HLS Accelerated Noise Reduction approach using Image Stacking on Xilinx PYNQ

Dimitris Tsiktsiris
Department of Informatics and Telecommunications Engineering
University of Western Macedonia
Kozani 50131, Greece
Email: dtsiktsiris@uowm.gr

Dimitris Ziouzios
Department of Informatics and Telecommunications Engineering
University of Western Macedonia
Kozani 50100, Greece
Email: dziouzios@uowm.gr

Minas Dasygenis
Department of Informatics and Telecommunications Engineering
University of Western Macedonia
Kozani 50131, Greece
Email: mdasyg@ieee.org

***Abstract*—Eliminating noise from the original signal is still a demanding problem for researchers. There has been a number of published algorithms and each approach has its limitations, advantages, and constraints. Here we present our own contribution on the noise reduction problem by proposing an intermediary step between the image sensor and the post-processing software in the image capturing process. Our accelerated method, compared to other researchers, can be used either as a standalone solution or combined with an existing algorithm to further results improvement. Our solution matches the discrete samples between multiple frames and averages the pixel values. The output image maintains its structural integrity, holds better color accuracy and incurs less noise than others.**

***Index Terms*—SNR, sampling enhancement, noise reduction, acceleration, image enhancement, FPGA**

I. Introduction

One practical obstruction to superior photography is the absence of light [1]. Sometimes, inadequate light during indoor or night-time shots can deteriorate image. A typical solution to this problem would be either an increase in exposure time, which causes motion blur due to a possible subject motion or camera shake, or analog/digital gain amplification, intensifying the noise. Surprisingly enough, even photographs taken in the day can be afflicted by a shortage of light. Particularly, with short exposure time, we prevent blowing out highlights, but the dim areas appear even darker and noisy. One way to brighten those areas would be using local tone-mapping, but that would cause the noise to intensify, as well. Therefore, some suggested ways to collect more light include the use of exposure bracketing, optical image stabilization, a larger-aperture lens, or even a flash.

However, each one of these aforementioned techniques is a trade-off. For example, the camera of any cell phone is thickness-restricted; thus, it is quite a task to make its aperture larger. Another way to reduce noise relies on a synthetic aperture formation by increasing the number of cameras [2]. This method is not suitable, as such devices are also power-restricted. Optical image stabilization offers increased exposure, while minimizing blur caused by camera motion, but the blur caused by subject motion can still not be controlled. When using exposure bracketing followed by image fusion, the scene is represented by different parts of the fused image, at different times, causing difficulty for the single self-consistent frame to be achieved. Due to the struggle of aligning images captured at different times, the most frequent artifact caused by false fusion is ghosting [3]. Some sensors, designed with increased exposure times in mind, use alternation between adjoining scanlines to improve that ghosting. However, in this case, details are being sacrificed and the formation of accurate demosaicing [4] is difficult. To many photographers, an on-camera flash is the least acceptable option. It may add the necessary light, but the scene is changed in a very disagreeable way. This issue is addressed by flash/no flash photography [5], but still, it is not sufficiently robust.

In this research, our goal is to define an appropriate camera system, able to address such complications by simply capturing a burst of images and merging them, in order to refine color accuracy and decrease the noise. We also present an accelerated hardware implementation of the proposed approach. Finally, we present some experimental results and discuss them.

II. Related Work

An interesting approach has been proposed by Sarode *et al.* [6]. They proposed a similar filtering technique for the removal of speckle noise from the digital images. However, their implementation is focused on medical and satellite images and sometimes the noise variance cannot be estimated by their methods.

Another interesting approach to noise reduction was introduced by Hedaoo *et al.* [7]. In their approach, they attempt to recover a function of unknown smoothness from noisy, sampled data. Their procedure, SureShrink, suppresses noise by thresholding the empirical wavelet coefficients. However, this denoising scheme tends to kill too many wavelet coefficients that might contain useful image information.

Lastly, Zhang *et al.* [8] have presented a method for enhancing images acquired under very low light conditions where the features of images are nearly invisible with very large amounts of noise. By applying an improved and effective image de-hazing algorithm to the inverted input image, they amplify the intensity, so that the dark areas become bright and the contrast gets enhanced. Then, the joint bilateral filter with

978-1-7281-1184-1/19/$31.00 ©2019 IEEE

the original green component as the edge image is introduced to suppress the noise. This method is not suitable for our work as it tends to create non-natural photographs. Also, the hardware implementation constraints are not considered by the authors.

III. Methodology

The following design principles were found to be important during our building of our hardware/software co-synthesis system:

- Immediateness. In less than a second, a photograph must be produced and displayed on the camera, even in cases where the camera is disconnected - wired or wirelessly. Therefore, there should be no offloading processing to the cloud or a network computer.
- Automation. Our method should be both fully automatic and parameter-free. This means that a photographer taking a picture should not be aware of the strategy used during image processing or capture.
- Conservation. Our system should be possibly used as a default picture-capturing mode; thus, there should be no artifacts in the final picture, which must be as good as conventional photographs, at the very least.

The key assumption of burst photography is the noise reduction apprehension, when merging various observations of the view over a period of time. In our experiments we captured images with lower exposure to avoid highlights clipping. This technique is used for the capture of a more dynamic range. Shorter than typical exposure times were also selected, so that the camera shake blur be alleviated, regardless of the scene content [9]. Even though noise should be expected to worsen in lower exposures, this effect was balanced out when we captured and combined multiple frames.

From our conservative constraint, a second design decision involves the selection as a "reference" frame of one of the images in the burst, following the alignment and merging of patches from "alternative" frames into this frame; the "alternative" frames depict the same portion of the scene. Moreover, only a single patch from each "alternative" frame is merged, so that the computational complexity of our design reduced. The final image may appear noisier than others, due to our conservative merging in cases of camera motion; however, it is hardly ever noticeable.

Note that by merging multiple frames, an intermediate image of reduced noise, compared to the input frames, as well as of a higher bit depth and dynamic range is created. This contribute to a high-quality (even though underexposed) photograph, assembled due to the discarding of the low-order bits. Nonetheless, one of our goals involves the production of natural-looking photographs, even though our design constraints conflict. This is the reason why we choose to boost shadows instead, maintaining local contrast, while carefully giving up global contrast.

The number of frames captured N is a trade-off. This means that in case of low-light circumstances, where the shadows will be enhanced later on, even more frames are needed for the signal-to-noise ratio improvement; these frames, though, demand more memory and time to capture, buffer and process, as well. Normally, a burst of 8 frames in a bright scene is adequate. However, the more images captured, the easier the camera shake blur will be tackled by our design.

Our merge process functions on image tiles in the spatial frequency domain. We split each frame into $3x3$ (this parameter can be changed in the design) tiles and for each one we try to match the corresponding tiles across the burst. Then we calculate their relative x,y planes of 2D Discrete Fourier transform (DFTs) as $T_z(\omega)$, where $\omega = (\omega_x, \omega_y)$ defines spatial frequency, z is the frame index. Without loss of generality, we assume frame 0 as the reference. The averaged final frame $\tilde{T}_0$ is given in Eq. 1.

$$\tilde{T_0}(\boldsymbol{\omega}) = \frac{1}{N} \sum_{z=0}^{N-1} T_z(\boldsymbol{\omega}) \tag{1}$$

IV. Hardware Implementation

For our implementation, we selected a Xilinx HLS approach on a Pynq-Z1 board from Xilinx. This board is designed to be used with PYNQ, an open-source framework that enables embedded programmers to exploit the capabilities of Xilinx Zynq All Programmable SoCs (APSoCs). The APSoC is programmed using Python and the code is developed and tested directly on the PYNQ-Z1. The programmable logic circuits are imported as hardware libraries and programmed through their APIs in essentially the same way that the software libraries are imported and programmed. The board is based on ZYNQ XC7Z020-1CLG400C and includes a 650MHz dual-core Cortex-A9 processor, 13.300 logic slices, each with four 6-input LUTs and 630 KB of fast block RAM. We used C++ for HLS of the HDL blocks and at the same time we configured the peripheral blocks for the system integration.

A. Processing Core

We exploit HLS capabilities to generate the Intellectual Property (IP) cores, by using C++ code written by us. In order to exploit the hardware parallelism, we need each application's timing diagrams for various inputs, function calls, as well as an in-depth code analysis. Our core consists of two main components:

1) Tile Merging: The HLS implementation for the Tile Merging IP consists of 4 stages. The first (AXIS2ImgArray) and the last one (ImgArray2AXIS) are responsible for transferring the image data through the AMBA AXI4-Stream Protocol Specification. We combine the array conversion into the first stage to simplify the design. We are storing the images as FIFO (First In First Out) queues and at a second stage we split each image to a predefined set of $3x3$ tiles. We are trying to match features between coherent image tiles in order to minimize ghosting and provide more accurate results. Based on some heuristics (color channel isolation, brute-search neighboring pixels etc.) we are trying to match at least one 3x3 pixel grid with minimal color variation.

TABLE I
UTILIZATION SUMMARY

Resource	Utilization	Available	Utilization %
LUT	23804	53200	44.74
LUTRAM	2117	17400	12.16
FF	24031	106400	22.58
DRAM	82	140	58.57
DSP	5	220	2.27

2) Int2RGB: The second IP helps to improve image reconstruction performance by offloading CPU calculations to Programmable Logic (PL), as the three RGB channels are encoded into a 32-bit unsigned integer. This is a crucial step, because, as we discovered in our experiments, performing this computation on the CPU incurred 280 seconds extra delay for every image of 4000x3000 resolution (12 megapixels). Using our optimization we reduce this operation's latency to under one second. We implement it through AXI4-Stream, furthermore optimizing the transfer of the data by using a bidirectional DMA (read and write), benefiting our architecture with reduced resource usage, in contrast to other implementations, which use one DMA for reading and one for writing.

Pynq-Z1 is a Zynq-7000 ARM/FPGA System on Chip (SoC), which utilizes the AMBA [10] architecture; therefore for the communication, we use AXI [11], an interface standard that allows communication of different components with each other. An AXI link consists of the AXI master, capable of initiating transactions, and the AXI slave, which responds to the master's initiated transactions [12]. AXI4-Stream allows an unlimited size of burst data and removes the memory address phase requirement altogether; but due to their lack of address phases, AXI4-Stream interfaces and transfers are not considered to be memory-mapped [13]. To achieve high performance, we use AXI Direct Memory Access (DMA) IPs (Fig. 1), which provide a direct memory access of a high bandwidth between the AXI4-Stream IP and AXI4 memory mapped interfaces [14].

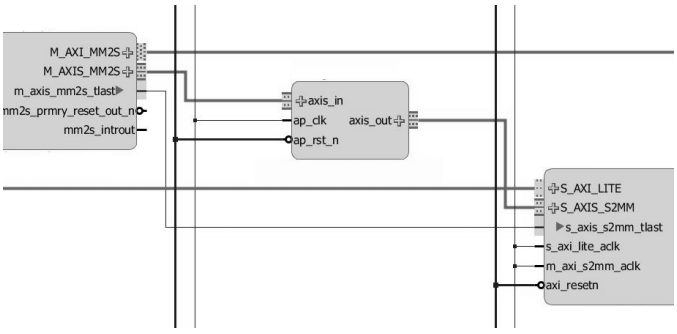

Fig. 1. Our design utilizies AXI Direct Memory Access (AXI DMA) connections, which provide a high-bandwidth direct memory access between memory and AXI4-Stream-type edge detection peripheral.

Fig. 1 depicts a block diagram with the interface connections of the Tile Merging IP with the DMA input and output of data. Researches have reported that at a clock frequency of 100MHz, data transitions may be established from both AXI4 master and AXI-Stream slave to AXI4 master at a data rate of 400MBps and 300MBps, respectively, being a quota of the theoretical bandwidths of 99.76% and 74.64% [15]. These numbers are used in order to help us find the overhead of the data transfer between the Zynq device's Processing System (PS) and PL.

B. Firmware

We are using the Python libraries provided with the base image of Pynq-Z1, for the firmware. We download the overlay on the board and we initialize the DMAs for interaction. Then we stream each input buffer to the PSPL DMA, wait for the DMAs to assert the finish signal and read the output buffer from the PLPS DMA.

V. EXPERIMENTAL MEASUREMENTS

Our system has been evaluated using devices with 13 megapixels sensors, on which we capture bursts of up to 15 frames. Therefore, there may be a need to process 195 megapixels to produce the final image. Our implementation does not consist of a complete standalone system (integrated with the camera sensor and accelerator altogether), but we can estimate the total processing time of a complete system.

Our software implementation of the merging approach, running on an Intel Core i5-4460 at 3.2 GHz with 16 gigabytes of RAM, averaged about 1700ms for a burst of 15 frames. Our hardware accelerator is 8.5x times faster and requires about 200ms on average to merge all the image buffers and output the final image. Therefore, we can safely assume that the largest delay in the capture progress is the sensor's shutter lag. To obviate this lag, many cameras utilize zero shutter lag (ZSL), in which the sensor constantly captures frames, caching them in a circular buffer, and on shutter trigger, one of them is selected.

$$PSNR = 20\log_{10}\left(\frac{MAX_f}{\sqrt{MSE}}\right) \tag{2}$$

We compared the results using the peak signal-to-noise ($PSNR$) ratio and the Mean Square Error (MSE), as shown in Eq. 2, 3 respectively [16]. The PSNR is often used to measure the quality between a compressed and the original image, denoting a peak error measure, whereas the MSE is used to measure the cumulative squared error between the original and the compressed image. Note that the higher the PSNR, the better the quality of the reconstructed or compressed image, while the lower the MSE value, the lower the error, respectively.

$$MSE = \frac{1}{mn}\sum_{0}^{m-1}\sum_{0}^{n-1}\|f(i,j) - g(i,j)\|^2 \tag{3}$$

In Eq. 2, 3, f denotes the original image data matrix and g the data matrix of our degraded image in question. Moreover, in Eq. 3, m and n are used to represent the number of rows and columns of the images' pixels, respectively, and i and j denote the index of that row or that column, respectively.

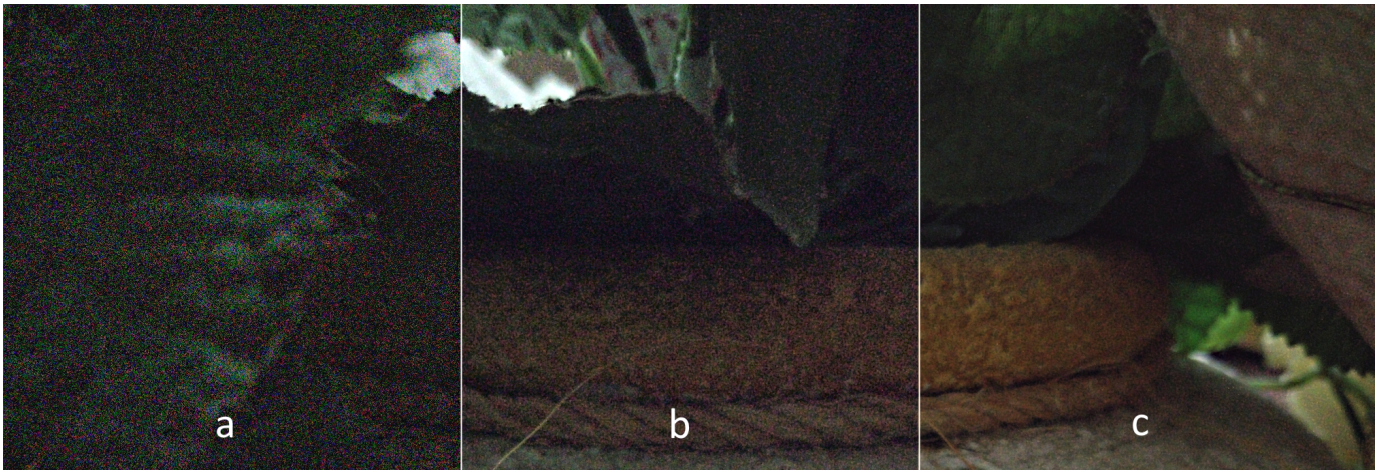

Fig. 2. Combined image results from a mobile device with Sony IMX363 sensor (f/1.9 3.94mm) in low light (1 lux). At the left (a) is the original image captured (1/10 sec. ISO 9600). In the middle (b) we have an FPGA processed result (1/10sec. ISO 3200, 8 frames). Finally, on the right side (c), the output of the FPGA accelerator is shown (1/10sec. ISO 3200, 15 frames). Clearly the output result (c) is very acute, bright and crisp.

Finally, $MAXf$ is the maximum signal value existing in the original known to be a high-quality image.

We use the MSE metric for practical purposes, being the comparison between the true pixel values of a high-quality image, captured with a professional camera, and the respective ones of the algorithm. Our goal is to minimize the MSE between the two aforementioned images, with respect to the image's maximum signal value.

Our results have less noise than the images derived by a typical imaging pipeline, especially in low-light scenes. In Fig. 2 we can clearly see the benefits of this approach. We managed to decrease the ISO by a factor of three and still capture a better exposed and without noise image. In our experiment, we used a shutter speed of 1/10 sec per frame, which we decided that is a "safe" value also for hand-held photography. In a complete camera system those parameters will be automatically adjusted in real-time using sophisticated algorithms, to fully exploit the hardware capabilities.

VI. Conclusions

Image Stacking technique is an essential step in noise reduction and image enhancement. Many existing noise reduction techniques rely on software processing and have mixed results. We implemented an FPGA accelerated approach, which provides excellent results and outstanding computation performance. Future work includes an improved image alignment algorithm, in order to compensate for both camera and subject motion. The feature matching algorithm at this time is very simple and sometimes inaccurate. Also, we are planning to further exploit the benefits of our approach by applying algorithms for HDR (High Dynamic Range) imaging and other image enhancing techniques.

References

[1] B. Moomaw, "Camera technologies for low light imaging: overview and relative advantages," in *Methods in cell biology*. Elsevier, 2013, vol. 114, pp. 243–283.

[2] B. Wilburn, N. Joshi, V. Vaish, E.-V. Talvala, E. Antunez, A. Barth, A. Adams, M. Horowitz, and M. Levoy, "High performance imaging using large camera arrays," in *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 765–776.

[3] S. Wu, S. Xie, S. Rahardja, and Z. Li, "A robust and fast anti-ghosting algorithm for high dynamic range imaging," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 397–400.

[4] R. Kimmel, "Demosaicing: image reconstruction from color ccd samples," *IEEE Transactions on image processing*, vol. 8, no. 9, pp. 1221–1228, 1999.

[5] G. Petschnigg, R. Szeliski, M. Agrawala, M. Cohen, H. Hoppe, and K. Toyama, "Digital photography with flash and no-flash image pairs," in *ACM transactions on graphics (TOG)*, vol. 23, no. 3. ACM, 2004, pp. 664–672.

[6] M. V. Sarode and P. R. Deshmukh, "Reduction of speckle noise and image enhancement of images using filtering technique," *International Journal of Advancements in Technology*, vol. 2, no. 1, pp. 30–38, 2011.

[7] P. Hedaoo and S. S. Godbole, "Wavelet thresholding approach for image denoising," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 3, no. 4, pp. 16–21, 2011.

[8] X. Zhang, P. Shen, L. Luo, L. Zhang, and J. Song, "Enhancement and noise reduction of very low light level images," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 2034–2037.

[9] J. Telleen, A. Sullivan, J. Yee, O. Wang, P. Gunawardane, I. Collins, and J. Davis, "Synthetic shutter speed imaging," in *Computer Graphics Forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 591–598.

[10] A. AMBA, "Protocol specification," *ARM, June*, 2003.

[11] ——, "3.0 AXI specification," 2011.

[12] S. Neuendorffer, T. Li, and D. Wang, "Accelerating opencv applications with zynq-7000 all programmable SoC using vivado hls video libraries," *Xilinx Inc., August*, 2013.

[13] K. Neshatpour, M. Malik, M. A. Ghodrat, A. Sasan, and H. Homayoun, "Energy-efficient acceleration of big data analytics applications using FPGAs," in *Big Data (Big Data), 2015 IEEE International Conference on*. IEEE, 2015, pp. 115–123.

[14] A. Shrivastav, G. Tomar, and A. K. Singh, "Performance comparison of amba bus-based system-on-chip communication protocol," in *2011 International Conference on Communication Systems and Network Technologies*. IEEE, 2011, pp. 449–454.

[15] L. Stornaiuolo, M. Santambrogio, and D. Sciuto, "On how to efficiently implement deep learning algorithms on pynq platform," in *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 2018, pp. 587–590.

[16] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment," *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.