an Optimal Block Motion Estimation Algorithm for H.264/AVC, International Journal of Image and Graphics (IJIG). 2006.
[20] Z. Wang,A.C. Bovik,H.R. Sheikh and E.P. Simoncelli, Image quality assessment: From error visibility to structural similarity, IEEE Trans. on Image Processing. vol. 13, no. 4, 2004.