A highly flexible high-speed camera system with real-time noise cancellation

Dušan Majstorović[1], Boris Radin[2]

Abstract – **This paper describes embedded system which presents high definition high speed camera based on commercially available CMOS sensor. The main emphasis is on proposed noise cancelling functionality. The paper gives short introduction to CMOS sensor noise characteristics and presents in detail the proposed approach to noise cancelling in CMOS based high speed camera applications with result analysis report. Finally, implemented hardware architecture is presented with emphasis on the noise cancelling block.**

Keywords – **High speed camera, FPGA, noise cancelling, embedded system.**

I. INTRODUCTION

In current scientific research, as well as in many commercial applications, there is a growing need for high speed imaging. Also, certain amount of flexibility is expected in terms of frame rate, region of interest and exposure range. Such applications are very demanding on sensor sensitivity, windowing capabilities and readout speed. In practice, most high speed applications are based on CMOS imagers. Among other qualities such as low power consumption and operation with single power supply, they offer relative ease of parallel readout structures and region-of-interest capabilities. Despite it's strengths in high speed applications, CMOS technology is still behind the CCD sensor technology when it comes to noise performance. In certain operational points, noise generated by CMOS sensor can significantly degrade the image quality, which is especially notable in applications that require wide range of sensor parameter settings.

Today, sensors intended for use in high speed applications can produce amount of data per second measured in gigabytes. This is by far faster than today's non-volatile memories (hard disk, flash) can write. For that reason, frame buffers are usually implemented using DRAM storage capacities. Usage of dynamic memories for storing the data significantly limits the maximum time of interrupted recording in most applications. Storage capacity is implemented either on camera itself or on a PC based frame grabber. In this case, data is usually transferred to the PC using camera-link interface that can support data rates up to 680MB/s.

FPGA devices are usually used for sensor control and data acquisition. Today, FPGA technology can offer not only highly flexible interfacing capabilities, but also significant processing power comparable to the state of the art general-purpose processors. Certain FPGA families even feature built-in processing blocks making those perfect candidates for complete SoC (*System on Chip*) solutions. Together with matching software tools, this feature provides an easy way for developing powerful and flexible embedded systems. This approach is taken in developing of the described system.

In high-speed camera applications, usually, large amounts of image data are captured. That fact makes offline PC-based image processing highly time demanding. That was the motive for proposing the real-time noise cancelling hardware function to be implemented in the FPGA. With this approach, image processing is done before the images are stored in camera frame buffer with only minimal additional delay.

II. NOISE CANCELLING ALGORITH DESCRIPTION

Basic requirements on the noise canceling algorithm, besides efficient noise suppression, include suitability for parallel processing, modest computation needs and low memory requirements. The need for low memory requirement comes from the fact that in high speed camera applications almost all available DRAM storage bandwidth is utilized by image data storing. Therefore, only memory available for the image processing is the internal memory of the FPGA. That fact puts a lot of strain on algorithm's memory efficiency and practically eliminates the possibility of image processing in spectral domain [1]. Despite the significant processing power offered by today's FPGAs, number of operations per pixel also presents a significant limiting factor for the algorithm.

Implemented algorithm consists of two functional blocks:

1. uncorrelated noise reduction block, and
2. block for correction of column gain error.

A. Uncorrelated noise removal block

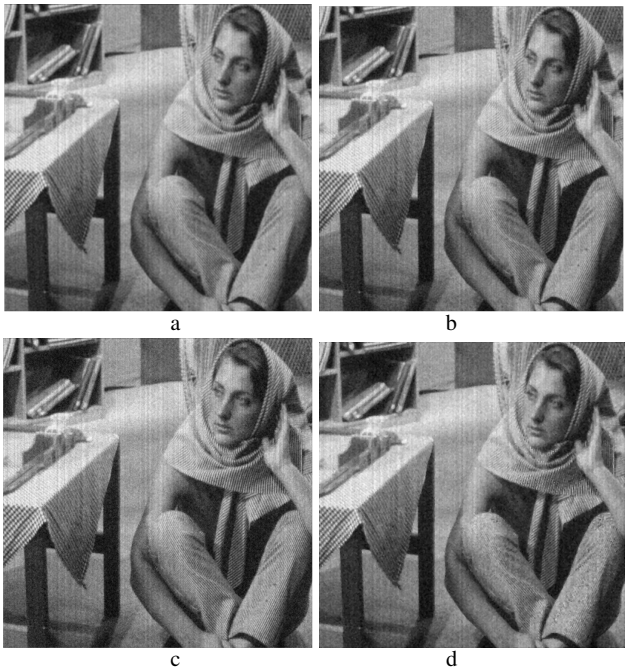
Uncorrelated noise reduction block performs uniform noise reduction targeting both temporal and fixed pattern noise (FPN). Image denoising in spatial domain is usually done by linear filtering. Besides linear, statistical filters are also commonly used, especially median filter. Usual kernel sizes for the mentioned filters are 3x3, 5x5 and 7x7. Since smaller block size requires less computation, the algorithm is based on the 3x3 block. Median and three linear filters: mean, cone and pyramidal are compared in order to find the optimal filter for the defined task. Kernels of used linear filters are shown in (1).

$$h=\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix} \quad h=\frac{1}{16}\begin{bmatrix}1&2&1\\2&4&2\\1&2&1\end{bmatrix} \quad h=\frac{1}{16}\begin{bmatrix}1&1&1\\1&8&1\\1&1&1\end{bmatrix} \tag{1}$$

[1, 2] All authors are with the Faculty of Technical Sciences at University of Novi Sad, Trg Dositeja Obradovića 6, Novi Sad 21000, Serbia
E-mail: dusan.majstorovic@rt-rk.com
E-mail: boris.radin@rt-rk.com

Filter performances are tested using referent image shown in figure 1(a). Image used as an input to the filters is first corrupted uniformly by Gauss noise of variance 0.005. The same procedure is than repeated on x-axis only, simulating the column gain variance (figure 1(b)). Filter output images are shown in figure 2.

Fig. 1. (a) Original image 512x512 (b) Noisy image



a b

c d

Fig. 2. Filter output images (a) Mean (b) Pyramidal (c) Cone (d) Median

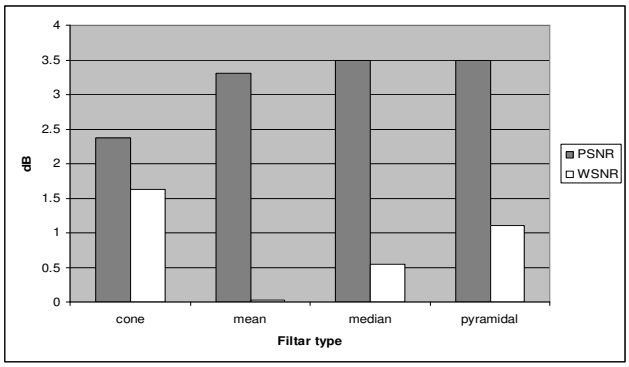


Fig. 3. Comparison graph of added PSNR and WSNR of different filters for the referent image

Filter performances are evaluated using both subjective and objective criteria. Subjective evaluation is performed by simple selecting the image closest to the original image. Objective evaluation is obtained using standard PSNR criteria as well as weighted signal to noise ratio proposed in [4]. The best results, based on both criteria, were achieved by using the pyramidal filter (figure 3).

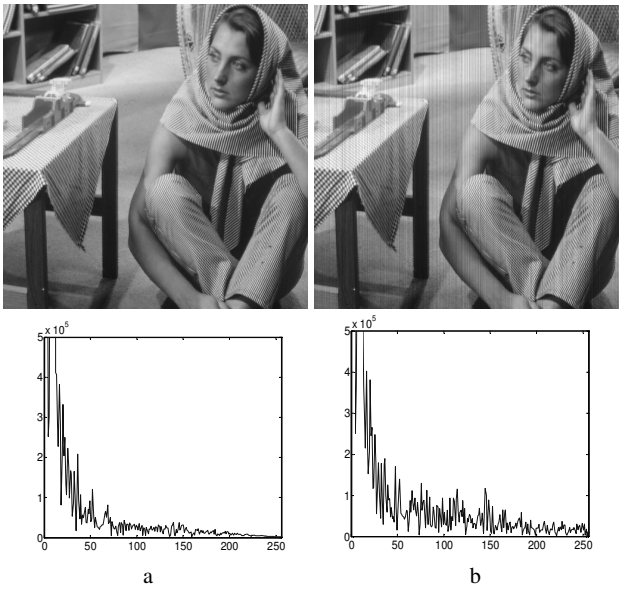
Analyzed convolution filters require less computation than the median filter. Pyramidal and cone filters are especially suitable for implementation in hardware due to specific kernel weight values (all values are powers of two). Based on all mentioned, the decision was made to implement pyramidal filter.

B. Column gain error compensation

Almost all high speed CMOS imagers feature column parallel readout structure. With that type of readout structure FPN generated by sensors is dominated by column-to-column variations [2].

Today's most efficient algorithms for FPN reduction are based on optical flow analysis [3]. Despite the unparalleled results of this approach, high computation needs (more than 1000 operations per pixel) makes it unsuitable for real-time implementation in high speed applications. For that reason the idea was to develop much simpler solution that will target only the dominant source of FPN.

The idea is to base the algorithm on information gathered from column sums of the image. Figure 4 shows original image (a) and noisy image with simulated column gain variation (b). Below, spectral contents of column sums for each image are shown.



a b

Fig. 4. Original (a) and noisy (b) images with matching column sum spectral contents

From figure 4 it is easy to notice that the column gain variation produces higher spectral components. That fact is

used for calculating the column correction coefficients. Referent column sum array is formed by low pass filtering of the original image column sum array. Column correction coefficient array is formed as shown in (2).

$$c(i) = \frac{sum_{ref}(i)}{sum(i)} \quad (2)$$

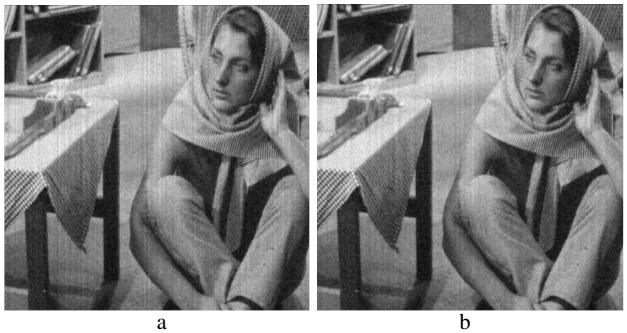
The column gain compensation defined like this has a serious drawback. In parts of image with great horizontal dynamics it produces artifacts in form of light and dark vertical stripes. To overcome this problem the column gain compensation defined this way must be attenuated in regions of high horizontal dynamics. The horizontal image dynamics is estimated for each column using the following formula.

$$D_h(i) = \frac{abs(0.5 * sum(i+2) + sum(i+1) - sum(i-1) - 0.5 * sum(i-2))}{\max(D_h)} \quad (3)$$

Final array of gain compensation coefficients is calculated as follows:

$$c_k(i) = \frac{D_h(i)}{D_{h\max}}(c(i)-1)+1 \quad (4)$$

Results of the column gain compensation formed like this are shown in figure 5.



a b

Fig. 5 Images before and after correction. (a) Before correction (b) After correction

III. SYSTEM STRUCTURE

The developed high-speed camera system structure is depicted in figure 6. Sensor is mounted on separate PCB which is connected to the main board using flexible connections allowing different sensor positions for future camera applications. Sensor in use is Micron MT9M413 [5]. It is a 1.3MP (1280x1024) CMOS sensor capable of 500 frames per second data rate at full resolution. The sensor has column parallel readout structure with on-chip 10 bit AD converters (one per every column). Developed system supports sensor's windowing capabilities allowing very high frame rates for small regions of interest (e.g. 4000 fps for 1280x128). Region of interest can be selected arbitrarily by dragging the mouse pointer over preview image area.

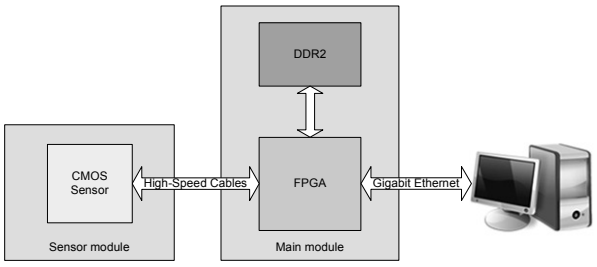


Fig. 6 Embedded system structure

The camera system is based on Xilinx Virtex4FX family FPGA [6]. Hardware/software support is integrated in the PowerPC based embedded system with structure depicted in figure 7.

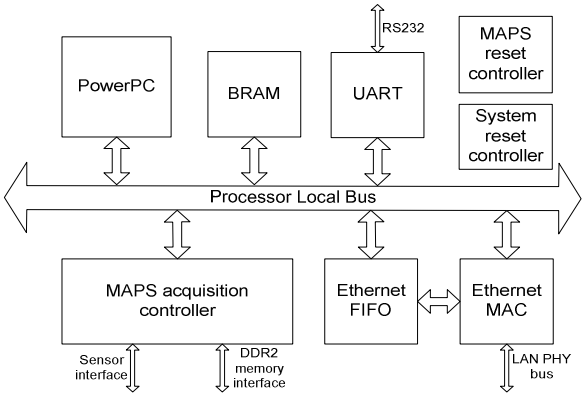


Fig. 7 Embedded system structure

Data acquisition subsystem (MAPS acquisition controller block in figure 7) is implemented as a Processor Local Bus (PLB)[7] compatible peripheral. DDR2 controller is implemented as integral part of the peripheral in order to overcome the problem of limited bandwidth of the PLB. The developed peripheral poses a bus master property maximizing the throughput to the rest of the system.

IV. NOISE CANCELLING BLOCK

The noise canceling block implements two noise canceling mechanisms described earlier in the text. The block has a pipeline structure and is inserted directly in the data path. The block has no access to external memory for the reasons mentioned previously. Noise canceling block structure is depicted in figure 8.

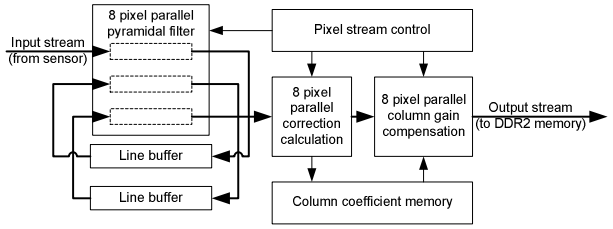


Fig. 8 Noise cancellation block structure

Pyramidal filter block uses the standard structure for convolution filters in FPGA [8]. To achieve the throughput required by the system the block is parallelized. This way, 8 pixels can be processed in a single clock cycle. Due to specific values of selected filter kernel coefficients, multiply and divide operations are replaced with arithmetic shift operations leading to lower utilization of available logic resources.

Column gain compensation functionality is divided in two functional blocks. Block for calculating compensation coefficients consists of 16^{th} order FIR filter and horizontal dynamic estimator. Column sums and compensation coefficients array are stored in the local dual-port memory.

The second block executes gain compensation on each pixel and is based on parallel fixed-point multiplier.

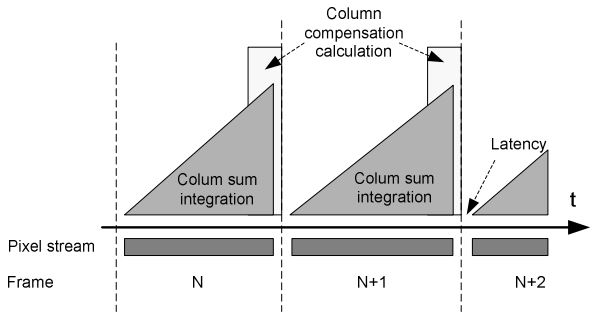


Fig. 9 Time diagram showing basic execution stages in noise cancellation block

Noise cancellation hardware block utilizes around 10% of the used XC4VFX40 FPGA device. Programmable matrix utilization is 7% for LUTs and 8% for FFs while DSP block utilization is over 40% due to intensive usage of dedicated multipliers. Thanks to the deep pipeline structure and the usage of dedicated multipliers the system can operate at 200MHz clock rate even on the slowest speed grade FPGA device.

Having in mind the system's parallel structure, the peak throughput is 1.6×10^9 pixels per second which is enough for real-time operation on described system. The issue is latency introduced by column gain compensation block (figure 9). It makes the sustained system performance dependable on image resolution while the maximum sensor throughput is constant with resolution. Due to that fact, column gain compensation operation in real-time is possible only on a subset of available image resolutions.

V. CONCLUSION

In this paper, the flexible high speed camera system with real time noise cancellation functionality is described. The implemented camera system was subject of thorough testing. The system demonstrated highly reliable operation.

During the system development the emphasis was placed on its flexibility. Sensor position can be customized for each application due to flexible connection between PCBs. With embedded hardware structure smaller changes in the system functionality can be done in PowerPC software without the need for hardware redesign.

Noise removal functionality consists of two functional blocks that can be independently enabled. Uncorrelated noise removal block provides efficient noise suppression while the resulting image blurring is modest.

Column gain compensation method shows certain dependence on image content. Latency introduced by the correction calculation block limits the real time operation on the upper range of supported image resolutions. Still, in some specific applications, this functionality can offer noticeable image quality improvement.

REFERENCES

[1] Mihajlo Katona, Aleksandra Pižurica, Nikola Teslić, Vladimir Kovačević, Wilfried Philips, "A Real-Time Wavelet-Domain Video Denoising Implementation in FPGA", Proceedings of the SPIE, Volume 5607, pp. 63-70 (2004)

[2] Abbas El GamaVL, Boyd Fowlera, Hao Minb, Xinqiao Liua, "Modeling and Estimation of FPN Components in CMOS Image Sensors", in Proceedings of SPIE, vol. 3301, pp. 168-177, April 1998.

[3] SukHwan Lim; A. El Gamal, "Gain fixed-pattern-noise correction via optical flow", *IEEE Transactions on Circuits and Systems I,* Volume 51, Issue 4, pp. 779 – 786, April 2004.

[4] T. Mitsa and K. Varkur, "Evaluation of contrast sensitivity functions for the formulation of quality measures incorporated in halftoning algorithms", ICASSP '93-V, pp. 301-304.

[5] "High Speed CMOS Image Sensor - Datasheet", Micron Technology, 2007.

[6] "Virtex-4 Family Overview", Xilinx, 2007.

[7] "Processor Local Bus (PLB) v4.6", Datasheet, Xilinx, 2008.

[8] C.J. Chang, P.Y. Hsiao and Z.Y. Huang, "Integrated operation of image capturing and processing in FPGA", IJCSNS International Journal of Computer Science and Network Security 6 (2006), pp. 173–180.