Chapter 1

CUSTOMIZABLE AND REDUCED HARDWARE MOTION ESTIMATION PROCESSORS

Nuno Roma, Tiago Dias, Leonel Sousa

Abstract New customizable and reduced hardware core-based architectures for motion estimation are proposed. These new cores are derived from an efficient and fully parameterizable 2-D systolic array structure for full-search block-matching motion estimation and inherit its configurability properties in what concerns the macroblock and search area dimensions and parallelism level. A significant reduction of the hardware resources can be achieved with the proposed architectures by reducing the spatial and pixel resolutions, rather than by restricting the set of considered candidate motion vectors. Low-cost and low-power regular architectures suitable for field programmable logic implementation are obtained without compromising the quality of the coded video sequences. Experimental results show that despite the significant complexity level presented by motion estimation processors, it is still possible to implement fast and low-cost versions of the original architecture using general purpose FPGA devices.

Keywords: Motion Estimation, Customizable Architectures, Configurable Structures

1. Introduction

Motion estimation is a fundamental operation in motion-compensated video coding [1], in order to efficiently exploit the temporal redundancy between consecutive frames. Among the several possible approaches, block-matching is the most used in practice: the current frame is divided into equally sized $N \times N$ pixel blocks that are displaced within $(N+2p-1) \times (N+2p-1)$ search windows defined in the previous frame; the motion vectors are obtained by looking for the best matched blocks in these search windows. In this procedure, the Sum of the Absolute Differences (SAD) is the matching criteria that is usually used by most systems, due to its efficiency and simplicity.

Among the several block-matching algorithms, the Full-Search Block-Matching (FSBM) method is the one that has been used by most VLSI motion estimation architectures that have been proposed over the last few years. The main reason for this is not only related to the better performance levels generally achieved by exhaustively considering all possible candidate blocks, but is mainly due to the regularity properties that it also offers. In fact, not only does it lead to much more efficient hardware structures, but it also requires significantly simpler control units, which is always a fundamental factor towards a real-time operation based on hardware structures. However, it requires a lot of computational resources. As an example, FSBM motion estimation can consume up to 80% of the total computational power required by a video encoder. This fact often prevents its implementation using low cost technologies with restricted amounts of hardware (such as Field Programmable Logic (FPL) devices) and usually demands the usage of technologies with higher densities of gates, such as ASIC or Sea-of-Gates. As a consequence, several fast block-matching motion estimation algorithms have been proposed over the last years. Most of them restrict the search space to a given search pattern, providing sub-optimal solutions (e.g. [2, 3]). Nevertheless, they usually apply non-regular processing and require complex control schemes, making their hardware implementation difficult and rather inefficient.

Hence, a tradeoff is frequently raised between the required computational resources, the implementation complexity and the precision of the obtained motion vectors. The new VLSI architectures that are proposed allow the design of array processors based on the FSBM that can be implemented in FPL devices. These architectures are based on a highly efficient core, that combines both pipelining and parallel processing techniques to design powerful motion estimators [4]. This original core is used to derive simpler structures with reduced hardware requirements. The reduction of the complexity of these architectures is achieved by decreasing the precision of the pixel values and / or the spatial resolutions in the current frame, while maintaining the original resolution in the search space. The pixel precision is configured by defining the number of bits used to represent the input data and by masking or truncating the corresponding Least Significant Bits (LSBs). The spacial resolution is adjusted by sub-sampling the blocks of the current frame. By doing so, the best candidate block in the previous frame is still exhaustively searched but the SAD of each candidate block is computed by using only sparse pixels. Assuming a typical setup with blocks of 16×16 pixels, the number of considered pixels decreases by $1/4$ and $1/16$ by applying alternate sub-sampling schemes of $2:1$ or $4:1$, respectively.

The efficiency of the proposed structures was evaluated by implementing these customizable core-based architectures in Field Programmable Gate Arrays (FPGA). It is shown that the amount of hardware required when sub-

sampling and truncation techniques are applied is considerably reduced. This fact allows the usage of a common framework for designing a wide range of motion estimation processors with different characteristics that fit well in current FPGAs, being a real alternative to those fast motion estimation techniques that apply non-regular processing. Moreover, experimental results obtained with benchmark video sequences show that the application of these techniques does not introduce a significant degradation in the quality of the coded video sequences.

2. Base FSBM Architecture

Several FSBM structures have been proposed over the last few years (e.g.: [5, 6, 4]). Very recently, a new class of parameterizable hardware architectures that is characterized by offering minimum latency, maximum throughput and a full and efficient utilization of the hardware resources was presented in [4]. This last characteristic is a fundamental requisite in any FPL system, due to the limited amount of hardware resources. To achieve such performance levels, a peculiar and innovative processing scheme, based on a cylindrical hardware structure and on the zig-zag processing sequence proposed by Vos [6], was adopted (see fig. 1.1). With such a scheme, not only is it possible to minimize the processing time, but it also provides the ability to prevent the usage of some hardware structures – the so called *passive processor elements (PEs)*, that do not carry useful information in many clock cycles.

Besides this set of performance and implementation characteristics, this class of processors features also a scalable and configurable architecture, making it possible to easily adapt the processor configuration to fulfill the requisites

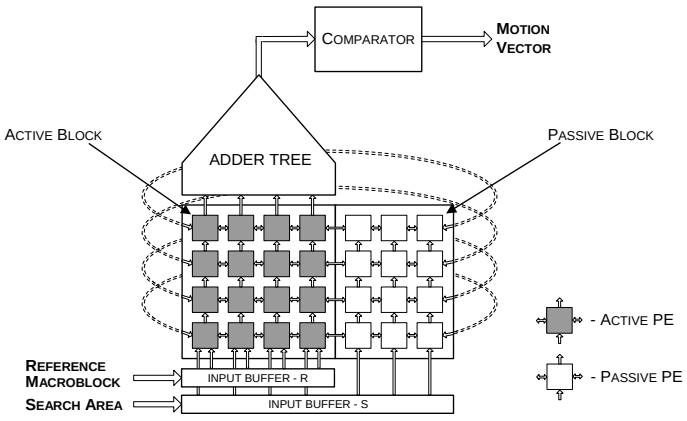


Figure 1.1. Processor array proposed in [4] based on an innovative cylindrical structure and adopting the zig-zag processing scheme proposed by Vos [6] ($N = 4$, $p = 2$).

of a given video coder. By adjusting a small set of implementation parameters, processing structures with distinct performance and hardware requirements are obtained, providing the ability to adjust the required hardware resources to the target implementation technology. While high performance processors, that inevitably require more resources, are more suited for implementations using technologies such as ASIC or Sea-of-Gates, those low-cost processors, that are meant to be implemented in FPL devices, with limited hardware resources, should use configurations requiring reduced amounts of hardware.

3. Architectures for Limited Resources Devices

Despite the set of configurable properties offered by FSBM architectures and, in particular, by the class of processors proposed in [4], restricted hardware technologies, such as FPL devices, often do not provide enough hardware resources to implement such processors. In such cases, sub-optimal motion estimation algorithms are usually adopted, which provide faster processing and require reduced amounts of hardware. Different categories of sub-optimal motion estimation algorithms have been proposed, based on three main techniques:

- *Reduction of the set of considered candidate motion vectors*, by restricting the search procedure in the previous frame to a given search pattern using hierarchical search strategies [2, 3];
- *Decimation at the pixel level*, where the considered similarity measure is computed by using only a subset of the $N \times N$ pixels of each reference macroblock [7–9];
- *Reduction of the precision of the pixel values*, where the similarity measure is computed by truncating the LSBs of the input values to reduce the hardware resources required by the arithmetic units [10, 9].

The main drawback of these solutions is a corresponding increase of the prediction error that inevitably arises as result of using less accurate estimations. This tradeoff usually leads to a difficult and non-trivial relationship between the final picture quality and the prediction accuracy that can not be assumed to be linear. In general, a larger prediction error will lead to higher bit rates, which will lead to the usage of greater quantization step sizes to compensate this increase, thus affecting the quality of the decoded images.

Up until now only a few VLSI architectures have been proposed to implement fast motion estimation algorithms, by restricting the search positions according to a given search pattern [9]. In general, they imply the usage of non-regular processing structures and require higher control overheads, which complicates the design of efficient systolic structures. Consequently, they have been extensively used in software applications, where such restrictions do not usually apply so strictly.

The set of architectures proposed herein try to combine the advantages offered by the regular and efficient FSBM structures proposed in [4] with the several strategies to reduce the amount of hardware that are usually offered by sub-optimal motion estimation algorithms. To implement such architectures on FPL devices, such as FPGAs, the original FSBM architecture will be adapted to apply two of the three decimation categories described above: decimation at the pixel level and reduction of the precision of the pixel values.

Decimation at the pixel level

By applying the decimation at the pixel level, the image data of the current frame is sub-sampled, by considering alternate pixels in each orthogonal direction. This scheme corresponds to using a lower resolution version of the reference frame in the search procedure that is carried out within the previous full-resolution frame. The SAD measure for a configuration using a $2^S : 1$ sub-sampling in each direction is given by (considering N a power of 2):

$$SAD(l,c) = \sum_{i=0}^{\frac{N}{2^S}-1} \sum_{j=0}^{\frac{N}{2^S}-1} \left| x_t(i.2^S, j.2^S) - x_{t-1}(l + i.2^S, c + j.2^S) \right| \quad (1.1)$$

The FSBM circuit proposed in [4] can be easily adapted to carry out this type of sub-sampling. In fact, considering that the computation of the SAD similarity measure is performed in the active block of the processor, the decimation can be implemented by replacing the corresponding set of active PEs by passive PEs. By doing so, only the pixels with coordinates $(i.2^S, j.2^S)$ will be considered and significant amounts of hardware resources can be saved.

Two different decimation patterns can be applied to the active block of the processor: *i)* an *aligned pattern*, in which the active PEs are aligned in rows and columns, as shown in fig. 1.2a) and *ii)* a *checkerboard pattern*, in which the active PEs are placed as it is shown in fig. 1.2b). Although this latter pattern may seem to provide better estimations, since it reduces the correlation degree of the pixel values, experimental results have shown that the usage of this pattern may cause a Peak Signal to Noise Ratio (PSNR) reduction of the coded video of about 0.5-1.0 dB. On the other hand, by adopting the sub-sampling pattern presented in fig. 1.2a) the degradation of the PSNR is negligible and significant amounts of hardware resources can be saved. In fact, not only do the passive PEs require considerably fewer hardware resources, but also the number of inputs of the adder tree block is reduced by the sub-sampling factor (S), which further reduces the amount of required hardware (see fig. 1.2).

According to the classical signal processing theory, this pixel decimation approach can be seen as a reduction of the sampling frequency of the image. Hence, it would be reasonable to expect that the PSNR of the coded video could be improved if anti-aliasing filtering were applied to the search or/and

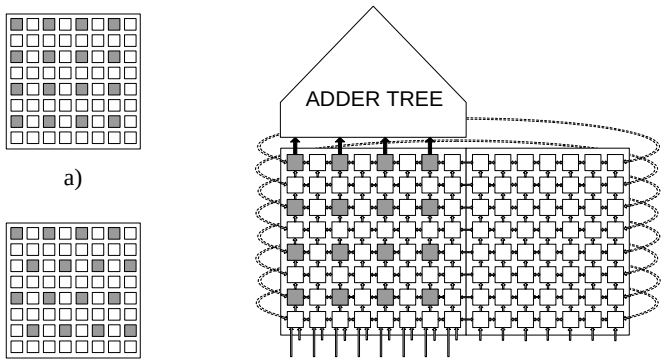


Figure 1.2. *a)* Aligned pattern. *b)* Checkerboard pattern. *c)* Modified processing array to carry out a 2:1 decimation function using the architecture proposed in [4] ($N = 8$, $p = 4$).

reference areas of the image. However, experimental results showed that the usage of such filters causes the opposite effect. Whether the filter is only applied to the search area, to the reference area or to both areas, the PSNR of the coded video always suffers a reduction of 0.5-5.0 dB, depending on the type of video sequence, i.e., on the amount of movement. Furthermore, since this PSNR reduction was obtained using both 3×3 and 5×5 Gaussian filters, it can be concluded that the PSNR reduction is independent of the filter type. Moreover, significant hardware resources would have to be wasted in the implementation of these 2D filters. As a result, the pixel decimation approach was implemented using the pattern depicted in fig. 1.2a) and not using any anti-aliasing filtering. The block diagram of the modified processing array is presented in fig. 1.2c).

Reduction of the precision of the pixel values

Another strategy to decrease the amount of hardware required by FSBM processors is to reduce the bit resolution of the pixel values considered in the computation of the SAD similarity function by truncating the LSBs. By adopting this strategy alone ($S = 0$) or in conjunction with the previously described sub-sampling method ($0 < S < \log_2 N$), the SAD measure is given by:

$$SAD(l,c) = \sum_{i=0}^{\frac{N}{2^S}-1} \sum_{j=0}^{\frac{N}{2^S}-1} \left| \left\lfloor \frac{x_t(i.2^S, j.2^S)}{2^T} \right\rfloor - \left\lfloor \frac{x_{t-1}(l+i.2^S, c+j.2^S)}{2^T} \right\rfloor \right| \quad (1.2)$$

$$\equiv \sum_{i=0}^{\frac{N}{2^S}-1} \sum_{j=0}^{\frac{N}{2^S}-1} \left| x_t(i.2^S, j.2^S)_{7:T} - x_{t-1}(l+i.2^S, c+j.2^S)_{7:T} \right| \quad (1.3)$$

where T is the number of truncated bits and $x_t(i,j)_{7:T}$ are the $(8-T)$ most significant bits of a pixel value of the t^{th} frame.

The adaptation of the original FSBM architecture [4] to apply this bit truncation scheme is straightforward: it is only necessary to reduce the operand widths of the several arithmetic units implemented in the active PEs, in the adder tree blocks and in the comparator circuits. Such modifications potentially increase the maximum frequency of the pipeline and will significantly reduce the amount of required hardware, thus providing the conditions that will make it possible to implement the motion estimation processors in FPL devices.

4. Implementation and Experimental Results

Several different setups of the proposed customizable core-based architectures for motion estimation were synthesized and implemented in a general purpose VIRTEX XCV3200E-7 FPGA using the Xilinx Synthesis Tool from ISE 5.2.1. The considered set of configurations assumed each macroblock composed by 16×16 pixels ($N = 16$) and a maximum displacement in each direction of the search area of $p = 16$ pixels. These configurations were thoroughly tested using sub-sampling factors (S) varying between 0 and 2 and a number of truncated bits (T) of 0, 2 and 4. The experimental results of these implementations are presented in tables 1.1 and 1.3.

The proposed core-based architectures can provide significant savings in the required hardware resources. From the set of configurations presented in table 1.1, one can observe that reduction factors of about 75% can be obtained by using a sub-sampling factor $S = 2$ and by truncating the 4 LSBs. However, this relation should not be assumed to be linear. By considering only the pixel level decimation mechanism ($T = 0$), it can be shown that a reduction of about 38% is obtained by using a $2 : 1$ sub-sampling factor ($S = 1$), while a $4 : 1$ decimation will provide a reduction of about 51%. The same observation can be made by considering only the reduction of the precision of the pixel values ($S = 0$). While using 6 representation bits ($T = 2$) a reduction of the number of CLB slices of about 31% is obtained (a reduction of about 23% of the number of Look-up Tables (LUTs)), by using only 4 representation bits ($T = 4$) a reduction of about 51% of the CLB slices is achieved (a reduction of about 47% of the number of LUTs). Table 1.2 presents the set of FPGA devices that should be used by each configuration in order to maximize the efficiency of the hardware resources used by each processor.

Table 1.3 presents the variation of the maximum operating frequency of the considered configurations with the number of truncated bits (T). Contrary to what could be expected, the reduction of the operands width of the several arithmetic units does not significantly influence the processors performance. This fact can be explained if one takes into account the synthesis mechanism that is used by this family of FPGAs to synthesize and map the logic circuits using built in fast carry logic and LUTs.

Table 1.1. Percentage of the CLB slices and LUTs that are required to implement each configuration of the proposed core-based architecture for fast motion estimation in a VIRTEX XCV3200E-7 FPGA ($N = 16; p = 16$).

	T=0		*T=2*		*T=4*	
S	*CLB Slices*	*LUTs*	*CLB Slices*	*LUTs*	*CLB Slices*	*LUTs*
0	90.7%	30.7%	62.6%	23.5%	43.8%	16.3%
1	56.2%	19.0%	38.2%	14.6%	26.6%	10.7%
2	44.7%	14.8%	30.1%	11.4%	21.0%	7.8%

Table 1.2. Alternative FPGA devices to implement each of the considered configurations ($N = 16; p = 16$).

S	*T=0*	*T=2*	*T=4*
0	XCV3200	XCV2600	XCV1600
1	XCV2000	XCV1600	XCV1000
2	XCV1600	XCV1000	XCV600

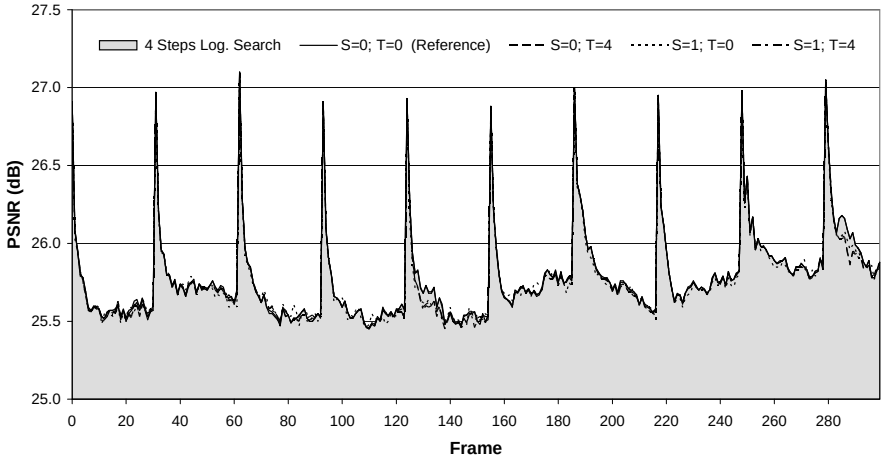
Table 1.3. Variation of the maximum operating frequency with the number of truncated bits (T) ($N = 16; p = 16$).

T=0	*T=2*	*T=4*
76.1 MHz	77.8 MHz	79.8 MHz

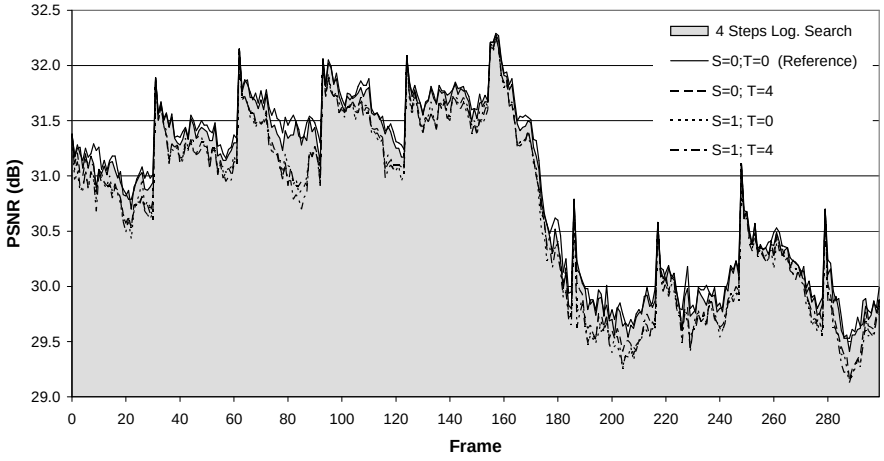
To assess and evaluate the efficiency of the synthesized processors in an implementation based on FPL devices, they were embedded as motion estimation co-processors in the H.263 video encoder provided by Telenor R&D [11], by transferring the estimated motion vectors to the video coder. Peak signal-to-noise ratio (PSNR) and bit-rate measures were used to evaluate the performance of each architecture. These results were also compared with those obtained with a sub-optimal 4-step logarithmic search algorithm [1], implemented in software.

The first 300 frames of several QCIF benchmark video sequences with different spatial detail and amount of movement were coded in interframe mode, by considering a GOP length of 30 frames and a quantization step with intermediate size of $\Delta = 30$ to keep the quantization error as constant as possible.

Fig. 1.3 presents the PSNR values obtained for the *carphone* and *mobile* video sequences, characterized by the presence of high amounts of movement and high spatial detail, respectively. Several different setups in what concerns the number of truncated bits (T) and the sub-sampling factor (S) for the decimation at the pixel level were considered. The PSNR value obtained for the INTER type frames of the *mobile* sequence is almost constant ($25 - 26$ dB), which demonstrates that the degradation introduced by using the reduced hardware architectures is negligible: less than 0.15 dB when the 4 LSBs are truncated and the sub-sampling factor is $2 : 1$. In contrast, it can be seen that the PSNR values obtained for the *carphone* sequence varies significantly along the time. The main reason for this fact is the amount of movement present in this sequence that is also varying. Even so, the proposed reduced hardware



(a) Mobile.



(b) Carphone.

Figure 1.3. Comparison between the PSNR values obtained for three different setups of the proposed reduced hardware architecture, for the original reference FSBM configuration and for the 4-steps logarithmic search algorithm.

architectures only introduce a slight degradation in the quality of the coded frames, when compared with the performances of both the reference FSBM architecture ($S = T = 0$) and of the 4-steps logarithmic search algorithm. A maximum reduction of about 0.5 dB is observed in a few frames when the PSNR value obtained with the original (reference) architecture is greater than 30 dB.

As it was previously noted, these video quality results were obtained with a constant quantization step size. The observed small decrease of the PSNR can be explained by the slight increase of the quantization error, as a consequence

Table 1.4. Variation of the output bit rate to encode the considered video sequences by using different setups of the proposed core-base architecture and the 4-steps logarithmic search algorithm.

(a) Miss America.

S	***T=0***	***T=2***	***T=4***
0	**31.9 kbps**	-0.8%	+3.4%
1	+1.7%	+3.0%	+3.8%
2	+3.9%	+3.8%	+3.8%
Four-step logarithmic search			+0.3%

(b) Silent.

S	***T=0***	***T=2***	***T=4***
0	**49.9 kbps**	+0.0%	+2.7%
1	+4.5%	+4.1%	+8.0%
2	+10.7%	+8.4%	+8.6%
Four-step logarithmic search			+0.8%

(c) Mobile.

S	***T=0***	***T=2***	***T=4***
0	**271.4 kbps**	+0.0%	+0.3%
1	+1.1%	+0.3%	+0.5%
2	+6.7%	+0.6%	+0.5%
Four-step logarithmic search			-0.1%

(d) Carphone.

S	***T=0***	***T=2***	***T=4***
0	**79.3 kbps**	-0.8%	+4.8%
1	+7.3%	+5.8%	+11.6%
2	+14.5%	+11.6%	+12.0%
Four-step logarithmic search			+1.7%

of the inherent increase of the prediction differences in the motion compensated block, obtained with these sub-optimal matching processors. The required bit-rate to store or transfer the two video sequences considered above and two other additional sequences, (*Miss America* and *Silent*) is presented in table 1.4. The relative values presented in table 1.4 demonstrate the corresponding increment of the average bit-rate when the number of truncated bits and the sub-sampling factor increase. This increment reaches its maximum value of 12% for the *carphone* sequence and for a processor setup with only 4 representation bits and a pixel decimation of $2:1$, due to the presence of a lot of movement. Nevertheless, the values obtained for the remaining three video sequences, with less amount of movement, are quite smaller and similar to those ones obtained with the 4-step logarithmic search algorithm.

Consequently, from these results it can be concluded that the confi guration using a $2:1$ decimation $(S=1)$ and 4 representation bits $(T=4)$ presents the best tradeoff between hardware cost (a reduction of about 70%) and video quality. Moreover, this confi guration can be implemented by using the lower-cost XCV1000 FPGA, which has only about 40% of the total number of system gates provided by the XCV3200 FPGA.

5. Conclusion

New customizable core-based architectures are proposed herein to implement real-time motion estimation processors on FPL devices, such as FPGAs. The base core of these architectures is a new 2-D array structure for FSBM mo-

tion estimation that leads to an efficient usage of hardware resources, which is a fundamental requisite in any FPL based system. The proposed architectures consist of a wide range of processing structures based on the FSBM algorithm with different hardware requirements. The reduction in the amount of required hardware is achieved by applying decimation at the pixel and quantization levels, but still searching all candidate blocks of a given search area.

The proposed core-based architectures were implemented on FPGA devices from Xilinx and their performance was evaluated by including the motion estimation processors on a complete video encoding system. Experimental results were obtained by sub-sampling the block of the current frame with $2:1$ and $4:1$ decimation factors and by truncating 2 or 4 LSBs of the representation. From these results it can be concluded that a significant reduction of the required hardware resources can be achieved with these architectures. Moreover, neither the quality of the coded video is compromised nor the corresponding bit-rate is significantly increased. One can also conclude from the obtained results that the configuration using a $2:1$ decimation $(S = 1)$ and 4 representation bits $(T = 4)$ presents the best tradeoff between hardware cost (a reduction of about 70%) and video quality.

Acknowledgments

This work has been supported by the POSI program and the *Portuguese Foundation for Science and for Technology* (FCT) under the research project *Configurable and Optimized Processing Structures for Motion Estimation* (COSME) POSI/CHS/40877/2001.

References

[1] Vasudev Bhaskaran and Konstantinos Konstantinides, *Image and Video Compression Standards: Algorithms and Architectures*, Kluwer Academic Publishers, second edition, June 1997.

[2] T. Koga, K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro, "Motion-compensated interframe coding for video conferencing," in *Proc. Nat. Telecomm. Conference*, New Orleans, LA, November 1981, pp. G5.3.1–G5.3.5.

[3] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, volume COM-29, no. 12, pp. 1799–1808, December 1981.

[4] Nuno Roma and Leonel Sousa, "Efficient and configurable full search block matching processors," *IEEE Transactions on Circuits and Systems for Video Technology*, volume 12, no. 12, pp. 1160–1167, December 2002.

[5] Y. Ooi, "Motion estimation system design," in *Digital Signal Processing for Multimedia Systems* (edited by Keshab K. Parhi and Takao Nishitani), Marcel Dekker, Inc, chapter 12, pp. 299–327, 1999.

[6] L. Vos and M. Stegherr, “Parameterizable VLSI architectures for the full-search block-matching algorithm,” *IEEE Transactions on Circuits and Systems*, volume 36, no. 10, pp. 1309–1316, October 1989.

[7] B. Liu and A. Zaccarin, “New fast algorithms for the estimation of block matching vectors,” *IEEE Transactions on Circuits and Systems for Video Technology*, volume 3, no. 2, pp. 148–157, April 1993.

[8] E. Ogura, Y. Ikenaga, Y. Iida, Y. Hosoya, M. Takashima and K. Yamash, “A cost effective motion estimation processor LSI using a simple and efficient algorithm,” in *Proceedings of International Conference on Consumer Electronics - ICCE*, 1995, pp. 248–249.

[9] Seongsoo Lee, Jeong-Min Kim, and Soo-Ik Chae, “New motion estimation algorithm using adaptively-quantized low bit resolution image and its vlsi architecture for MPEG2 video coding,” *IEEE Transactions on Circuits and Systems for Video Technology*, volume 8, no. 6, pp. 734–744, October 1998.

[10] Z. L. He, K. K. Chan, C. Y. Tsui and M. L. Liou, “Low power motion estimation design using adaptative pixel truncation,” in *Proceedings of the 1997 international symposium on Low power electronics and design*, Monterey - USA, August 1997, pp. 167–171.

[11] Telenor, *TMN (Test Model Near Term) - (H.263) encoder/decoder - version 2.0 - source code*, Telenor Research and Development, Norway, June 1996.