A LIGHT VIEWFINDER PIPELINE FOR CONSUMER DEVICES APPLICATION

Sebastiano Battiato, Alfio Castorina, Mirko Guarnera, Filippo Vella

Advanced System Technology Catania Lab - STMicroelectronics
E-mail: {Sebastiano.Battiato, Alfio.Castorina, Mirko.Guarnera, Filippo.Vella}@st.com

ABSTRACT

The paper describes an image generation pipeline able to realize a “viewfinder”. The viewfinder of a typical handset device allows user to track in real time the scene under detection. The pipeline is composed by a set of blocks implementing demosaicing and image enhancement algorithms with new and efficient techniques able to reduce considerably the computational overhead. Experiments show how modest computational resources can be coupled with acceptable perceived image quality for this particular target.

1. INTRODUCTION

Consumer devices acquire digital images through digital sensors (e.g. CCD/CMOS) and input data is typically acquired by a Color Filtering Array (CFA) in the classical Bayer Pattern format [1]. High quality images are obtained with high-resolution sensor and using sophisticated image-processing algorithms ([4], [5] and [8]). The overall Image Generation Pipeline (IGP), aimed to reconstruct the final image, is complex and is oriented to exploit all the information acquired by sensor to achieve the as “best” as possible image [2], [7]. The light pipeline described in this paper is instead aimed to obtain, from the sensor data, a quick thumbnail of the final image to be shown on a display used as viewfinder. The proposed method works directly on CFA Bayer matrixed video sequences in order to save resources in terms of time and space. These features are particularly relevant for applications with real time constraints. The algorithm is tuned to be quick and to perform a reduced number of operation maintaining a good quality for the final image. A new IGP has been generated in order to obtain a suitable trade-off between perceived quality (in terms of sharpness, color hue & saturation, frame rate) and computational complexity. Typical LCD displays, used in consumer devices for a viewfinder application, use only a fraction (e.g. 1/3, 1/4...) of the original sensor resolution employing a reduced pixels depth. The proposed pipeline adapts computation to the final viewing conditions carrying out a suitable data elaboration just after the acquisition process. For example, scaling is not applied to the RGB planes at full resolution but is applied in a proper way to sensor data [3]. Experiments show the real effectiveness of the methodology with a great saving in terms of computation.

The rest of the paper is organized as follows. Next section after a brief overview of the overall system, describes the technical details of each single step involved in the pipeline. In section 3 experimental results show the effectiveness of the proposed technique. A conclusion section closes the paper tracking directions for future works.

![Figure 1 – Overall pipeline description.](image)

2. LIGHT VIEWFINDER

In order to implement a viewfinder pipeline able to guarantee a good “preview”, final sequence quality has been considered both in terms of refresh-rate and perceived image appearance. The overall computation is devised to be oriented to the final display resolution (depth/size): the final output is only a thumbnail. At the same time the viewfinder should give to the user a good perceived quality in terms of color rendition as balancing and matrixing. The underlying ideas in the proposed
pipeline group together different computations that can be executed as a single step and avoid redundant processing (e.g. benefits coming from an anti-aliasing algorithm would be negligible for such low resolution final images). The main proposed improvements could be summarized in the following list:

- White Balancing (WB) gains are not evaluated for each frame. The relative statistics collected are estimated sub-sampling the input data;
- Join together Scaling and Color Interpolation;
- Join together, in a single step, Color Matrixing and White Balancing;
- Gamma correction is done with a Look-up-table;
- The classical artifacts introduced by hard quantization are avoided using a smart anti-contouring technique.

The overall pipeline is schematically described in Figure 1. When the sensor acquires a scene, the White Balancing block processes the raw data. Considering that, between consecutive frames, light condition does not change significantly, the WB gain estimation is performed every $K$ frames (8, for instance). The raw data are then scaled and interpolated in a single step, deciding only the scaling factor (we have assumed as possible factors 3, 4 and 5). The colors of the scaled image are then corrected, adapting the spectrum to the human sensitivity, with a matrixing operation. Both Matrixing and White Balancing are linear algebraic operators, thus it is possible to combine them in a single step. At this point, a gamma operation is performed to adapt the colors to the display response, and finally a quantization with anti-contouring reduces the color planes.

### 2.1. White Balancing Gains computation

White Balancing algorithms take into account color casting in scenes taken under non-white illumination. The algorithm divides the Bayer pattern in 4-pixels blocks (each of them containing 1 R, 1 B and 2 G pixels) and considers for computation purpose one pixel for each channel. As displayed in Figure 2 this means that the G pixel ($\times$) will not be considered. White Balancing is achieved forcing the chromatic channel energy values to equal the maximum energy. In order to avoid excessive saturation of original colors, while computing the energy for a channel, values are weighted with the minimum value in the 4-pixel block and furthermore near saturation values are discarded. In detail three channel energies are computed using the following:

$$ce_i = \sum_{j=0}^{B} c_{ij} \cdot \min(c_{rj}, c_{bj}, c_{gj}) \cdot \alpha(T, c_{rj}, c_{gj}, c_{bj})$$

where $c_{ij}$ is the value of the $i$-channel of block $j$, $B$ is the number of total 4-pixels blocks, and $\alpha$ is a 4-ary operator whose value is 0 if all channels are greater than an user fixed threshold $T$, 1 otherwise. Adopting both spatial and temporal sampling a considerable amount of computations are avoided [2], [4]. In fact, since illumination is usually uniformly distributed across the image, the aforementioned elaboration is done sub-sampling the 4-pixel blocks across vertical and horizontal directions (see Figure 3). Due to the data pattern the sampling step in terms of pixels must be in the form $2N$. Furthermore in video sequences illumination conditions are expected to vary smoothly from frame to frame and channel gains update can be done only every $K$ frames.

$$B = \frac{M}{K} \times \frac{H \times V}{16 \times N^2}$$

### 2.2. Scaling and Color Interpolation

For a sequence composed by $M$ frames of dimension $H \times V$, combining the two speeding techniques the number of considered blocks $B$ is only:

Starting from the sensor raw data (i.e. in CFA format) this step builds a full color image, at a given scaled resolution. Conventional scaling techniques do not perform any type of color interpolation. Such classical techniques are usually applied on full RGB color images. Averaging or decimation techniques, that throw away original pixels in the scaled image, are not a good choice if applied directly to CFA images. We propose to realize scaling and color interpolation in a simultaneous step. In order to manage different display resolutions, several scaling factors have been considered. Changing the scaling factor, the kernel dimension for the interpolation operator must be changed, according to the Shannon’s theorem. Considering the
Bayer structure shown in Figure 4, in the case of a scaling x3 (using 3x3 kernel), from each quadrant \( Q_i \), \( i = 1, \ldots, 4 \) a triplet of values (RGB) will be calculated by the following formulas:

For Q1  \[
\text{Red}_1 = B2 \\
\text{Green}_1 = \frac{(B1 + A2 + C2 + D3)}{4} \\
\text{Blue}_1 = \frac{(A1 + C1 + A3 + C3)}{4} \tag{3}
\]

For Q2  \[
\text{Red}_2 = \frac{(D2 + F2)}{2} \\
\text{Green}_2 = \frac{(D1 + F1 + D3 + F3)}{4} \\
\text{Blue}_2 = \frac{(E1 + E3)}{2} \tag{4}
\]

For Q3  \[
\text{Red}_3 = \frac{(B4 + B6)}{2} \\
\text{Green}_3 = \frac{(A1 + C1 + A6 + C6)}{4} \\
\text{Blue}_3 = \frac{(A5 + C5)}{2} \tag{5}
\]

For Q4  \[
\text{Red}_4 = \frac{(D4 + F4 + D6 + F6)}{4} \\
\text{Green}_4 = \frac{(E4 + D5 + F5 + E6)}{4} \\
\text{Blue}_4 = E5 \tag{6}
\]

For each of these four cases the result is an usual RGB pixel.

The previous example could be used, for instance, to obtain a QCIF preview from a VGA sensor (with a scaling by 3). Scaling by 4 (from VGA to 160x120) and by 5 (from VGA to 128x96) are analogous. In the last case a 5x5 kernel is used (see figure 5). Similarly to the previous case, the corresponding formulas to obtain the missing data can be easily derived. It is important to note how in both cases, 3x3 and 5x5 kernel, the formulas implicitly implements suitable low pass filters for the image sub-sampling, taking into account for each channel, only the real data.

![Figure 4 – Bayer scaling in the x3 case.](image)

![Figure 5 – Bayer scaling in the x5 case.](image)

2.3. Matrixing and White-balancing

Color Matrixing is aimed to improve the color rendition and saturation of the image. In particular it corrects the spectral sensitivities of the image sensor accordingly to the chromaticities of the display and the characteristics of the human eye. This enhancement is achieved manipulating the color matrix table coefficients to be applied to each R-G-B color pixel. On computational side, Matrixing is a matrix product between the input color channels and a correction matrix, whose coefficients are fixed referring to the specific display characteristics. White Balancing correction and Matrixing have been coupled together, and thus performed as a single step by multiplying the Matrixing matrix by a diagonal one. Elements of the diagonal matrix are the White Balancing channel gains computed as described in the previous paragraph. Formulas (7) and (8) show how a single White Balancing/Matrixing matrix is derived and applied.

\[
\begin{bmatrix}
\begin{bmatrix}
G_r & 0 & 0 \\
0 & G_g & 0 \\
0 & 0 & G_b
\end{bmatrix} \cdot \\
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
1,1 & c_{1,2} & c_{1,3} \\
c_{2,1} & c_{2,2} & c_{2,3} \\
c_{3,1} & c_{3,2} & c_{3,3}
\end{bmatrix}
\end{bmatrix} = \\
\begin{bmatrix}
m_{1,1} & m_{1,2} & m_{1,3} \\
m_{2,1} & m_{2,2} & m_{2,3} \\
m_{3,1} & m_{3,2} & m_{3,3}
\end{bmatrix}\end{bmatrix}
\end{bmatrix} \tag{7}
\]

\[
\begin{bmatrix}
\begin{bmatrix}
R_g & G_g & B_g
\end{bmatrix} = \\
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
R_o & G_o & B_o
\end{bmatrix} \cdot \\
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
G_r & 0 & 0 \\
0 & G_g & 0 \\
0 & 0 & G_b
\end{bmatrix} \cdot \\
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
1,1 & c_{1,2} & c_{1,3} \\
c_{2,1} & c_{2,2} & c_{2,3} \\
c_{3,1} & c_{3,2} & c_{3,3}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} = \\
\begin{bmatrix}
m_{1,1} & m_{1,2} & m_{1,3} \\
m_{2,1} & m_{2,2} & m_{2,3} \\
m_{3,1} & m_{3,2} & m_{3,3}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix} \tag{8}
\]

Ideally values coming from White Balancing processing should be clipped in the output range before Matrixing. Avoiding the clipping phase between these two corrections, out of range values could be processed by Matrixing step creating false color effects in the final image. Actually this problem affects the less significant bits of pixels and this piece of information will anyway be cut by the quantization thus avoiding unpleasant effects in the final displayed image.

2.4. Gamma Correction

The luminance generated by a physical device is generally not a linear function of the applied signal. This non-linearity must be compensated in order to achieve correct reproduction of luminance. The Gamma correction is a nonlinear transformation applied to each pixel. To apply this function with a low computational cost a look-up-table is used where the \( i \)-th element is generated considering the relation:

\[
GC(i) = 255 \cdot \left( i \over 255 \right)^\gamma \tag{9}
\]

with a device-specific \( \gamma \).

2.5. Improved Gray Scale Quantization

A simple quantization of a channel from 8 to 4 bit, or less, produces a series of noticeable artifacts in the final image. A suitable algorithm able to avoid a brutal truncation of
preventing the creation of artifacts in the image is needed. To each pixel value is added a pseudo-random noise before truncation. In this way spatially near pixels with same values can have slightly different values in the output image breaking false contours due to the loss of the smooth degradation. The pseudo-random noise to be added is given by the less significative bits of the sum between the pixel value and the previously evaluated noise. The result of the improved quantization is an image where false contours are highly reduced. See also [6], [7] for more details.

3. EXPERIMENTAL RESULTS

In order to validate the proposed pipeline plenty of tests have been made on a real time acquisition system. In particular the overall scheme was implemented in a real-time framework, using a CMOS-VGA sensor on the “STV6500 - E01” Evaluation Kit equipped with a “502 VGA sensor” [9]. Besides considering the complexity of a typical IGP for High End Digital still cameras in term of simple operations such as: shift, add, mac, compare, multiply, it is correct to consider that about 300ops/pixel are needed, from the acquisition to the final RGB image (before the compression). Thus, for a VGA image about 90 Mops, are needed. Using our pipeline we have obtained (for a VGA image) in the case of x3 scaling about 310Kops., for x4 scaling about 950Kops. and for x5 scaling about 620Kops. The achieved performances allow embedding the overall processing in a real-time system. A snapshot of the system is is shown in Figure 6 where the application was implemented using a C-code interface to the sensor. The final quality of the image is not affected by color artifacts due to the ligther computation implemented while the relative frame-rate is sensibly improved.

![Figure 6 - Snapshot of the Viewfinder implementation.](image-url)

4. CONCLUSIONS

The proposed pipeline processes image from a sensor with Bayer CFA and swiftly creates, with a low computational cost a preview of the image. It can be used in any device to pre-view real time image sequences acquired by a sensor and it is suited to mobile device for its low power consumption. The view-finder has been achieved avoiding all redundant processing. The overall system is also optimized (e.g. White Balancing gains estimation are spatially and temporally subsampled). Some typical IGP steps are coupled and melted in a single block (e.g. Scaling and Color Interpolation are carried out together and the WB is applied during the Color Matrixing elaboration). Experiments have confirmed the effectiveness of the proposed pipeline.

Future works will include objective tests based on no-reference metrics to evaluate the quality loss with respect to a full IGP + Scaling.

5. REFERENCES

[5]. S. Battiato, A. Castorina, M. Mancuso, “High Dynamic Range Imaging: Overview and Application”, Accepted for publication: SPIE Journal of Electronic Imaging, November 2002;