Optimal DSP Based Integer Motion Estimation Implementation for H.264/AVC Baseline Encoder

Imen Werda[1], Haithem Chaouch[1], Amine Samet[2], Mohamed Ben Ayed[2], and Nouri Masmoudi[1]
[1] National School of Engineering, University of Sfax, Tunisia
[2] High Institute of Electronics and Communication, University of Sfax, Tunisia

Abstract: *The coding gain of the H.264/AVC video encoder mainly comes from the new incorporated prediction tools. However, their enormous computation and ultrahigh memory bandwidth are the penalties. In this paper we present an approach supporting efficient data reuse process to avoid unnecessary memory accesses and redundant motion estimation computations combined with a novel fast algorithm. A merging procedure joining search origin, search pattern and new variable block size motion estimation for H.264/AVC is detailed in this paper. Those approaches yield good tradeoffs between motion estimation distortion and number of computations since they invest and exploit the centre-biased characteristics of the real world video sequences: a reliable predictor determines the search origin, localizing the search process. An efficient search pattern exploits structural constraints within the motion field. A new fast block size selection DSP-based algorithm allows simultaneous fidelity of the video quality and the reduction of the computational cost. Extensive experimental work has been done, results of which show that our approach gives a speed up of 1.14 times over that of the recent fast algorithms and 10 times over the spiral search algorithm on average, with a negligible degradation in peak signal-to-noise ratio. In addition, interesting memory bandwidth is further saved with the proposed data reuse techniques at architecture level.*

Keywords: *H.264/AVC, search centre, block matching algorithm, pattern search, variable block size, complexity, video quality, PSNR, SSIM.*

Received March 7, 2008; accepted September 1, 2008

1. Introduction

Motion estimation and compensation play key roles in video compression systems due to the ability of realizing high compression rates achieved by exploiting the inter frame correlation between successive frames. Several video coding standards [1], incorporate sophisticate ME and motion compensation technologies to improve coding efficiency. In the basic motion-compensated predictive coding structure, motion estimation is carried out using the current frame and previously decoded one. The motion vectors are then used to construct the prediction frame. However, the enormous computations of these techniques [2] prevent the H.264/AVC standard [3] from any real-time application possibility. Therefore, it would be greatly beneficial to optimize as much as possible motion estimation tools, which takes the main share of the computational time. Several contributions in the literature aiming the reduction of the computational motion estimation cost were adopted.

Nowadays, Block Matching Algorithm (BMA) is the most popular technique adopted for motion estimation due to its simplicity for implementation. Generally, the most straightforward BMA called Full Search (FS) [4], which determines all the candidates in search window to obtain the best matching macroblock from reference frame, is not fit for real-time applications because of its unacceptable computational cost. Therefore, many well-known fast BMAs such as Three Step Search (TSS) [5], the HEXagon-Based Search (HEXBS) [6] and Diamond Search (DS) [7] have been proposed to reduce computational amount, based on fixed search pattern and greedy search method. As viewed from optimization theory, these traditional fast BMAs are based upon the following two assumptions: the error function has only one global optimum solution and the matching error decreases monotonically as the search point moves towards it. However, since the two assumptions can hardly be satisfied in most real-world videos, abovementioned fast BMAs are liable to get trapped in local minima resulting in degradation on video quality to some extent after decoding.

The purpose of this survey is to develop new methods that are pertinent to the processor-based implementation of the ME tools in the H.264/AVC encoder supporting efficient data reuse techniques to reduce memory bandwidth. Limited by the computing power and memory size of Digital Signal Processor (DSP), proposed ME tools have to reduce its computational complexity when matching the DSP computing architecture. Proposed DSP-based approaches, joining faithful predicted motion vector determination, flexible search pattern strategy and optimal Variable Block Size (VBS) motion estimation, have a good potential in

constructing a real time DSP-based encoder without sacrificing the compression efficiency.

The rest of the paper is organized as follows. Integer motion estimation implementation design is detailed in section2. Section 3 presents the new predicted motion vector approach. Section 4 gives a brief overview of several block matching algorithm strategies and our proposed algorithm Line Diamond Pattern Search (LDPS). We describe a brief overview of several fast VBS motion estimation algorithms along with our proposed Fast Block Size selection Algorithm (FBSA) in section 5. A DSP-core mapping of the SAD engine is developed in section 6. Our comparative simulation results and analysis are presented in section 7. Finally, conclusion is proposed in section 8.

2. Integer Motion Estimation Design

For a baseline H.264/AVC video encoder, the most critical component should be Integer Motion Estimation (IME). The frame memory, which is typically too large to fit into the embedded memory and usually located in external memory, stores reconstructed anchor frames (I- or P-frame) for reference; it dominates total memory usage in the encoder due to high video compression rate. For instance, frame memory accesses are responsible for approximately 98% of the total memory accesses. Such a large data transfer becomes the dominant part in the cycle's consumption of a video encoder. On the other hand, motion estimation is also a major consumer of overall system cycles, consuming about 50% of the total computational resources available to the encoder [8]. Compared with the previous standards, the IME of H.264/AVC is almost ten times more complex than that in MPEG-4 [2]. Therefore, it would be greatly beneficial to optimize as much as possible this module, which still takes the lion's share of the computational load. When low activity videos (e.g., video conference and video surveillance) which contain highly correlated images and large static video objects are encoded, there exist many stationary macroblocks. Proposed design exploits the stationary macroblock characteristic to avoid unnecessary frame memory accesses and redundant ME computations.

In ME, in order to find the best matched candidate of one current MB, a SW for reference frame has to be searched. The data transfer would be very heavy if all required pixels are loaded into internal memory. It consumes too much power and is not achievable in the platform-based system where the memory bandwidth is limited. To minimize the data/memory transfer time, we propose an approach that takes advantage of the DM642 architecture. More specifically, we develop a better design for data transfer via a DMA controller, reducing substantially memory I/O stalls. Since the CPU can do the processing in parallel with the DMA controller (different address and data buses), the encoder implementation can be structured to maximize DMA transfer in particular situations. However, we may have conflicts when performing simultaneous data accesses. Thus, it is necessary to understand the data flow and properly schedule it.

In the proposed IME design, local buffers are designed to store reusable data. At the architecture level, because the reference pixels of neighbouring candidates are considerably overlapped, a two-stage process is used to speed search region loading. At the beginning of each frame, three rows of 24 macroblocks wide (22 plus the left and right frame extension) is loaded directly from the external frame to an internal search buffer and completed immediately. On subsequent loads, tow row of 24 macroblocks wide are shifted to the beginning of the internal buffer, then later the load of the third row of 24 macroblocks take place as shown in Figure 1. The additional step avoids reloading macroblocks already existent in the local buffer. By means of local buffers access, the memory bandwidth can be greatly reduced when alleviating the data transfers.

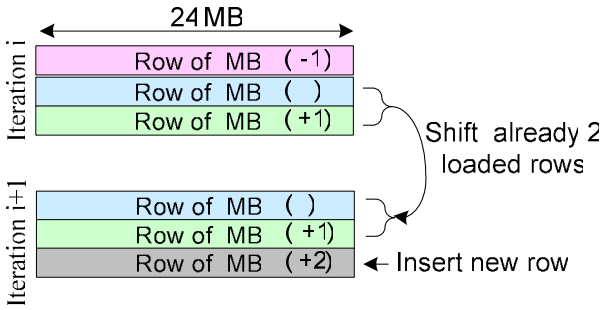

Figure 1. Reference search window load.

Reconstructed macroblocks are then written to the external frame buffer one row at a time. Each write begins at the end of the row, and waits for completion immediately.

The cycle's consumption of the IME module mainly comes from two parts. The first one is the data transfer consumption to read reference pixels from external to internal memories. The other is computational cycles to calculate matching costs with processing elements. Several techniques are used to reduce cycle's consumption. At the algorithm level, fast algorithms are applied to reduce the computational complexity. Both the data access power and the computational power are thus saved.

3. Potential Predicted Motion Vector Determination

In this section we study the effectiveness of different motion vector predictor algorithms in order to verify that their use is justified. Moreover, we propose a new

predictor, which may substitute the median predictor used in [9].

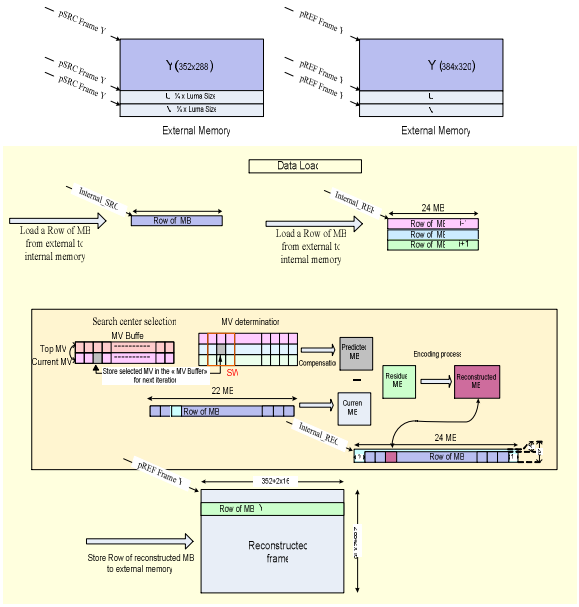

Figure 2. Integer motion estimation implementation design.

Motion in most videoconference image sequences involves a few blocks and lasts for a few frames. Therefore, spatially or temporally adjacent blocks often have similar motion vectors. So, the neighbouring motion activities can be exploited to achieve a good trade-off between the compression performance and the number of searched candidates.

Taking the advantage of the correlation among neighbouring motion vectors, an interesting approach were proposed to determinate the search centre before applying the search strategy. It considers the median value between three motion vectors of spatially adjacent blocks MB0, MB1, and MB2 as shown in Figure 3. The MV Predictors (MVP) is the median of left, up, and up-right blocks' MVs. This algorithm can successfully detect motion changes and effectively reduces the residual energy at the expense of increased algorithmic complexity.

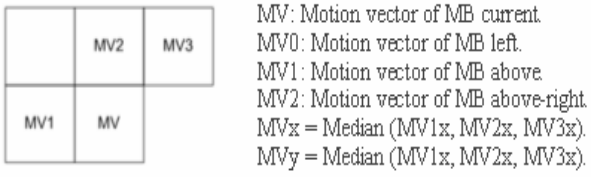

Figure 3. PMV determination approach.

Inspired from the previous approach, we propose a new software-based methodology which is more adapted to the DSP core architecture. In fact, conditional instructions used to extract the median in the previous approach, are not adapted to an optimal software implementation. When the CPU executes a conditional instruction, the pipeline must be "flushed". It must then start over with a new fetch and discard instructions that have already been fetched. Minimizing the pipeline "flush" overhead is critical to any pipelined architecture since conditional instructions causes' pipeline discontinuity which increase motion estimation module cycles consumption. In the proposed approach, both spatially and temporally previously coded motion vectors representing neighbouring macroblocks are used to predict the motion vector. As illustrated in Figure 4, an additional motion vector of temporally adjacent blocks (collocated) MBc is considered as a forth candidate of the selection. Four coasts are computed to choose the best candidate for the search centre. The candidate motion vector that yields to the minimum coast is chosen to be the PMV. The coast is calculated as Sum of Absolute Differences (SAD) between every alternate pixel in the corresponding blocks of reference frame and current one:

$$SAD(x,y) = \sum_{i=0}^{15}\sum_{j=0}^{15} \left| Ycurr(i,j) - Yprev(x+i, y+j) \right| \quad (1)$$

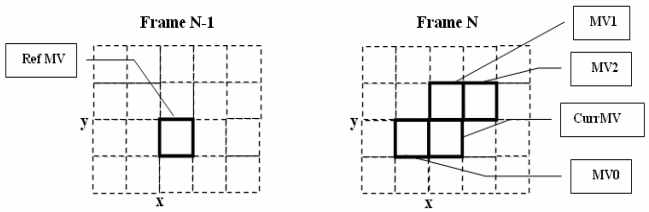

Figure 4. New PMV determination approach.

In order to find the best matched candidate of one current MB, a SW for reference frame has to be searched. However, after the search centre selection step, the search window can be constrained to a small clique surrounding this candidate position predicated based on the known neighbouring motion vectors which reduce considerably the computational time. Analysis provided in [10] shows that the search window dimension can be restricted to (horizontal: [-9, 9] and vertical: [-7, 7]) assuming that the objects move in a translational manner in a video conference sequence. The search window restriction, from (horizontal: [-15, 15] and vertical: [-15, 15]) to (horizontal: [-9, 9] and vertical: [-7, 7]), permits the highly localized search discussed next to originate from a more reliable location, as it distinguishes between motion and non-motion changes when deriving the predicted motion vector.

4. Block Matching Algorithm

Block Matching Algorithm (BMA) for motion estimation was been widely adopted by many video coding standard to reduce the temporal redundancy between different frames. The basic idea of the Block-matching process is to seek the best-matched block from the reference frame within certain search area. The matching criterion, usually called block distortion measure, is the Sum of Absolute Differences (SAD). There have been many Fast Motion Estimation (FME)

techniques proposed in the literature. FS [4] evaluates exhaustively all the possible candidate motion vector within the search window to find the globally best matched block in the reference frame. However, its very computationally intensive nature prevents it from practical implementation on a processor for real-time applications since it is considered to be the bottleneck of video coding systems. Hence, many kinds of fast motion estimation algorithms are now available. Typical fast algorithms available in the literature [8] can be grouped into four catalogues: (a) fast full search, (b) fast search by patterns, (c) fast search by schemes, and (d) Pixel decimation algorithms.

Gradient-based motion estimation algorithms, with carefully designed search patterns, have been developed to alleviate the computational load when limiting search points to a small subset of all possible candidates. This category is more adapted to a DSP implementation and includes the well-known TSS [5], DS [7], HEXBS [6], The Nearest Neighbours Search (NNS) [9] and the Horizontal Diamond Search (HDS) [11]. Although this category of algorithms may be trapped into a local minimum point and hence the efficiency of the motion compensation may drop, they can considerably reduce the number of block matching computations.

In order to design the optimal pattern and strategy search, two main proprieties were respected in the proposed approach:

- The distribution of the block motion fields is centre biased so the smaller displacement is more probable and the motion vector (0, 0) has the highest probability of occurrence [12]. This is illustrated in Figure 5, which shows sample SAD surfaces for a given target macroblock and each candidate macroblock within a (horizontal: [-15, 15] and vertical: [-15, 15]) search window. Thus, a small pattern is used in a fist step to verify this propriety an avoid loosing a computational time in stationary blocks.
- The SAD value will decrease to the minima in some direction [13] so the BMA should use a second pattern with higher dynamic than the first one in order to quickly determine the convenient area.

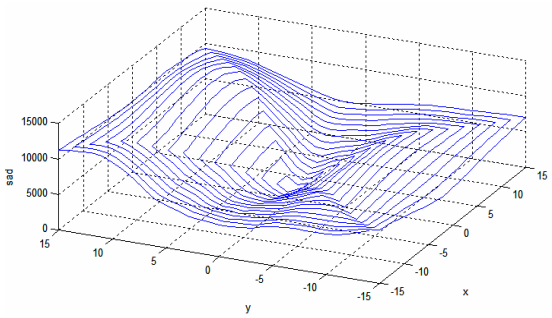

Figure 5. Sample Macroblock SAD distribution within a (horizontal: [-15, 15] and vertical: [-15, 15]) search window.

When proposing the new search algorithm Line Diamond Parallel Search (LDPS), we assume, as initial step, a simple search procedure, in which the LDPS exploits the centre-biased characteristics of the real world video sequences by using the Small Diamond Search Pattern (SDSP) as shown in Figure 6.

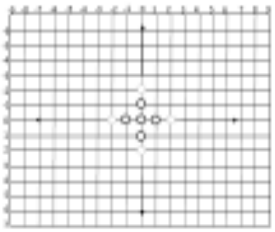

Figure 6. LDPS search patterns.

The checking points of the search pattern are examined one by one in the order from the centre point, and its points on the left, right, top and bottom. If the centre point gives the best match, it will be the search result. Else, a dynamic pattern is applied on the best match point, found from one of four surrounding centre points, to improve search on the horizontal or vertical motion component according to the best match of the previous iteration. Three directional candidates with steps of two pixels are then searched containing at most two untested candidate motion vectors. If the already tested point gives the best match, it will be the search result. Otherwise, a refinement is performed around the candidate who gives the best match when reapplying the first localized pattern as shown in Figure 7.

5. Mode Decision Algorithm

The computation burden is increased drastically for the H.264/AVC encoder because there are a number of combinations for partitioning a macroblock into sub-blocks ranging from 4x4 to 16x16 as shown in Figure 8. In a typical variable block-size motion estiation module, each MB can be split up in four kinds of partitions: 16x16, 16x8, 8x16, and 8x8. If the partition 8x8 is selected, each block can be further split into four kinds of subpartitions: 8x8, 8x4, 4x8, and 4x4. This tree-structured partition has 41 different blocks within one MB and gives rise to a large number of possible combinations.

Potentially, each sub-block can have its own motion vector. Such wide block choices greatly improve coding efficiency but at the cost of largely increased motion estimation time. Such high computational complexity is often a bottle-neck for realtime conversational applications. So, the mode decision algorithm is left as an open issue based on the fact that it is not specified in the H.264/AVC standard: With an optimal selection of the motion estimation tools the motion estimation algorithm could be designed to suit real time applications.

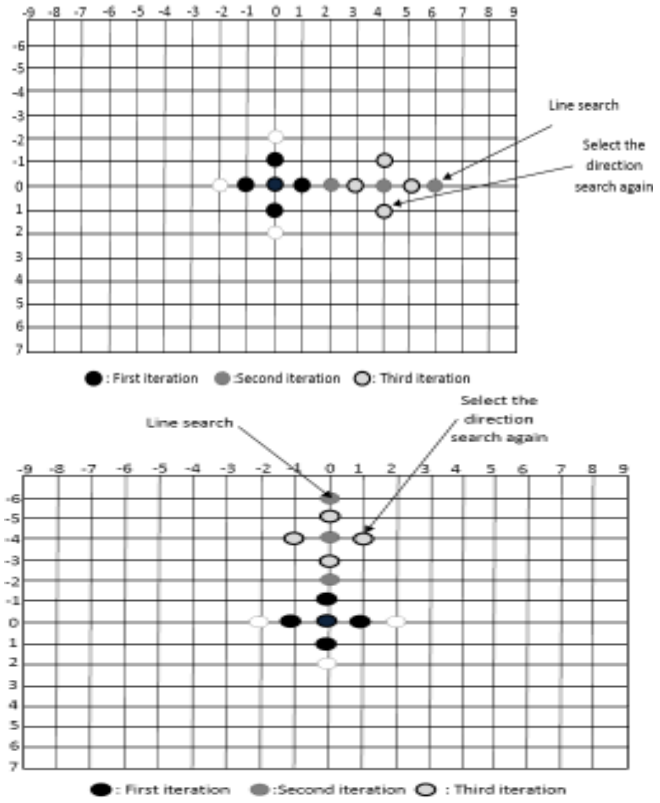

Figure 7. LDPS search strategy.

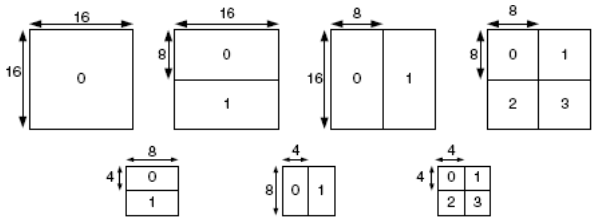

Figure 8. INTER modes with seven different block sizes ranging from 4x4 to 16x16.

Serious experiments on the test video sequences used in JVT test model Ad hoc group show that there is an average of 35% 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. In [11], many fast variable block-size motion estimation algorithms were tested before extracting the Zoom Motion Estimation (ZME) algorithm [11] arranged as show in Figure 9.

We implement the ZME algorithm on the TMS320C6416 DSP to study its effectiveness in order to find out whether it truly fit to the DSP architecture. When searching the best matched candidate of one current MB in a reference frame SW, matching costs are calculated and stored on the fly for each tested candidate. Each coast is stored as sixteen 4x4 SADs in a "historic search" buffer. This allows reusing locally SAD values for finding best match in sub-macroblock motion estimation at 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4 block levels. When extracting the best block size, there is no need to recompute SAD's for all iteration since they were carefully stored in a "historic search" buffer.

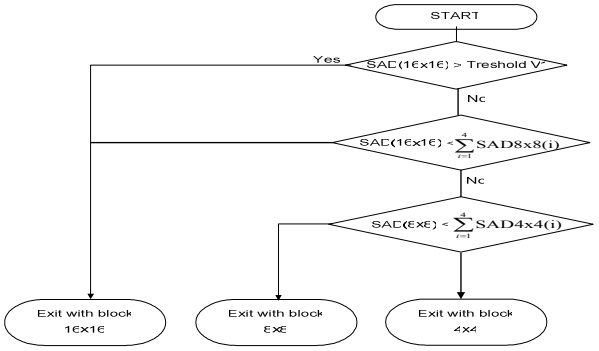

Figure 9. ZME algorithm.

We have just to sum respective SAD's to select the minimum coast and then the best block size. This approach reduces considerably computational complexity cycles at the expense of increased Memory Consumption (Mc).

Mc= 16*32*W*H= 64 Kbytes where W=32 and L=32: respectively width and high of the search window. The proposal of the new typical DSP-based VBS motion estimation algorithm 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 we are eliminating three modes, without affecting the overall visual video quality. For better understanding, the following organigram explains in detail our proposed algorithm steps:

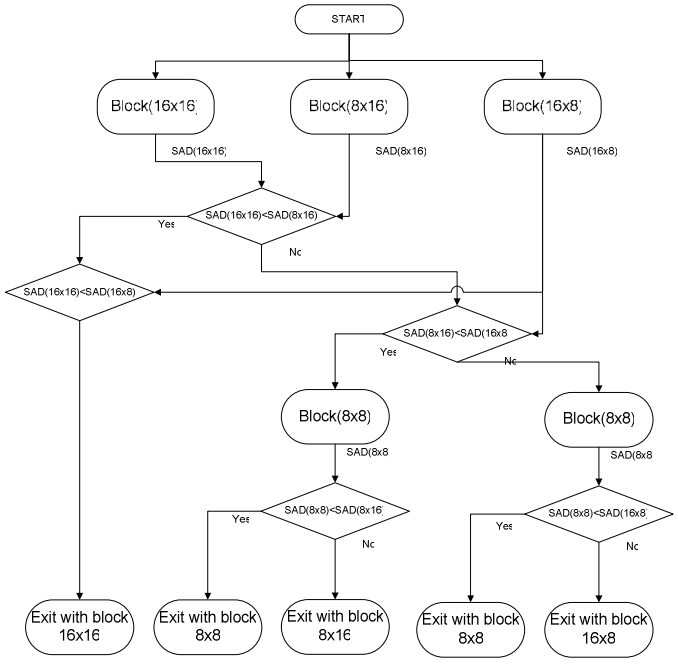

Figure 10. Fast block size selection algorithm FBSA organigram.

When adopting this algorithm, matching costs of corresponding candidates searched in a reference frame SW, are stored as four 8x8 SADs in the "historic search" buffer. Additional internal memory consumption is thus saved: only 16 Kbytes of internal memory is used for the "historic search" buffer against 64 Kbytes used for the ZME algorithm.

The best coast of 16x16 block is computed as the minimum sum of four 8x8 SADs computed at the same iteration. The best coast of 16x8 or 8x16 block is the minimum sum of four respective 8x8 SADs each tow of them are computed at the same iteration. To extract the best 8x8 block size, we need just to find out the minimum 8x8 SADs. In order to maintain the performance of the DSP-based VBS algorithm, the SAD engine is further optimized in the next section.

6. Architectural Optimization

To optimize the SAD engine, the latter is mapped carefully onto the C64 CPU core architecture proposed in Figure 11. Since the C64 provides multiple buses and separate functional units, parallel fetch, decode and execute operations can be over-lapped for maximum pipelining efficiency.

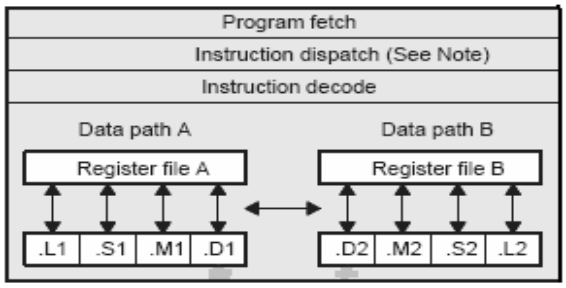

Figure 11. DSP core architecture.

To optimize the SAD engine, a Standard Assembly (SA) code is proposed. It takes advantage of the instruction set provided by the C64 through the use of instructions that operate directly on packed data. For more efficient parallelism, we perform the same process on different pixels by exploiting the *A* and *B* sides of the DSP when balancing the computational load on the four units of each side.

The SAD engine is implemented using a well-optimised SA as shown in Figure 12 code that enables us to benefit from the internal architecture of C64, which is considered to be the most suitable for any real-time multimedia application. Up to 75% reduction in terms of cycle count is obtained compared to the *C* code.

7. Simulation Results and Analysis

We study the effectiveness of the new motion estimation tools implemented in order to find out whether all of them truly contribute to the decrease of motion estimation time. Implementation discussed in the present paper has been developed on the TMS320C6416 DSP of Texas Instruments [18] based on LETI encoder [19].

To evaluate the new motion estimation tools impact on the encoding process, several analyses on a large number of sequences with the CIF format (352x288) and different quantization parameter values were performed. For limited space, only three sequences with low, medium and high motion activities shown as shown in Figure 13, are proposed in this section.

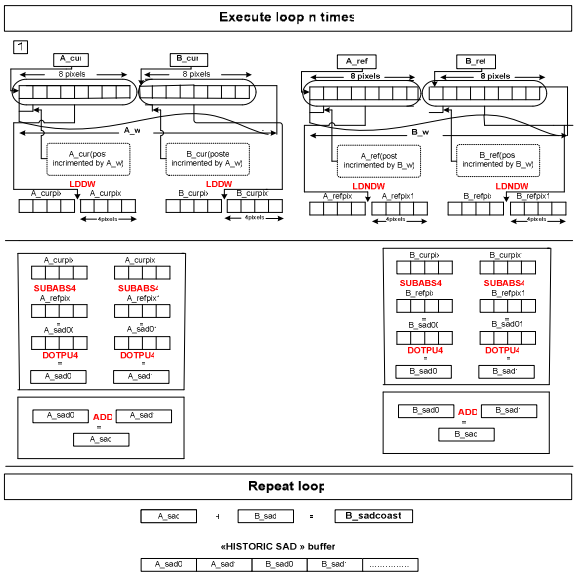

Figure 12. SAD software design diagram.

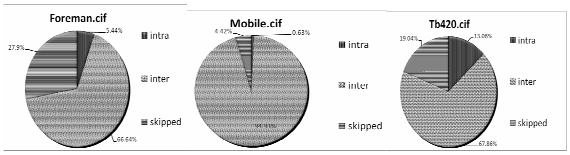

Figure 13. Percentages of ME (inter) intra and skip used in foreman, mobile and Tb420 CIF sequences.

Performance of proposed algorithms is compared with several common algorithms available in the literature. We first present the experimental results of the new approach used to determinate the predicted motion vector compared to the referenced one using the Spiral Search (SS) algorithm as a search strategy. We then evaluate the Fast Block Size Selection Algorithm (FBSA) improvement when compared to the ZME approach. Finally, we evaluate the performance of the new LDPS search strategy compared to DS, HEXBS, NNS, and HDS one, when combined to "min SAD (MV0, MV1, MV2, Ref MV)" and FBSA algorithms.

7.1. Experimental Results For The Proposed PMV Approaches

In this sub section, we will compare the MEDIAN (MV0, MV1, MV2) search centre performance to the proposed min SAD (MV0, MV1, MV2, Ref MV), with

search window (Horizontal: [-15, 15] and vertical: [-15, 15]). Table 1 and Table 2 shows that the new predicted motion vector provides a complexity reduction up to 1.7%, with no video quality degradation.

7.2. Experimental Results for Block Size Prediction

To illustrate the effect of the new variable block size algorithm, we illustrate the average PSNR and SSIM [20] for 50 predicted frames of three CIF sequences with QP =38 and search window (horizontal: [-15, 15] and vertical: [-15, 15]) as shown in Tables 3 and 4. The frames are predicted using spiral search algorithm to compare the ZME block mode selection algorithm performance to the proposed FBSA. In addition to the 25% memory saving, Tables 3 and 4 shows that the elimination of 4x8, 8x4, and 4x4 block size from the mode selection affects slightly the video quality (PSNR and SSIM). The FBSA algorithm provides better subjective and objective quality compared to the ZME approach with complexity reduction up to 44% as shown in Table 5.

7.3. Experimental Results for Search Pattern

Performance of the LDPS algorithm is compared with several common fast search algorithms available in the literature: The SS, TSS, HS, DS, NNS and HDS. The PSNR listed in the Table 6 is the average PSNR value of the luminance component of the decoded frames proposed for two search window dimensions when using the min SAD (MV0, MV1, MV2, MVc) approach to predict the search centre and block size selection ranging from 16x16 to 4x4. It can be noted that the search window reduction does not affect the video quality. In addition the LDPS algorithm provides the best objective quality compared to others implemented approach. Sp is the speed-performance on the average number of cycles needed to compute the ME engine. Table 7 show that the speed performance of the LDPS algorithm is very interesting compared to the other motion estimation algorithm at the exception of the NNS where a 4% loss of in speed performance is obtained.

7.4. Experimental Results When Combining Three Approaches

When combining our proposed algorithms, we can explain the lost in terms of complexity for the LDPS compared to the NNS algorithm shown in Table 7. In fact, when combined to the 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, and 4x4 mode selection, the code size of the merged ME module exceeds the cache memory size causing a cache miss overhead that affects the speed performance. When merged with the new approach, the code size of the ME module is reduced which alleviate the memory loads and show the right speed performance of the LDPS algorithm. As shown in Tables 8, 9 and 10, the merge of new DSP-based motion estimation tools approaches have been able to eliminate memory loads and reduce cycle's consumption of motion estimation module without any loss in video quality. Comparing our algorithm LDPS with the NNS, the speed performance of our algorithm is better for all the 3 sequences. The increases of the SP at the same PSNR vary from 8.92% to 11.17%. The overall performance of our algorithm is usually much better than that of other fast search algorithms; especially it is the fastest one. Results illustrated in Table 11, demonstrate the viability of the proposed algorithms in low bit rate video coding applications: Video conference. The proposed motion estimation algorithms provide substantially higher encoding speed as well as graceful computational degradation capabilities Figure 14 shows the speed improvements for a number of popular sequences and illustrate the consistent advantages of the CPU-specific SAD module optimizations: the architectural features of the TMS320C6416 are exploited to reduce the total number of ME cycles by 36% without any loss in video quality.

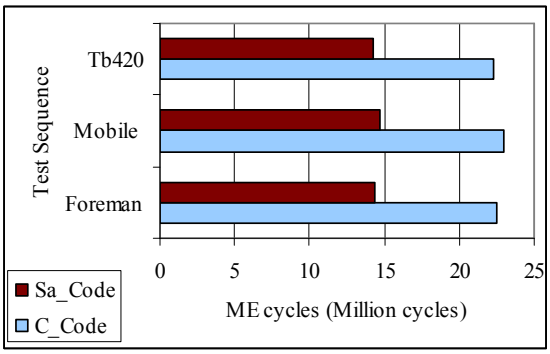

Figure 14. ME cycles consumption for foreman, mobile and Tb420 CIF sequences.

8. Conclusion

In this paper, we proposed DSP based specific methods to decrease the motion estimation module complexity in the H.264/AVC Baseline encoder, enabling interactive real-time video applications for CIF resolution on a single TMS 320C6416 DSP. The implementation of combined new PMV method, LDPS search pattern, and the FSBA decomposition strategy, is evaluated with various video CIF sequences. This approach is very effective for low motion video sequences. On the average, this work has obtained about 25% saving in frame memory accesses and reduced from 10% and 90% in ME computations for low motion video when compared respectively to NNS and SS algorithms with slightly video quality loss. The results illustrated in this paper lead us to conclude that if we use an accurate PMV with an appropriate LDPS search pattern combined with an optimal VBS algorithm, a better

image quality with less important computing time will be obtained. Additional CPU-specific optimizations of the SAD module exploiting the architectural features of the TMS320C6416 conduct to emphasize the superiority of our proposed algorithms for different video sequences. The optimized LETI's encoder is now able to encode ~22 frames per second for the CIF Foreman sequence as compared to 0.2 frames per second when using the spiral search algorithm, while of course maintaining the same video reproduction quality. Further analysis of the optimized LETI's encoder shows that substantial improvements, in terms of encoding speed, can be obtained through optimizing heavily some cycles consuming modules. This research work is on going, and the results will be presented in future publications.

References

[1] Al-Mualla E., Canagarajah N., and Bull R., "Simplex Minimization for Single and Multiple Reference Motion Estimation," *in Proceedings of IEEE Transaction Circuit and Systems for Video Technology*, pp. 1220-2001, 2001.

[2] Ben Ayed A., Samet A., and Masmoudi N., "Toward an Optimal Block Motion Estimation Algorithm for H.264/AVC," *International Journal of Image and Graphics*, vol. 5, no. 2, pp. 23-26, 2006.

[3] Circuit and System Group, http: //www.csgroup.tunet.tn, 2002.

[4] Chaouch H., Werda I., Samet A., and Massmoudi N., "Search Window Impact on H.264/AVC Motion Search Implementation on TMS320C6416 DSP," *in Proceedings of IEEE Conference on Communication and Signal Processing SSD*, USA, pp. 19-22, 2007.

[5] Gallant M., Côté G., and Kossentini F., "An Efficient Computation Constrained Block-Based Motion Estimation Algorithm for Low Bit Rate Video Coding," *in Proceedings of IEEE Transaction Image*, Chicago, pp. 8-11, 1999.

[6] Horowitz M., Joch A., Kossentini F., and Hallapuro A., "H.264/AVC Baseline Profile Decoder Complexity Analysis," *in Proceedings of IEEE Transaction Circuits System Video Technol*, UK, pp. 704-716, 2003.

[7] Joch A., Kossentini F., Schwarz H., Wiegand T., and Sullivan J., "Performance Comparison of Video Coding Standards Using Lagrangian Coder Control," *in Proceedings of IEEE International Conferences Image Processing,* USA, pp. 501-504, 2002.

[8] Jiang Y., Li S., and Goto S., "A Low Complexity Variable Block Size Motion Estimation Algorithm for Video Telephony Communication," *in Proceedings of 47th IEEE International Midwest Symposium on Circuits and Systems*, Japan, pp. 465-468, 2004.

[9] Jiang Y., Li S., and Goto S., "A Low Complexity Variable Block Size Motion Estimation Algorithm for Video Telephony Communication," *in Proceedings of IEEE International Midwest Symposium on Circuits and Systems*, Japan, pp. 465-468, 2004.

[10] Li R., Zeng B., and Liou L., "A New Three-Step Search Algorithm for Block Motion Estimation," *in Proceedings of IEEE Transactions Circuits System Video Technology*, USA, pp. 438-442, 1994.

[11] Richardson I., *Full Search Motion Estimation*, Washington, USA, 2002.

[12] Tham Y., Ranganath S., Ranganath M., and Kassim A., "A Novel Unrestricted Center-Biased Diamond Search Algorithm for Block Motion Estimation," *in Proceedings of IEEE Transactions Circuit and Systems for Video Technology*, USA, pp. 369-377, 1998.

[13] Thomas W., "Study of Final Committee Draft of Joint Video Specification," *Technical Report H.264 | ISO/IEC 14496-10 AVC*, 2002.

[14] Texas Instruments, "TMS32064xx Video/ Imaging Fixed-Point Digital Signal Processor," *Technical Literature Number: SPRS200E*, 2004.

[15] Tu Y., Yang F., and Sun T., "Fast Variable-Size Block Motion Estimation Using Merging Procedure with an Adaptive Threshold," *in Proceedings of IEEE International Conference on Multimedia and Expo*, Baltimore, pp.789-792, 2003.

[16] Wu D., Hou T., and Zhang Q., "Transporting Real-Time Video over the Internet: Challenges and Approaches," *in Proceedings of the IEEE INFOCOZ*, USA, pp. 1855-1874, 2000.

[17] Zhou Z., Sun T., and Hsu F., "Fast Variable Block-Size Motion Estimation Algorithms Based on Merge and Split Procedures for H.264/MPEG-4 AVC," *in Proceedings of IEEE International Symposium on Circuits and Systems*, ISCAS, pp. 23-26, 2004.

[18] Zhu C., Lin X., and Chau P., "Hexagon Based Search Pattern for Fast Block Motion Estimation," *in Proceedings of IEEE Transactions Circuit and Systems for Video Technology*, USA, pp. 349-355, 2002.

[19] Zhu S. and Ma K., "A New Diamond Search Algorithm For Fast Block-Matching Motion Estimation," *in Proceedings of IEEE Transaction Image Processing*, Japan, pp. 287-290, 2000.

Imen Werda received her degree in electrical engineering and her DEA in electronic engineering from Sfax National School of Engineering, Tunisia, in 2003 and 2004, respectively. Currently, she is assistant professor at the High Institute of Sciences and Technological Studies of Sousse ISSATS.

Haithem Chaouch received his degree of Diplôme d'ingénieur in electrical engineering in electronic engineering at National School of Engineering, Tunisia, in 2007.

Amine Samet received his degree of Diplôme d'Ingénieur in electrical engineering, his DEA in electronic engineering, and his PhD in electronic engineering from Sfax National School of Engineering, Tunisia, in 2001 2002, and 2006, respectively.

Mohamed Ben Ayed received his BS degree in computer engineering from Oregon State University in 1988, his MS degree in electrical engineering from Georgia Institute of Technology in 1990, his DEA degree and PhD in electronic engineering from Sfax National School of Engineering, Tunisia, in 1998 and 2004, respectively.

Nouri Masmoudi received his electrical engineering degree from the Faculty of Sciences and Techniques in Sfax, in 1982, the DEA degree from the National Institute of Applied Sciences- Lyon and University Claude Bernard-Lyon, France in 1984. From 1986 to 1990, he prepared his thesis at the Laboratory of Power electronics at the National School Engineering of Sfax. He received his PhD degree from the National School Engineering of Tunis, Tunisia in 1990.