Optimizing subpixel rendering using a perceptual metric

Joyce Farrell
Shalomi Eldar
Kevin Larson
Tanya Matskewich
Brian Wandell

Abstract — ClearType is a subpixel-rendering method designed to improve the perceived quality of text. The method renders text at subpixel resolution and then applies a one-dimensional symmetric mean-preserving filter to reduce color artifacts. This paper describes a computational method and experimental tests to assess user preferences for different filter parameters. The computational method uses a physical display simulation and a perceptual metric that includes a model of human spatial and chromatic sensitivity. The method predicts experimentally measured preferences for filters for a range of characters, fonts, and displays.

Keywords — *Image-quality metrics, subpixel rendering, display simulation.*
DOI # 10.1889/JSID19.8.513

1 Introduction

Recent advances in display technology take advantage of the arrangement of the display primaries within a pixel (subpixels) to enhance the spatial resolution of the rendered image. Novel display architectures have been proposed that vary the number of subpixel colors and their spatial arrangements.[1] Because there are many design options, evaluation is an important and challenging task.[2] The ability to evaluate display architectures and corresponding subpixel-rendering algorithms using a computational method would simplify the design process and enable exploration of many more architectures. This paper reports the results of computational and behavioral experiments that evaluate the use of a spatio-chromatic perceptual metric (S-CIELAB) to optimize a subpixel text-rendering method (ClearType).

ClearType is a subpixel-rendering method designed to increase the horizontal resolution of rendered text on certain displays. Specifically, pixels in many color displays are composed of three horizontally adjacent subpixels that emit the red, green, and blue (RGB) primary lights. Conventional display algorithms treat the subpixels as spatially coincident and forfeit the potential resolution enhancement in the horizontal dimension. ClearType uses the subpixel elements to triple the horizontal spatial resolution during the font rasterization process.

The subpixels are colored, so without further processing ClearType fonts would have color artifacts. A second element of rendering a ClearType character, then, is to reduce the visibility of the color artifact by spatially averaging in the horizontal dimension. This averaging slightly reduces the horizontal resolution. Platt[3] used principles from vision science to quantify the perceptual tradeoff between spatial resolution and color errors. Specifically, he used the S-CIELAB spatial filters[4–6] as an approach for selecting spatio-chromatic filters to minimize the visibility of color artifacts. The initial work suggested applying a total of nine spatial filters corresponding to each of the three input and output channels. For example, the filters applied to create the red color channel output are denoted by R → R, G → R, and B → R. The outputs of these three filters are summed to create a filter that is centered on the red pixel. Similarly, there are three color filters for the green and blue channels.

Subsequently, Betrisey *et al.*[7] found that the cross-channel filters (R → G, R → B, G → B, *etc.*) have relatively little power, and the three within-channel filters (R → R, G → G, and B → B) are nearly identical but centered at different subpixels. Hence, Betrisey *et al.* replaced the nine filters with one filter and referred to this approximation as RGB decimation with displaced filters. These simplifications are used in the real-time implementation of ClearType.

There are significant differences between the ClearType rendering and conventional anti-aliasing. Anti-aliasing refers to the process of removing high-resolution information from a signal prior to sampling at a lower resolution in order to remove the possibility that the high-resolution information becomes "aliased" into a lower-frequency signal. ClearType does not remove high-resolution information prior to sampling.

The ClearType process begins with a very high – essentially continuous – representation of the desired font that is sampled by the lower-resolution pixel grid. Unlike anti-aliasing, the ClearType filtering does not prevent the artifacts caused by sampling the high-resolution image onto the low-resolution chromatic display grid. Rather, it blurs the signal after sampling to reduce the visibility of the sampling artifacts. The visibility of blur and color artifacts generated by different ClearType rendering depend on the physical characteristics of the display, the ClearType filters, and the viewing conditions (*e.g.*, viewing distance).

Received 02/02/11; accepted 06/09/11.
J. E. Farrell is with Stanford University, Stanford Center for Image Systems Engineering, 350 Serra Mall, Stanford, CA 94305, USA; telephone 650/736-8030, e-mail: joyce_farrell@stanford.edu.
S. Eldar and B. Wandell are with Stanford University, Psychology, Stanford, CA, USA.
K. Larson and T. Matskewich are with Microsoft Corp., Advanced Reading Technology, Stanford, CA, USA.

The sampling theorem guides the design of anti-aliasing filters, but says nothing about visibility. Because there is no theorem to guide the design of ClearType filters, we created a simulation of the display[8] and evaluated the visibility of blur and color artifacts using S-CIELAB.[5] To check the simulation, we measured subjects' preferences between high-contrast text (black/white) rendered on different displays with a range of filter parameters. We compared the measured subject preferences with the size of the artifacts predicted by S-CIELAB.

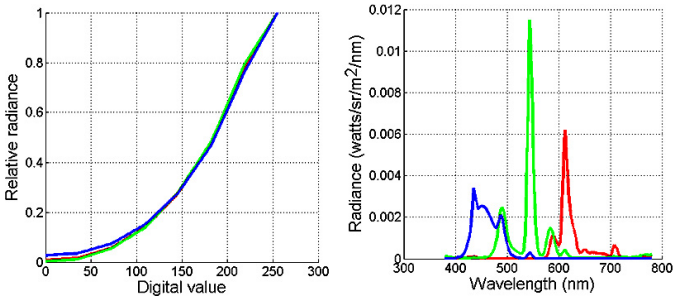

FIGURE 2 — Display gamma (left) and spectral power distributions (right) for the two calibrated displays used in this study.

2 General methods

2.1 Display measurement and modeling

The display simulation toolbox (DST) provides a framework that guides the estimation and simulation of the spatial-spectral radiance emitted from a display by any image.[8] Calculating the spectral radiance image is essential for predicting the visibility of blurring and color artifacts; unlike the digital image values (RGB) usually used for image-quality predictions, the spectral radiance image is the stimulus that actually reaches the eye.

The DST uses three functions to predict the spatial-spectral radiance emitted by a display. First, the DST converts digital values into a measure of the linear intensity (display gamma). Second, the DST models the spatial spread of light using a point-spread function for each color component (subpixel point spread function). Third, the DST uses the spectral power distributions of the display color primaries to calculate the spectral composition of the displayed image. These three functions – the display gamma, the subpixel point-spread functions, and the spectral power distributions – are sufficient to characterize the performance of displays with independent pixels.[9]

Figure 1 shows the subpixel arrangements for two LCD monitors. These calibrated measurements were used to generate point-spread functions for the red, green, and blue subpixel components of the two different displays. Figure 2 shows the display gamma and spectral power distributions that were, in this case, matched for the two different displays. Elsewhere, we demonstrate that a model with independent pixels accurately characterizes the properties of these displays.[8]

In the simulation, the displayed letter is represented as a spatial array of spectral radiance functions (photons/sec/nm/sr/m^2), sometimes called a hyperspectral data cube. Figure 3 shows a magnified view of a character rendered on one of the displays (Dell LCD 1907FPc). Two graphs represent quantities that can be calculated from the hyperspectral data. The top graph shows the luminance profile along a horizontal line. The bottom graph plots the spectral power distribution of the white background.

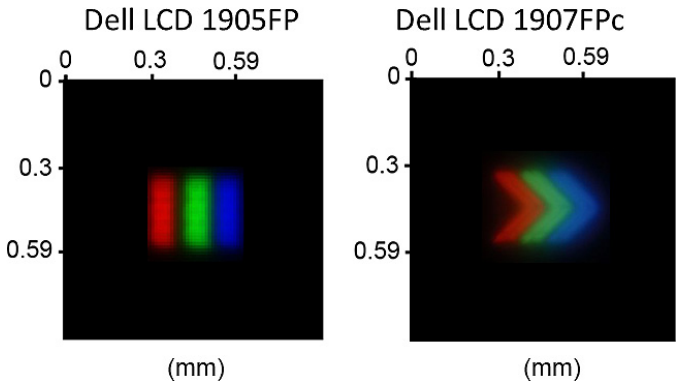

FIGURE 1 — Images of white pixels for the two different LCD monitors, the Dell 1905FP (left) and the Dell 1907FPc (right).

2.2 Subpixel rendering

ClearType renders text at subpixel resolution and then applies a symmetric one-dimensional mean-preserving filter to reduce color artifacts. We investigate the effects of varying the parameters of a five-tap filter (a, b, c, b, a). To preserve the mean, the filter parameters sum to one ($2a + 2b + c =$

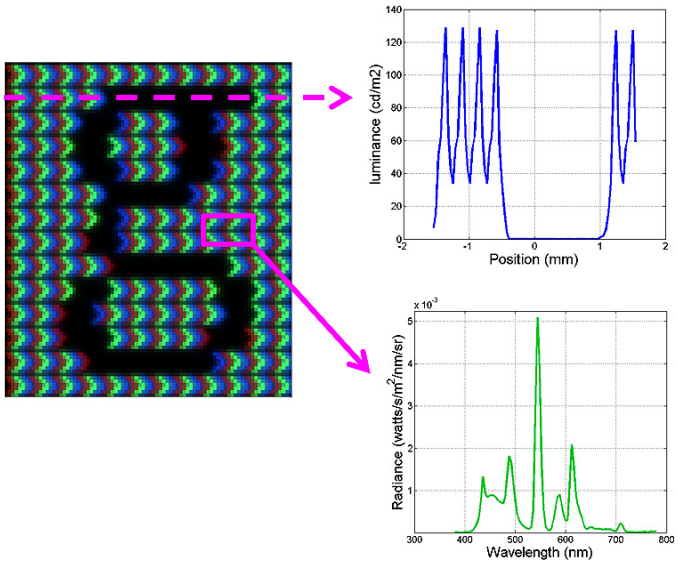

FIGURE 3 — Magnified view of a character rendered on one of the displays (Dell LCD 1907FPc) with luminance profile along a horizontal graph (top graph) and spectral power distribution of the white background (bottom graph).

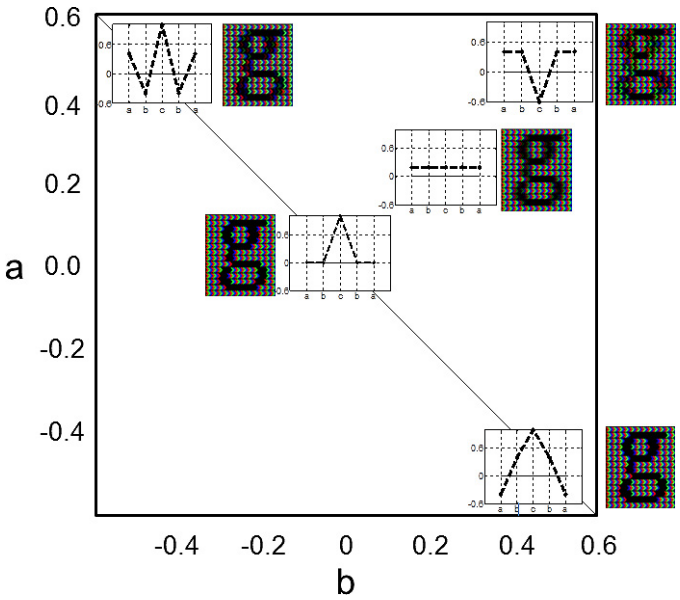

FIGURE 4 — Illustration of ClearType five-tap filters with varying (*a*, *b*) parameters. Filters along the negative diagonal have *c* = 1. The insets show the filters along with color images that illustrate the effect of blurring the character with these different filters.

1). Consequently, the filters can be described by a two-dimensional parameterization (*a*, *b*).

The variation in the filters in the (*a*, *b*) space is illustrated in Fig. 4. The filter parameters (0, 0) and thus *c* = 1 represent the impulse function; this filter produces the highest spatial resolution rendering. Subjects often prefer (*a*, *b*) filter parameters different from (0,0) because some spatial blurring can reduce the visibility of chromatic artifacts.

The computational and experimental methods assessed artifacts and preferences for filter values in a restricted range, –0.6 < *a*, *b* < 0.6, and subject to the constraint that filter values with *c* > 1 are not allowed because they produce many out-of-range display values.

2.3 Predicting artifact visibility

The visibility of spatio-chromatic artifacts is evaluated by comparing the test character rendered on a real display to an ideal reference. We define the ideal reference as a black/white test character rendered on a monochrome display matched in resolution to the subpixel resolution of the RGB display. It is possible to use other reference images, such as a character rendered on a higher-resolution display, but we chose not to do this because ClearType fonts adjust the letter shape given the available resolution. Using the matched monochrome reference image, S-CIELAB quantifies the visibility of differences in contrast and color without confounding differences in font outline.

To compute the visible error, we model the display and then apply the S-CIELAB metric (Fig. 5). The first step is to calculate the displayed spatial-spectral radiances of a letter rendered on a color display and the corresponding ideal

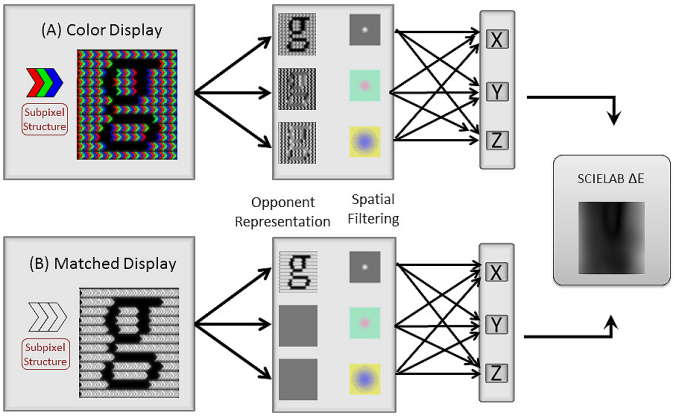

FIGURE 5 — Calculation of the S-CIELAB difference between the displayed radiance of a letter rendered on a color display (A) and the displayed radiance of the same letter rendered on a monochrome display matched in resolution (B). The error map is the S-CIELAB difference (Δ*E*) between the two radiance images. The mean Δ*E* (averaged across the error map) is used as a summary statistic.

reference. Then, the point-by-point S-CIELAB difference (Δ*E*) between the two radiance images is calculated and an error map is generated. We use the mean S-CIELAB difference (Δ*E*) (averaged across the error map) as an estimate of the visible difference between the ClearType rendering and the ideal reference. A Δ*E* value of 1 is designed to be near perceptual threshold. Hence, when the Δ*E* value for two filters is near 1, the effects of the filters should be quite similar.

3 Computational experiments

In the computational experiments described below and in the perceptual experiments described later, we analyzed the letters g, s, and v in a serif (Georgia) and sans-serif (Arial) font and rendered on the two different displays (Fig. 1). The difference between the ideal reference and the rendered image has a mean ΔE of 2 or more. Hence, the metric predicts that the ideal reference will appear different from the rendered image.

3.1 Filter parameters for different letters, fonts, and displays

The iso-Δ*E* contours in the filter parameter space, (*a*, *b*) are shown in Fig. 6. The six plots show the visible errors for g, s, and v, in Georgia and Arial fonts, rendered on an LCD (Dell 1907FPc). Each contour encloses the (*a*, *b*) filters that produce a visible error less than or equal to a specific S-CIELAB Δ*E* level; the contours are separated by a Δ*E* = 0.5. Simulations show that there are regions within the (*a*, *b*) plane that differ by less than one Δ*E* value.

Across all conditions iso-Δ*E* contours have similar orientation. The (*a*, *b*) coefficients that generate the smallest S-CIELAB Δ*E* values fall along a negative diagonal (constant *a* + *b* values). This negative diagonal that minimizes the Δ*E* value differs from the *c* = 1 negative diagonal.

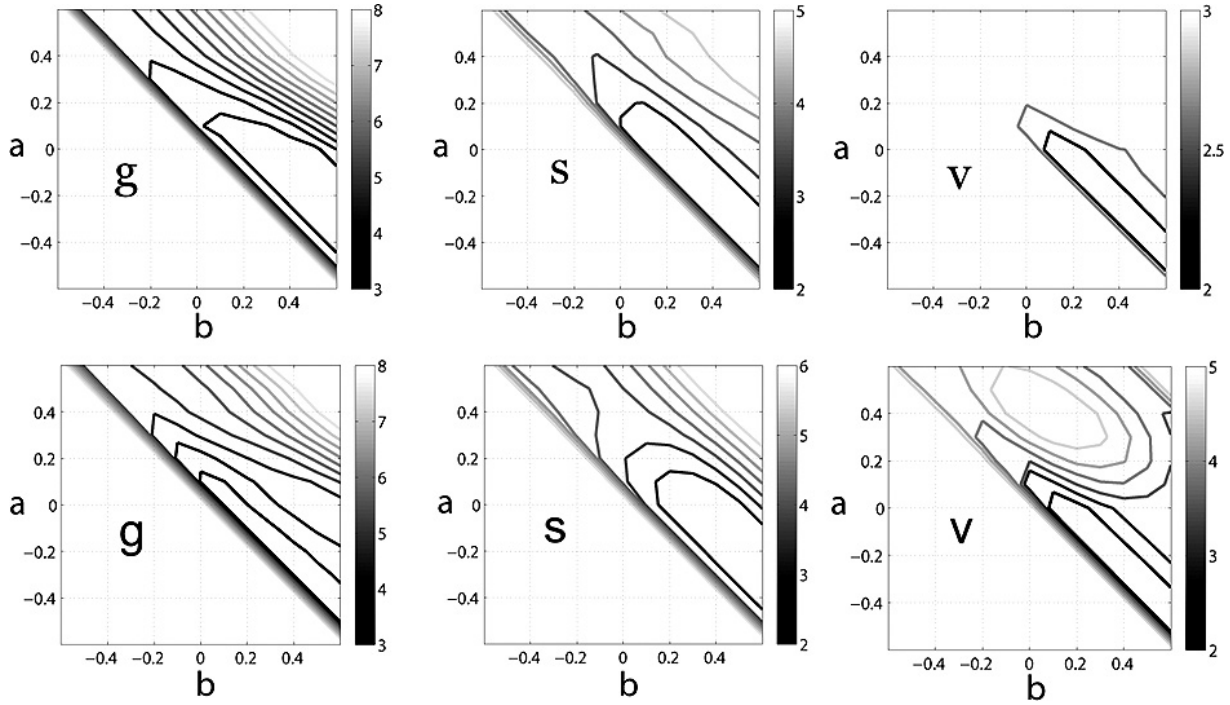

FIGURE 6 — Predicted iso-Δ*E* contours for the letters g, s, and v from the Arial-font family, rendered on the Dell 1907FPc display (see Fig. 1) and viewed from a distance of 15 in.

Hence, S-CIELAB predicts that blurring will reduce the visibility of the color artifacts.

Across conditions the smallest visible errors occur when the value of the *a* filter coefficient is small. Hence, the filters with smallest color artifacts have small values of *a* and large values of *c*. This region in the *a*, *b* parameter space, near (–0.1, 0.2) and *c* ~ 0.8 represents a perceptual compromise between preserving resolution and reducing chromatic artifacts. Note that when *a* = 0, the ClearType filter is a simple three-tap filter.

Figure 7 shows the iso-Δ*E* contours for the same letter rendered on two different displays. There are small differences in the absolute Δ*E* levels across the conditions but again the ClearType filters that minimize the visibility of artifacts are near (–0.1, 0.2).

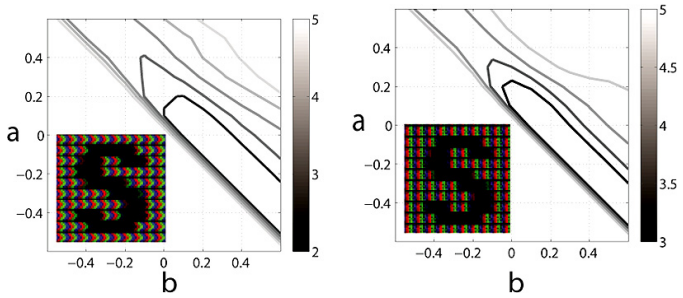

FIGURE 7 — Predicted iso-Δ*E* contours for the letter s from the Georgia-font family, rendered on the Dell 1907FPc (left) and Dell 1507FP (right) displays and viewed from a distance of 15 in.

3.2 PSNR

The peak signal-to-noise (PSNR) metric is widely used to evaluate image quality. The PSNR is calculated from the digital RGB values that represent an image. In this application any calculation of error based on RGB values is inappropriate for several reasons. First, the ClearType representation is based on the spatial array of single primary subpixels. The RGB representation does not account for the spatial position of the subpixels that are at the heart of the subpixel-rendering architecture.

Further, the values of the ideal reference are always R = G = B = (255 or 0), while the ClearType representation always has at least two of the R, G, and B values equal to 0. Consequently, the PSNR value will inescapably be very low and without real meaning.

Finally, the PSNR calculation has no room for critical variables, such as viewing distance, subpixel structure, and visual resolution. All of these factors matter in the calculation, and none of these factors are accounted for by the PSNR metric. In this application, it is essential to use a visibility metric that incorporates features of human color-pattern visibility.

4 Perceptual experiments

A series of visual psychophysical experiments were conducted in which subjects chose a preferred rendering from a set of alternatives.

4.1 Experimental procedures

4.1.1 Subjects

Three subjects (two males and one female) participated in this experiment. All subjects had normal or corrected to normal vision.

4.1.2 Stimuli

Stimuli were the letters g, v, and s in Georgia 12-point font and Arial 11-point font. They were rendered on two different displays that have different pixel structures, as shown in Fig. 1. Subjects viewed the stimuli using a chin rest placed 0.38 m (15 in.) from the display.

Different versions of the letter were created by systematically varying the (a, b) parameters of a one-dimensional symmetric mean-preserving five-tap filter (a, b, c, b, a). The parameter values (a, b), were constrained such that c would never be greater than 1.0. We used a GAST procedure, described next, to find the preferred (a, b) parameters.

4.1.3 The GAST procedure

Even for a single display, and a letter with fixed font size and family, the number of different versions of the same letters (created with different filter parameter values) is quite large. For example, 88 different filters represent the possible ClearType versions when $-0.6 \le a, b \le 0.6$ and $|c| \le 1$, and a, b are sampled at a rate of 0.1. There are $(88 \times 87)/2$ or 3828 possible pairwise comparisons of these different versions. Each pairwise comparison should be presented at least 10 times, yielding 38,280 trials for one letter. Clearly, a full pairwise comparison task is not feasible, even for one letter.

In this experiment, the number of pairwise comparisons was reduced by using the GAST (Gradient Ascending Subjective Testing) method introduced by Voran and Catellier.[10] The GAST procedure combines pairwise stimulus comparisons with gradient ascent optimization algorithms to search efficiently for stimuli that optimize perceived image quality. In this experiment, the GAST procedure was used to find the (a, b) parameter values that subjects prefer the most.

In each trial, subjects were presented with pairs of filtered versions of the same character. They were asked to select one of five possible responses: the left version is much better than the right version, the left version is slightly better than right version, the two versions look the same, the right version is slightly better than the left version, or the right version is much better than left version.

Each block of trials began by randomly selecting a point in the (a, b) filter parameter space, referred to hereafter as the starting point. This point defined the filter parameter values that were used to generate one of the ClearType versions presented to subjects in the first pairwise comparison task. The other ClearType version was created by increasing or decreasing the **a** parameter value. After recording the subjects preference judgment, the next trial presented the Cleartype version defined by the starting point with a version created by increasing or decreasing the **b** parameter value. The GAST algorithm used the subjects' responses in these two trials to estimate the direction in the two-dimensional a, b filter space in which preference increases. This direction defined a line or vector of filter values beginning with the original a, b starting point and ending at the boundary of the defined parameter space. The GAST procedure then presented a series of pairwise ClearType renderings that had (a, b) points along this line until it found the parameters that maximized preference. This became a new starting point and the two adjacent and perpendicular filter values were selected for the following trials. The algorithm iterated between finding directions in the a, b filter space and finding maximum preference along a direction in filter space until there was no change in direction or location that increased preference. The endpoint defined the optimal filter coefficients (a, b) for this series of trials.

In each block of trial, subjects completed 10 series of trials for each letter. The 10 series were interleaved throughout the block of trials. Each block of trials generated 10 ending points. In each experimental session, subjects completed six blocks of trials, one for each of the letters g, v, and s in Georgia 12-point font and Arial 11-point font, respectively, generating a total of 60 GAST series for each session. Each GAST run started from a random starting point in the (a, b) parameter space and "ascended," based on subject's responses, until it reached an (a, b) endpoint. Over the course of several weeks, each of the three subjects completed six experimental sessions during which the stimuli were presented on one of two displays – three sessions for a Dell LCD 1905P display and three sessions for a Dell LCD 1907FPc. Subjects completed 30 GAST runs for each of the three letters shown in two different fonts and on each display, yielding a total of 360 GAST runs.

5 Results

Figure 8 shows the 1080 GAST endpoints (three letters × two fonts × 2 displays × three subjects × 30 GAST runs). The data points in each panel are colored coded so that the data can be seen as grouped by fonts (A), letters (B), displays (C), and subjects (D). Comparing the data across panels, it is apparent that the variability between subjects is larger than the variability due to letters, fonts, and displays.

The variability in subject preference judgments can be quantified by calculating the bootstrap distribution of mean c parameters for each condition of subject, letter, font and display.[11] In 59 of the 72 comparisons (82%), the mean c parameter for one subject was more than two standard deviations from the bootstrap distribution of mean c values for another subject. The variability in c values was greater across subjects than across letters, fonts, or displays. Within

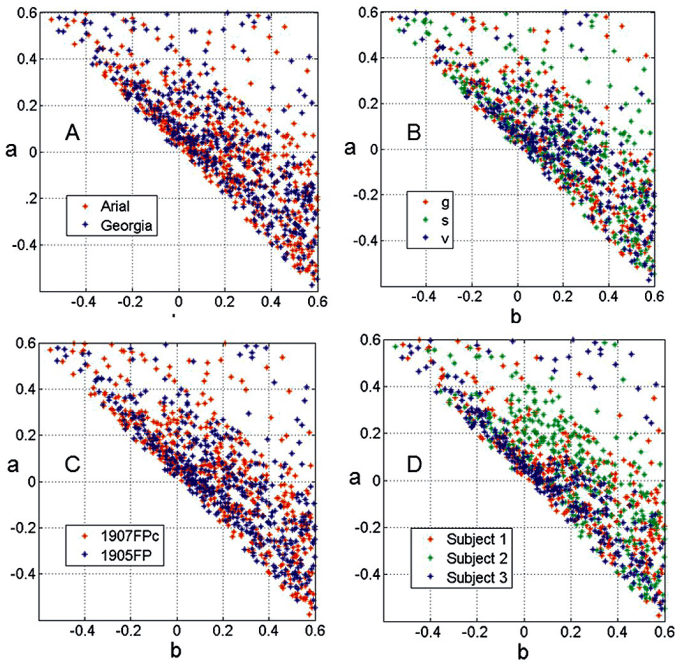

FIGURE 8 — 1080 GAST endpoints (3 letters * 2 fonts × 2 displays × 3 subjects × 30 GAST runs). The data points in each panel are colored coded so that the data can be seen as grouped by fonts (A), letters (B), displays (C), and subjects (D).

subjects, 50% of the comparisons between different letters were significant, 50% of the comparisons between different fonts were significant, and 11% of the comparisons between displays were significant ($p < 0.05$).

The value of the GAST procedure is that it generates preference estimates from a large set of comparisons; but the method also has limitations. Specifically, subjects can become stuck in a local preference maximum rather than ascending to a global preference maximum. Some of the variability across subject preference judgments may be due to this GAST limitation.

Figure 8 plots all 1080 GAST end points in the (a, b) parameter space (three letters × two fonts × two displays × three subjects × 30 GAST runs). Despite the variability both within and between subjects, the data share important similarities with one another and with the region identified by S-CIELAB as having the smallest color artifacts. The GAST endpoints fall in a region that covers a negative diagonal and is near (–0.1, 0.2). Subjects chose filters that produce some blurring; less than 4% of the endpoints fall within the highest resolution region in which $c > 0.95$.

6 Discussion

The main purpose of this investigation was to build and evaluate a computational method to predict subject preferences for simple colored patterns. We created a model of the display stimuli (Figs. 1–3), and we adapted a color difference metric for the task of making visible artifact predictions (Figs. 5–7) for individual characters on specific displays. For each of these cases, the S-CIELAB measure identifies a region in the (a, b) plane that minimizes the visibility of the difference between an ideal and rendered font. The region within $\Delta E = 1$ of the minimum tends to fall along a negative diagonal centered near (–0.1, 0.2); subjects preferences generally align with this region. Next, we consider how the computational methods can be used to optimize filter parameters across a larger set of characters and displays.

6.1 Optimizing over a larger character set

We can use simulations to find preferred filter parameters for a larger collection of letters by searching for the filter parameters that minimize the total error $\mu = \sum_{i=1}^{26} \Delta E_i$, where the sum is across all 26 letters. The set of acceptable filters will be those with a sum within $\mu + \theta$, where θ is a threshold constant. For example, across all 26 letters in the Georgia 12-point font rendered on the Dell LCD 1905FP display, the set of filters with a total error less than $\mu + 1$ fall near the line defined by $c = 0.4$ with $a \leq 0.1$. Hence, this method predicts that the three-tap filter (0.3, 0.4, 0.3) will be close to the filter with the smallest total error. The data in Fig. 8 show that both filters fall within the preference range in the test conditions used here.

Selecting the best three-tap filter across the larger set is a compromise whose cost can be quantified. Specifically, each letter has a filter that minimizes ΔE. We can measure the difference between the best ΔE for this letter-specific optimal filter and the ΔE value for the three-tap filter optimized across the entire set. Using this measure for the three-tap filter (0.3, 0.4, 0.3), the largest ΔE difference is $0.84\Delta E$ (for the letter h).

The same analysis can be used to optimize a three-tap filter for a collection of letters, fonts, and two displays. For example, the three-tap filter for the 26 letters in the two fonts rendered on the two displays is (0.2, 0.6, 0.2), and the largest difference is $2\Delta E$.

6.2 Application to the design and evaluation of novel displays

Silverstein *et al.*[2] used physical simulations to evaluate the quality of color lines and curves rendered on displays with different color subpixel arrays. Using physical simulations and subjective evaluation, they quantified the effects that subpixel rendering had on perceived image quality.

The results of our study suggest that computational methods can potentially replace physical simulations and subjective ratings. Substituting computational methods for experimental methods in the design and evaluation process can greatly increase efficiency. Computational methods can be used to understand the effects of display systems with novel pixel structures, such as the RGBG pixel arrays evaluated by Silverstein *et al.*,[2] the RGBW pixel arrays used in the

Kodak AMOLED displays,[12] and the many different PenTile™ pixel arrays.[1] Furthermore, these methods also can be used to optimize subpixel-rendering algorithms needed for implementing these novel displays.

Finally, because the method uses the S-CIELAB metric, the computation takes into account the viewing conditions and specific features of the human visual system. This method can be used to adjust parameters that account for viewers with vision impairments including common ailments of an aging population, such as presbyopia.

Acknowledgments

The authors thank Greg Hitchcock for his advice and encouragement and the Microsoft Corporation for supporting this research.

References

1 C. H. B. Elliott, "Co-optimization of color AMLCD subpixel architecture and rendering algorithms," *SID Symposium Digest* **33**, 172–175 (2002).
2 L. D. Silverstein *et al.*, "Effects of spatial sampling and luminance quantization on the image quality of color matrix displays," *J. Opt. Soc. Am. A (Optics and Image Science)* **7**(10), 1955–1968 (1990).
3 J. C. Platt, "Optimal filtering for patterned displays," *IEEE Signal Processing Lett.* **7**(7), 179–181 (2000).
4 X. Zhang and B. A. Wandell, "A spatial extension to CIELAB for digital color image reproduction," *SID Symposium Digest* **27**, 731–734 (1996).
5 X. Zhang, "S-CIELAB: A spatial extension to the CIE *L*a*b** DeltaE Color Difference Metric" (1998), retrieved http://white.stanford.edu/~brian/scielab/scielab.html.
6 X. Zhang *et al.*, "Applications of S-CIELAB: A spatial extension to CIELAB," *Proc. {IS&T/SPIE} 9th Annual Symposium on Electronic Imaging* (1997).
7 C. Betrisey *et al.*, "Displaced filtering for patterned displays," *SID Symposium Digest* **31**, 1–4 (2000).
8 J. Farrell *et al.*, "A display simulation toolbox for image quality evaluation," *IEEE/OSA J. Display Technol.* **4**(2), 262–270 (2008).
9 D. H. Brainard, "Calibration of a computer controlled color monitor," *Color Res. Appl.* **14**, 23–34 (1989).
10 S. Voran and A. Cattellier, "Gradient ascent paired-comparison subjective quality testing," *Presented at the Proceedings of the First International Workshop on Quality of Multimedia Experience*, San Diego, CA, July 29–31 (2009).
11 B. Efron, "Better bootstrap confidence intervals," *J. Am. Statistical Assoc.* **82**(397), 171–185 (1987).
12 A. Arnold *et al.*, "Full-color AMOLED with RGBW pixel pattern," *J. Soc. Info. Display* **13**(6), 525–535 (2005).

Joyce Farrell is a Senior Research Associate at the Stanford School of Engineering and Executive Director of the Stanford Center for Image Systems Engineering (SCIEN). She earned her Ph.D. from Stanford University in psychology, specializing in visual perception and psychophysics. Her research focuses on methods for quantifying human perception of image quality.

Shalomi Eldar received his B.Sc. degree in computer science from Haifa University, Israel, in 2009, and his B.Ed. degree in 2007. He is currently a Software Engineer at Lab126 (Amazon.com subsidiary) and a Judaic Studies researcher. In 2010, he was a Research Assistant with the VISTA Lab at Stanford University.

Kevin Larson received his Ph.D. degree in cognitive psychology from the University of Texas in 2000 for his studies on reading acquisition. He is a researcher on Microsoft's Advanced Reading Technologies team. He works with type designers, psychologists, and computer scientists on improving the on-screen reading experience.

Tanya Matskewich received her Ph.D. degree in computer science from the Hebrew University of Jerusalem. She works on the Microsoft Advanced Reading Technologies team on improving on-screen reading experience by developing mathematical models for text rendering and understanding aspects of visual perception.

Brian A. Wandell is the first Isaac and Madeline Stein Family Professor. He joined the Stanford Psychology faculty in 1979 and is a member, by courtesy, of Electrical Engineering, Ophthalmology, and Radiology. He is also the Director of the Center of Cognitive and Neurobiological Imaging (CNI). His research projects center on how we see, spanning topics from visual disorders, reading development in children, to digital imaging devices and algorithms.