

Scene-based noise reduction on a smart camera

Faouzi Hamdi*, Tomasz Toczek*, Barthélemy Heyrman* and Dominique Ginjac*[†]

*Le2i UMR 6306, University of Burgundy, Dijon, France

[†]Email: dginjac@u-bourgogne.fr - Corresponding author

Abstract—Raw output data from CMOS image sensors tends to exhibit significant noise called Fixed-Pattern Noise (FPN) due to on-die variations between pixel photodetectors. FPN is often corrected by subtracting its value, estimated through calibration, from the sensor’s raw signal. This paper introduces an on-line scene-based technique for an improved FPN compensation which does not rely on calibration, and hence is more robust to the dynamic changes in the FPN which may occur slowly over time. Development has been done with a special emphasis on real-time hardware implementation on a FPGA-based smart camera. Experimental results on different scenes are depicted showing that the proposed correction chain induces little additional resource use while guarantying high quality images.

I. INTRODUCTION

CMOS image sensors are well-know to be prone to noise, especially under low illumination conditions [1]–[3]. Among them, Fixed-Pattern Noise (FPN) is the pixel-to-pixel output variation under uniform illumination due to mismatched pixels and associating readout circuits [4]. Since noise deteriorates imaging performance, noise reduction techniques must be performed to circumvent these sensor limitations. Despite the integration of FPN-reduction circuits, such as Correlated Double Sampling, CMOS image sensors still have FPN noise problems, especially in extreme light conditions. So, it is of the utmost importance to consider efficient image processing techniques in order to achieve higher-quality images.

The most common way to deal with FPN is through a calibration step at the factory [5]. The "FPN frame" is stored in a ROM and subtracted to each captured frame. However, since FPN varies slowly over time, this makes an one-time calibration ineffective and poses additional challenges to achieve a high quality image. Scene-based methods [6] are natural candidates because they do not require initial conditions for the purpose of calibration. FPN is dynamically updated from the real-time analysis of the scenes captured by the sensor. It is obvious that such techniques increase system reliability but have higher computational complexity [7] requiring significant hardware resources to be processed at the sensor frame rate. Scene-based methods include two distinct categories: (1) algebraic techniques, and (2) statistical techniques. As defined in [8], the algebraic techniques make use of global motion between the frames without making statistical assumptions about the FPN. Such methods offer good robustness but are computationally heavy and not really suitable for real-time correction. On the other hand, statistical techniques model the FPN as a random spatial noise and estimate the statistics of the noise to remove it [9]–[11]. Compared with registration-based methods, statistical approaches have been more widely studied

because of their relatively lower computational complexity, smaller storage demands, and better realtime performance.

The remainder of this paper is organized as follows. Section II deals with an improved algorithm based on the most well-known statistical method developed by Harris and Chiang [9]. Hardware implementation on a FPGA-based platform is then discussed in Section III. Finally, experimental results on a set of different realistic scenes and comparisons in terms of image quality with original Harris and Chiang algorithm are presented in Section IV.

II. IMPROVED CONSTANT STATISTICS METHOD

We propose to extend the approach developed by Harris and Chiang [9]. Our objective is to improve the method so as to make it more flexible regarding the input scene. The original paper models the non-linear behavior of each photodetector as an affine function. Each photodetector k is associated with a pair of scalars a_k and b_k such that the actual luminance \tilde{I}_k received by the photodetector can be obtained from the measured one I_k through the relation:

$$I_k = a_k \tilde{I}_k + b_k \quad (1)$$

Harris and Chiang assume that the measured luminance *over time* has the same mean and the same variance among all the photodetectors of the sensor. Under these conditions, it can be shown that for each photodetector k , assuming m_k is its average measured luminance and σ_k the measured luminance variance, the following holds:

$$\tilde{i}_k = \frac{I_k - m_k}{\sigma_k} \quad \text{and} \quad \tilde{I}_k = A \tilde{i}_k + B \quad (2)$$

There are several easy ways to iteratively estimate m_k and σ_k , so \tilde{i}_k can be cheaply computed. The \tilde{i}_k signal mean is zero and its variance is unitary, but it is otherwise proportional to the luminance value plus an offset. \tilde{I}_k can be obtained by knowing the factors A and B , which can be plausibly estimated by using the least squares method so as to minimize the error between the measured and corrected pixel values.

Harris and Chiang recommend to move the camera or otherwise move the objects of the scene during calibration, but this implies the approach cannot be used to continuously calibrate the sensor. Whats more, even when the camera is moving and a dynamic scene is being shot, the mean and variance hypotheses may not hold. We have observed that making accurate hypotheses regarding the FPN is much easier than regarding the scene. So, we propose to adapt this

approach so as to make it work whenever the following is true for each photodetector k :

- 1) FPN has only an offset component (no gain component);
- 2) the mean value of the FPN offset is locally zero;
- 3) the high spatial frequency component at the corresponding pixel has an average value of zero over time.

Concerning the validity of the first hypothesis, we can notice that FPN is almost only seen on low luminosity and/or long exposure shots. If a significant gain component exists, its influence should be noticeable regardless of luminosity or exposure. Since it is not the case for almost any visible light sensor we studied, it means that either no FPN gain component exist at the photodetector level, or that current compensation mechanisms for this component work just fine, unlike those aiming to correct the offset component. The second hypothesis is more or less justified by assuming that manufacturing defects occur randomly. The last assumption is the only one concerning the scene. Since the FPN is mostly a high frequency component, it is enough to average over time the high spatial frequencies at each pixel to determine the noise value, should this assumption be satisfied.

Our method is based on the following principles: 1) remove the low spatial frequencies from the current frame, 2) apply the Harris and Chiang method on them, and 3) add them back to obtain the final image. When applying the Harris and Chiang method, we will assume that $a_k = 1$ for each k . Benefits of our method are various: 1) range of scenes correctly corrected is much wider and, 2) the post-processing dynamic range adjustment step is no longer necessary. Indeed, for each photodetector k , let us separate the high frequency component I_k^H from low frequency one I_k^L . Using the previous notations, we will have:

$$I_k^H + I_k^L = a_k \tilde{I}_k + b_k \quad (3)$$

Assuming $a_k = 1$ (assumption 1), using the same reasoning as in the original article but on I_k^H instead of I_k , we get:

$$\tilde{i}_k^H = I_k^H - m_k^H \quad (4)$$

Where m_k^H is the average over time of the high spatial frequency component of the photodetector k . In practice, it is interesting to consider the column noise as a special case and to correct it separately, in order to speed up convergence. This can be done in the same way as pixel correction, but using a separate estimator for each column j , called m_j^H , and defined as the average over time of the high spatial frequency component of the photodetectors of the column j . Assuming $col(k)$ is the column index of the photodetector k , the final correction equation is:

$$\tilde{i}_k^H = I_k^H - m_k^H - m_{col(k)}^H \quad (5)$$

When using this equation, the values (m_k^H) averages must be the already column-corrected ones. Let us notice that the I_k^L component can be considered as noise free according to the assumption 2. While (\tilde{i}_k^H) is zero-meaned temporally just as in the original method, so is (\tilde{I}_k^H) , the high frequency component

of the actual luminance the detector would measure. This is a consequence of the assumption 3. So, we get:

$$\tilde{I}_k = \tilde{i}_k^H + I_k^L \quad (6)$$

This can be computed as we go, confirming that no further post-processing is needed. Low-pass filtering aside, we will just need to update the (m_k^H) and (m_j^H) estimates at each pixel read. The cheapest method to do this is to use an exponential window [12]. We will therefore take a pair of scalars α and α' , such as, if $\hat{m}_k^H(n)$ is the m_k^H estimate and $\hat{m}_{col(k)}^H(n)$ is the $m_{col(k)}^H$ estimate for the k -th pixel at the n -th iteration:

$$\begin{aligned} \hat{m}_k^H(n) &= \alpha I_k^H + (1 - \alpha) \hat{m}_k^H(n-1) \\ \hat{m}_{col(k)}^H(n) &= \alpha' I_k^H + (1 - \alpha') \hat{m}_{col(k)}^H(n-1) \end{aligned} \quad (7)$$

III. HARDWARE IMPLEMENTATION

As described on Fig. 1, the correction chain can be implemented by a simple pipeline consisting of a low-pass filter followed by two correction blocks, working respectively at the column and pixel levels. The high-frequency data is obtained by subtracting the low frequencies from the raw signal. Those low frequencies are later added back to the corrected high-frequency data. The low-pass filter and the correction block have an average throughput of one datum per cycle. The correction block has a constant latency of several cycles. Consequently, using an appropriate delay is required to synchronize the low and the frequency components before adding them back together. Since the pixel correction block needs to store the temporal average value estimate m_k^H for each pixel k , it needs an access to the external memory if the input resolution is too high. Assuming a resolution of 1280×720 pixels and using 16 bits per estimator, 1.84 MB of memory are required, i.e. more than most FPGAs can provide internally. It is possible to prefetch the estimator values and store them back once updated, using a small on-chip scratch pad, without affecting the overall correction block latency or throughput. Conversely, no access to external memory is required for the column corrector. Indeed, for a resolution of 1280×720 pixels, all of the (m_j^H) take only about 3.2 KB and can therefore be stored in on-chip RAM.

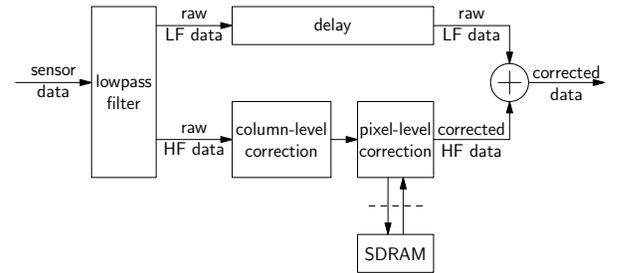


Fig. 1: Proposed FPN correction pipeline

The overall principle of the correction block is shown on Fig. 2. The main issue when implementing this block is the storage of the temporal averages for each photodetector, which

must be read and written in a timely manner. This is achieved by using a small cache capable of prefetching the data from the SDRAM. The column corrector is very similar, but the cache is replaced by an on-chip memory containing all the column average values, and the estimator address counter k is reset on horizontal synchronization instead of vertical synchronization. If α (or α') is a negative power of two, the two multipliers can be replaced with two shifts and a subtractor.

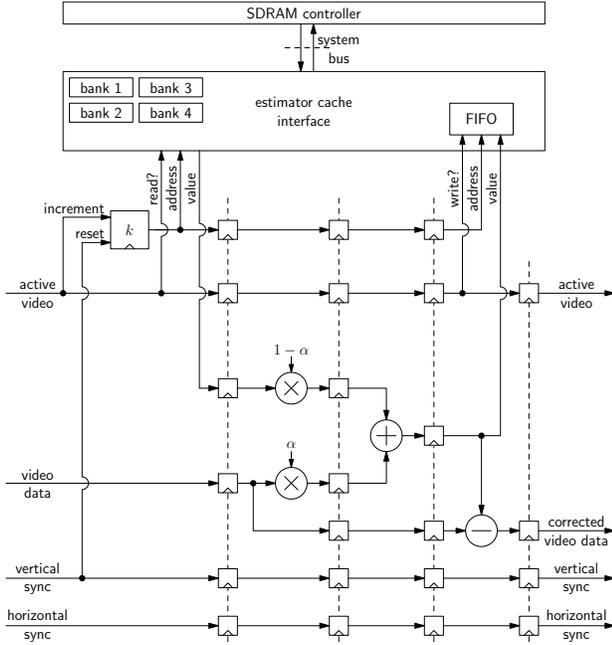


Fig. 2: Pixel correction block principle schematic

We have tested two low-pass filter architectures: a classical rectangle window low-pass and a block computing the average of all the pixels of a scanline. Our implementation of the rectangle window low-pass requires 10 adders/subtractors and two comparisons on values whose sizes are at most 20 bits, no matter the filtering window size. It also requires an amount of on chip memory proportional to the height of the filtering window. The alternative filtering architecture we propose is a simple set of two counters used to sum all the pixels of a line. One of them holds the sum of all the pixels of the previous line, the other holds the intermediate sum of the pixels of the current line read so far. At the end of each line, the roles of the two counters are swapped. The “filtered output” is simply the value of the counter corresponding to the previous scanline pixel sum, multiplied and shifted so as to get the average pixel value. It can be thought of as a $\infty \times 1$ rectangle window filter. While much simpler, such a filter decreases the PSNR by at least 5 dB on our test sequences.

IV. RESULTS

We have tested the proposed method by using video sequences tainted with artificial noise. The artificial offset noise used is a uniform pixel noise within $\pm 12.5\%$ of the dynamic range, in addition to an uniform column noise in this range

as well. An image comparison metric is used to estimate how much the corrected version differs from the original sequence. We have chosen to use both the PSNR and the UQI [13] as such metrics. The UQI roughly corresponds to the human perception of distance between images. With a dynamic range of $[-1, 1]$, the best value is 1 whenever the original image and the test image are identical. In order to work correctly, statistics based methods need some time for the estimator values to converge. Therefore, in the rest of this section, the evaluations of efficiency will be given by averaging the UQI on the last 12 frames of each test sequence. Similarly, the mean square error measure used in the PSNR calculation is also computed on the same frames.

Table I gives correction quality estimations for seven test sequences. A comparison with a simplified version of the Harris and Chiang algorithm, altered not to perform any gain correction, is also provided. Since there is no gain component to compensate to in our artificial noise, this simplification is justified. The parameters used for our method are $\alpha = 2^{-5}$, $\alpha' = 2^{-9}$, 64×32 low-pass filter, 16-bit pixel and 20-bit column estimators. Sequences 1 and 2 use a stationary camera while some elements of the scenes are moving. Sequence 1 has an uniform background, while sequence 2 has a very detailed background. Sequences 3 to 7 vary by the level of luminosity and of contrast. Sequences 3 and 4 are rather bright, while sequences 5 and 6 are almost pitch dark. High contrast elements are found on the sequences 4 and 6. The sequence 7 has a varying degree of illumination over its course. As expected (the third operating assumption is not met), the sequence 2 is the least effectively corrected but the proposed algorithm still outperforms the Harris and Chiang method. According to PSNR measurements, the sequences 3 and 6 are also poorly corrected, while they are satisfying using a visual criterion. On the contrary, the corrected version of the sequence 4 shows visual shortcomings while yielding descent PSNRs. Globally, the proposed correction method does not seem to specifically favor bright sequences over dark ones, and works for different levels of contrast.

We have implemented the proposed corrector on a basic FPGA-based smart camera built around the general-purpose Avnet Spartan-6 LX150T development board (based on the Xilinx XC6SLX150T-3FPG676 FPGA) and an Omnivision OV9715 image sensor [14] capturing images at resolutions up to 1280×800 at 30 fps. We have implemented the different parts of the proposed correction pipeline and assembled them into a pcore for use with the Xilinx EDK. The implementation itself has been done in VHDL and in Haskell. VHDL was used for the control-dominated components, for those dealing with more than one clock domain (typically, at the interface with the memory controller), and whenever the critical path length was not a limiting factor. We have used an in-house Haskell based hardware description embedded language to generate pipelined netlists from combinatorial datapaths. This allowed us to easily adjust the pipeline depth, and measure resource usage and maximum frequency depending on it.

The results are shown on Table II. The version evaluated

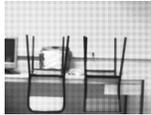
| | Correction quality (UQI / PSNR) | | | | | | |
|-----------------------------|---|---|---|---|---|---|---|
| | seq. 1 | seq. 2 | seq. 3 | seq. 4 | seq. 5 | seq. 6 | seq. 7 |
| |  |  |  |  |  |  |  |
| Our method | 0.885 / 34.25 dB | 0.250 / 20.10 dB | 0.940 / 22.23 dB | 0.700 / 31.04 dB | 0.907 / 36.72 dB | 0.827 / 22.22 dB | 0.888 / 33.54 dB |
| Harris and Chiang method | 0.881 / 27.18 dB | 0.203 / 14.59 dB | 0.510 / 12.88 dB | 0.417 / 23.76 dB | 0.901 / 34.29 dB | 0.795 / 16.47 dB | 0.850 / 25.89 dB |

TABLE I: Correction quality for seven test sequences

uses a point-to-point (NPI) connexion to the Xilinx MPMC memory controller, and employs the scanline-averaging version of the low-pass filter. Since those are the simplest possible versions for those two components, the results shown are upper bound for working frequencies and lower bound for slice usage. As expected, the higher the pipeline stage count, the higher the register count and the higher the slice usage, as more of them are used either as registers without accompanying logic, or simply as route-thrus. The highest reachable frequency is around 169 MHz, when using 6 pipeline stages for the low pass filter and 4 stages per corrector (the pixel and column correctors share the same architecture, so their pipelines have the same length). The maximum frequency goes down when using deeper pipelines due to routing issues. It is interesting to note that in our test system, we could have afforded to use single stage correctors, as the sensor input clockrate is 40 MHz. This is not the case for systems using better cameras: with a sensor capable of 1080p capture at 60Hz, a working frequency of 125 MHz or higher is necessary, requiring moderately deep correction pipeline use.

| # pipeline stages | | Frequency | Slice usage | |
|-------------------|-----------|-------------|-------------|--------|
| Lowpass | Corrector | | # registers | # LUTs |
| 1 | 1 | 80.652 MHz | 580 | 289 |
| 2 | 2 | 99.502 MHz | 748 | 280 |
| 4 | 3 | 140.076 MHz | 994 | 762 |
| 5 | 4 | 149.410 MHz | 1099 | 745 |
| 6 | 4 | 169.319 MHz | 1172 | 596 |
| 7 | 5 | 129.232 MHz | 1258 | 786 |
| 8 | 6 | 158.278 MHz | 1348 | 854 |
| 11 | 8 | 148.170 MHz | 1546 | 671 |
| 22 | 16 | 134.391 MHz | 1878 | 1124 |

TABLE II: FPGA implementation performance and resource usage, depending on pipeline depths.

V. CONCLUSION

We have proposed an improved algorithm for FPN correction in visible light sensors and an associated architectural pipeline for a FPGA-based smart camera. Its hardware complexity is marginally higher than that of a simple reference image while it significantly achieves improved quality of resulting images. Contrarily to the original approach we improve on, it is capable of handling a wide variety of scenes correctly by separating the signal from the noise in a more reliable fashion. This is achieved by taking into account information

from the spatial neighborhood of the pixels being corrected. Finally, our correction pipeline adds little latency in most cases and hence is suited for incorporation in applications where this is an issue. A future work perspective is the automatic detection of static scenes. Indeed, a mechanism to temporarily prevent the update of estimators when two consecutive frames are too similar would allow to preserve their correctly computed values when the convergence hypotheses are not met, further increasing correction robustness.

ACKNOWLEDGMENT

The authors would like to thank the DGCIS (French Ministry for Industry) for funding the project HiDRaLoN.

REFERENCES

- [1] H. Tian, B. Fowler, and A. Gamal, "Analysis of temporal noise in cmos photodiode active pixel sensor," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 1, pp. 92–101, jan 2001.
- [2] J. Nakamura, *Image Sensors and Signal Processing for Digital Still Cameras*. Boca Raton, FL, USA: CRC Press, Inc., 2005.
- [3] S. Morfu, P. Marquie, B. Nofiele, and D. Ginhac, "Chapter 3: Nonlinear systems for image processing," in *Advances in Imaging and Electron Physics*, P. W. Hawkes, Ed. Elsevier, 2008, vol. 152, pp. 79–151.
- [4] A. El Gamal and H. Eltouky, "Cmos image sensors," *IEEE Circuits and Devices Magazine*, vol. 21, no. 3, pp. 6–20, may-june 2005.
- [5] G. C. Richard M. Malueg, "Detector array fixed-pattern noise compensation," US Patent US 3 949 162, 04 06, 1976.
- [6] B. Ratliff, M. Hayat, and J. Tyo, "Generalized algebraic scene-based nonuniformity correction algorithm," *J. Opt. Soc. Am. A*, vol. 22, no. 2, pp. 239–249, Feb 2005.
- [7] T. Huang, L. Xiao-Lin, and S. Yang, "Fixed pattern noise suppression algorithm based on background modeling," *Procedia Engineering*, vol. 29, no. 0, pp. 884–888, 2012.
- [8] U. Sakoglu, R. Hardie, M. Hayat, B. Ratliff, and J. Tyo, "Non algebraic restoration method for estimating fixed pattern noise in infrared imagery from a video sequence," in *9th Annual Meeting of the SPIE: Applications of Digital Image Processing XXVII*, vol. 5558, Aug 2004, pp. 69–79.
- [9] J. Harris and Y.-M. Chiang, "Nonuniformity correction using the constant-statistics constraint: Analog and digital implementations," in *Proceedings of SPIE*, vol. 3061, 1997, pp. 895–905.
- [10] C. Zhang and W. Zhao, "Scene-based nonuniformity correction using local constant statistics," *J. Opt. Soc. Am. A*, vol. 25, no. 6, pp. 1444–1453, Jun 2008.
- [11] C. Zuo, Q. Chen, G. Gu, X. Sui, and W. Qian, "Scene-based nonuniformity correction method using multiscale constant statistics," *Optical Engineering*, vol. 50, no. 8, p. 087006, 2011.
- [12] A. Kumar, S. Sarkar, and R. Agarwal, "Fixed pattern noise correction and implementation for infrared focal plane array based staring system using scene statistics," *International Journal of Industrial and Systems Engineering*, vol. 1, January 2007.
- [13] Z. Wang and A. C. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, 2002.
- [14] "Ov9715 1-megapixel product brief," March 2011, omniVision Technologies, Inc.