Crayon: Saving Power through Shape and Color Approximation on Next-Generation Displays

Phillip Stanley-Marbell
MIT
psm@mit.edu

Virginia Estellers
UCLA
vestellers@cs.ucla.edu

Martin Rinard
MIT
rinard@csail.mit.edu

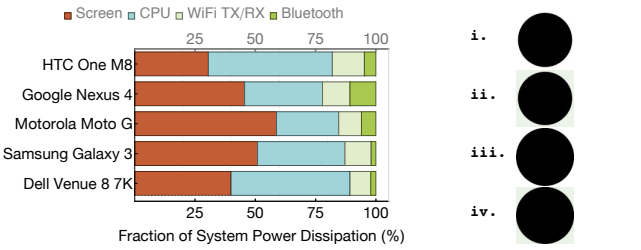
Abstract

We present Crayon, a library and runtime system that reduces display power dissipation by acceptably approximating displayed images via shape and color transforms. Crayon can be inserted between an application and the display to optimize dynamically generated images before they appear on the screen. It can also be applied offline to optimize stored images before they are retrieved and displayed. Crayon exploits three fundamental properties: the acceptability of small changes in shape and color, the fact that the power dissipation of OLED displays and DLP pico-projectors is different for different colors, and the relatively small energy cost of computation in comparison to display energy usage.

We implement and evaluate Crayon in three contexts: a hardware platform with detailed power measurement facilities and an OLED display, an Android tablet, and a set of cross-platform tools. Our results show that Crayon's color transforms can reduce display power dissipation by over 66% while producing images that remain visually acceptable to users. The measured whole-system power reduction is approximately 50%. We quantify the acceptability of Crayon's shape and color transforms with a user study involving over 400 participants and over 21,000 image evaluations.

1. Introduction

Displays account for a significant fraction of total system power dissipation in mobile platforms such as smart watches, phones, and tablets [12, 14, 54, 72] (Figure 1(a)). Because display power dissipation is dominated by analog electronic and optoelectronic effects which do not scale with improvements in digital semiconductor processes, the relative fraction of system power consumed by displays is likely to remain constant or even increase in the future.



(a) Displays dissipate a large fraction of total system power when active.

(b) Example: Shape and color transforms.

Figure 1. (a): Displays dissipate a large fraction of system power. **(b)** Changes in shape (**b.i** versus **b.iii**) are barely perceptible; **b.iii** however causes 44% lower power dissipation than **b.i** on OLED displays. With both shape and color transforms, **b.iv** reduces display power by 67% compared to **b.i**.

Modern organic light-emitting diode (OLED) [8] displays and digital light processing (DLP) [66] pico-projectors can offer significant advantages over traditional display technologies such as LCDs. Because OLED displays do not include a separate backlight, they are thinner and lighter than LCDs [1]. OLED displays support up to three orders of magnitude higher refresh rates than LCDs [44], while DLP pico-projectors can enable fundamentally new applications such as automotive heads-up displays [62] and other portable and head-mounted display applications [63]. And OLED and DLP displays can provide better power efficiency for many use cases. Because of these advantages and others, they are increasingly deployed in production devices.

In contrast to traditional display technologies, the power dissipation of OLED and DLP displays depends on the specific displayed colors (blue pixels in OLED displays typically dissipate twice as much power as green, with red in between).[1] At the same time, the human visual system exhibits great tolerance for certain kinds of shape and color changes (Figure 1(b)). Over the last six decades, researchers have established that most people resolve limited levels of hue and of brightness [26] and easily tolerate changes in the areas

[1] Because power dissipation in traditional display technologies (i.e., LCDs) is dominated by backlights [13, 16–18], their power dissipation is largely independent of displayed image color content.

EuroSys '16, April 18-21, 2016, London, United Kingdom

DOI: http://dx.doi.org/10.1145/http://dx.doi.org/10.1145/2901318.2901347

of graphical objects [25]. The human visual system also exhibits great perceptual flexibility to a variety of other visual changes [7, 20, 28, 33, 42, 67]. This flexibility, in combination with the color-dependent power dissipation properties of modern OLED and DLP displays, opens up new opportunities to trade small color changes in return for large reductions in display power dissipation.

1.1 Crayon

We present Crayon, a new system that exploits the flexibility of the human visual system to reduce display power dissipation while preserving acceptable display color accuracy. Unlike previous work, which is based on algorithms that explicitly target offline image optimization [2, 13, 16–19, 22–24, 35, 36, 51, 53, 68], Crayon is designed for both static offline and dynamic online optimization:

Efficiency: Crayon exploits the fact that the power dissipation function for OLED displays can be modeled with a low-order polynomial to obtain a closed-form representation of the optimum color transform (Section 2). The closed-form representation is orders of magnitude more efficiently-computable than previous techniques: Crayon can transform an image in milliseconds as opposed to hours as reported for previous systems [24]. This efficiency makes it possible to use Crayon not just for offline image optimization but also for online optimization of dynamically-generated bitmap and vector drawing content. Our successful integration of Crayon into the Firefox web browser (via the Cairo C/C++ graphics library [9, 70, 71]) demonstrates the viability of this approach.

Shape and Color: Because Crayon intercepts standard drawing API calls (as opposed to only working with images after rasterization), it has access to shape and color information available via these API calls. Crayon exploits this information to implement a new class of shape transforms unavailable to previous algorithms that operate at the level of discrete pixels [23]. Because the Crayon transforms operate before rasterization, the transformed images are still rendered by the GPU subsystem.

Static Offline and Dynamic Online Optimization: Figure 2 illustrates both static offline and dynamic online Crayon optimization. For offline optimization, Crayon takes an image file as input and produces a transformed image file that reduces display power dissipation. For online optimization, Crayon intercepts Cairo API calls to transform the color and shape content of graphics.

1.2 Evaluation

Our evaluation considers several aspects of Crayon: the display power reductions it enables, the compute overhead required to obtain these reductions, the subjective acceptability of the transformed images (as measured by participants in an Amazon Mechanical Turk user study), and the relationship between subjective human image evaluations and quantitative image quality measures such as MSE, PSNR, and SSIM [69].

Display Power Reductions: We work with a dedicated hardware platform to obtain a model for display power dissipation. This platform contains dedicated onboard power monitors that enable us to measure the display power dissipation as a function of the displayed color. We characterize Crayon's power savings by applying this detailed display power model to Crayon-optimized images in our user study.

Crayon Overheads: We measure the Crayon transform overhead with a combination of DTrace [10] and logging. Our measurements indicate that the Crayon image color transform requires less than 1 μs of compute time per pixel on a current-generation Android tablet [21]. The Crayon shape transforms impose a less than 14% compute time overhead (measured against a baseline Cairo implementation). We note that because Crayon intercepts and modifies all drawing API calls before GPU rendering, Crayon still realizes all of the benefits of GPU-accelerated graphics for shape transforms.

Our measurements indicate that for an XGA image (1024×768 pixels) displayed on a current-generation Android tablet [21], the display energy savings more than offset the Crayon color transform energy cost when the image is displayed for at least two seconds. For shape and color transforms on vector images, the energy break-even display time can be as short as one millisecond. Crayon is therefore currently effective for content that is displayed for milliseconds to seconds at a time, as typically occurs in interactive use. For these devices and use scenarios, the display energy savings more than offset the energy required to perform the Crayon computations (Section 5).

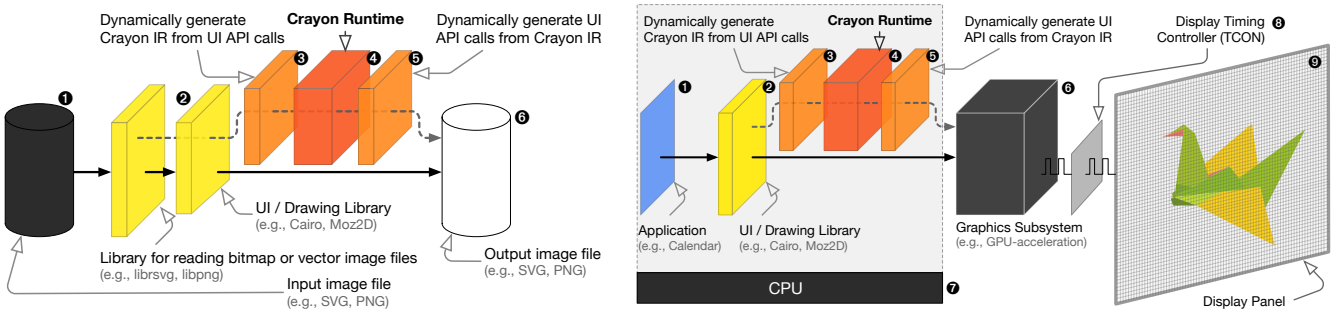
User Study: Our user study involved over 400 human participants (Section 4), who together performed over 20,000 evaluations of Crayon-transformed images. The results show that Crayon delivers significant power savings (typically 40–60% display power savings) with acceptable image color approximation.

Quantitative Image Quality Metrics: We compare the user image quality evaluations to three quantitative measures of image quality applied to the same images used in the study: MSE, PSNR, and SSIM [69]. The results show that, in general, the three quantitative measures are correlated with the user study results. But the correlation is far from exact — the user evaluations and quantitative measures disagree for many transformed images.

1.3 Contributions

This work makes the following five contributions:

❶ **Efficient Color Transforms:** It presents efficient closed-form transforms for acceptable color approximation (Section 2). These transforms are the first transforms efficient enough to enable dynamic online image optimization for OLED display power reduction.



(a) Static offline optimization: Crayon transforms bitmap and vector image files.

(b) Dynamic online optimization: Crayon intercepts drawing API calls to transform shape, color, and bitmap image content.

Figure 2. (a): Static offline optimization applied to bitmap and vector content in files. (b): Dynamic online optimization applied by intercepting drawing API calls for vector and raster drawing.

❷ **Shape Transform:** It presents a technique for perception-aware shape approximation. The technique is the first to exploit object shape and color information to reduce display power dissipation. The experimental results show that this transform is effective for both online and offline optimization of drawing API calls and vector image files.

❸ **Architecture and Implementation:** It presents an architecture and implementation for dynamically intercepting drawing API calls to apply perception-aware shape and color approximation. Our implementation of the entire Crayon system is currently ~14 k lines of C code. The current implementation runs on desktops and on an Android tablet. As one demonstration of the architecture, we show how to integrate Crayon into the Cairo drawing API [70] via the addition of approximately 50 lines of glue logic C code (Section 3).[2]

❹ **User Study:** It presents a comprehensive user study involving over 400 participants (Section 4). The study involves more than an order of magnitude more participants than previous work [2, 24, 30, 35, 61]. We augment the qualitative user study results with three quantitative measures for image quality assessment (PSNR, MSE, and SSIM [69]).

❺ **Power and Energy Savings:** It presents experimental results that characterize the power savings from Crayon's transforms. The results show that Crayon can acceptably reduce display power dissipation by over 66% and whole-system power dissipation by approximately 50%.

We also present experimental results that characterize the computation overhead of the current Crayon implementation. For color transforms of XGA-sized bitmap images, the display energy savings more than offsets the Crayon transform overhead when the image is displayed for at least two seconds. For transforms of vector images, the display energy savings more than offset the Crayon transform overhead when the image is displayed for as little as one millisecond.

Energy consumption and battery lifetime are central issues for current and future mobile devices. Display power dissipation is one of the primary contributors to the energy consumption of these devices. By enabling the first dynamic online optimization of display content for display power reduction (as well as fast and effective static offline image optimization), Crayon can significantly extend the utility of modern mobile devices.

2. Crayon Color Transforms

The power dissipation of OLED displays and DLP projectors is a function of the displayed image. Crayon exploits this fact to trade image fidelity for power savings by approximating the original image to minimize display power dissipation while preserving image acceptability.

2.1 Formulation of the color transform

Let v be an N-pixel RGB image with color channels r, g, and b. We model its power dissipation with the cost function:

$$P(v) = \sum_{l \in \{r,g,b\}} \sum_{i=1}^{N} \frac{1}{2} \alpha_l v^l[i]^2 + \beta_l v^l[i] + \gamma_l, \qquad (1)$$

where $v^l[i]$ is the image intensity of channel l at pixel i and α_l, β_l, and γ_l are power model parameters obtained through power measurements detailed in Section 5. We choose a quadratic cost function for four reasons: 1) it provides a good fit to the power measurement data observed in practice, 2) it is amenable to efficient optimization, 3) its smoothness regularizes the power measurements of Section 5 to reduce the effects of noise, and 4) its simplicity avoids over-fitting and produces reliable estimates of the model parameters α_l, β_l, and γ_l with a few measurements.

Given an image v, the goal is to find an image u that approximates v but dissipates less display power. We formulate

[2] Including our Cairo-specific front- and back-ends, only about 1% of the Crayon code base depends on Cairo.

this goal as the minimization problem

$$\min_u \; P(u) \;\text{ s.t. }\; \phi(u-v) < \epsilon, \tag{2}$$

where ϕ is a convex function that measures the distance between images and ϵ quantifies acceptable approximation (as measured by ϕ). The function ϕ should simplify the optimization while providing a meaningful comparison for images. For this reason we evaluate the ℓ_2 and ℓ_2^2 distances commonly used in image denoising. Typically, the ℓ_2^2 distance produces a solution with many small color approximations while the ℓ_2 distance produces a solution with a few large color approximations.

With $\alpha_l > 0$, $P(v)$ is a convex function and the minimization problem (2) is convex and has a unique minimizer. Moreover, there exists a Lagrange multiplier λ, whose value depends on ϵ, that defines an unconstrained problem with the same minimizer:

$$\min_u \; P(u) + \lambda\phi(u-v). \tag{3}$$

Without loss of generality, we adopt the unconstrained formulation (3) and investigate the minimization solutions for ℓ_2^2 distance (Section 2.2) and the ℓ_2 distance (Section 2.3).

2.2 Least-squares color approximation

If we measure the distance between the two images with the squared Euclidean norm, $\phi(u-v) = \frac{1}{2}\|u-v\|_2^2$, the minimization problem becomes

$$\min_u \sum_{l\in\{r,g,b\}} \sum_{i=1}^{N} \frac{1}{2}\alpha_l u^l[i]^2 + \beta_l u^l[i] + \gamma_l + \frac{\lambda}{2}(u^l[i]-v^l[i])^2.$$

The problem above is decoupled for each pixel and image channel. As a result, the minimizer is obtained by independently minimizing

$$\min_{u^l[i]} \frac{1}{2}\alpha_l u^l[i]^2 + \beta_l u^l[i] + \gamma_l + \frac{\lambda}{2}(u^l[i]-v^l[i])^2 \tag{4}$$

for each pixel i and image channel l. The optimality conditions of (4) are obtained by differentiation with respect to $u^l[i]$ and give us the closed-form solution

$$u^l[i] = \frac{\lambda v^l[i] - \beta_l}{\lambda + \alpha_l}. \tag{5}$$

The squared Euclidean norm constrains all the pixels and color channels of the transformed image to be at small distances from their counterparts in the original image.

2.3 Euclidean-distance color approximation

If we measure the distance between images with the Euclidean norm $\phi(u-v) = \|u-v\|_2$ and simplify the power-dissipation model of (1) by setting $\beta_l = 0$, the minimization problem becomes

$$\min_u \sum_{l\in\{r,g,b\}} \sum_{i=1}^{N} \frac{1}{2}\alpha_l u^l[i]^2 + \gamma_l + \lambda\|u^l[i]-v^l[i]\|_2. \tag{6}$$

Because of the square root in the last term, although the objective function of (6) is decoupled for each pixel, it is not decoupled for each channel. The minimization solution is therefore obtained by independent minimization for each pixel i of the problem in the vector variable $\vec{u}[i] = (u^r[i], u^g[i], u^b[i]) \in \mathbb{R}^3$:

$$\min_{\vec{u}[i]\in\mathbb{R}^3} \frac{1}{2}\vec{u}[i]^T D_\alpha \vec{u}[i] + \lambda\|\vec{u}[i]-\vec{v}[i]\|_2, \tag{7}$$

where D_α is the diagonal matrix with $\alpha_r, \alpha_g, \alpha_b$ as diagonal elements. With a change of variables $\vec{z} = \vec{u}[i] - \vec{v}[i]$ we obtain the following problem:

$$\min_{\vec{z}\in\mathbb{R}^3} \frac{1}{2}(\vec{z}+\vec{v}[i])^T D_\alpha(\vec{z}+\vec{v}[i]) + \lambda\|\vec{z}\|_2. \tag{8}$$

The second term in the objective functional (i.e., the function from the RGB vector space to the scalar minimization value) depends only on the length of $\vec{z}$ and takes the same value for all possible orientations of $\vec{z}$, while the first term is minimized when $\vec{z}$ has the same orientation as $\vec{v}[i]$. As a result, the minimizer satisfies $\vec{z} = \mu\vec{v}[i]$, for some $\mu > 0$. The problem is then reduced to a minimization in $\mu \in \mathbb{R}$:

$$\min_\mu \frac{1}{2}\vec{v}[i]^T D_\alpha \vec{v}[i](1-\mu)^2 + \lambda\|\vec{v}[i]\|_2\mu. \tag{9}$$

The problem is again differentiable and we can solve it by differentiating and equating to zero. Doing so gives us a closed form for the color of transformed pixel $u[i]$:

$$u[i] = (\mu+1)v[i] \;\text{ with }\; \mu = \max(1 - \lambda\frac{\|v[i]\|_2}{v[i]^T D_\alpha v[i]}, 0). \tag{10}$$

The ℓ_2 distance model constrains the transformed image to differ from the original only in a reduced set of pixels and color channels. Thus, while the ℓ_2^2 transform of (5) results in many small color approximations, the ℓ_2 transform of (10) results in a small number of approximations that might be large.

2.4 Color distances and color spaces

The RGB color space is the native color space in which content is sent to the display. It therefore directly reflects the physical properties of the display. We also explore color transforms in the CIE LAB color space [38], which was designed to reflect the way humans perceive color differences. We apply the power model in (1) to the CIE LAB color space by adapting the model parameters α_l, β_l, and γ_l to the corresponding power measurements. We limit ourselves to the RGB and CIE LAB spaces because more sophisticated spaces [38, 43] require more computationally-complex optimizations and are less widely used.

2.5 Implementing the color transforms

Crayon applies the color transform equations (5) and (10) to pixel values to reduce display power dissipation with a

Table 1. Mapping to Crayon's IR of the Cairo [70, 71] API calls that cause changes to the drawing surface (`cairo_surface_t`).

Cairo API calls	Crayon IR Ops
`cairo_arc`, `cairo_arc_negative`	`arc`
`cairo_rectangle`	`polygon`
`cairo_fill`, `cairo_fill_preserve`	`composite`
`cairo_line_to`, `cairo_rel_line_to`	`polygon`
`cairo_curve_to`, `cairo_rel_curve_to`	`beziercurve`
`cairo_paint`, `cairo_paint_with_alpha`	`composite`
`cairo_glyph_path`	`shape`

bounded amount of color approximation. The parameter λ determines the tradeoff between power reduction and color approximation. These transforms can be implemented efficiently: We show in Sections 4 and 5 that the transform of (5) produces images acceptable to participants in a large user study and at the same time significantly reduces display power dissipation. The transforms (5) and (10) have low computational costs: The color transform of (5) requires only four arithmetic operations per color channel.

3. Crayon IR and Shape Transform

In addition to bitmap images, display content is often generated from vector graphics operations, which specify geometric, location, layering, and color information for image components [3, 5, 34, 70]. Crayon exploits this additional drawing information to apply *shape transforms* that change not just the color, but also the shapes of image components.

Crayon uses an image content representation, the Crayon intermediate representation (IR), to capture the drawing information necessary to enable shape transforms. The Crayon IR comprises seven types of components: pens, shapes, images, colors, patterns, gradients, and operators. Pens (with type **`pen`**) have a defined color and width and may be used to generate contiguous outlines. Shapes (type **`shape`**) are contiguous outlines of a pen and may optionally be *closed*. Collections of shapes make up images (with type **`image`**). Colors (type **`color`**) are values taken from a given color space. Patterns and gradients are also types of images, comprising repeated scaled images and linear or radial blends of color, respectively. Operators act on existing shapes and images or create new ones.

Crayon builds the IR from information contained in on-disk vector and bitmap image files or captured dynamically from intercepted drawing API calls. We used Cairo [70] as a host UI library to evaluate the dynamic interception of drawing API calls. Integrating Crayon with Cairo required adding about 50 lines of C code to the Cairo implementation and adding an additional state pointer to extend Cairo's **`cairo_t`** structure. Table 1 presents the mapping from Cairo API calls to Crayon IR operators. Similar mappings are possible for other drawing APIs such as Skia [3] and vector file formats such as PDF [34].

3.1 The Crayon shape transform

The Crayon shape transform either grows or shrinks a shape along its perimeter by g pixels. Growing the shape replaces background colors with colors from the shape; shrinking the shape replaces colors from the shape with colors from the background. The relative power dissipation of the background and shape colors determines the desirable direction (grow or shrink) of the transform and the power savings that the transform can deliver. The area and color of a shape determines its contribution to display power dissipation (Equation 1). The results in Section 4.2.2 show that small changes in linear dimension are largely imperceptible to users.

We illustrate the effect of small linear dimension changes by considering the lower bound on the change in area of a shape as a result of growing its border by g pixels. In two dimensions, the shape with the smallest perimeter for a given area is the circle. Thus, for a fill region of area A and color κ, the lower bound on the relative change in power dissipation as a function of g, $\Delta_{\substack{\text{display}\\\text{power}}}(\kappa, g)$, is given by simplifying the corresponding algebraic expressions from the radius and area of a circle:

$$\Delta_{\substack{\text{display}\\\text{power}}}(\kappa, g) \geq \left(2g\sqrt{\pi A} + \pi g^2\right) P(\kappa). \tag{11}$$

This property is important because, as a lower bound, it dictates that small visual changes of g pixels around the border of a shape will be accompanied by changes in display power dissipation that grow (or shrink) at least as fast as g^2.

Whether this change causes an overall reduction in display power depends on whether the added perimeter of pixels replaces background pixels for which the per-pixel-power function, P, evaluates to a higher value. The Crayon IR (Section 3.2) encodes information about the fill regions and the colors which lie immediately below a given shape. This information enables Crayon to determine whether a given shape should grow or shrink to reduce display power dissipation.

3.2 Building the IR by intercepting API calls

Crayon can generate its IR from information contained in the procedure call sequences of existing 2D graphics libraries. We employ the Cairo library in our first implementation because it is mature, has a stable API, and is also the sole candidate for the proposed C++ 2D drawing standard [40].

Algorithm 1 presents the mechanism for generating Crayon from Cairo API calls. We modified the development version of the Cairo library to shunt API calls through the Crayon runtime to allow us to intercept Cairo API calls. We link (unmodified) applications which use the Cairo API against this modified version of the Cairo library.

The Crayon runtime buffers procedure identifiers and parameters of the Cairo API calls it intercepts in a *call sequence buffer* ***csb*** until it reaches a call in the set of defined *boundary points* ***bp***. Boundary points are API calls which change implicit state (the drawing context, **`cairo_t`**

Algorithm 1: Sketch of algorithm for generating Crayon IR from Cairo API calls.

```
1  {csb} ← ∅                          /* csb: call sequence buffer */
2  ci ← 0                             /* ci: call index */
   Emit stub Crayon program with empty init
   /* bp: boundary points that trigger processing csb. */
   while call ∉ bp do
       if call causes explicit change of Cairo state then
           Emit shape s using curPen, curColor
           s.order ← ci + 1
           ci ← s.order
           csb ← csb ∪ s
       else if call causes implicit change of Cairo state then
           /* E.g., cairo_t change via cairo_set*.   */
           Emit new pen or color blocks
           curPen ← newPen
           curColor ← newColor
       else
           Pass call through to Cairo
3  Emit csb into init using composite operator
   return CrayonIR
```

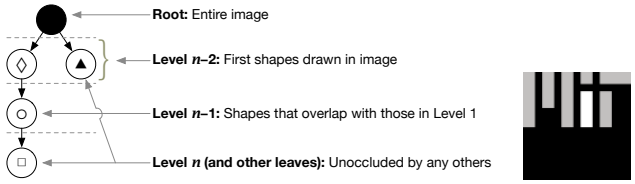
in Cairo). A conservative set of boundary points includes all Cairo APIs with prefixes **cairo_set**, **cairo_reset**, **cairo_save**, **cairo_restore**, **cairo_push**, and **cairo_pop**. Crayon then converts the ***csb*** (line 3) to the Crayon IR based on the mappings in Table 1.

3.3 IR passes to enable shape transform

After Crayon generates the IR, it next applies three passes that together set up the shape transform.

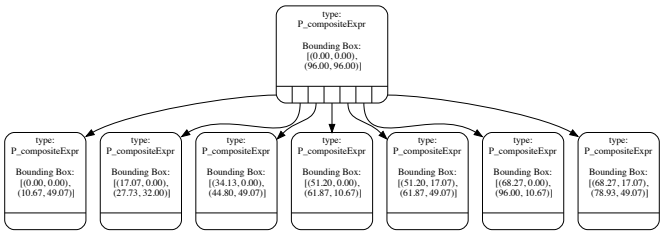
Pass 1: Pen and color de-duplication. In practice, drawing programs may contain redundant color adjustments, resulting in redundant **pen** and **color** nodes in the Crayon IR. This redundancy occurs because, in stateful APIs like Cairo, programs often reset the current stroke and color at the start of drawing each new object. A traversal of the IR in program order, maintaining two stacks for **pen** and **color** statements, removes the resulting redundant IR nodes. The cost of this pass is linear in the number of Crayon IR statements (**pen** and **color** definitions do not occur nested).

Pass 2: Bounding box labeling. This pass computes the bounding box for each Crayon **shape**. Crayon uses the bounding boxes to compute the layer ordering in the shape dependence labeling, which are in turn used to determine whether the shape transform should grow or shrink transformed shapes. The pass is applied to each **shape** or **image** in the IR and is linear in the number of sides of each such shape. The result of this pass is the Crayon compositing-dependence graph. Figure 3(a) illustrates the structure of the Crayon compositing-dependence graph. Figure 3(b) and Figure 3(c) present an example image and its Crayon compositing-dependence graph.



(a) The structure of the compositing-dependence graph for sets of drawing operations.

(b)



(c) The compositing-dependence graph for the drawing (b), generated by the Crayon runtime.

Figure 3. The structure of the compositing-dependence graph (a), an example for a simple drawing with two overlapping layers (grey and white objects on top of a black background) (b), and its compositing-dependence graph (c).

Algorithm 2: Sketch of algorithm for compositing-dependence labeling nodes in Crayon IR.

```
for (node ∈ CrayonIR) ∩ (node.type == shape) do
    O ← {n  s.t.  (n.bBox ∩ node.bBox) ≠ ∅}
    n ← a  s.t.  a ∈ O && a.order == min_{i∈O} (i.order)
    node.parent ← n
    n.child ← node
return CrayonIR
```

Pass 3: Shape dependence labeling. For each **shape** or **image** node in the IR, this pass (Algorithm 2) annotates the corresponding node with references to nodes whose bounding boxes it intersects. The bounding boxes of two shapes intersect when one shape in the pair either partly or completely occludes the other shape.

3.4 The shape transform

Crayon's shape transform iterates over the shape dependence labeled IR to shrink or grow shapes relative to their bounding boxes. The transform is based on:

❶ the specified area growth or shrinkage;

❷ the average fill color of the shape;

❸ the average fill color of the background.

Crayon's shape transform grows shapes within an image if their colors cause lower display power dissipation than the colors they occlude and shrinks shapes within an image if their colors cause greater display power dissipation than the colors they occlude.

Shapes grow or shrink based on the information of their computed bounding boxes in the second transform pass. The transform moves the points on the border either outward from or inward to the center of the bounding box. The *shape scale factor* is the factor by which the transform scales the side of the shape's bounding box. Shape scale factors of less than one shrink shapes, while shape scale factors greater than one grow shapes. Crayon chooses shape scale factors based on the colors of the scaled shapes and the colors they occlude in the compositing-dependence graph as described above.

We have three backends in our current implementation that may process the Crayon IR. One backend regenerates API calls corresponding to the transformed IR. A second renders the image that the API calls generate and stores the image on disk. The third backend renders a depiction of the IR itself (for debugging).

4. User Study

We ran a user study to quantify acceptable shape and color transforms. The study starts with a set of images, then applies color and shape transforms to obtain a set of transformed images. It presents matched pairs of original and transformed images to human participants on Amazon Mechanical Turk and asks the participants to rate the acceptability of each transformed image. The goal is to discover the minimum values of the color transform tradeoff parameter λ (Sections 2.2 and 2.3) and the shape scale factor (Section 3.4) that deliver acceptable images. These parameter values enable us to determine how much display power savings Crayon can achieve while delivering acceptable images (Section 5).

4.1 Methodology

We start with three bitmap images and two vector images as our baselines. The bitmap images are standard images used in image processing research [64]. The vector images are variants of the MIT logo. We generate variants of each image as follows.

Matched pairs for shape transform evaluation: We generate variants of the vector images by applying Crayon's shape transforms. We choose 20 uniformly-sampled shape scale factors in the range of 0.6 to 1.4. For each scale factor, we obtain the transformed image by scaling the size of each shape in the image by the scale factor (scale factors less than one shrink shapes; scale factors greater than one grow shapes). We obtain a total of 40 matched shape-transform pairs (20 for each of the two vector images). Each pair consists of an original vector image and a corresponding shape-transformed image.

Matched pairs for color transform evaluation: We use the baseline bitmap images and the shape transformed vector images as the original images. We generate variants of these original images by applying the different color transforms from Section 2 to these original images.

A combination of a distance function and a color space determines each color transform. The distance function is either the ℓ_2^2 distance function from Section 2.2 or the ℓ_2 distance function from Section 2.3. The color space is either the RGB or CIE LAB color space. Each color transform also takes a tradeoff parameter λ. For each combination of distance function and color space we choose a range for λ, with the maximum λ producing transformed images indistinguishable from the original and the minimum λ producing almost unacceptably approximated images. We then choose 40 uniformly-sampled λ values within the range. We apply each transform to each of the original images, with λ set to each of the sampled values in the range (obtaining 800 matched color-transformed pairs, 160 for each original image and 40 for each combination of original image and color transform).

It is also possible to reduce power dissipation by simply making the image darker. We therefore also evaluate two darkness transforms. The first scales the luminance component in the CIE LAB space. The second scales the R, G, and B channels equally. We choose 40 uniformly-sampled scaling values between zero and one, with zero producing a completely black image and one producing a completely unchanged image. We apply each of the sampled scaling values to each of the bitmap images (obtaining 400 matched darkness-transformed pairs, 80 for each original image and 40 for each combination of original image and darkness transform).

Image pairs for perceptual evaluation: Participants rated groups of ten matched pair images. We construct each group of ten images by pseudo-randomly selecting ten matched pairs, without replacement, from all of the matched shape, color, and darkness transformed pairs. As a control, we randomly select one of the ten pairs, then replace the transformed image with a control image. The control is either the original image or a completely black image.

Running the study: We obtained exempt-status authorization from our institution's Committee on the Use of Humans as Experimental Subjects to run the user study. We performed the study using workers on Amazon Mechanical Turk (AMT). The study employed a total of 440 unique participants (we consider each unique AMT worker ID as a unique participant). All of the study's evaluation questions ask participants to compare an original reference image with a corresponding transformed image. With appropriate study safeguards [39] (which we apply), AMT can deliver sufficient numbers of workers for conducting effective full-reference image comparisons.

We presented participants with ten matched pairs of images as described above. We asked participants to rate each image pair as either *identical* (score 3), *minor difference* (score 2), *significant difference* (score 1) or *completely different* (score 0) [37]. For each pair, we also asked participants to describe the rationale for their score in words. We

discarded all groups of ten that did not meet this requirement for every pair in the group. Following accepted practice for conducting perceptual studies with potentially-unreliable participants [39], we also discarded all results for each group of ten images if the study participant incorrectly rated the control pair. We furthermore discarded all results in a group if the participant gave all ten pairs the same rating. We paid participants 0.2 USD per group of ten images rated if they passed the checks listed above. We ran the study until we had over 20,000 image evaluations.

4.2 Results

We analyze the study data to quantify acceptable color transforms (Section 4.2.1), acceptable shape transforms (Section 4.2.2), characteristics of the study participants (Section 4.2.3), and the relationship between subjective perceptual scores and objective quantitative image quality metrics (Section 4.2.4).

4.2.1 Color transforms

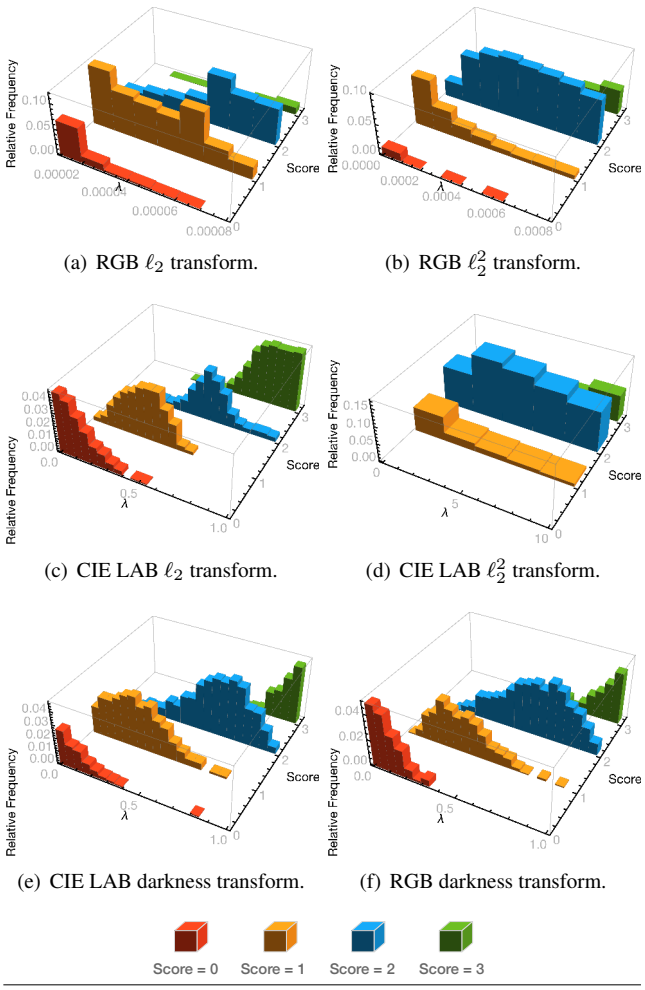
Figure 4 presents the histogram of the number of responses with a given score as a function of the color transform tradeoff parameter λ for the six color transforms we evaluated. In trading color accuracy for display power dissipation, larger values of λ favor color accuracy over lower display power dissipation. Participants unsurprisingly gave higher ratings to images transformed with larger values of λ.

To quantify the effect of the tradeoff parameter λ and the color transform model (i.e., ℓ_2 versus ℓ_2^2 in the RGB and CIE LAB color spaces), we performed a two-way ANOVA analysis on the perceptual score data in Figure 4. Because the tradeoff parameter λ differs across the color transform models, we normalize the range of λ for each model and discretize the resulting normalized values to 5 levels. Using these normalized λ values, we test the null hypothesis that the mean perceptual scores across color transform models and λ levels do not significantly differ. The two-way ANOVA analysis rejects both hypotheses with p values less than 0.01. This analysis indicates that both the choice of color transform model and tradeoff parameter λ affect the perceptual score.

4.2.2 Shape transforms

Figure 5 presents the scores that participants gave to shape-transformed images as a function of the shape scale factor. Figure 5(a) plots the histogram of responses per score as the shape scale factor varies from 0.6 to 1.4. As expected, the scores decrease as the shape scale factor moves away from one (either shrinking or growing).

Figure 5(b) plots the fraction of evaluations with a given minimum score as a function of the shape scale factor. For shape transforms with shape scale factors between 0.92 and 1.08 (i.e., up to 8% shrinkage or growth), a majority of the participants (greater than 90%) rated the images either *minor difference* (score 2) or *identical* (score 3).



(a) RGB ℓ_2 transform. (b) RGB ℓ_2^2 transform.

(c) CIE LAB ℓ_2 transform. (d) CIE LAB ℓ_2^2 transform.

(e) CIE LAB darkness transform. (f) RGB darkness transform.

Figure 4. Distribution of perceptual scores as a function of color accuracy tradeoff parameter λ. The range of values for λ differs across the transform methods, but larger λ always places greater emphasis on color accuracy. As λ increases, participants give higher scores to transformed images.

We used the non-parametric Kruskal-Wallis test to evaluate the null hypothesis that the mean perceptual scores for different shape scale factors are not significantly different.[3] The result of the test is the rejection of the null hypothesis with a p-value of less than 0.01.

4.2.3 Rating behavior across study participants

Participants who evaluated shape transforms evaluated an average of 17.5 matched shape transform pairs, with a standard deviation of 11.5 matched shape transform pairs. No individual participant evaluated more than 40 matched shape transform pairs. Participants evaluating color transforms evaluated 56.7 matched pairs on average, with a standard

[3] Because the data do not pass normality tests, we opt for a parametric Kruskal-Wallis test instead of the more common ANOVA analysis which assumes normally-distributed data.

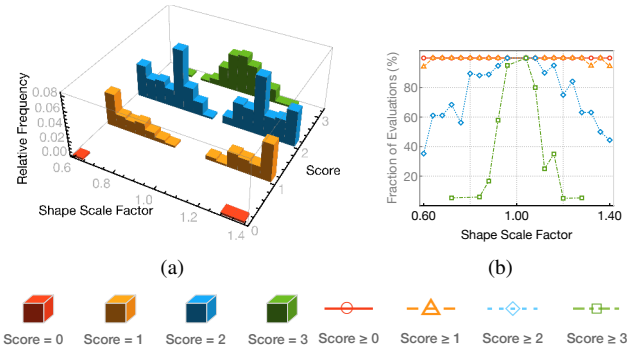


Figure 5. A majority of participants in the study rated shape changes by a factor of up to $\pm 8\%$ to be either *identical* or *minor difference*.

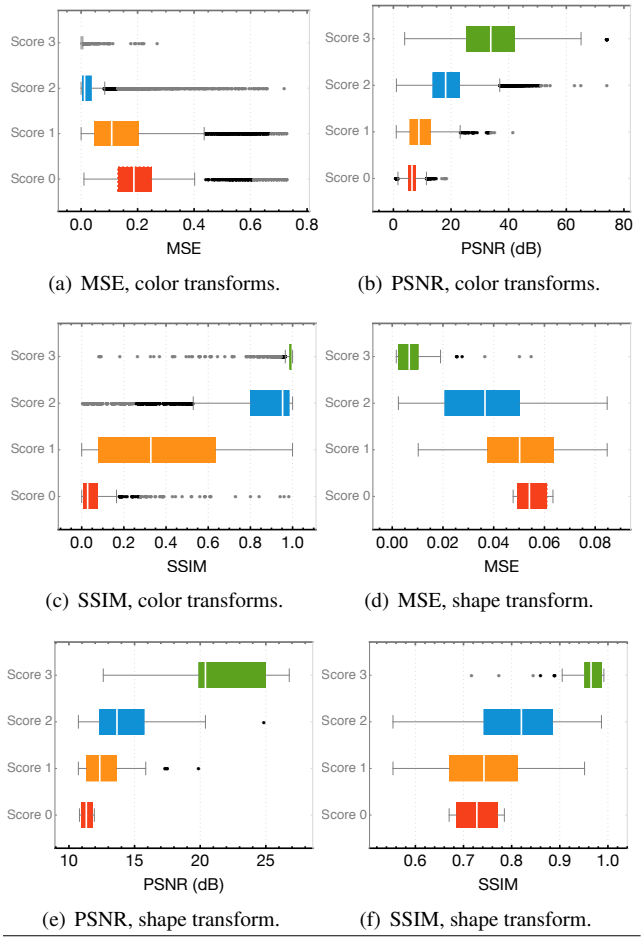


Figure 6. Box plots of perceptual scores versus quantitative image quality metrics for Crayon's color transforms (a–c) and shape transforms (d–f).

deviation of 125.2 matched pairs. There were four participants who evaluated more than 500 matched pairs each.

4.2.4 Quantitative metrics versus perceptual scores

We evaluate how well quantitative techniques capture perceived color and shape differences by analyzing the correlation between scores from the user study and three quantitative image quality assessment metrics: mean squared error (MSE), peak signal-to-noise ratio (PSNR), and structural similarity (SSIM) [69], computed in the RGB color space.

MSE computes the mean squared error between two images. Smaller values are better with an MSE of zero indicating the two images are identical. PSNR captures the ratio of the maximum possible signal (pixel) intensity to the MSE. Larger values of PSNR indicate a closer match between transformed and reference images. MSE and PSNR do not capture structural changes in image content. SSIM is a recent-developed and widely-used image quality metric that does capture structural changes.

Figure 6 presents box-and-whisker plots of MSE, PSNR, and SSIM for both shape and color transforms as a function of perceptual score. The boxes span the 25th percentile to the 75th percentile and the white line on each box indicates the median value. Figure 6 shows how, for both color and shape transforms, the scores reported by participants in the user study are correlated with all three quantitative metrics.

To quantitatively determine whether the four levels of the perceptual scores (0–3) correspond to different clusters in the quantitative metrics, we evaluate the Kruskal-Wallis test[4] on values of PSNR, MSE, and SSIM computed for each matched image pair employed in the user study, with the null hypothesis that the mean quantitative metrics for different scores are not significantly different. We reject the null hypothesis with a p-value of less than 0.01.

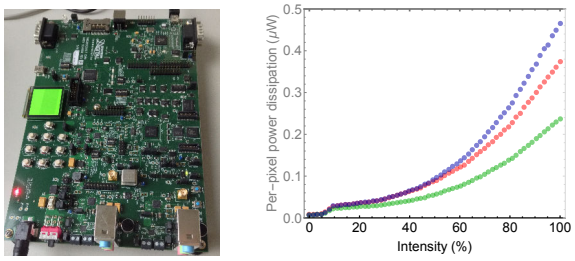
We then compute a non-parametric Spearman correlation coefficient ρ between the perceptual scores and the quantitative metrics for both shape and color transforms. For shape transforms we obtain a ρ of -0.65 for the correlation between the MSE and perceptual score, indicating that the MSE is correlated with perceptual score and decreases as the score increases. Similarly, we obtain a ρ of 0.60 for PSNR and a ρ of 0.65 for SSIM: PSNR and SSIM are correlated with the perceptual score and increase as the perceptual score increases. For color transforms we obtain a ρ of -0.76 for MSE, a ρ of 0.60 for PSNR, and a ρ of 0.44 for SSIM. Transformed images with score 0 may however have good MSE, PSNR, and SSIM values (see Figure 6). Quantitative metrics on their own are therefore not sufficient to evaluate techniques that may adversely affect image acceptability.

4.3 Discussion

The color transform results show that the mean perceptual scores differ across transform methods (i.e., different color distance functions and color spaces) and differ across values of the tradeoff parameter λ at a statistically significant level.

The study results show that perceptual scores are correlated with the shape scale factor, but shape transforms with shape scale factors of 0.92 to 1.08 (i.e., less than 8% change

[4] Because the groups do not pass normality tests or the variances differ, we use a parametric Kruskal-Wallis test instead of ANOVA analysis. The results of the test (the rejection of the null hypothesis) remain the same.



(a) Platform for OLED display power measurements.

(b) OLED display power characterization measurements.

Figure 7. Hardware for display power model measurements and power measurement data of OLED panel. Each point in (b) corresponds to the average of 100 measurements of the display power at the given intensity level.

in the linear dimension of the bounding boxes of shapes being transformed) are typically rated as *identical* or *minor difference*.

The user study results enable us to determine the values of the color transform tradeoff parameter λ and shape scale factor at the threshold of what study participants find acceptable. In our display power savings evaluation in Section 5, we use the lowest value of λ for which a majority of the participants rate a transform with score 2. For color transforms, this threshold is the lowest value of λ for which the sub-histogram for score 2 in each of the sub-plots of Figure 4 has the highest bin count among the four possible scores. The threshold values of λ are 0.00004 for the ℓ_2 color transform in the RGB color space, 0.00004 for the ℓ_2^2 color transform in the RGB color space, 0.5 for the ℓ_2 color transform in the CIE LAB color space, 1.0 for the ℓ_2^2 color transform in the CIE LAB color space, 0.43 for the RGB darkness transform, and 0.40 for the CIE LAB darkness transform.

5. Power Savings and Overhead Evaluation

We evaluate Crayon to answer three questions:

❶ **How much power can color and shape transforms save?** We answer this question using measurements on a hardware evaluation platform with an OLED display (Section 5.1) and by characterizing the power savings for shape transforms (Section 5.4) and color transforms (Section 5.2).

❷ **What is the shape transform overhead?** We evaluate the Crayon shape transform overhead using programs that employ the Cairo API linked against a Cairo library modified to transparently pass drawing calls through Crayon (Section 5.5).

❸ **What is the color transform overhead?** We evaluate the color transform overhead by measuring the time required to apply the transforms to bitmap images (Section 5.3).

5.1 Power characterization

We used the hardware evaluation board shown in Figure 7(a) as our measurement platform to build a detailed display power dissipation model. The measurement platform contains an OLED display, a processor for controlling the display, and built-in power measurement circuits [65].

The red, green, and blue sub-pixels in current OLED displays are made from different electroluminescent or phosphorescent organic compounds, different dopants, or both [4, 41]. Different color sub-pixels therefore generate light with differing efficiencies. For this reason OLED display power dissipation depends on color content.

We measured the average power dissipation of the OLED display for each of the possible 6-bit[5] intensity values for red, green, and blue independently. Figure 7(b) presents the results, which show that blue pixels dissipate about twice the power of green pixels, with red in between the two.

We fit the measurement data of Figure 7(b) to the power model of Equation 1 to obtain the parameters α, β, and γ of Equation 1. These parameters also serve as input to the color transforms of Equation 5 and Equation 10. Given any OLED display panel, a similar calibration process can deliver the data required to obtain the model parameters of Equation 1.

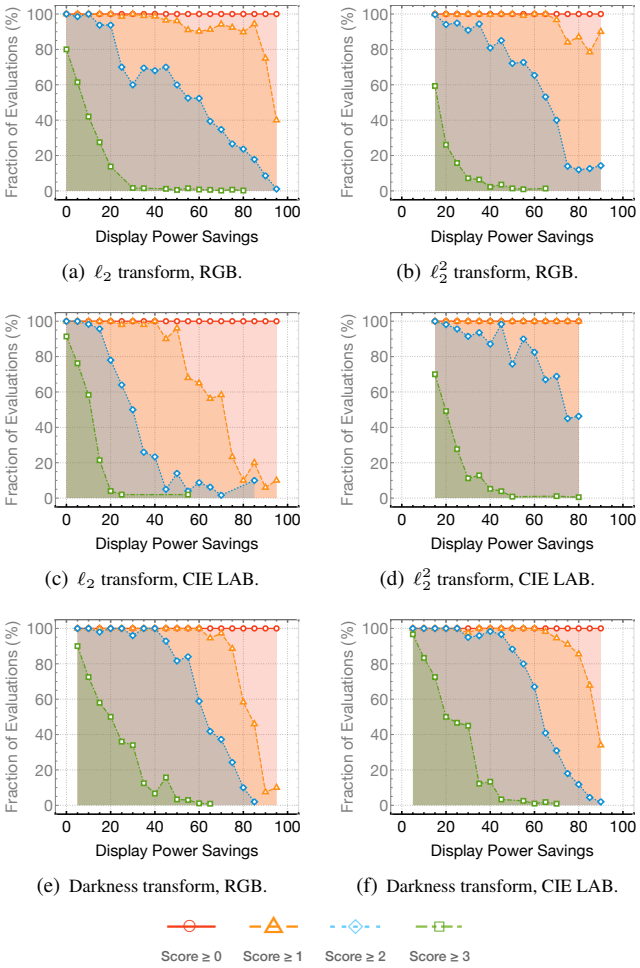
5.2 Color transform power savings

We computed the cumulative fractions of the participants that rated each transform with a given score or better as a function of display power savings (we discretized the display power savings percentage for transformed images to 20 levels). Figure 8 presents the results. The results show that for display power savings below 40%, the Crayon CIE LAB ℓ_2^2 transform and the darkness transforms have the highest fraction of evaluations rated score 2 (*minor difference*) or better. For display power savings above 65%, the Crayon CIE LAB ℓ_2^2 transform has the highest fraction of evaluations rated *minor difference* or better. For display power savings above 65%, the Crayon RGB ℓ_2^2 transform also outperforms all the other transforms except the CIE LAB ℓ_2^2 transform.

Figure 9 presents the display power savings as a function of the tradeoff parameter λ. We compute the power savings using the calibrated power model from Section 5.1. At the lowest value of λ for which the most-frequent score in the user study was score 2 (*minor difference*), the corresponding display power savings across the transforms range from at least 25% display power savings (CIE LAB ℓ_2 transform) to at least 66% display power savings (RGB ℓ_2^2 transform).

Figure 10 presents images that highlight the visual effect of the Crayon color transforms. All of the transformed images deliver 25% display power savings. The ℓ_2 transform exhibits *channel distortion*, concentrating the color approximation into a few colors.

[5] This is the display's native color depth.

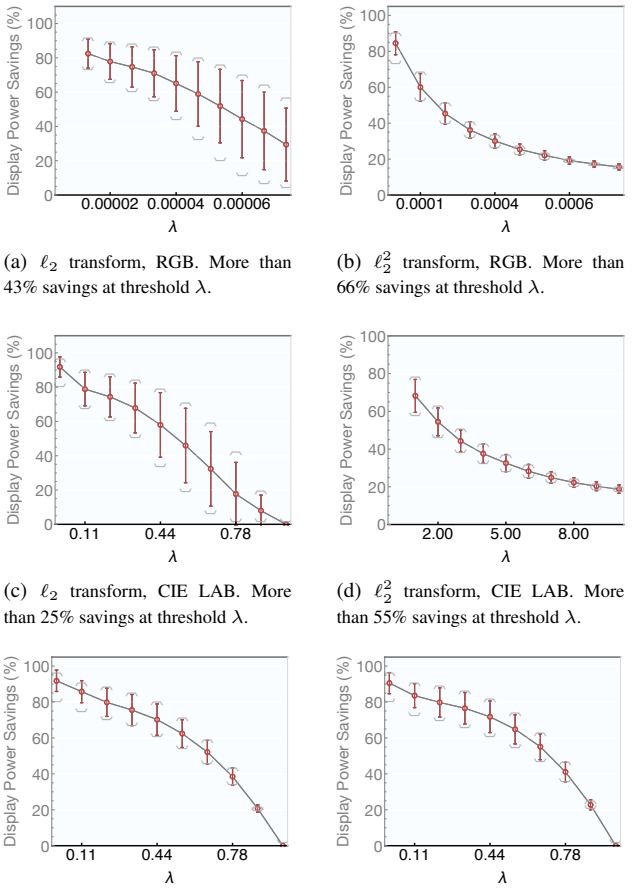


(a) ℓ_2 transform, RGB. (b) ℓ_2^2 transform, RGB.

(c) ℓ_2 transform, CIE LAB. (d) ℓ_2^2 transform, CIE LAB.

(e) Darkness transform, RGB. (f) Darkness transform, CIE LAB.

Figure 8. Cumulative fractions of participants rating a transform with a given score (or better), as a function the display power savings the transform causes.



(a) ℓ_2 transform, RGB. More than 43% savings at threshold λ.

(b) ℓ_2^2 transform, RGB. More than 66% savings at threshold λ.

(c) ℓ_2 transform, CIE LAB. More than 25% savings at threshold λ.

(d) ℓ_2^2 transform, CIE LAB. More than 55% savings at threshold λ.

(e) Darkness transform, RGB. More than 60% savings at threshold λ.

(f) Darkness transform, CIE LAB. More than 61% savings at threshold λ.

Figure 9. Effect of tradeoff parameter λ on display power savings. Across the different transforms, the display power savings at the threshold λ (Section 4.3) range from at least 25% to at least 66%.

5.3 Color transform overheads

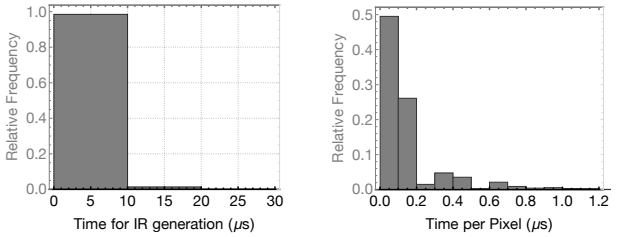
We evaluated the cost of computing Crayon's color transforms on a state-of-the-art mobile platform. For our tests, we used a Dell Venue 8 7000 tablet with an Intel Atom Z3580 processor. We integrated Crayon's color transforms into Firefox (Fennec) on Android, which uses Cairo for 2D graphics. With no modifications to the application source, we compiled it against a version of the Cairo library to funnel incoming Cairo API calls through Crayon, along with added timing instrumentation. Using this setup, we used Fennec to browse web pages until we had accumulated a trace of over 7500 image color transform execution times. Each entry in the trace contained sufficient information to compute two quantities:

❶ The time spent in the Crayon runtime system excluding time spent performing the image color transform.

❷ The image size and time overhead per image color transform computation, for the RGB ℓ_2^2 transform.

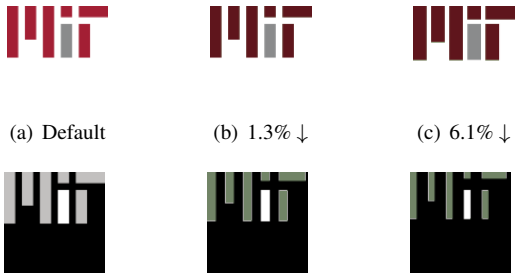
Figure 10. Effect of color transforms all configured to cause a 25% reduction in display power dissipation.

Figure 11 presents the results of the measurements. Figure 11(a) shows the time taken for generating the Crayon IR is always less than 20 μs. We calculated the cost per RGB ℓ_2^2 color transform from the image transform time and image sizes in the measurement traces. Figure 11(b) presents the



(a) (b)

Figure 11. IR generation and color transform overheads.



(a) Default (b) 1.3% ↓ (c) 6.1% ↓

(d) Default (e) 38%↓ (f) 54%↓

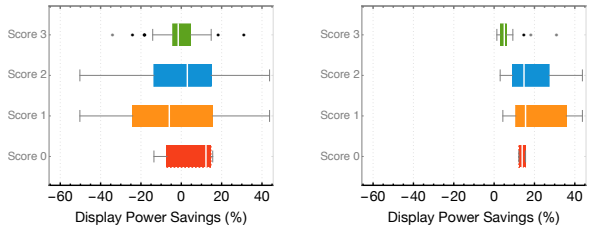
Figure 12. Shape and color transforms and their resulting display power savings. In (b), only color transforms have been applied. In (c) and (e), in addition to applying color transforms, shapes have been altered by 20%. For (f), shapes have been modified by 40% in addition to color transforms.

distribution of measured times per color transform computation. The color transform computation overhead is typically smaller than 1μs per color transform, with a mean value of 0.5μs and a standard deviation of 1.8μs.

5.4 Shape transform power savings

Figure 12 presents different shape- and color-transformed versions of the MIT logo and the resulting power savings. We compute the power savings using the calibrated power model from Section 5.1. From Figure 12, we observe that changes in both shape and color can cause significant display power savings (54% in the example).

The effect on display power dissipation of growing or shrinking a shape depends on the shape's color relative to the color of its surroundings. Figure 13 presents box-and-whisker plots for the change in display power dissipation as a function of participant's ratings of matched pairs of shape transforms from the user study. The boxes in Figure 13 span the 25th percentile of display power dissipation to the 75th percentile of display power dissipation. The white line on each box indicates the median value. Figure 13(a) presents the results for all of the shape-transformed images from the user study (Section 4). Figure 13(b) presents results only for images for which the shape transform reduces display power dissipation (Section 3.4). The results show display power savings of 5.5% on average and as high as 31.0% for shape-



(a) (b)

Figure 13. (a): Changes in shape can cause either increases or decreases in display power dissipation. (b) When restricted to only those shape transforms that either obscure pixels that dissipate more power, or expose pixels that dissipate less power, shape transforms can enable power savings with visually acceptable results.

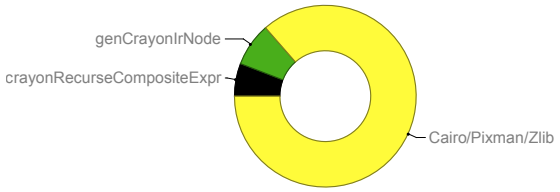


Figure 14. Shape transform overheads obtained by DTrace instrumentation, for the **`rsvg-view`** application using Crayon transparently through its use of Cairo. Crayon introduces less than 14% overhead to Cairo. The remaining Crayon-specific functions, not shown in the breakdown, take up less than 0.1% of the execution time.

transformed images rated *identical* to their originals (score 3). For shape-transformed images rated as having a *minor difference* from their originals (score 2), the results show display power savings of 17.9% on average, with savings as high as 43.8%.

5.5 Shape transform overheads

We use DTrace [10] to investigate the different sources of overhead in the Crayon shape transform implementation. We measure the time spent generating the Crayon IR, performing the shape transform, and regenerating Cairo API calls. We measured the **`rsvg-view`** application (an SVG viewer which uses Cairo for drawing) compiled against our modified Cairo library that passes API calls through Crayon.

Figure 14 presents the execution time breakdown for the **`rsvg-view`** application on a typical SVG input from the freely-available AIGA database of international signs. The Crayon dynamic online transforms take up 13.5% of the time spent in the Cairo library. Of that time, in our current implementation, a majority of the Crayon overhead (57%) is spent in building the IR. Most of the time in the **`rsvg-view`** application as a whole is spent in application logic.

5.6 Crayon whole-system power savings

We performed whole-system power measurements on an Android tablet with an OLED display [21]. We used the An-

(a) Original image, scaled to fit the aspect ratio of the measurement device's display.

(b) Image after applying the Crayon ℓ_2^2 RGB color transform with $\lambda = 0.0006$.

(c) Image after applying the Crayon ℓ_2^2 RGB color transform with $\lambda = 0.00004$.

Figure 15. Original and color-transformed images used to estimate Crayon's whole-system power savings.

droid battery fuel gauge interface for these power measurements. We first performed experiments to obtain the coefficients α_l, β_l, and γ_l for the display power model (Equation 1) and then built the Crayon color transform for this model. We selected a representative image (Figure 15(a)) and measured the whole-system power dissipation with and without Crayon color transforms applied to this image.

Display power model: Using power measurements from the fuel gauge interface, we measured the power dissipation of the whole device while displaying each one of eight levels between minimum and maximum intensity of red, green, and blue on the whole screen. We also measured power dissipation when displaying a completely black image and subtracted this measured power dissipation from the single-color measurements to obtain measurements for display-only power dissipation as a function of color. We then divided these whole-screen power measurements by the number of display pixels to obtain average per-pixel power measurements as functions of color. We fitted this per-color power data to the model of Equation 1 to obtain the coefficients α_l, β_l, and γ_l.

Whole-system power dissipation without Crayon: We next used the Android battery fuel gauge interface to measure the whole-system power dissipation with the screen displaying the unmodified representative image (Figure 15(a)). The image was scaled to fit the full screen.

Whole-system power dissipation with Crayon: We computed the mean λ value for Crayon's ℓ_2^2 color transform which participants in the user study rated as producing *identical* images (score 3). We applied the Crayon ℓ_2^2 color transform with this value of λ (0.0006) to the image in Figure 15(a) to obtain the color-transformed image in Figure 15(b). We used the Android battery fuel gauge interface to measure the whole-system power dissipation with the screen displaying the image in Figure 15(b) scaled to fit the full screen. We measured an approximately 12% reduction in whole-system power dissipation.

We next applied Crayon's ℓ_2^2 color transform with the smallest λ for which the most-frequent score in the user study was score 2 (*minor difference*). This λ value was 0.00004 (see Section 4.3). Figure 15(c) presents the resulting color-transformed image. We verified that the level of color approximation visually matched that obtained with the power model of Section 5.1. We used the Android battery fuel gauge interface to measure the whole-system power dissipation with the screen displaying the image in Figure 15(c) scaled to fit the full screen. We measured an approximately 50% reduction in whole-system power dissipation.

6. Related Research

Fundamental limits [6, 56, 57], economics, and engineering challenges constrain semiconductor process technology scaling and limit the possibility of faster and more energy-efficient computing systems. These challenges have motivated research into *approximate computing*, which trades fidelity of computation, storage, or communication in return for speed or energy efficiency [11, 27, 29, 46–50, 52, 55, 56, 58–60, 73]. Techniques can be applied individually or as part of a control system [31, 32, 55] to ensure that a target energy reduction or accuracy constraint is satisfied.

Displays constitute a large fraction of the power dissipation in mobile systems. A number of approximation techniques, targeted primarily at legacy backlit LCDs, have been developed to reduce display power dissipation [16, 45]. With the advent of organic light-emitting diode (OLED) displays, a number of research efforts [23, 24, 30, 35, 53, 61] have explored exploiting approximation of color content to reduce power dissipation in OLED displays.

To the best of our knowledge, Crayon is the most efficient system for reducing display power dissipation by color approximation. Crayon is also the first system that transparently allows shape approximation in addition to color transforms. Unlike prior work which targeted application-specific implementations, Crayon is exposed to all of a system's 2D drawing/GUI calls by virtue of its interposition into the high-level GUI drawing pipeline before GPU-accelerated rendering occurs. Crayon's static offline transform tools are the first set of techniques we know of for applying power-reducing color approximation transforms to vector image files.

Prior work on trading image fidelity for energy efficiency can be classified broadly into five directions: Color transforms by color remapping; color transforms by mathematical optimization; color transforms in restricted applications such as web browsers; selective dimming based on a user's visual focus; and image fidelity tradeoff analyses that employ perceptual studies. We review each of these in turn.

6.1 Color remapping

When color transforms are applied in restricted contexts such as in color schemes for infographics [19, 68] or in GUI color schemes [22–24], colors that are more power-expensive on OLED displays may be substituted for ones that cause lower display power dissipation. Crayon's bitmap transforms, in contrast to these restricted use cases, can be applied to any display content including images of natural scenes, not just to infographics.

6.2 Mathematical optimization

Dong *et al.* [23] focus on *usability* rather than color transform *acceptability*. They therefore employ an approach that may completely remap colors regardless of the perceptual distance between the original and distorted colors. They formulate color transforms as an optimization problem that minimizes power under the constraint that all pairs of pixels in the transformed image are at the same (or greater) ℓ_2 distance in CIE LAB space compared to the untransformed image. They observe that the optimal solution of this problem is exponential in the number of display colors supported. Reported execution times for a QVGA (320×240) display are in the range of 1000 to 10,000 seconds. To address this cost, they propose a polynomial-time greedy heuristic that still requires up to 10 seconds for displays that support as few as 16 colors. Crayon's ℓ_2^2 color transform in the RGB color space, in contrast, only requires one multiplication, one division, and two additions per channel, yet achieves similar display power savings (Section 5).

6.3 Application-specific color transforms

Another approach directly modifies individual applications such as games [2], web browsers [24, 36], and web servers [35]. Application-specific tradeoff techniques have the disadvantage that modifications must be repeated for each new application. Unlike these application-specific techniques, Crayon's dynamic online transforms can benefit any application that uses the operating system platform's drawing library. Application-specific techniques can also be complex: For example, the Chameleon web browser [24] employs several techniques including designing color schemes for specific popular websites, inverting colors in web pages, and requiring users to explicitly select schemes. Chameleon requires color maps to be calculated offline, using an optimization method which the authors themselves describe as "compute-intensive". Like in previous work [23], Chameleon's color transform is an optimization formulated over all pairs of pixels and over all colors. Both the optimal solution and approximate heuristics are therefore computationally expensive. Crayon's color transforms, in contrast, require only three parameters, but achieve average display power reductions ranging from over 25% for the CIE LAB ℓ_2 color transform to over 66% for the RGB ℓ_2^2 color transform (Section 5.2). These power reductions are in line with Chameleon's reported 64% display power reduction. Crayon's power reductions are supported by a detailed user study involving over 400 participants. The Chameleon evaluation involved 20 participants [24].

6.4 Selective area dimming

A number of research efforts selectively dim portions of an OLED display panel based on heuristics of a user's focus of attention [61]. The techniques are obtrusive and, when they guess the user's focus of attention incorrectly, can render a device unusable. Other research efforts have used heuristics to guess which part of a display is occluded by a user's hand [15]; these latter techniques must, among other things, guess whether a user is left- or right-handed, how large their hands are, whether they are using a stylus, and so on.

6.5 User studies

Several studies of the tradeoffs between image quality and power dissipation of displays have employed perceptual studies. These studies have all involved only a small number of participants. For example, Harter *et al.* [30] employed 12 users in their analysis of the effects of selective display area dimming for OLED displays. Tan *et al.* [61] employed 30 users in evaluating a similar technique. To evaluate their color-adaptive server-side color transforms, Li *et al.* [35] conducted a perceptual study with 17 users. Dong *et al.* [24] employed 20 participants to evaluate their color-adaptive web browser. Anand *et al.* [2] conducted a user study with 60 users to evaluate a display brightness and image tone mapping technique. All of these prior efforts provided valuable insight into the challenges and benefits of performing perceptual user studies. Our Crayon evaluation builds on these prior efforts with a large user study comprising 440 participants. We further use the results of the study to gain insight into the relationship between perceptual scores obtained from the study and quantitative image quality metrics such as PSNR, MSE, and SSIM (Section 4.2.4).

7. Conclusions

Power dissipation and energy consumption are fundamental concerns for mobile devices (and other battery-powered devices). For many such devices, display power dissipation constitutes a significant fraction of the power dissipation of the complete system. Unlike previous-generation LCD displays, the power dissipation of new display technologies such as OLED displays and DLP pico-projectors is a function of the displayed colors.

We present efficient new color-aware image transforms and a new system, Crayon, that applies these transforms to reduce display power dissipation while preserving acceptable image quality. A comprehensive user study of transformed image acceptability in combination with power dissipation measurements from both an isolated display and a complete system highlight the significant power and energy savings that Crayon can deliver.

Acknowledgments

We thank Fan Long for providing and running the Python scripts for uploading the human interface tasks (HITs) to Amazon's Mechanical Turk, as well as scripts for randomizing HITs and downloading HIT results. V. Estellers was supported by the Swiss National Science Foundation grant P300P2_161038.

References

[1] 4D Systems. Introduction to OLED Displays Design Guide for Active Matrix OLED (AMOLED) Displays, 2008.

[2] B. Anand, K. Thirugnanam, J. Sebastian, P. G. Kannan, A. L. Ananda, M. C. Chan, and R. K. Balan. Adaptive display power management for mobile games. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 57–70, New York, NY, USA, 2011. ACM.

[3] Android. Skia Graphics Engine, October 2015.

[4] M. A. Baldo, D. O'brien, Y. You, A. Shoustikov, S. Sibley, M. Thompson, and S. Forrest. Highly efficient phosphorescent emission from organic electroluminescent devices. *Nature*, 395(6698):151–154, 1998.

[5] Bell-Labs. Libmemdraw, October 2014.

[6] C. H. Bennett and R. Landauer. The fundamental physical limits of computation. *Scientific American*, 253(1):48–56, 1985.

[7] V. K. Bowden, J. E. Dickinson, A. M. Fox, and D. R. Badcock. Global shape processing: A behavioral and electrophysiological analysis of both contour and texture processing. *Journal of Vision*, 15(13):18–18, 2015.

[8] J. Burroughes, D. Bradley, A. Brown, R. Marks, K. Mackay, R. Friend, P. Burns, and A. Holmes. Light-emitting diodes based on conjugated polymers. *Nature*, 347(6293):539–541, 1990.

[9] Cairo. Cairo 2D graphics library, development version 1.12.16, November 2014.

[10] B. M. Cantrill, M. W. Shapiro, and A. H. Leventhal. Dynamic instrumentation of production systems. In *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, ATEC '04, pages 2–2, Berkeley, CA, USA, 2004. USENIX Association.

[11] M. Carbin, S. Misailovic, and M. C. Rinard. Verifying quantitative reliability for programs that execute on unreliable hardware. In *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications*, OOPSLA '13, pages 33–52, New York, NY, USA, 2013. ACM.

[12] A. Carroll and G. Heiser. An analysis of power consumption in a smartphone. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, USENIXATC'10, pages 21–21, Berkeley, CA, USA, 2010. USENIX Association.

[13] N. Chang, I. Choi, and H. Shim. Dls: dynamic backlight luminance scaling of liquid crystal display. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12(8):837–846, Aug 2004.

[14] X. Chen, Y. Chen, M. Dong, and C. Zhang. Demystifying energy usage in smartphones. In *Proceedings of the 51st Annual Design Automation Conference*, DAC '14, pages 70:1–70:5, New York, NY, USA, 2014. ACM.

[15] X. Chen, K. W. Nixon, H. Zhou, Y. Liu, and Y. Chen. Fingershadow: An oled power optimization based on smartphone touch interactions. In *6th Workshop on Power-Aware Computing and Systems (HotPower 14)*, Broomfield, CO, 2014. USENIX Association.

[16] W.-C. Cheng, Y. Hou, and M. Pedram. Power minimization in a backlit tft-lcd display by concurrent brightness and contrast scaling. In *Proceedings of the Conference on Design, Automation and Test in Europe - Volume 1*, DATE '04, pages 10252–, Washington, DC, USA, 2004. IEEE Computer Society.

[17] W.-C. Cheng, C.-F. Hsu, and C.-F. Chao. Temporal vision-guided energy minimization for portable displays. In *Proceedings of the International Symposium on Low Power Electronics and Design*, ISLPED'06, pages 89–94, Oct 2006.

[18] I. Choi, H. Shim, and N. Chang. Low-power color tft lcd display for hand-held embedded systems. In *Proceedings of the 2002 International Symposium on Low Power Electronics and Design*, ISLPED '02, pages 112–117, New York, NY, USA, 2002. ACM.

[19] J. Chuang, D. Weiskopf, and T. Möller. Energy aware color sets. In *Computer Graphics Forum*, volume 28, pages 203–211. Wiley Online Library, 2009.

[20] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American statistical association*, 79(387):531–554, 1984.

[21] Dell. Dell Venue 8 7000 Series Android™ Tablet, 2014.

[22] M. Dong, Y.-S. K. Choi, and L. Zhong. Power modeling of graphical user interfaces on oled displays. In *Proceedings of the 46th Annual Design Automation Conference*, DAC '09, pages 652–657, New York, NY, USA, 2009. ACM.

[23] M. Dong, Y.-S. K. Choi, and L. Zhong. Power-saving color transformation of mobile graphical user interfaces on oled-based displays. In *Proceedings of the 2009 ACM/IEEE International Symposium on Low Power Electronics and Design*, ISLPED '09, pages 339–342, New York, NY, USA, 2009. ACM.

[24] M. Dong and L. Zhong. Chameleon: A color-adaptive web browser for mobile oled displays. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services*, MobiSys '11, pages 85–98, New York, NY, USA, 2011. ACM.

[25] C. W. Eriksen and H. W. Hake. Absolute judgments as a function of stimulus range and number of stimulus and response categories. *Journal of Experimental Psychology*, 49(5):323, 1955.

[26] C. W. Eriksen and H. W. Hake. Multidimensional stimulus differences and accuracy of discrimination. *Journal of Experimental Psychology*, 50(3):153, 1955.

[27] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger. Architecture support for disciplined approximate programming. In *Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XVII, pages 301–312, New York, NY, USA, 2012. ACM.

[28] M. D. Fairchild. *Color appearance models*. John Wiley & Sons, 2013.

[29] J. George, B. Marr, B. E. S. Akgul, and K. V. Palem. Probabilistic arithmetic and energy efficient embedded signal pro-

cessing. In *Proceedings of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, CASES '06, pages 158–168, New York, NY, USA, 2006. ACM.

[30] T. Harter, S. Vroegindeweij, E. Geelhoed, M. Manahan, and P. Ranganathan. Energy-aware user interfaces: An evaluation of user acceptance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 199–206, New York, NY, USA, 2004. ACM.

[31] H. Hoffmann. Jouleguard: Energy guarantees for approximate applications. In *Proceedings of the 25th Symposium on Operating Systems Principles*, SOSP '15, pages 198–214, New York, NY, USA, 2015. ACM.

[32] H. Hoffmann, S. Sidiroglou, M. Carbin, S. Misailovic, A. Agarwal, and M. Rinard. Dynamic knobs for responsive power-aware computing. In *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS XVI, pages 199–212, New York, NY, USA, 2011. ACM.

[33] X. Hou and L. Zhang. A time-dependent model of information capacity of visual attention. In *Neural Information Processing*, pages 127–136. Springer, 2006.

[34] ISO, International Organization for Standardization. ISO 32000-1: 2008 Document Management—Portable Document Format—Part 1: PDF 1.7, 2008.

[35] D. Li, A. H. Tran, and W. G. J. Halfond. Making web applications more energy efficient for oled smartphones. In *Proceedings of the 36th International Conference on Software Engineering*, ICSE 2014, pages 527–538, New York, NY, USA, 2014. ACM.

[36] D. Li, A. H. Tran, and W. G. J. Halfond. Nyx: A display energy optimizer for mobile web apps. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ESEC/FSE 2015, pages 958–961, New York, NY, USA, 2015. ACM.

[37] F. Long, V. Ganesh, M. Carbin, S. Sidiroglou, and M. Rinard. Automatic input rectification. In *Proceedings of the 34th International Conference on Software Engineering*, ICSE '12, pages 80–90, Piscataway, NJ, USA, 2012. IEEE Press.

[38] M. R. Luo, G. Cui, and B. Rigg. The development of the CIE 2000 colour-difference formula: CIEDE2000. *Color Research & Application*, 26(5):340–350, 2001.

[39] W. Mason and S. Suri. Conducting behavioral research on amazon's mechanical turk. *Behavior research methods*, 44(1):1–23, 2012.

[40] M. B. McLaughlin, H. Sutter, and J. Zink. A Proposal to Add 2D Graphics Rendering and Display to C++. Technical Report N3888, ISO/IEC JTC1/SC22/WG21 Programming Language C++/SG13 - Graphics, January 2014.

[41] A. Mikami, T. Koshiyama, and T. Tsubokawa. High-efficiency color and white organic light-emitting devices prepared on flexible plastic substrates. *Japanese journal of applied physics*, 44(1S):608, 2005.

[42] G. A. Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

[43] C. Oleari, M. Melgosa, and R. Huertas. Euclidean color-difference formula for small-medium color differences in log-compressed osa-ucs space. *JOSA A*, 26(1):121–134, 2009.

[44] Qualcomm. Competitive Display Technologies, June 2009.

[45] P. Ranganathan, E. Geelhoed, M. Manahan, and K. Nicholas. Energy-aware user interfaces and energy-adaptive displays. *Computer*, 39(3):31–38, Mar. 2006.

[46] M. Rinard. Probabilistic accuracy bounds for fault-tolerant computations that discard tasks. In *Proceedings of the 20th Annual International Conference on Supercomputing*, ICS '06, pages 324–334, New York, NY, USA, 2006. ACM.

[47] M. Samadi, J. Lee, D. A. Jamshidi, A. Hormati, and S. Mahlke. Sage: Self-tuning approximation for graphics engines. In *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-46, pages 13–24, New York, NY, USA, 2013. ACM.

[48] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman. Enerj: Approximate data types for safe and general low-power computation. In *Proceedings of the 32Nd ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '11, pages 164–174, New York, NY, USA, 2011. ACM.

[49] A. Sampson, J. Nelson, K. Strauss, and L. Ceze. Approximate storage in solid-state memories. In *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-46, pages 25–36, New York, NY, USA, 2013. ACM.

[50] J. Sartori and R. Kumar. Exploiting timing error resilience in processor architecture. *ACM Trans. Embed. Comput. Syst.*, 12(2s):89:1–89:25, May 2013.

[51] M. Schuchhardt, S. Jha, R. Ayoub, M. Kishinevsky, and G. Memik. Optimizing mobile display brightness by leveraging human visual perception. In *Proceedings of the 2015 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, CASES '15, pages 11–20, Piscataway, NJ, USA, 2015. IEEE Press.

[52] N. Shanbhag. Reliable and energy-efficient digital signal processing. In *Proceedings of the 39th Annual Design Automation Conference*, DAC '02, pages 830–835, New York, NY, USA, 2002. ACM.

[53] D. Shin, Y. Kim, N. Chang, and M. Pedram. Dynamic voltage scaling of oled displays. In *Proceedings of the 48th Design Automation Conference*, DAC '11, pages 53–58, New York, NY, USA, 2011. ACM.

[54] A. Shye, B. Scholbrock, and G. Memik. Into the wild: Studying real user activity patterns to guide power optimizations for mobile architectures. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO 42, pages 168–178, New York, NY, USA, 2009. ACM.

[55] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard. Managing performance vs. accuracy trade-offs with loop perforation. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, ESEC/FSE '11, pages 124–134, New York, NY, USA, 2011. ACM.

[56] P. Stanley-Marbell. How device properties influence energy-delay metrics and the energy-efficiency of parallel computations. In *Proceedings of the Workshop on Power-Aware Computing and Systems*, HotPower '15, pages 31–35, New York, NY, USA, 2015. ACM.

[57] P. Stanley-Marbell, V. Caparros Cabezas, and R. Luijten. Pinned to the walls: Impact of packaging and application properties on the memory and power walls. In *Proceedings of the 17th IEEE/ACM International Symposium on Low-power Electronics and Design*, ISLPED '11, pages 51–56, Piscataway, NJ, USA, 2011. IEEE Press.

[58] P. Stanley-Marbell and D. Marculescu. A programming model and language implementation for concurrent failure-prone hardware. In *2nd Workshop on Programming Models for Ubiquitous Parallelism, PMUP'06*, 2006.

[59] P. Stanley-Marbell and M. Rinard. Efficiency limits for value-deviation-bounded approximate communication. *IEEE Embedded Systems Letters*, 7(4):109–112, Dec 2015.

[60] P. Stanley-Marbell and M. Rinard. Value-deviation-bounded serial data encoding for energy-efficient approximate communication. Technical Report MIT-CSAIL-TR-2015-022, MIT Computer Science and Artificial Intelligence Laboratory (CSAIL), June 2015.

[61] K. W. Tan, T. Okoshi, A. Misra, and R. K. Balan. Focus: A usable & effective approach to oled display power management. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 573–582, New York, NY, USA, 2013. ACM.

[62] Texas Instruments. Enabling the Next Generation of Automotive Head-Up Display Systems. Literature number DLPA043, October 2013.

[63] Texas Instruments. DLP Technology for Near Eye Display. Literature number DLPA051, September 2014.

[64] The University of Southern California Signal and Image Processing Institute. The USC-SIPI Image Database, 2016.

[65] TI. INA219 Zero-Drift, Bi-Directional Current/Power Monitor with I2C Interface, 2011.

[66] P. F. Van Kessel, L. J. Hornbeck, R. E. Meier, and M. R. Douglass. A mems-based projection display. *Proceedings of the IEEE*, 86(8):1687–1704, 1998.

[67] P. Verghese and D. G. Pelli. The information capacity of visual attention. *Vision research*, 32(5):983–995, 1992.

[68] J. Wang, X. Lin, and C. North. GreenVis : Energy-Saving Color Schemes for Sequential Data Visualization on OLED Displays. Technical report, Department of Computer Science, Virginia Tech, Blacskburg, VA, 2012.

[69] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *Trans. Img. Proc.*, 13(4):600–612, Apr. 2004.

[70] C. Worth and K. Packard. A Realistic 2D Drawing System, October 2003.

[71] C. Worth and K. Packard. Xr: Cross-device rendering for vector graphics. In *Linux Symposium*, page 480, 2003.

[72] L. Zhong and N. K. Jha. Energy efficiency of handheld computer interfaces: Limits, characterization and practice. In *Proceedings of the 3rd International Conference on Mobile Systems, Applications, and Services*, MobiSys '05, pages 247–260, New York, NY, USA, 2005. ACM.

[73] Z. A. Zhu, S. Misailovic, J. A. Kelner, and M. Rinard. Randomized accuracy-aware program transformations for efficient approximate computations. In *Proceedings of the 39th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '12, pages 441–454, New York, NY, USA, 2012. ACM.