FPGA ARCHITECTURE FOR REAL-TIME VIDEO NOISE ESTIMATION

François-Xavier Lapalme, Aishy Amer, and Chunyan Wang

Concordia University , Electrical and Computer Engineering
Montréal, Québec, Canada
email: {f_lapalm,amer,chunyan}@ece.concordia.ca

ABSTRACT

This paper proposes a hardware architecture of a video noise estimation algorithm capable of real-time processing. The objectives consist of adapting a computationally demanding noise estimation algorithm to a synthesizable VHDL implementation and achieving real-time performance. This Structure-oriented noise estimation method considers image structure to find intensity-homogeneous blocks. Subsequently, these blocks are included in the averaging process to estimate the noise variance. Generating worst-case estimation error of 3 dB, this spatial noise reduction method is reliable for highly noisy and textured images. The proposed architecture provides a satisfactory compromise between area and processing speed. Furthermore, parameterization of the architecture allows additional flexibility with the scaling of mask sizes that can operate on 3x3 or 5x5 blocks of pixels. The proposed design is targeted to an FPGA device and estimates the noise variance over an interlaced PAL video sequence.

Index Terms— Video signal processing, Field programmable gate arrays, White noise, Estimation

1. INTRODUCTION

1.1. Motivation

Digital video processing algorithms, such as noise estimation, are computationally intensive. The algorithm's performance worsens dramatically as image resolution and pixel data size grow larger. Effective techniques are required to contend with this shortcoming in performance. One solution is to make use of a fast-prototyping, flexible and reprogrammable Field Programmable Gate Array (FPGA) technology.

The current high density FPGAs offer compelling platforms for hardware acceleration of software-based DSP algorithms. Moreover, recent technological development in the field of FPGAs has increased the logic density and clock speeds to comparable ASIC performances at a much lower cost. However, complex algorithms implemented on resource constrained systems can cause serious speed and quality downgrades. Hence, in depth analysis and adaptation of the algorithms is required to attain real-time performances.

Noise can significantly effect the performance of video processing algorithms. A noise estimation technique calculates the level of white noise (e.i., which is the most commonly assumed noise type in video processing applications [1, 2, 3]) contained in a corrupted video signal. When noise variance becomes available, video processing algorithms (e.g., noise reduction or motion estimation) can be adapted to the amount of noise for improved performance [1]. This paper proposes an efficient and flexible FPGA architecture of a fast and reliable spatial video noise estimation technique [1].

The remainder of Section 1 discusses related work and introduces the noise estimation algorithm. Section 2 describes the hardware architecture and Section 3 illustrates the simulations and details the synthesis results. Section 4 concludes this paper and outlines future work.

1.2. Related Work

Video noise can be estimated spatially or temporally. A widely used spatial noise estimation method calculates the variance (as a measure of homogeneity) over a set of image blocks and averages the smallest block variance as an estimate of the image noise variance [2]. Spatial variance-based methods tend to overestimate the noise in less noisy images and underestimate it in highly noisy and textured images. Therefore, measures other than the variance were introduced in [1] to determine homogeneous blocks. Temporal noise estimation [4] evaluate noise using motion information. Such approach is very expensive for hardware implementations with estimation accuracy not significantly more precise than spatial methods. Thus, for the FPGA implementation described in this paper, we select the spatial method in [1] which outperforms the other noise estimation methods presented in [2] and [4].

To the best knowledge of the authors, no literature exist to FPGA architecture to video noise estimation. Closest hardware related work include [5],[6] and [3]. In [6], to eliminate additive noise, an iterative image restoration method demonstrates the complexity of mapping DSP algorithms onto programmable logic. In [6], it is shown that resource planning and proper algorithm adaptation help find the optimal FPGA design with respect to the trade-off between run time and result quality. Nonetheless the adaptation effort set forth in [6] to reach high quality results lack the speed to yield real-time performance.

The paper in [5] concentrates on a spatial-based adaptive and reusable hardware architecture for image enhancement. The mean and the variance are calculated over an entire frame and used in a transformation function (histogram) to enhance the image. The reusability is realized through the use of VHDL generics where image resolution can be easily modified without changing the RTL behavior while still being capable of producing real-time results.

In [3], a combined spatial-temporal 3D noise filter IC for television receivers uses a noise estimation circuit to improve its efficiency. ASIC's costly static implementations can support temporal filters design due to their greater memory capacity. Although memory is limited, reconfigurable FPGA chip development is less expensive and its flexibility is well suited for spatial filter design.

In this paper, we propose an FPGA architecture for video noise estimation where contributions include i) defining a computation procedure for spatial noise estimation that can be effectively be synthesized for FPGA implementation, ii) developing a flexible scalable

architecture with such attributes as various filter sizes to adaptively estimate noise in good quality and highly structured images, and iii) successfully developing a FPGA-oriented architecture capable of reaching real-time processing performance.

1.3. Structure-oriented noise estimation

The noise estimation algorithm used is a fast, reliable, automated method for estimating the variance of additive white noise in video frames [1]. This technique finds homogeneous regions in the noisy images and estimates the noise variance by taking image structure into account. It sub-divides the frame into evenly sized blocks. Eight high-pass operators measure the high-frequency image components to calculate the homogeneity measure, ξ_{Bh}. Illustrated in Eq. 1 is the mask of the horizontal direction scanning window for 5x5 pixel blocks. The operators compensate for the noise along eight directions and stabilize the selection of homogeneous blocks. The goal is to find the noise variances of blocks showing similar homogeneities.

$$I_o(i) = -I(i-2) - I(i-1) + 4 \cdot I(i) - I(i+1) - I(i+2). \quad (1)$$

The sample mean, μ_{Bh}, calculated in Eq. 2 for each block, is used to generate the variance, σ^2_{Bh} (Eq. 3), for each individual block. The 10% of the sorted block variances with respect to the block's homogeneous measure are selected. This intricate sorting and block selection procedure is the main complexity of this noise estimation algorithm. The middle row in Fig. 1 represents the elements $(\xi_{Bh_1}, \xi_{Bh_2}, \cdots, \xi_{Bh_n}$ where n is the total number of blocks in an image field[1]) by which the blocks are sorted. The complexity resides in the selection of the corresponding variances (seen in the last row $\sigma^2_{Bh_{223}}, \sigma^2_{Bh_{5809}}, \cdots, \sigma^2_{Bh_{4199}}$) used to generate the final variance.

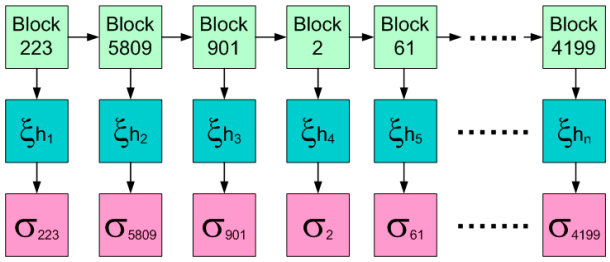


Fig. 1. Homogeneous measure sorting.

A reference variance, σ^2_{REF}, is calculated from the median of the variances of the three most-homogeneous blocks. Valid variances are selected by the difference between the logarithmic value of the reference variance and the individual block variances. This difference is compared to a threshold, t_σ (Eq. 4), which determines the similar homogeneities between blocks and is chosen as the maximum-affordable difference between the true variance and the estimated variance [1]. The frame variance, σ^2_n, is produced from the average sum of the valid variances.

$$\mu_{Bh} = \frac{\sum_{(i,j)\in W_{ij}} I(i,j)}{W \times W}. \quad (2)$$

$$\sigma^2_{Bh} = \frac{\sum_{(i,j)\in W_{ij}} (I(i,j) - \mu_{Bh})^2}{W \times W}. \quad (3)$$

$$|log(\sigma^2_{Bh}) - log(\sigma^2_{REF})| < t_\sigma. \quad (4)$$

[1]For a 5x5 and 3x3 filter configuration, $n \approx 8200$ and $n \approx 23000$ blocks per field, respectively.

2. PROPOSED FPGA ARCHITECTURE

2.1. Constraints analysis

Analyzing the noise estimation algorithm with respect to the logic resource availability is imperative to establish whether a hardware design can be realized onto an FPGA platform. Thus, the adaptation of the algorithm requires several modifications to conform to the VHDL hardware description language limitations and to comply with the resource constraints of an FPGA device.

The first compromise in accurately implementing the noise estimation algorithm in [1] consists of reducing the precision of the algorithm from floating-point to fixed-point arithmetic. The fixed-point calculations that use multiplying, dividing and squaring functions produce truncated fractional parts. Nevertheless, this study shows that the quality of hardware-based results in comparison to the software-based results are almost similar (Fig. 3 part c)); furthermore, the hardware-based results present a worst-case estimation error of less than 3 dB with respect to the noise level introduced in the frames, which is suitable for video broadcasts [1].

The second compromise consists of implementing the logarithmic function (Eq. 4) using a look-up table. Since logarithmic series have an exponential order of complexity, it would require excessive amount of logic to implement. This technique consequently reduces the accuracy of the results because the look-up table logarithmic values are abbreviated due to FPGA memory constraints. Since the accuracy level is directly proportional to the size of the look-up table a concession was approved to use 16 bit logarithmic data values which spawned acceptable results with respect to the software results (Section 3).

The main bottleneck in terms of timing in this algorithm was the sorting of the homogeneous measures in real-time (Fig. 1). Attempts were made to eliminate the sorting step of the algorithm but the quality of the results were significantly affected. Popular sorting algorithms were evaluated (e.g., *bubble sort* and *quick sort*) that process data lists in a time-consuming sequential fashion by comparing and swapping data positions or by using recursiveness. Alternatively, a hardware-oriented *counting sort* algorithm best suited to order large data lists quickly (see footnote for relative size of lists) was implemented. Thus, this algorithm sustained real-time performance by sorting the block variances in less than a time frame using four distinct sequential steps and a single internal BRAM memory. The development of this optimum architecture is described in the following section.

2.2. Design architecture

This section describes the proposed design's architecture; the data flow and internal module description. The top-level view is illustrated in Fig. 2. This architecture is designed to process two consecutive fields simultaneously by acquiring data within Module A and by processing data within Module B and C. This parallel processing concept is designated as a "ping-pong" structure. The term "ping-pong" refers to the back and forth processing scheme that is used to enable rapid parallel processing. Thus, simultaneous but separate accesses to two external memory components (Bank 1 and 2) are being used to store and read data concurrently.

The 8 bit pixel data make up the 720x288 video input field. The frame rate is 27 MHz according to the ITU-R BT 601 standards. The noise estimation output is 22 bits of fixed-point data which includes a 16 bit decimal part and 6 bit fractional part. The final noise variance output is latched on a 22 bit bus which can be used in, e.g., noise filters [3] or object segmentation [7].

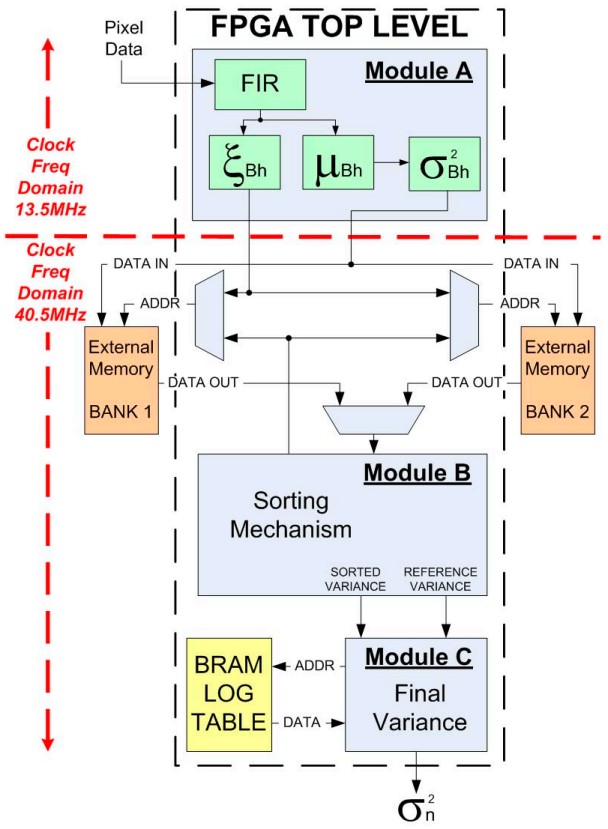


Fig. 2. Block Diagram of the proposed FPGA Design.

Module A of Fig. 2 generates the variance, σ^2_{Bh}, the sample mean, μ_{Bh}, and the homogeneity measures, ξ_{Bh}, of each block contained in a field. Each block make up the 5x5 windows that subdivide the fields. The FIR filter samples the input pixels to generate the block pixels. These blocks of pixel data are then used to produce the homogeneous measure, sample mean and variance. This module sends the generated data to external memory banks via large multiplexers that determine which field is being processed. Module B is responsible for sorting the homogeneity values of the previous field. This sorting mechanism is controlled by an FSM which builds, in a sequential fashion, the list of variances with respect to the ordered homogeneity measure data list. These variances are stored in internal BRAMs where only the 10% most homogeneous blocks are kept. Module C works to find the logarithmic value of the variances by querying the logarithmic look-up table. The logarithmic values are compared to an application-dependent threshold as seen in Eq. 4. The remaining valid variances are added and averaged to generate the final variance, σ^2_n. This design is uniformly pipelined, DCMs are used to accelerate the sorting mechanism's clock frequency, and maximum parallelism of arithmetic operations (e.g., "ping-pong" structure) is applied to help reach real-time. Furthermore, the proposed architecture is scalable in that it can accommodate different filter size configurations (e.g., 3x3 and 5x5) through the use of VHDL generics and packages.

3. PROTOTYPING RESULTS

Commonly referenced video sequences (*Train*, *Flower_Garden* and *Prlcar*) were applied to the proposed design. These sequences contain complex image structure and motion. The tests include a range of different noise levels which are evaluated on the design's 3x3 and 5x5 filter configurations.

Simulations were done using *Synopsys VHDL Simulator*^{TM} tool. The simulator tool acquired pixel from PAL video files of 60 frame. Part a) and b) of Fig. 3 illustrates the average hardware noise estimation results using a 5x5 filter hardware configuration compared to the original software results for video sequences with noise levels of 20dB and 40dB PSNR. Part c) presents the difference between software-based and hardware-based PSNR results.

The synthesis was done using *Synopsys Design Compiler*^{TM} tool and the place and route was carried out with the *Xilinx Project Navigator*^{TM} tool. This design was implemented on a Virtex II XC2V4000 FPGA. Its resource consumption is illustrated in Table 1. As can be seen, only 10% of LUTs and only 20% of SLICEs are being used. As many as 60% of BRAMs are being used to construct the log look-up table.

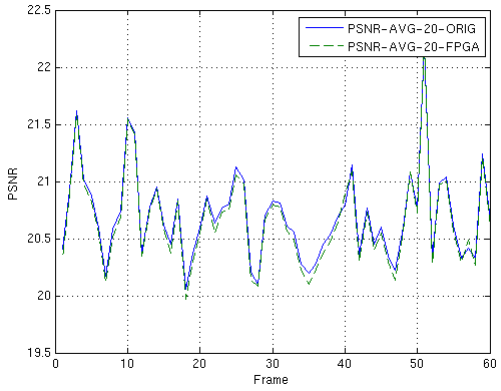
Type	FF	LUT	Slices	BRAM	MULT18X18
Percent	14%	10%	20%	60%	20%

Table 1. Resource utilization for XC2V4000 Xilinx FPGA.

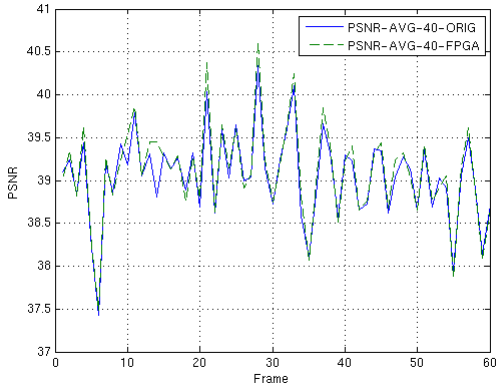
The Virtex XC2V4000 chip has been chosen because it contained the required amount of internal BRAM memory needed to build the look-up table, the line delays (FIR filter) and the FIFOs (sorting module). Hence 60% of the BRAM resources have been utilized while only a fraction of the Configurable Logic Blocks (CLB) are consumed. Small area consumption is advantageous in that the remaining slices can be employed to upgrade the data formats or to incorporate other processing algorithms (e.g., video noise reduction or motion estimation) that use noise estimated variance values. The pipelined design is portrayed in time in Fig. 4 to fully illustrate the timing distribution of the various processes. The time that is consumed by the sequential sorting algorithm is clearly shown as the bottleneck of the proposed design. Knowing that the proposed design processes two fields at a time (current and previous fields are processed concurrently), the variance is only required to be generated before the input of the third field. Hence, from this figure, the real-time performance is validated (designated by the positive slack) as it can be seen that the first field's variance is generated before the third incoming field.

4. CONCLUSION

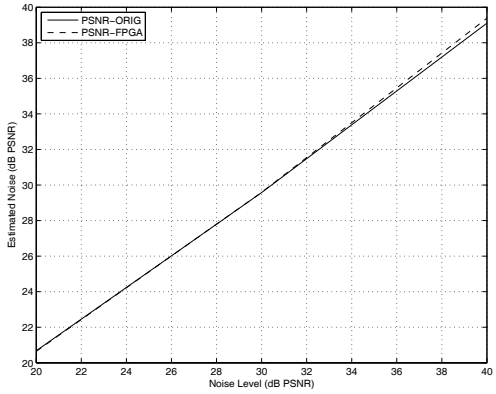
This paper introduced the realization of a noise estimation implementation onto an FPGA. The main purpose of attaining real-time processing speeds was met. The algorithm [1] has been successfully adapted into a synthesizable hardware implementation. Some design feasibility issues had to be overcome in the transformation of the original algorithm. Thus, this noise estimation has reached real-time performances by efficiently pipelining the VHDL design, promptly parallelizing the architecture and adequately adapting the software algorithm. The noise variance produced was proved to be accurate compared to the original software results. Finally, the scalability of modules that allow different filter sizes adds flexibility to the implementation.



(a) Noise level : 20 dB PSNR



(b) Noise level : 40 dB PSNR



(c) Difference between Software-based and Hardware-based PSNR results

Fig. 3. Software versus hardware implementation; average estimated PSNR using 5x5 filter size.

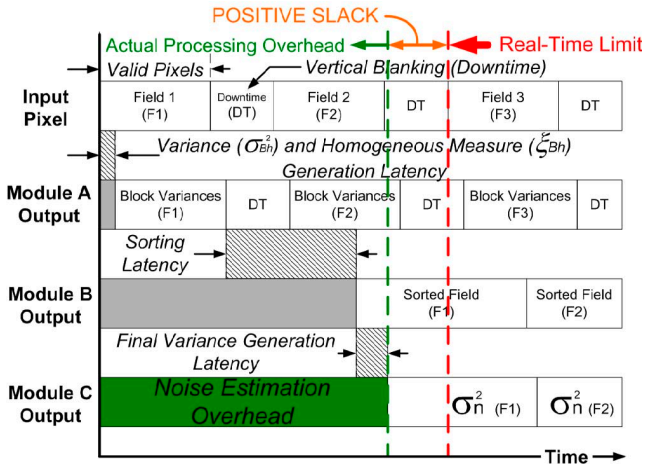


Fig. 4. Design Timing Diagram.

Future work includes speeding up the processing time by increasing the number of pipeline stages and by using external memory to implement the logarithmic look up table.

5. REFERENCES

[1] A. Amer and E. Dubois, "Fast and reliable structure-oriented video noise estimation," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 15, no. 1, pp. 113–118, Jan. 2005.

[2] S.I. Olsen, "Estimation of noise in images: An evaluation," *CVGIP: Graphical Model and Image Processing*, vol. 55, no. 4, pp. 319–323, July 1993.

[3] G. de Haan, O. A. Ojo, and T. G. Kwaaitaal-Spassova, "Automatic 2D and 3D noise filtering for high-quality television receivers," in *Proc. of 7th Int. Work. on HDTV*, Turin, Italy, Oct. 1994, Session 4B.

[4] B. C. Song and K. W. Chun, "Noise power estimation for effective de-noising in a video encoder," *IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 357–360, March 2005.

[5] U. Bidarte, J.A. Ezquerra, A. Zuloaga, and J.L. Martin, "VHDL modeling of an adaptive architecture for real-time image enhancement," in *Proc. of VHDL International Users Forum (VIUF)*, Orlando, FL, USA, Oct. 1999, pp. 94–100.

[6] S. Memik, A. Katsaggelos, and M. Sarrafzadeh, "Analysis and FPGA implementation of image restoration under resource constraints," *IEEE transactions on Computers*, vol. 52, no. 3, pp. 390–399, Mar. 2003.

[7] A. Amer, "Memory-based spatio-temporal real-time object segmentation," in *Proc. SPIE Int. Conf. on Real-Time Imaging*, Santa Clara, California, USA, Jan. 2003, vol. 5012, pp. 10–21.