Low-Overhead Adaptive Contrast Enhancement and Power Reduction for OLEDs

Daniele Jahier Pagliari, Massimo Poncino, Enrico Macii
Dipartimento di Automatica e Informatica, Politecnico di Torino, Turin, Italy
Email: {daniele.jahier, massimo.poncino, enrico.macii}@polito.it

Abstract—Organic Light Emitting Diode (OLED) display panels are becoming increasingly popular especially in mobile devices; one of the key characteristics of these panels is that their power consumption strongly depends on the displayed image. In this paper, we propose a new methodology to reduce the energy consumed by OLED displays that relies on image-specific pixel-by-pixel transformations, aimed at preserving the contrast of the image as much as possible while reducing the overall power. Unlike previous approaches, our method focuses specifically on the minimization of time and power overheads to implement the image transformation at runtime. To this end, we propose a transformation that can be executed online in real time, either in software, with low time overhead, or in a hardware accelerator with a small silicon footprint.

Despite the great reduction in complexity, our results are comparable to those achieved with more complex approaches in terms of image quality. Moreover, our method allows to easily explore the full quality-versus-power tradeoff by acting on a few basic parameters; thus, it enables the runtime selection among multiple display quality settings, according to the status of the system.

I. INTRODUCTION

The energy consumption of the display subsystem is known to be a significant contributor to the total energy consumed by embedded devices [1]. Organic Light Emitting Diode (OLED)-based displays are increasingly used as an alternative to classic Thin Film Transistor (TFT) LCDs for a variety of reasons, including better viewing angles, higher brightness, and the possibility of building thinner and flexible panels [2]. Perhaps the most important advantage of OLEDs is that these devices are emissive, and therefore do not require an external light source. This opens a new optimization dimension for the reduction of energy consumption, as the latter becomes strongly dependent on the images being displayed [3].

Several research works have focused on exploiting the image-dependent power consumption property of OLED panels [3]–[8]. Some, being specifically targeted at Graphical User Interfaces, are not applicable to other use cases such as displaying pictures or videos [3], [4]. Others require modifications of the analog hardware of the panel, thus are not suitable for systems using off-the-shelf displays [6]. Finally, some techniques are appropriate for general images and can be implemented in software or accelerated in hardware [5], [7], [8]. However, all of them involve computationally intensive algorithms.

The latter group of techniques is based on transforming an input image I into an output image O = T(I) so that the power consumed by O is less than that consumed by I, while visual quality is preserved as much as possible. In these approaches, very little attention has been devoted to the analysis of the time and energy overheads for implementing the transformation T. Since the parameters of T are image-dependent, an offline implementation is inherently suboptimal, unless the content to be displayed is known a priori for the entire lifetime of the system. For an online application, instead, the parameters of T have to be determined in real time, i.e. every time a new image is stored in the frame buffer. Since most modern displays are refreshed at a frequency of 50-60 Hz, the entire process has to be repeated every 15-20 ms. Moreover, since OLED panels allow faster response times, time constraints can be even tighter [2].

Previous approaches in this category involve either iterative nonlinear optimizations [5] or complex image histogram processing [7], [8]. Implementing these operations in software could consume a significant percentage of CPU time, eating up processing power for other tasks. Alternatively, a dedicated processor could be used, but in both cases, the energy overhead for the computation of parameters and evaluation of T could drastically reduce the total achievable savings.

In this paper we propose a low-cost alternative for OLED displays energy optimization, that can be implemented in software with low overhead, or accelerated in hardware with a small silicon footprint. Our approach is based on two phases (see Fig. 1). A first, computationally intensive training phase is performed offline; the information extracted during training is then used to reduce the complexity of the second phase, which consists in the application of a simplified image transformation. This operation is performed online, but has a linear complexity with respect to the size of the panel. The novelties introduced in this paper are the following:

- We demonstrate that the proposed low-cost image transformation algorithm is able to reach similar savings to those
achieved with more complex techniques (e.g. [5]), with comparable visual quality.

- Unlike previous works, we thoroughly analyze the time and power overheads to implement the image transformation at runtime. In particular, we quantitatively measure the area, time and power overheads that result from a hardware implementation of the proposed transformation.

II. BACKGROUND AND RELATED WORK

In transmissive TFT-LCDs the main source of power consumption is the Cold Cathode Fluorescent Lamp (CCFL) used as backlight, which accounts for 80% or more of the total display subsystem power [9]. The intensity levels and pixel colors of the image being displayed have a small impact. Consequently, most energy reduction techniques for TFT-LCDs rely on some form of backlight scaling [9]–[12]. Image transformations are commonly used, but their purpose is to compensate for the brightness reduction due to the changed backlight level. OLED pixels, conversely, are formed by the combination of three types of emissive devices, corresponding to the red (R), green (G) and blue (B) components of the RGB color space [3]. The power consumed by each device increases with the intensity of the corresponding component. Consequently, the power consumption of an OLED panel depends on the brightness of the displayed image, and secondarily on the balance between color components. Measurements performed on real panels in [3] allowed to build a non-linear model for the power consumption of an OLED display:

\[ P_{tot} = \sum_{i=0}^{W} \sum_{j=0}^{H} (w_0 + w_r \cdot R_{i,j}^3 + w_g \cdot G_{i,j}^2 + w_b \cdot B_{i,j}^3) \] (1)

In Eq. 1, \( W \) and \( H \) are the width and height of the panel respectively, and the triplet \((R_{i,j},G_{i,j},B_{i,j})\) represents the sRGB components of the pixel at position \((i,j)\). The coefficients \( w_r, w_g, w_b \) are display-dependent and can be obtained via characterization [3]. In most cases \( \gamma \geq 2 \).

The authors of [3] proposed several techniques based on this model to optimize consumption in mobile GUIs. When building a transformation for GUIs, the most important metric to preserve is usability, e.g. text elements must be distinguishable from the background, rather than visual fidelity. Therefore, their approach is based on drastically changing colors, and is not applicable to general images and videos, for which fidelity is fundamental for the perceived quality. Similar considerations apply to [4], where power reduction is obtained by selectively dimming the pixels outside the area of user interest, (e.g. the window with active focus in a desktop).

In [6] concepts similar to LCD backlight dimming are applied to OLED displays; power is reduced by applying Dynamic Voltage Scaling (DVS) to the OLED panel, using an appropriate driver circuit. Since DVS limits the maximum brightness that can be produced by the emissive devices, a compensation must be performed acting on the pixel values. Although this technique is effective and applicable to any kind of image, it requires custom analog drivers and control circuits, which are not available in off-the-shelf OLED displays.

In [5] and [7] two generally applicable methods that only leverage image transformations are presented. [5] uses non-linear optimization algorithms to find a pixel transformation that concurrently reduces power and improves contrast. Although this method is effective, the operations involved in the identification of the optimal transformation are computationally expensive (e.g. solution of Karush-Kuhn-Tucker conditions, computation of the secant formula, etc.). The work in [7] uses Histogram Shrinking (HS) combined with contrast enhancement to obtain a visually good image with reduced consumption. Also in this case, the technique involves expensive iterative procedures, with a high cost in terms of power and time overheads.

III. LOW-COST LUMINANCE TRANSFORMATION

We propose an alternative to the techniques mentioned in Sec. II, showing that a simple transformation \( T \) that maps each input pixel to a corresponding output pixel according to a polynomial function can achieve significant power reduction with acceptable visual quality preservation. We focus only on the luminance of the pixels of a color image, without altering the chroma components, because especially for pictures and videos, color alterations affect too much the perceived visual quality. Most systems internally represents images in RGB format. Therefore, the complete transformation is composed of three steps: (1) the image is converted from RGB to YUV color space; (2) the “Y” component of each pixel (i.e., the luminance) is transformed according to a polynomial function; (3) the resulting image is converted back to RGB. RGB-YUV conversions can be performed with a linear operation [13]:

\[ (Y_{i,j},U_{i,j},V_{i,j})' = C \cdot (R_{i,j},G_{i,j},B_{i,j})' \] (2a)

\[ (R_{i,j},G_{i,j},B_{i,j})' = K \cdot (Y_{i,j},U_{i,j},V_{i,j})' \] (2b)

where \( C, K \in \mathbb{R}^{3 \times 3} \) are constant coefficients.

A. Derivation of the Generic Transformation Function

Power reduction in OLED displays can be trivially achieved by decreasing the pixels luminance. In fact, according to Eq. 1 and 2a, it follows that \( P(Y) \propto Y^\gamma \). However, pure “scaling” of pixel values degrades significantly the perceived quality. For this reason, most literature works combine a reduction of the total luminance with an enhancement of the contrast [5]–[7]. Our approach pursues a similar objective. We first observe that the intensity transformation curves \( Y_i = T(Y) \) obtained in previous works all exhibit a non-monotonic concavity [5], [7]. This happens because of the need to alter in different ways the dark and bright areas of the image. Secondly, we consider the fact that a polynomial transformation uses only additions and multiplications, which are among the least expensive operations to be implemented either in HW or SW.

We therefore select as generic pixel-by-pixel intensity transformation law, the simplest polynomial that can vary in concavity, i.e. a third order one. In other words, we transform the luminance of each pixel in the target image according to the following equation:

\[ Y_i = T(Y_{i,j}) = a_3 Y_{i,j}^3 + a_2 Y_{i,j}^2 + a_1 Y_{i,j} + a_0 \] (3)
This transformation requires only 6 multiplications and 4 additions per pixel.

The coefficients $a_x$ in Eq. 3 must be set to a numeric value that minimizes power while enhancing contrast. To reduce the degrees of freedom, the general expression for $T$ can be simplified by imposing some additional constraints, similar to those in [5]. Since we want to preserve contrast, we can impose that the transformation spans the full luminance range:

$$T(Y_{\text{min}}) = Y_{\text{min}}, \quad T(Y_{\text{max}}) = Y_{\text{max}}$$

where $Y_{\text{min}}$ and $Y_{\text{max}}$ are the minimum and maximum luminance values. In YUV space, luminance spans the $[0:1]$ interval. Therefore, Eqs. 4 yield:

$$a_0 = 0, \quad a_3 = (1 - a_2 - a_1)$$

$$Y_t = T(Y_{t,j}) = (1 - a_2 - a_1)Y_{t,j}^3 + a_2Y_{t,j}^2 + a_1Y_{t,j}$$

Another important property of $T$, as pointed out in [5], is that it should be monotonically increasing, to avoid the creation of artifacts due to inversions of luminance relations between the input and the output images. This constraint sets a relation between the two remaining coefficients $a_1$ and $a_2$, obtained imposing that the first order derivative $T'$ is always positive or zero in the entire luminance range. $T'$ is identified geometrically by a parabola; the four non-degenerate sets of curves that correspond to a monotonically increasing $T$ are shown in Fig. 2, where $y_1$ and $y_2$ indicate the two solutions of the parabola equation.

All four curves satisfy the imposed constraints. However, deriving $a_1$ and $a_2$ from $y_1$ and $y_2$ in CASE1 and CASE4 yields unbounded intervals (semiplanes). Therefore, those cases do not define a finite domain for the search of the optimal transformation coefficients. Consequently, we limit our optimization to the domains defined by the two curves labeled CASE2 and CASE3, which instead impose bounded intervals on both $a_1$ and $a_2$. Calculations result in the following regions:

$$R2 : \{0 \leq a_1 \leq 2 \quad 1 - a_1 < a_2 < 3 - 2a_1\}$$

$$R3 : \left\{\begin{array}{l}
0 \leq a_1 \leq 4 \\
-3a_1 - \sqrt{3(4a_1 - a_1^2)} \\
-3a_1 + \sqrt{3(4a_1 - a_1^2)} \leq a_2 \leq -3a_1 + \sqrt{3(4a_1 - a_1^2)}
\end{array}\right. \quad (7b)$$

But all considerations can be easily extended to equivalent color spaces such as YCbCr, in which luminance has a different range of values.

The union of R2 and R3 is the largest bounded area of $R^2$ for which $T$ is monotonically increasing.

B. Offline Training Algorithm

Having determined the equation for the generic transformation $T$, an exhaustive search algorithm can be used to identify the optimal values of the free parameters $a_1$ and $a_2$ for a given image. We use this approach to analyze a set of training images, and to identify a potential relation among the optimal parameters of $T$ and some simple quantitative features of the image. In this phase, we minimize a cost function $F$ which is the weighted sum of a power component $P$ (Eq. 1) and a contrast component $\sigma$. As simple measure of the contrast, we use the standard deviation of the luminance histogram [14].

In the search for the optimal $a_1$ and $a_2$, we must also impose a constraint on the maximum image alteration allowed. To measure it, we use the Mean Structural Similarity Index (MSSIM) [14] between the input and output images. During training, we consider valid only those solutions for which $\text{MSSIM} \geq \text{MSSIM}_{\text{min}}$. The pseudocode for the training phase is reported in Fig. 3, where $a = [a_2 \ a_1]$, $w_p$ is a weighting coefficient that allows to balance the power and contrast components, and $\mu$ represents the mean of the luminance histogram. The additional constraint $P(Y_t) \leq P_{\text{opt}}$ in Line 9 ensures positive power savings.

C. Linear fitting of the Transformation Parameters

In Sec. III-B we computed the optimal parameters of $T$ for a set of training images. To obtain an adaptive transformation that is flexible but simple enough to be amenable for online usage, these parameters have to be put in relation with elementary information about the image being processed. In particular, we select $\mu$ and $\sigma^2$, which represent the brightness and contrast of the image respectively [14].

We propose to compute a linear fitting that relates $a$ to $\mu$ and $\sigma^2$ using as data points the results of the training phase. A linear model is chosen because of its low evaluation complexity. The Least Squares Method yields coefficients $P \in R^{2 \times 2}$ and $q \in R^{2 \times 1}$ that identify the regression plane:

$$a = P \cdot \begin{bmatrix} \mu \\ \sigma^2 \end{bmatrix} + q$$

1\text{But all considerations can be easily extended to equivalent color spaces such as YCbCr, in which luminance has a different range of values.}
As shown in Sec. IV this simple model, although subject to a certain amount of error, is sufficient to obtain transformations that are very similar to the optimal ones, from both visual inspection and quantitative analysis.

D. Online Image Transformation

Having determined the fitting coefficients that link the basic features of an image \((\mu, \sigma^2)\) to the optimal values of \(a\), we can adapt the pixel-by-pixel polynomial transformation expressed by Eq. 6 to the content of the OLED display at runtime. The sequence of operations that transforms an input image \(I\) into an energy-efficient output image \(O\) is shown in Fig. 4. Notice that each block in the diagram requires a number of operations which linearly depends on the number of pixels in the panel, except for the computation of \(a\), which requires a single evaluation of Eq. 8 per image. Also, all blocks perform only additions and multiplications; the divisions required to single evaluation of Eq. 8 per image. Also, all blocks perform only additions and multiplications; the divisions required to inversion, since the divisor is constant \((W \cdot H)\). Moreover, if we accept to convert the images from RGB to YUV before storing them in the frame buffer, the online transformation does not require additional memory, except for a few registers to store intermediate coefficients. Overall, such a scheme is significantly simpler than those proposed in previous works [5], [7].

The only inputs to the architecture in Fig. 4 are the original intermediate coefficients. Overall, such a scheme is significantly simpler than those proposed in previous works [5], [7]. The only inputs to the architecture in Fig. 4 are the original image \(I\) and the fitting coefficients \(P\) and \(q\). The latter can be changed at runtime to obtain different transformations, according to the conditions of the system (e.g. battery status).

IV. EXPERIMENTAL RESULTS

We implemented the training algorithm (Fig. 3) in C, while fitting coefficients (Eq. 8) have been determined in MATLAB. For estimating the power consumption in the panel (Eq. 1) we used the same coefficients of [5], i.e. \((\gamma, w_0, w_r, w_y, w_h) = (2.2, 0, 70, 115, 154)\). In all training runs, we set:

\[
w_p = \frac{1}{W \cdot H(w_0 + w_r + w_y + w_h)255^\gamma}
\]

(9)

where 255 is the maximum value representable in 8-bit RGB. This weighting coefficient normalizes the power component with respect to the maximum power of an image of the same size \((W \cdot H)\), and guarantees a good balance between power reduction and contrast enhancement. We explored regions \(R_2\) and \(R_3\) in discrete steps of 0.05 for both dimensions \((a_1\) and \(a_2\)), since smaller discretizations do not produce a significant difference in terms of power saving and visual quality.

### TABLE I: Comparison with PCCE [5]

<table>
<thead>
<tr>
<th>Image</th>
<th>Saving [%]</th>
<th>MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCCE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Babaon</td>
<td>61.6</td>
<td>0.692</td>
</tr>
<tr>
<td>Lena</td>
<td>59.5</td>
<td>0.799</td>
</tr>
<tr>
<td>F-16</td>
<td>60.3</td>
<td>0.795</td>
</tr>
<tr>
<td>Ideal a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Babaon</td>
<td>69.5</td>
<td>0.694</td>
</tr>
<tr>
<td>Lena</td>
<td>51.9</td>
<td>0.870</td>
</tr>
<tr>
<td>F-16</td>
<td>60.3</td>
<td>0.833</td>
</tr>
<tr>
<td>Fitted a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Babaon</td>
<td>69.4</td>
<td>0.691</td>
</tr>
<tr>
<td>Lena</td>
<td>55.1</td>
<td>0.860</td>
</tr>
<tr>
<td>F-16</td>
<td>67.4</td>
<td>0.751</td>
</tr>
</tbody>
</table>

A. Comparison with PCCE

One of the most effective algorithms for the combined power reduction and contrast increase in OLED display panels is the Power-Constrained Contrast Enhancement (PCCE), proposed in [5]. As mentioned in Sec. II, the main drawback of this algorithm is its computational complexity. Figs. 5b show the outputs of PCCE, as reported in [5], when applied to three popular benchmark images: Baboon, Lena and F-16. The main PCCE parameter, \(\beta\), is set to 1.5. We computed the MSSIM between these outputs and the correspondent unprocessed images (Figs. 5a). Then, we set those MSSIM values as constraints in the training phase of our algorithm, to compare it with PCCE in similar quality conditions.

Results are shown in Figs. 5c and 5d. In the first set of images, the transformation coefficients \((a)\) have been computed using directly the algorithm of Fig. 3; therefore, they are the ideal parameters for each of the three images. For Figs. 5d, instead, we used the values of \(a\) produced by the regression equation (Eq. 8). Regression coefficients \(P\) and \(q\) were computed fitting the results of the training algorithm, run on a set of 40 standard images (from the Kodak database [15]) that did not include the three targets, under the same MSSIM constraint. From visual inspection of Fig. 5 the similarity between PCCE and our algorithm is evident. Moreover, the polynomial transformations with ideal and fitted parameters are almost identical, except for the F-16 image, where in the fitted version some details in the mountains area are lost. This happens because that image has very peculiar characteristics, i.e. a large brightness \((\mu)\) and relatively small contrast \((\sigma^2)\). Due to the scarcity of similar images in the training set, fitting yields a larger error in these conditions.

Both PCCE and our algorithm are visually superior to a simple uniform scaling (multiplication with a \(k < 1\)) of all luminance values. To show this, we computed the scaling factor \(k\) that for each image produces a power saving approximately equal to that achieved with our algorithm. The scaled images look dull and unclear, as shown in Figs. 5e.

The quantitative data reported in Table I also confirm the goodness of our approach. For similar values of MSSIM index, our algorithm produces power savings which are comparable to those of PCCE, and superior in some cases, despite the significant difference in complexity. Notice that this accuracy would not be achievable with an image-independent transformation, that uses the same \(a\) coefficients for all images (e.g. the average of training results). We verified that such transformation fails both in respecting the imposed quality.
constraint and in maximizing the power saving, when tested on images outside of the training set.

B. Quality versus Power Tradeoff

PCCE and other popular algorithms do not allow to directly set a quality constraint for the image transformations [5], [7]. Different quality settings can be obtained acting on the algorithms parameters, but this requires a non-trivial tuning. Instead, in our algorithm, a quality metric is directly set as input constraint to the training phase (MSSIM). This allows to easily build a set of Pareto points in the quality vs. power design plane, corresponding to different values of the fitting coefficients. Quality/power settings can be switched at runtime, according to the system status (e.g. battery status).

To highlight this flexibility, we trained the algorithm with the same 40 images of the previous experiment, setting three MSSIM constraints: 0.95, 0.85 and 0.75. We calculated regression coefficients for each of these conditions ($P_{95}$, $P_{85}$, $P_{75}$, and $q_{95}$, $q_{85}$, $q_{75}$). Fig. 6 shows the results of applying the polynomial transformation to two images excluded from the training set. The above-mentioned regression coefficients have been used to obtain a different $a$ for each image and MSSIM condition. As before, the last column contains the output of a simple luminance scaling, with a $k$ that guarantees the same power saving as the case MSSIM=0.75. Notice how the contrast and details (e.g. in the butterfly wings) are preserved better using the third-order transformation.

The actual MSSIM indexes (computed after the transformations) and the corresponding power savings are reported in Table II. Desired and actual MSSIM can differ because of the errors introduced by fitting. However these errors are negligible, confirming that our algorithm allows to obtain a measured quality which is very close to the desired value. The second image of Fig. 6 (Android home screen) highlights the generality of our approach, which yields good results also when applied to a GUI. In particular, the icons and text are much more contrasted and visible in Fig 6d than in Fig 6e, for identical power savings. Notice that in this experiment, the transformation has been trained with general purpose images, characterized by a spread intensity histogram (typically bell-shaped). This screenshot has a relatively similar histogram, hence we obtain good results. Clearly, quality will worsen if the input GUI has a strongly multimodal histogram, with few distinct intensity values [12]. This is mainly due to the absence of training data with that type of histogram, and could be overcome by running another training phase with sample GUIs. However, we don’t expect to obtain better power savings than those produced by approaches specifically targeted at that type of images, such as [3], [4]. In fact, due to the continuous nature of the third order polynomial on which our transformation is based, it alters ranges of pixel intensities rather than single values, preserving fidelity and avoiding artifacts. While this might reduce the power saving opportunities on some GUIs, in which most consumption is due to few histogram bars, it makes our strategy generally applicable also to pictures and videos.

C. Implementation of the Online Intensity Transformation

To better show the advantages of our approach, we implemented the operations depicted in Fig. 4 both in software and hardware. The software version was written in C language, as a single-threaded sequential program, without any optimization. We compiled it with GNU C Compiler for a high-end x86_64 platform (Intel Xeon E3-12, 8MB L3, Linux Kernel v. 2.6). The execution time has been measured with the Linux real time clock library, averaging the results of 10000 runs. To meaningfully compare the execution time with that of [5] we considered a 512 x 512 image, and we only measured the steps of the polynomial transformation (purple and orange)

### Table II: Quality versus power tradeoff.

<table>
<thead>
<tr>
<th>Image</th>
<th>MSSIM</th>
<th>Saving [%]</th>
<th>Out MSSIM</th>
<th>Saving [%]</th>
<th>Out MSSIM</th>
<th>Saving [%]</th>
<th>Out MSSIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Butterfly</td>
<td>0.95</td>
<td>21.3</td>
<td>0.960</td>
<td>45.2</td>
<td>0.843</td>
<td>59.7</td>
<td>0.773</td>
</tr>
<tr>
<td>Android</td>
<td>0.85</td>
<td>25.4</td>
<td>0.977</td>
<td>45.3</td>
<td>0.849</td>
<td>54.4</td>
<td>0.777</td>
</tr>
</tbody>
</table>

2All operations in Fig 4 have data independent complexity, so these results are valid for any image of the same size.
blocks in Fig. 4), neglecting color space conversions. As shown in Table III, under these conditions the transformation takes less than 1 ms to complete, a significant speedup with respect to the 6.23 ms reported by [5]. Obviously, for larger images, the execution time of the SW increases proportionally; a 1280x720 HD image (not considered in [5]) requires 5.41 ms, still below the time reported for PCCE. In a realistic scenario, however, the transformation would not be executed on a high-end processor. Therefore, we designed a dedicated hardware module to implement it. The design was specified in RTL using VHDL and synthesized with Synopsys DC v2011.09; we set the clock frequency to 500MHz, and targeted a 45nm CMOS standard-cell library by STMicroelectronics. The execution time of the synthesized HW was evaluated by means of simulation using Mentor Modelsim SE 6.4a. Power consumption was estimated loading the switching activity annotated by Modelsim in Synopsys PT F-2011.12. Results are shown in Table III. In this case, to obtain representative data of a realistic application, we considered also the conversion from/to RGB to/from YUV (which is included in the hardware module). The power consumption considers both dynamic and leakage contributions. In the HW, we used 16-bit fixed-point representation for non-integer quantities (a, P, q, etc.). With a benchmark of images, we confirmed that this representation ensures an average error of less than 0.5% on the transformed pixels with respect to the SW version leveraging double precision floating-point variables. To minimize power consumption and silicon footprint, the hardware architecture processes a single pixel at a time. Still, it is able to complete the transformation in less than 2ms for HD images. This result is sufficient to guarantee real-time applicability with standard display refresh rates (see Sec. 1). Moreover, since operations on different pixels are mostly independent, hardware parallelization can be easily exploited to reach much better performance if needed, at the expense of an increase in power consumption.

**V. CONCLUSIONS**

We have shown a potential alternative to the currently available algorithms for OLED displays power optimization, that poses particular attention to the overheads of an online implementation. We have demonstrated the competitiveness of this solution through comparison with state-of-the-art algorithms on standard general purpose test images. Moreover, we have demonstrated that a low-cost hardware accelerator for the online part of the algorithm can be synthesized, to be inserted in embedded SoCs for systems that leverage OLED technology. Future work will include the evaluation of the complexity versus accuracy tradeoff obtainable using a more complex polynomial for the luminance transformation, or higher order models for the fitting of transformation parameters.

**REFERENCES**


