Fully Parameterizable VLSI Architecture for Sub-Pixel Motion Estimation with Low Memory Bandwidth Requirements

Tiago Dias
ISEL - DEETC
Rua Conselheiro Emdio Navarro, 1, 1950-007 Lisbon, Portugal

Nuno Roma, Leonel Sousa
IST / INESC-ID
Rua Alves Redol, 9, 1000-029 Lisbon, Portugal

Abstract

This paper proposes a new scalable and efficient VLSI type-II architecture for real-time motion estimation optimized for sub-pel refinement algorithms. Based on the proposed architecture, which provides minimum latency, maximum throughput, and full utilization of the hardware resources, the implementation of a dedicated motion estimation coprocessor is also presented in this paper. This circuit is characterized by low memory bandwidth requirements, a modular and highly flexible structure and is capable of estimating motion vectors with half-pixel accuracy using the bilinear interpolation algorithm. Experimental results for implementations on ASIC and FPGA devices show that by using the proposed architecture it is possible to estimate motion vectors up to the 16CIF image format in real-time, with any given sub-pixel accuracy.

1. Introduction

Motion compensation (MC) is a fundamental technique used in most hybrid video coding systems to exploit the temporal redundancy between frames in video sequences [1]. Nevertheless, it is also the most computationally intensive task in video coding, involving up to 65% of the total computational resources of the video coder and requiring high power, throughput and memory utilization. All these factors are critical design metrics for most novel video applications, aimed at portable and battery supplied terminal devices using limited bandwidth communication channels. In such scenarios, modern video codecs not only must provide higher compression rates by computing better predictions for motion in video sequences, but also have to make use of efficient power management techniques in order to comply with the requirements of the new multimedia applications. As a result, modern video coding systems use Application-Specific Instruction-Set Processors (ASIPS) to do all the general data processing and include specially designed coprocessors to perform the motion estimation algorithm.

Among the several possible approaches, block matching [1] is the most popular method for motion estimation (ME) and despite the several search algorithms that have been proposed, the Full-Search Block-Matching (FSBM) ME algorithm [1] still remains widespread in hardware video coding applications, mainly due to its regularity and optimality. In this approach, the current frame is divided into blocks of pixels (macroblocks) and a motion vector (MV) is estimated for each macroblock. The MV coordinates define the displacement between the macroblock under processing and the best matching block of pixels defined within a search region in the previous frame using a given matching criteria. The Sum-of-Absolute-Differences (SAD) [2] is usually adopted for this criteria due to its simplicity and satisfactory results.

Although most video coding applications only estimate MVs with integer-pixel accuracy (IPA), i.e, the MVs point to candidate blocks spaced by an integer number of pixels, in both natural and synthetic video sequences the true frame-to-frame displacement of moving objects is seldom an integer number of pixels. Hence, in order to comply with the low bandwidth requirements of new multimedia applications, ME must be performed over a search area with greater pixel resolution so as to increase the accuracy of MVs and therefore fully exploit temporal redundancy. In most of these applications, MVs point to candidate blocks placed at half, quarter or eighth-pixel locations of the search area and can be computed using any of the existing ME algorithms, either at the whole sub-pixel level or by means of a hierarchical approach. Nevertheless, two-step approaches are often preferred, due

to their lower storage requirements and smaller number of computations, which for hardware implementations reduces the latency and the power consumption of the circuits. In this approach, the best matching block at integer locations of the search area is found in a preliminary step of the search procedure. Then, in a second step, the search area surrounding the selected integer resolution candidate block is interpolated into a higher resolution and the integer-pixel MV is refined into sub-pixel accuracy (SPA).

This paper proposes a new and highly scalable VLSI architecture for real-time ME, optimized for sub-pixel resolution. This innovative structure is characterized by a full and efficient use of the few hardware resources required for its implementation and has low power consumption and memory bandwidth requirements. Moreover, the proposed architecture only requires three sets of data for its operation: *i)* the initial coarser MV coordinates, that can be pre-computed in any other hardware or software application; *ii)* the search area pixels surrounding that location; and *iii)* the reference macroblock pixels. Consequently, it can be used both in hardware or hybrid software-hardware video coding systems, either to improve the accuracy of the ME process or to estimate local motion based on MVs predicted from previous frames.

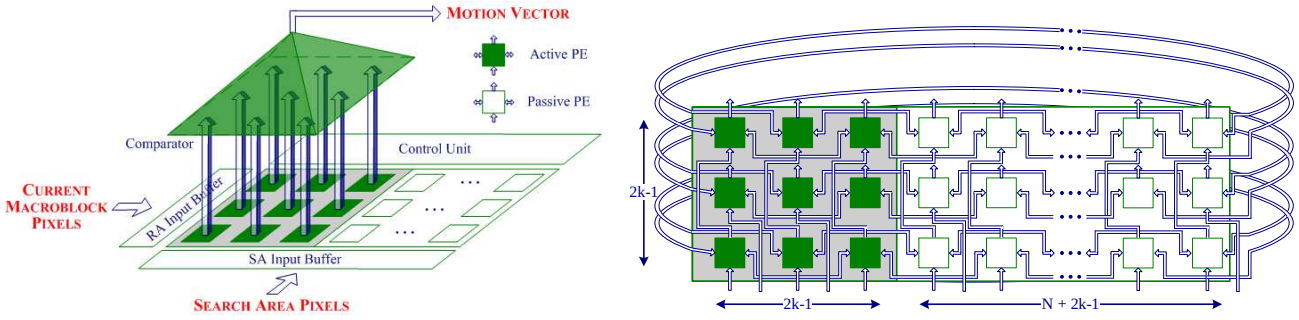
2. Improved Type-II Architecture for Sub-Pixel ME

ME with IPA is usually performed over a large search region to guarantee that the best MV is always found, regardless of the amount of movement present in the video sequence [1]. However, the refinement of such MVs into SPA involves only a small search region surrounding the best matching block at the coarser resolution in the search frame. As a result, the maximum displacement of the candidate blocks in each direction of the search area (p) is very small for search areas with sub-pixel resolution ($-k+1 \leq p < k-1$, where k denotes the SPA factor: $k = 2$ for half-pixel, $k = 4$ for quarter-pixel, and so on).

From a comparative analysis of the main array structures that have been proposed for ME in the latter years [3], it is possible to conclude that the type-II architecture [4] is one of the structures for ME that best minimizes the hardware requirements and provides a constant and relatively low processing time for small search ranges ($p \leq \frac{N}{2}$, where N represents the macroblock width). Consequently, this architecture proves to be well suited for refining the precision of IPA MVs into half ($p = 1$), quarter ($p = 3$) or even eighth-pixel accuracy ($p = 7$), considering macroblocks with 8×8 ($N = 8$) or 16×16 pixels ($N = 16$), the more frequently adopted parameters by the ITU-T H.26x and ISO MPEG-x video coding standards. Nevertheless, since the type-II architecture was firstly designed to estimate IPA MVs, several changes have to be performed in its datapath and control unit so that it can be used in sub-pixel refinement algorithms. Thus, the proposed SPA ME architecture is based on the original type-II architecture [4], but presents significant improvements in what concerns its structure, as well as several optimizations that improve its efficiency in terms of hardware resources, power consumption and throughput.

A. Datapath

In the proposed type-II architecture, whose generic block diagram is depicted in Fig. 1(a), the cost functions for all candidate blocks are computed in parallel in a systolic processing array. Consequently, after N^2 clock cycles all pixels of the reference area (RA) have been processed and $(2k-1)^2$ cost functions become simultaneously available at the output of the array. To obtain the optimum MV, i.e., the one holding the lowest cost function, this set of values is subsequently compared against each other in a dedicated comparison unit. Such unit consists



(a) Processing structure. (b) PEs interconnections considering half-pixel accuracy ($k = 2$).

Figure 1. Proposed type-II architecture for ME with sub-pixel accuracy.

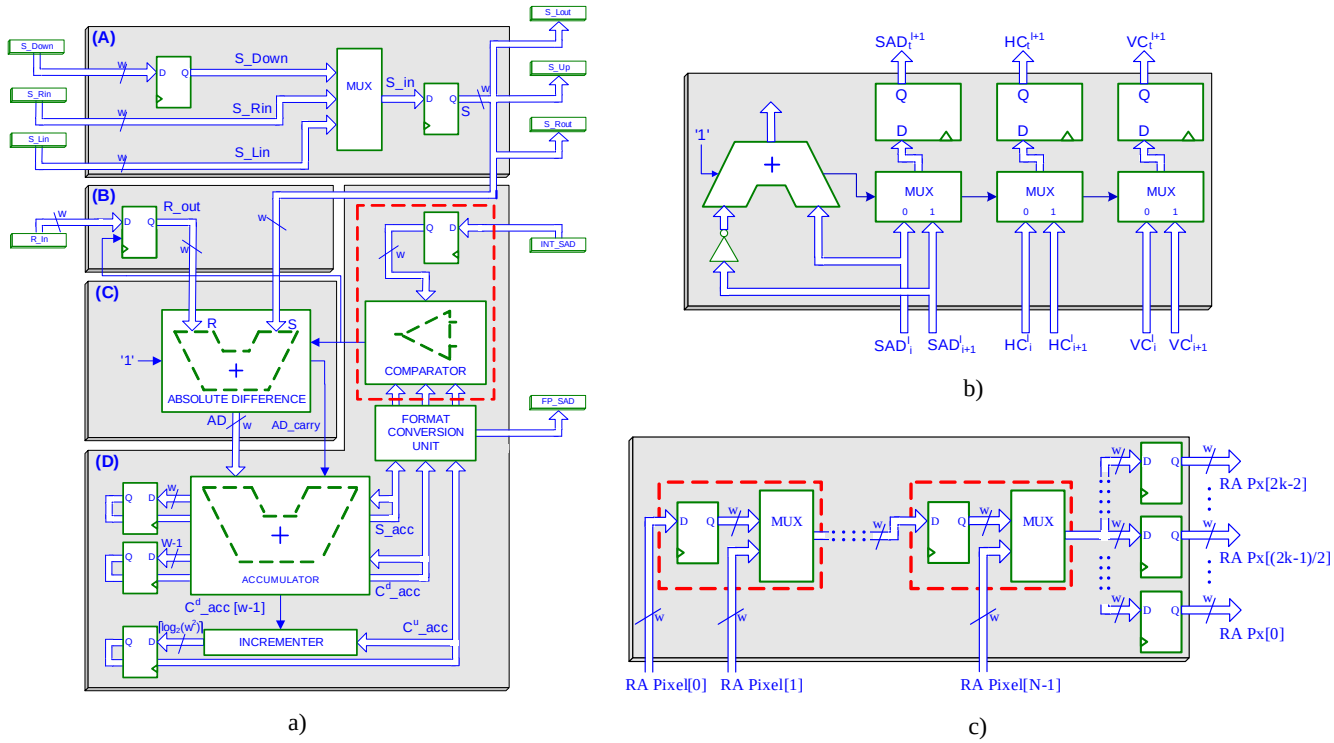


Figure 2. ***a)*** **Active processor element circuit: (A) SA displacement unit; (B) Reference pixel storage unit; (C) Absolute difference arithmetic unit; (D) Accumulation and format conversion unit.** ***b)*** **PE of the comparison unit.** ***c)*** **RA input buffer.**

of a binary tree structure with $(2k-1)^2$ inputs, optimized to minimize both its processing time and its hardware requirements. Each level of the binary tree structure is composed by a set of two input comparators based on Sklansky prefix-adder structures, that compare two cost function values obtained from the level bellow and output the lowest one to the level above. The coordinates of the MV corresponding to the highest similarity measure are also bypassed to the level above using $2:1$ multiplexers, as shown in Fig. 2(b).

The processing array is composed by two sets of processing elements: the *active processing elements (PEs)* and the *passive PEs*. The passive PEs are mainly composed by data registers and are used to store and displace the search area (SA) pixels of the candidate blocks within the processing array, therefore avoiding redundant access to the main picture memory and significantly reducing the memory bandwidth requirements of the architecture. The internal structure of a passive PE is the same as the SA transfer circuit of an active PE (block A in Fig. 2(a)). The active PEs are composed by four main blocks, as shown in Fig. 2(a), and are used to compute the cost function for a given displacement vector. On each processing cycle all active PEs are fed with the same pixel of the reference macroblock and a different SA pixel corresponding to the same relative location for each of the $(2k-1)^2$ candidate blocks being processed. The SA pixel used in the computation of the distance measure is selected from the set of three pixels supplied by the PEs located below, on the left and on the right of the considered active PE and is transferred to an internal standing-data register. In contrast, the RA pixel is transferred from the running-data registers that feed the input lines of the active PEs to the standing-data register inside the active PEs. The absolute difference arithmetic unit computes the absolute difference value between the reference and search area pixels and the obtained result is added to the partial sum computed in the previous clock cycles in the accumulation unit. To reduce the power consumption of the proposed structure, the accumulation unit also includes a power-saving module that dynamically controls the computation of the cost function. This circuit consists on a standing-data register to store a threshold value, such as the cost function of the best integer-pixel candidate block obtained so far, and comparison circuitry to disable this computation whenever the partial accumulated value exceeds the threshold one. In such situation, the threshold value is outputted by the active PE and fed into the binary tree comparator for comparison with the remaining cost function values.

To guarantee that the SA pixels are displaced over a sampling grid with $\frac{1}{k}$-pixel resolution, all PEs of the proposed structure are connected to each other in an interleaved manner in groups of N in the horizontal direction and in groups of k in the vertical direction, being spaced apart from $k-1$ PEs in both directions, as shown in

Fig. 1(b). Unlike other architectures [5], in the proposed structure there are few PEs in each line of the processing array ($N + 3$ PEs for half-pixel accuracy), and thus the delays associated to the longer interconnections at the vertical borders of the array do not greatly influence the critical path of the circuit. To avoid redundant passive PEs, and therefore minimize the hardware requirements of the proposed architecture, the passive PEs located in the right margin of the processing array are also connected to the active PEs in its left margin, creating a cylindrical structure. As a result, within the processing array SA pixels are displaced in three different directions: upwards, to the right and to the left. Whenever a set of k lines of SA pixels is fed into the array through its lower inputs, all SA pixels within the structure are simultaneously shifted one position upwards. In the subsequent $N - 1$ clock cycles SA pixels are shifted to the right, one position per clock cycle. As soon as all these $N - 1$ shift-right operations have been carried out, the SA data is shifted upwards once again and a new set of k lines of SA pixels is fed at the bottom of the array through its lower inputs. During the next $N - 1$ clock cycles, SA data is shifted to the left in a similar manner as described above, being shifted upwards after the $(N - 1)^{th}$ clock cycle. Hence, by using the adopted cylindrical structure and this zig-zag processing scheme, it is possible to maximize the efficiency of the proposed architecture: all active PEs are kept busy at any time instant and it avoids both redundant accesses to the frame memory and the need for dummy clock cycles between any two adjacent rows of SA pixels. Nevertheless, the usage of such zig-zag processing scheme requires the use of dedicated input buffers in order to maintain data consistency inside the array.

The reference and search area pixels are loaded into the processor array by means of one parallel-in-serial-out (PISO) and one parallel-in-parallel-out (PIPO) input buffers, respectively. The purpose of the PIPO buffer is to hold the line of SA pixels being fetched from the picture memory, either in line-scan or block-scan mode [4], until these pixels can be transferred, in parallel, to the processing array. In contrast, the PISO buffer stores the RA pixels fetched from the picture memory and transfers them serially to each of the $(2k - 1)^2 - 1$ active PEs. Furthermore, to avoid broadcasting the RA pixels to all active PEs in the array, the RA input buffer was designed using N cascaded registers that are loaded in parallel with an entire line of reference area pixels, and are transferred serially to the active PEs through a set of $2k - 1$ redundant registers, as depicted in Fig. 2(c). Each of these output registers contains a copy of the pixel value to be fed to the active PEs, but is only responsible for feeding the active PEs of a specific line of the processing array, thus minimizing the required fan-out for the RA input buffer.

B. Control Unit

The signals which control all blocks of the proposed type-II structure are generated by a central control unit. This unit consists of three individual and simpler blocks that are synchronized by a reduced number of signals: a SA line counter, a reference pixel counter and a state machine. The SA line counter and the reference pixel counter units monitor the loading of SA lines and RA pixels to the array, respectively, signalling the state machine unit whenever the last line of the SA or the last pixel of a RA line is loaded. The state machine controls the data flow inside the array and the selection of the best MV inside the comparison unit. This controller was implemented by means of a nine state Moore sequential circuit to guarantee that the output signals of the control unit do not depend on its input signals, thus minimizing the critical path of the control circuit and, consequently, optimizing the performance of the global architecture.

3. Implementation and Experimental Results

To evaluate the performance of the proposed architecture, a sub-pixel refinement ME coprocessor was developed and embedded in a hybrid hardware-software video coding system [6]. This coprocessor, whose simplified block diagram is depicted in Fig. 3, consists of a control circuit and four functional units (the ME architecture described in Section 2.; an interpolation unit that implements the bilinear interpolation algorithm using a high throughput 4-tap FIR filter with low hardware requirements [6]; and two input buffers) interconnected in a pipeline processing scheme. The circuit is fed with the coordinates of an IPA MV pre-computed in the software module of the video coder, as well as with all the required reference and search area pixels, and refines such coordinates into half-pixel accuracy, outputting them back to the software module.

The proposed sub-pixel ME coprocessor was completely described using both behavioural and fully structural parameterizable IEEE-VHDL. Several setups of these descriptions were synthesized and implemented in

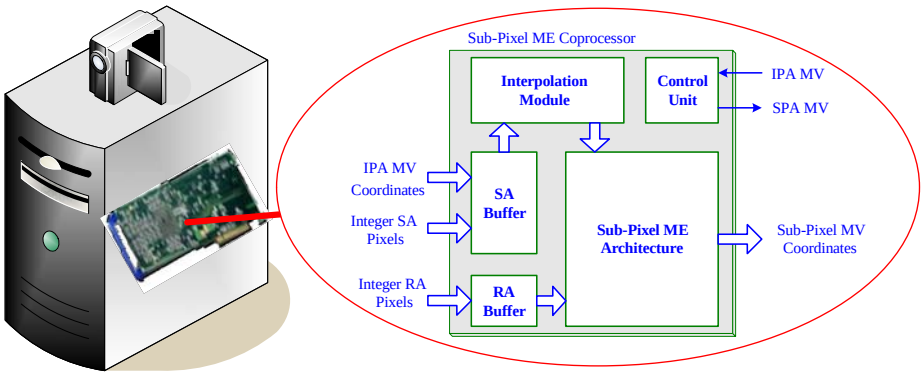


Figure 3. Block diagram of the proposed ME coprocessor.

ASIC and FPGA considering the typical parameters currently adopted in videoconferencing applications. The implementation setup presented in this paper considers 8-bit to represent pixel values and macroblocks with 8×8 ($N = 8$) and 16×16 pixels ($N = 16$), the set of parameters more frequently adopted by the ITU-T H.26x and ISO MPEG-x video coding standards.

A. FPGA based implementation

Table 1(a) presents the experimental results obtained for the two considered implementations using a *Xilinx* Virtex-E XCV3200E-7 [7] device, and the *Xilinx* synthesis tool from ISE 6.1.3i. The hardware resources required by the proposed sub-pixel ME coprocessor were assessed in terms of the percentage of CLB slices and LUTs that were required for each implementation. Such results evidence the very low hardware requirements of the proposed ME architecture. Moreover, considering that in the proposed structure the pixel rate equals the clock rate, the maximum operating frequency values depicted in Table 1(a) evidence that the proposed architecture is able to refine MVs into half-pixel accuracy in real-time for the 4CIF image format. By taking into account the pipeline systolic structure of the processing array, one can also predict that more accurate MVs can be obtained by merely increasing the number of PEs in the array, which will only change the latency of the circuit due to an augment in the number of levels of the binary tree comparator. Consequently, from the experimental results presented in Table 1(a) it can be concluded that the proposed sub-pixel ME architecture is able to estimate MVs with any given pixel accuracy in real-time for any x-CIF image format, with a minor increase of its latency.

B. ASIC based implementation

The implementation in an ASIC was performed using *Synopsys* synthesis tools and a high density StdCell library from *UMC*, which is based on a $0.13\mu m$ CMOS process from *Virtual Silicon Technology Inc* [8]. Table 1(b) summarizes the main characteristics of the implemented circuits for the two setups mentioned above. The obtained results evidence that by using the ASIC technology it is possible to operate at a frequency that is about 2.8 times faster than the one obtained with the FPGA, thus allowing the estimation of MVs with sub-pixel accuracy in real-time for any x-CIF image format.

Experimental results concerning the power consumption of the proposed sub-pixel ME coprocessor operating at the normalized frequency of 100MHz were also obtained using the *Synopsys* synthesis tools. The obtained values, summarized in Table 1(b), correspond to the worst case situation when all active PEs are kept computing the cost function for all N^2 iterations. Nevertheless, this scenario hardly ever happens in practice due to the power-saving control circuitry embedded in each active PE, that disables the computation of the cost function whenever the partial value exceeds the SAD value obtained for the coarser MV (the threshold value). In such situation, the power consumption of an active PE becomes equal to the one of a passive PE for the remaining iterations of the algorithm and therefore depends greatly on the video sequence being coded. To estimate the percentage of iterations avoided in the computation of the SAD measure, several QCIF benchmark video sequences, characterized by different spatial detail and amount of movement, were coded in interframe mode using half-pixel accuracy and the H.263 video encoder provided by Telenor R&D [9]. From Table 1 and Fig. 4, which depicts the estimates of the power saving rates obtained for six benchmark video sequences, one can conclude that not only does the proposed sub-pixel ME coprocessor provides significant power saving rates, but also that it imposes low power

Table 1. Implementation results for the ME coprocessor.

N	F $[MHz]$	# Slices	# LUTs
8	141.4	949	1736
16	138.8	1712	3197

(a) Using the *Xilinx* XCV3200E-7 FPGA device.

N	F $[MHz]$	Area $[\mu m^2]$	Power $[mW]$
8	408.16	265768.46	399.65
16	393.70	621455.86	801.57

(b) Using the StdCell library from *UMC*.

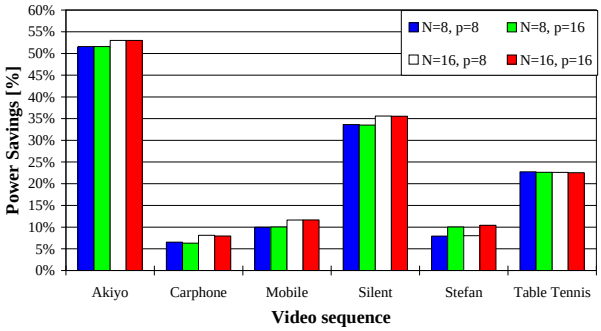


Figure 4. Power savings obtained by the ME coprocessor.

constraints when compared with other ME structures [10].

4. Conclusion

An innovative type-II architecture for real-time ME was presented in this paper. This architecture is fully parameterizable and optimized for sub-pixel ME, allowing the estimation of sub-pixel accurate MVs with any given accuracy in real-time. Moreover, due to its innovative cylindrical architecture and zig-zag processing scheme, the proposed structure makes a full and efficient use of the hardware resources and has maximum throughput and minimum hardware requirements.

The proposed structure was used as the basis for a dedicated ME coprocessor that provides sub-pixel accuracy. This circuit has a highly modular and flexible architecture that allows the development and integration of new modules and can be used both in hardware or hybrid software-hardware ME systems to refine the coordinates of MVs into sub-pixel resolution. Furthermore, it avoids redundant accesses to the main picture memory by being capable to retrieve reference and search area pixels from both local memories and the main picture memory.

Experimental results obtained for implementations on ASIC and FPGA devices of the proposed coprocessor using the more frequently adopted parameters of the ITU-T H.26x and ISO MPEG-x video coding standards, demonstrate that by using the proposed circuit in hybrid video coding systems it is possible to estimate MVs with half-pixel accuracy up to a rate of 30 fps for high-quality video (16CIF format).

Acknowledgment

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] P. M. Kuhn and K. P. M., *Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.

[2] S. Vassiliadis, E. A. Hakkennes, J. S. S. M. Wong, and G. G. Pechanek, "The sum-absolute-difference motion estimation accelerator," vol. 2, Aug. 1998, pp. 559–566.

[3] P. Pirsch, N. Demassieux, and W. Gehrke, "VLSI architectures for video compression - a survey," *Proceedings of the IEEE*, vol. 83, no. 2, pp. 220–246, Feb. 1995.

[4] L. Vos and M. Stegherr, "Parameterizable VLSI architectures for the full-search block-matching algorithm," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 10, pp. 1309–1316, Oct. 1989.

[5] N. Roma and L. Sousa, "Parameterizable hardware architectures for automatic synthesis of motion estimation processors," in *Proceedings of the IEEE Workshop on Signal Processing Systems - Design and Implementation (SiPS'01)*, Antwerpen - Belgium, Sept. 2001, pp. 428–439.

[6] T. Dias, "Dedicated processors for motion estimation in video sequences," Master's thesis, Instituto Superior Técnico, Lisbon, Sept. 2004.

[7] *Virtex-E 1.8V Field Programmable Gate Arrays Datasheet*, v2.3 ed., Xilinx Inc., July 2002.

[8] *UMC High Density Standard Cells Library - 0.13μm CMOS process*, v2.3 ed., Virtual Silicon Technology Inc., Dec. 1999.

[9] Telenor, TMN (Test Model Near Term) - (H.263) encoder/decoder (source code), v2.0 ed., Telenor Research and Development, June 1996.

[10] S. Yang and W. Wolf, "Power modeling of motion estimation VLSI architecture," in *Proceedings of the The 36th International Symposium on Microarchitecture (MICRO-36)*, San Diego, CA - USA, Dec. 2003.