USENIX Association

Proceedings of MobiSys 2003: The First International Conference on Mobile Systems, Applications, and Services

San Francisco, CA, USA
May 5-8, 2003

Energy-Adaptive Display System Designs for Future Mobile Environments

Subu Iyer† Lu Luo‡ Robert Mayo† Parthasarathy Ranganathan†

†*Hewlett Packard Labs*
1501 Page Mill Road MS 1177
Palo Alto California 94304 USA
{*Subu.Iyer,Bob.Mayo*}*@hp.com*
Partha.Ranganathan@hp.com

‡*School of Computer Science*
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213 USA
luluo@cs.cmu.edu

Abstract

The utility of a mobile computer, such as a laptop, is largely constrained by battery life. The display stands out as a major consumer of battery energy, so reducing that consumption is desirable. In this paper, we motivate and study **energy-adaptive display sub-systems** that match display energy consumption to the functionality required by the workload/user. Through a detailed characterization of display usage patterns, we show that screen usage of a typical user is primarily associated with content that could be displayed in smaller and simpler displays with significantly lower energy use. We propose example energy-adaptive designs that use emerging *OLED displays* and software optimizations that we call *dark windows*. Modeling the power benefits from this approach shows significant, though user-specific, energy benefits. Prototype implementations also show acceptability of the new user interfaces among users.

1 Introduction

With the increased acceptance and use of mobile devices such as laptops and pocket computers, mobile computing systems are rapidly becoming one of the key markets of interest for computing systems. Since the batteries on these mobile systems are typically limited in capacity, reducing the energy consumption is one of the key challenges in designing mobile systems.

Among the various components that contribute to the consumption of electrical energy, the display sub-system (the electronics associated with the visual representation of the data generated by the system - namely the display and the controller) often plays an important role. For example, Udani and Smith report that the display component of the system can easily take over half the total energy of a laptop system [7]. Similarly, Choi et al. report that the display component of the system consumed close to 61% of the total power of the system for a handheld device. Furthermore, unlike some of the other components of the system, display power consumption has traditionally been relatively invariant across technology shrinks and unamenable to opportunities to exploit "slack," [5] making it a likely greater fraction of the total power of future systems.

Previous approaches to reducing display power consumption have either focused on aggressively turning off the entire display when it is not being used or have resorted to designing systems with lower-quality or smaller sized displays to minimize power. However, new technologies, such as Organic Light Emitting Diodes (OLEDs), are becoming available that allow lower power consumption when a reduced area of the screen is in use.

We propose utilizing this new flexibility to reduce energy consumption. Our work is based on the intuition that different workloads and users have varying display needs. Having a *"one-size-fits-all"* display targeted at the needs of the most aggressive workload/user often leads to large energy inefficiencies in the display energy consumptions of other workloads and users. Consequently, an *energy-adaptive system design* that consumes energy only on portions and characteristics of the screen that are being used by the application and are relevant to the user, can achieve energy benefits.

In this work, we make two key contributions. First, we perform a detailed characterization of the display screen usage of a representative test user population. Our results indicate that, on average, our users use only about 60% of the screen area available to them. Additionally, screen usage is often associated with content that could have been equivalently displayed, with no loss in visual quality, on much simpler lower-power displays (lower size, resolution, color, brightness, refresh rates, etc.) Using a detailed analysis of the user traces, we also correlate our results to user and application behavior. To the best of our knowledge, our work is the first to identify, quantify, and analyze such mismatch opportunities in workload/user needs and current display properties. Overall, our results indicate that an energy-adaptive system design that matches display power consumption to the functionality required by the workload/user can significantly reduce the energy consumption of future display sub-systems.

Building on the insights from the above analysis, we propose example energy-adaptive display system designs. At the hardware level, our designs leverage emerging Organic LED (OLED) displays [3] that use energy proportional to the overall light output of the display. At the software level, we propose *dark windows* optimizations that enable the windowing environment to change the brightness and color of areas of the screen that are not of interest to the user. We model the power benefits and study the user experience with our designs. Our results indicate significant, though user-specific, energy reductions with acceptable user interfaces.

The rest of the paper is organized as follows. Section 2 discusses our user study in greater detail. Section 3 discusses the example energy-adaptive display systems that we consider and our experiences with those. Section 4 further explores the design space with energy-adaptive displays and Section 5 discusses related work. Section 6 concludes the paper and discusses future work.

2 User Study

This section presents the results from our user study characterizing typical system usage of a representative user test population. Our goal was to understand the screen usage patterns and identify opportunities for power reduction. Section 2.1 discusses our methodology and Section 2.2 discusses our results.

2.1 Methodology for the user study

Our user study is based on usage of the Microsoft Windows environment by seventeen users. We chose Microsoft Windows environments because of their widespread acceptance and representativeness of the general mobile market. The users were chosen to include characteristics representative of typical usage and cover a cross section of mobile system usage (administrative tasks, code development, personal productivity, entertainment, etc.). Since we were interested in understanding both current and future user behavior when using mobile environments, we studied both laptop and desktop users. (Many of our laptop users used their machines as their main machine - both as a desktop and a laptop.) The systems used by the test users include a variety of screen sizes and display resolutions. Column 2 in Figure 1 summarizes the properties of the systems used by our test population.

An application-level logger program was run on the users' machines for times ranging from 1 to 14 days. The logger program was used to collect periodic information about (i) the current window of focus – its size, its location, and its title and (ii) the size of total screen area used (all non-minimized windows). Our sampling rate was set to once a second. Screen savers were set to turn on after a reasonable time (1-5 minutes) to allow us to isolate only the usage patterns when the user was active.

Column 3 in Figure 1 summarizes the length of the user traces. The traces range from 9 hours to 346 hours. The variation in the traces represent the differences in how individual users used their machines during their participation in the study. Overall, our samples represent close to 100 days of continuous computer usage time.

Column 4 in Figure 1 summarizes the length of the "active" user traces, after factoring out the time spent in the screen saver as an indication of the time the user was idle. Traces are still collected during the time it takes for the screen saver to be activated, but given the length of our logs, the effect of this is minor. The sizes of our active user logs range from about 6 hours to 61 hours of computer usage per user. Given that this is the time we are interested in, the rest of the paper will focus on the active window usage without considering the time spent in the screen saver. Existing technologies can save power spent in the idle time by turning off the displays.

2.2 User study results

Average screen usage. Figure 1 summarizes the information about the screen usage. Columns 5 and 6 present the mean and standard deviation, per user, for the screen usage of the window of focus. For this study, we define

User	Display (column 2)	Log length (column 3)	Active samples (column 4)	Screen usage for active samples: Window of focus – Mean	Window of focus – Std. dev	Background windows – Mean	Background windows – Std. dev
Desktop user population							
1	19” 1024x768	210 hours	33 hours	62.8%	38.5%	10.6%	21.2%
2	21” 1280 x 1024	346 hours	61 hours	57.2%	22.3%	11.6%	28.5%
3	19” 1280 x 1024	214 hours	31 hours	46.3%	19.7%	30.4%	19.7%
4	19” 1280 x 1024	64 hours	43 hours	36.7%	14.5%	34.1%	8.8%
5	19” 1280 x 1024	253 hours	27 hours	44.5%	22.7%	32.6%	21.1%
6	21” 1280 x 1024	229 hours	31 hours	55.5%	18.4%	24.7%	17.8%
7	21” 1280 x 1024	235 hours	30 hours	57.5%	19.2%	20.0%	18.8%
8	17” 1024 x 768	135 hours	13 hours	85.2%	26.2%	9.7%	24.4%
Laptop user population							
9	13” 1280 x 1024	42 hours	23 hours	61.8%	21.6%	25.1%	22.3%
10	14” 1024 x 768	98 hours	54 hours	71.1%	25.4%	22.4%	23.9%
11	14” 1400 x 1050	57 hours	57 hours	37.4%	20.3%	7.2%	15.1%
12	14” 1024 x 768	20 hours	13 hours	93.7%	12.3%	2.3%	12.2%
13	15” 1024 x 768	169 hours	154 hours	43.3%	38.9%	17.5%	24.3%
14	13” 800 x 600	132 hours	6.2 hours	71.1%	37.6%	3.0%	15.0%
15	14”1024 x 768	9 hours	6.4 hours	44.1%	21.4%	10.3%	15.3%
16	14” 1400 x 1050	69 hours	15 hours	54.6%	25.9%	18.5%	17.5%
17	14” 1024x768	10 hours	6.0 hours	77.3%	36.8%	5.0%	17.0%
Average screen usage – window of focus: 58.8%; background windows: 16.7%							

Figure 1: *Key statistics from user study. Column 3 summarizes the length of the user traces while column 4 summarizes the length of the* active *user traces after factoring out the time the user was idle. The* window of focus *columns summarize the percentage of screen area used by the active window while the* background windows *columns summarize the percentage of area used by other non-minimized windows not hidden under the active window.*

the window of focus as the window that accepts keyboard or mouse input. In determining the size of the window of focus, we include the title bar and the scroll bar and other menu bars that are embedded in the window. Columns 7 and 8 present the mean and standard deviation for the *additional* screen area used by other non-minimized windows in the system (i.e., the area not hidden under the window of focus).

Focusing on the average screen usage for the window of focus from Figure 1, we can see that our test population uses anywhere from 37% to 94% of the total screen area available to them. An additional 2% to 34% of the screen is used by other background windows that are not active, yet are not minimized. The last row of Figure 1 indicates the average usage across our user population. This average is obtained by computing the arithmetic mean of the averages of the individual users. This ensures that the average is not biased by users with larger log lengths. On average, across all our users, typically only about 59% of the entire screen area is used by the window of focus, the primary area of interest to the user. An additional 17% of the screen, on average, is used for background windows that are not minimized. In both these cases, however, the standard deviations are fairly high indicating a wide range in the screen usage values associated with each sample.

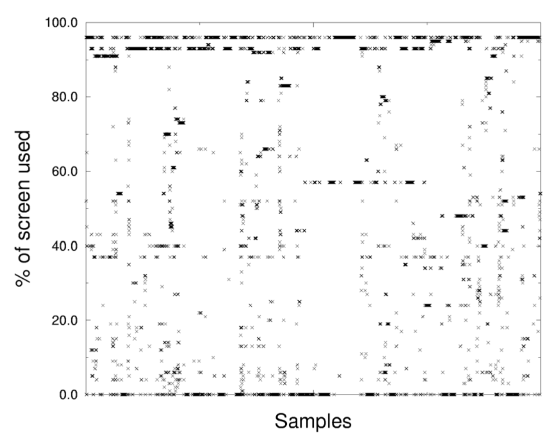

Figure 2: *Variation in the screen usage of the window of focus for typical user (User 1). Each point represents one data sample in the log.*

Screen usage distribution. To better understand the distribution of the screen usage characteristics, Figure 2 plots the variation in the screen usage of the window of focus for one sample user, over the log collection period.

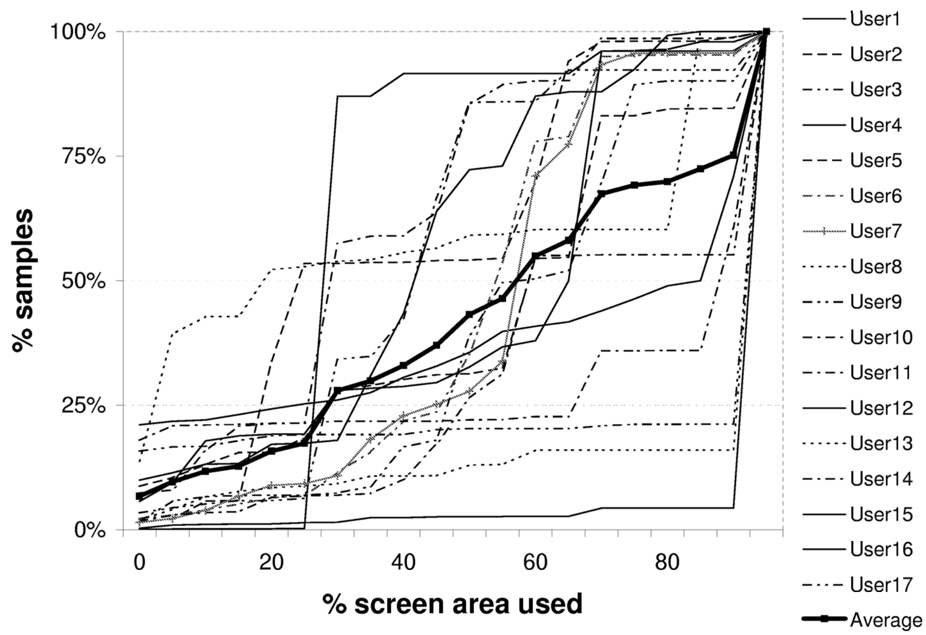

Figure 3: *Cumulative distributions of the active screen usages for the test population.*

Each point represents the average screen area associated with one data sample in the log. As can be seen from the figures, the percentage of screen usage varies significantly over the time the data was collected, all the way from near-0 to near-100% usage of the screen. Clustering of points at specific screen usage percentages can be correlated back to the continuous usage of key applications used by the user and their normal (or default) sizes.

Figure 3 presents the same data for all the users in a summarized manner. Each line in the graph represents one user from our test population and the thicker solid line represents values averaged over all the users. The X-axis represents the percentage of screen area used per sample and is divided into bins of 5 each. The Y axis represents the cumulative number of samples associated with each screen-area-percentage bin. For example, if we were to draw a vertical line from the 50% screen area point to intersect all the lines, that would give us the cumulative number of samples where each user uses less than 50% of the total available screen area. For example, focusing on *User 5*, this would mean that close to 54% of the samples use less than 50% of the screen area.

Summarizing the results in the graph, we can observe that, on average, across all our users, for almost 45% of the time, we end up using less than half the entire screen area. Some users spend more time in windows less than half the screen area (for example, *User 4* spends more than 90% of their time in windows that are typically less than 25% of the total screen area).

Screen usage corresponding to application behavior. In order to understand the relationship between the screen usage and the application behavior, we took the samples from each of our user logs and categorized them into four bins – (i) samples where the window of focus usage was between 0 and 25% of the total screen area, (ii) samples where the window of focus usage was between 25% and 50%, (iii) samples where the window of focus usage was between 50% and 75%, and (iv) samples where the window of focus usage was between 75% and 100%. For each bin, we then analyzed the key applications associated with the samples. Figure 4 summarizes our results. As before, we compute the arithmetic mean of the averages per individual users to avoid distortions due to trace lengths.

Overall, the workloads used by our user population span a range of applications representative of typical system usage. Broadly, they can be categorized into (i) access related - web browsing and e-mail (Internet Explorer, Netscape, Outlook, Pachyderm mail reader, Messenger), (ii) personal productivity and code development (Word, Emacs, Powerpoint, Excel, Visual studio, Dreamweaver, X-term, Realplayer, Image viewer, Acrobat reader, Ghostview), and (iii) system related and application control windows (File Explorer, navigation windows, taskbars, menus, status and properties messages, confirmation and password query windows).

Focusing on the windows associated with the various applications, we observe two interesting trends. First,

Active area is 0-25% *(23% of the time for typical user)*
Key applications: 20% Task bar, 15% Program Manager, 5% X-term, 60% miscellaneous windows (message composition, MSN messenger, real player, menu and message windows – properties, connection status, file downloads, alerts and reminders, volume control, printer status, find-and-replace, organizer preferences, file explorer, spell-check, wizards, status messages, file-find, password query windows, confirmation windows)
Active area is 25-50% *(22% of the time for typical user)*
Key applications: 19% X-term, 18% message composition, 6% internet explorer, 57% miscellaneous windows (mail-related windows, file explorer, emacs and notepad, MSN messenger chat windows, other status windows)
Active area is 50-75% *(28% of the time for typical user)*
Key applications: 33% Internet Explorer, 24% mail composition and reading, 43% miscellaneous windows (emacs and notepad, Image editor and photo viewer, messenger chat windows, Frontpage, Framemaker and ghostview, file explorer, Powerpoint, dreamweaver, winlogger)
Active area is 75-100% *(27% of the time for typical user)*
Key applications: 21% Outlook, 20% Internet Explorer, 7% Excel, 52% miscellaneous windows (Powerpoint, Framemaker, Acrobat reader, Word [various files], Visual C++ [various files]), Dreamweaver, Imageviewer)

Figure 4: *Understanding screen usage by application. Windows are classified based on their sizes into four bins, and for each of the four bins, the key applications dominating the samples in the bin are summarized.*

system-related status messages and query windows typically use small window sizes; in fact, these windows constitute a significant fraction of the samples associated with smaller size windows. Additionally, these windows usually display fairly low content that do not need the aggressive characteristics of the display – for example, a low-resolution display with support for a small number of colors would be adequate to obtain an equivalent user experience. Second, personal-productivity applications and development environments and web-browsing and e-mail applications typically use larger portions of the display area. The actual fraction of the screen area used appears to be highly dependent on individual user preferences for window size, fonts, etc. However, even with these large windows, characteristics of the displays such as resolution, brightness, and color are not used to their full capacity.

Screen usage corresponding to user behavior. Focusing on the individual user logs, we observe that individual user preferences and pre-set defaults tend to significantly influence the overall screen usage characteristics. For example, *User 1* who, on average uses 63% of the display area, has Internet Explorer set to use 96% of the screen area, while *User 5* who, on average uses 37% of the display area, has Internet Explorer set to use 67% of the display area. Similarly, *User 12*, who has the largest screen usage in our study, has a default mail composition window of 95% that dominates the traces. This user-specific sizing of windows appears to be particularly characteristic of web browsing, email, and editor applications. In contrast, for development applications (Visual Studio, and Dreamweaver, Powerpoint), most of our users prefer to have larger windows – possibly because of the multi-window content structure of these applications. Similarly, system-related and application-control messages typically use smaller windows irrespective of the user – mainly since the content in these windows is relatively low and in most cases the window sizes are pre-determined by the application. An illustrative examples is the case of *User 8* who maximized *all windows* as a matter of routine ("to be able to read better"). This user still consumes only 85% of the total screen area because of the smaller window sizes associated with system-related and application-control messages. Finally, while the laptop users have a slightly larger screen usage (62%) than the desktop users (56%), in general, the results are fairly similar over the laptop and desktop users.

2.3 Summary

Summarizing the results of our user study, overall, there is a significant mismatch between the properties supported in the display and the actual usage of these attributes by the users in our user study. The size of the display used exhibits the greatest mismatch – users use only about 60% of the screen area available. A large fraction of the smaller windows are typically associated with system-related and application-control windows that are independent of user preferences. User preference for smaller window sizes and font sizes can also translate into a greater use of smaller sized windows.

Similarly, there are significant mismatches between the actual screen usage and other attributes of the display such as resolution, brightness, color, refresh rate, etc. In particular, most of the smaller windows include content that could have been equivalently displayed, with no loss in visual quality, on much simpler lower-power displays (lower size, resolution, color, brightness, refresh rates, etc.) Many of the larger windows also do not use all the aggressive characteristics of the display.

Overall, these results indicate that energy-adaptive system designs that match display power consumption to the functionality required by the workload/user have significant potential to reduce the energy consumption of the display sub-system.

3 Energy-adaptive Display Sub-systems

This section studies some example energy-adaptive systems to evaluate how the potential benefits identified in the user study can be translated to energy reductions. Section 3.1 describes the hardware and software components of these designs. Section 3.2 discusses the experimental methodology used in prototyping and studying the user interfaces and the energy consumption for the different designs. Section 3.3 discusses our results and Section 3.4 summarizes.

3.1 System Design

Our designs use emerging display technologies and modified window system software to exploit the mismatches between workload/user requirements and display properties.

Hardware Support: OLED displays. To enable energy-adaptive designs, a key requirement is support at the display sub-system for variability in the display power based on the properties of the screen output. That is, it should be possible to change the energy consumption of regions of the display independent of each other. Several emerging display technologies support such variable power over different regions of the screen. This is preferable to existing technologies, where energy tradeoffs have to be made for the entire screen.

Organic Light Emitting Diode (OLED) displays [3] are a good example of this class. In OLEDs, the energy consumption of a pixel is related to its brightness and color. OLED displays are built from small organic molecules that efficiently emit light when stimulated by an electric field. More than 100 companies are developing aspects of OLED technology. Kodak, Sanyo, and Sony have shown prototypes from 5.5-inch displays to 13-inch displays at trade shows. In general, OLEDs have better image quality compared to conventional LCDs (better horizontal and vertical viewing angles, higher brightness, and faster response times) and do not need a separate backlight, resulting in lower power. As the technology matures, the biggest challenges are in overcoming yield problems and consequently reducing costs. Small OLED displays are currently available in devices like cell phones. Larger displays, for handhelds and laptops, are expected in 2004 [3].

For our prototype, we assumed a laptop system with a 15” AMOLED (Active Matrix Organic Light Emitting Diode) display. (Section 4 discusses other hardware approaches to implement the similar functionality.) The only hardware changes to a current commodity design will be to replace the conventional LCD panel, backlight, and controller with their OLED equivalents.

Software Support: *Dark Windows*. The software support for energy-adaptive displays can be implemented at one of several levels: the application level, the window system level, or the operating system level. In this paper, we focus on modifications to the windowing system. Specifically, the *“Dark-Windows”* modifications use the window of focus as an approximation of the active area of user interest to ensure that energy is spent mainly on portions of the screen that have the user's attention. For example, if a user is using an editor to take notes, her attention is typically on the screen area used by her editor, her current window of focus. Energy-aware changes to the brightness, color, or other properties of the remaining portions of the screen can result in energy benefits without impacting the user experience. Our discussions below focus on optimizations that automatically change the *luminescence* and *color* of non-active screen areas to reduce power. Section 4 discusses other properties of the background that can be changed and optimizations that focus on mismatches internal to the window of focus.

3.2 Prototyping and Evaluation Methodology

Our goals with the evaluation were two-fold. First, we wanted to understand the intrusiveness of the energy-adaptive user interfaces and the complexity and overhead associated with implementing the dark windows optimizations. Second, we also wanted to quantify the energy benefits in the context of one particular technology, namely OLED displays, and understand the design

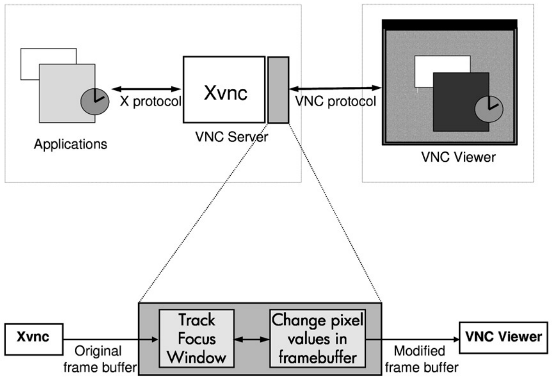

Figure 5: *Methodology used to prototype the user interface.*

Interface Name	Modification
HalfDimmed	Areas outside the window of focus are dimmed by 50%.
FullyDimmed	Areas outside the window of focus are fully dimmed (turned off).
GrayScale	Areas outside the window of focus are changed to gray, by setting red, green, and blue values to the average of the three.
GreenScale	Areas outside the window of focus are changed to green (lowest power color for OLEDs). The green value is set to the average of the three colors, and the red and blue values are zeroed. This also dims the screen by 67%.

Figure 6: *Summary of the* dark windows *optimizations considered. The window of focus is left untouched, while the background areas are modified in either brightness or color to save energy.*

tradeoffs between the various optimizations.

Prototyping the User Interface We implemented several prototypes in order to understand the impact of the dark-windows changes on the user interface. Given the difficulties with modifying the Microsoft windowing environments, we implemented our prototypes on the X Window System under Linux.

To enable a clean implementation of the changes, we used the open-source VNC (Virtual Network Computing) server [6]. VNC provides a virtual representation of the display hardware, i.e. a remote framebuffer, and allows one to run a server on a machine while viewing its display remotely from anywhere on the internet. While the remote viewing property was not of interest to us, by using VNC, we were able to gain two advantages. First, since it is a virtual frame buffer, we were able to easily manipulate the pixel values. Secondly, since VNC is linked with the X-Windows server, we had access to the window server data structures, such as the window of focus.

Figure 5 shows an overview of our changes. We modified the X-VNC server with two additional functions – to track the window of focus and other objects in the display area, and to change the values of the pixels in the frame buffer based on our dark-windows algorithms. To reduce the performance overhead from our software changes, our modifications closely track the incremental change update mechanism of the VNC protocol.[1] Each framebuffer update represents a region that has changed, and is sent to the viewer as required to keep the display up-to-date. Before each frame buffer update is sent to the viewer, we perform two functions. First, we query the X-server and record the window of focus – the window that keyboard events are directed to – including its origin, width, and height. We then divide the update region into two groups – those that are in the window of focus and those that are outside it. The pixels within the window of focus are sent unmodified, while the rest have the desired transformation applied to them. Overall, the performance overhead of our implementation was negligible with performance acceptable to users.

We experimented with several transformations, described in Figure 6. These included transformations that modified the brightness of the background (non-window-of-focus) regions (*HalfDimmed and FullyDimmed*[2]) as well as transformations that modified the color of the background regions (*GrayScale and GreenScale*). (Screenshots of these interfaces are shown in Figure 8 in the next section.) In all these cases, we also implemented a user-defined keyboard short-cut to be able to turn off the modifications if desired.

Quantifying energy benefits for typical users. Though we used VNC for implementation convenience, we still wanted to focus our benefits study on the commodity Microsoft Windows based systems used in our user study. We therefore constructed a synthetic trace that modeled the average behavior observed in our user study and re-

[1] We originally chose a simpler implementation approach where we simply traversed the finally-generated framebuffer to apply our modification heuristic on each pixel. However, this led to an inordinately large number of computations and caused a perceptible slowdown in the system. We chose to instead use the rectangular update mechanism to obviate this problem.

[2] Our optimization allows the brightness to be dimmed by any user-defined fraction; however, in this paper, for space reasons, we consider only two representative data points - half-dimmed and fully-dimmed.

Avg. window size Avg. background windows' size	59% 17%
0-25% screen usage (taskbar, xterms, miscellaneous) 25-50% screen usage (xterms, editors, mail readers, miscellaneous) 50-75% screen usage (web browsers, mail readers, editors, miscellaneous) 75-100% screen usage (web browsers, mail readers, miscellaneous)	23% 22% 28% 27%
Default screen background Default window color Default foreground font	Teal White Black

Figure 7: *Summary of properties of synthetic trace. The trace models the average behavior exhibited in the user study and was created to match the data from Figure 4.*

played it on our prototype systems. The trace runs for about 1000 seconds and uses a set of applications similar in nature to those used in the user study. The script for the trace models the initiation and termination of applications and varies the window sizes and the duration of time windows were open to obtain statistics consistent with the user study. While the users in our test population used a variety of backgrounds and window colors, a majority of them used the default windows settings. Our synthetic trace therefore predominantly uses those colors. To bracket the impact of specific user choices in this respect, we also performed experiments where the screen background and window colors are varied. Figure 7 summarizes the window properties of the synthetic trace and a comparison of the data with that in Figure 4 shows the close correlation between the behavior of the trace and the user study.

Given the current unavailability of 15" OLED displays (aside from rare prototypes), we decided to use a software power model representative of future systems. The model computes the display sub-system power as the sum of the controller power, the driver power, and the display panel power. The controller and driver power are modeled as constant values irrespective of what is displayed on the screen. The panel power represents the pixel array power and is the total of the power consumed by each pixel in the panel array. In turn, the power consumed by each pixel is proportional to the brightness of the red, green, and blue components.

$$DisplayPower = P_{controller} + P_{driver} + PanelPower$$
$$PanelPower = PixelArrayPower$$
$$= \Sigma P_{red} \times pixel_R + P_{green} \times pixel_G + P_{blue} \times pixel_B$$

The values of the parameters of the power model were chosen to represent likely future OLED displays, and were validated by discussions with researchers working in the area and by inspecting data sheets of current displays. For the 15" display we model, we set $P_{controller}$ to 0.5W and P_{driver} to 1W. The $PanelPower$ can be a maximum of 8.5W. Interestingly, with current OLED technologies, pixels consume different amounts of power based upon the particular color, with green typically being the lowest. For a 1024x768 resolution display with pixel values ranging from 0 (off) to 1 (fully on), the values for P_{red}, P_{green}, and P_{blue} are 4.3μW, 2.2μW, and 4.3μW.

3.3 User Experience and Energy Benefits

User Interface Intrusiveness. Figure 8 summarizes the various interfaces. (Color pictures are available in the electronic version of the paper.) The picture on the left represents the original configuration, while the two pictures on the top right represent dark-window configurations with modified background brightness, *HalfDimmed* and *FullyDimmed*. The two pictures on the bottom right represent the dark-window configurations with modified background colors, *GrayScale* and *GreenScale*. The performance overhead of the *dark windows* implementation was small and there was no significant difference in the response time of the user interfaces.

To determine the difference in the quality of the various interfaces, we informally surveyed 9 users. We asked each user two questions. First, *without describing any battery life issues*, we asked users to choose their best interface. Second, we showed the users the battery life advantages from the various dark-windows modifications and *then* asked users to choose their best interface again, with the assumption that they were in a situation that required long battery life, such as a meeting or an airplane trip. Most of the users indicated a willingness to use dark windows optimizations to tradeoff longer battery life for a different user interface. Of the 9 users we queried, 4 of them preferred *GreenScale*, 3 preferred *HalfDimmed*, and 2 preferred *FullyDimmed*. Most of the users expressed a desire to be able to see the contents of the background, even at the expense of some energy. Interestingly, even without an awareness of the energy benefits, four of the users chose some of the dark windows interfaces (*GrayScale* and *HalfDimmed*) as their preferred interface. Note that even without the software *dark windows'* optimizations, the base energy-adaptive display hardware still provides the ability to have lower power consumption based on what is displayed on the screen. Compared to LCDs, this can achieve some power

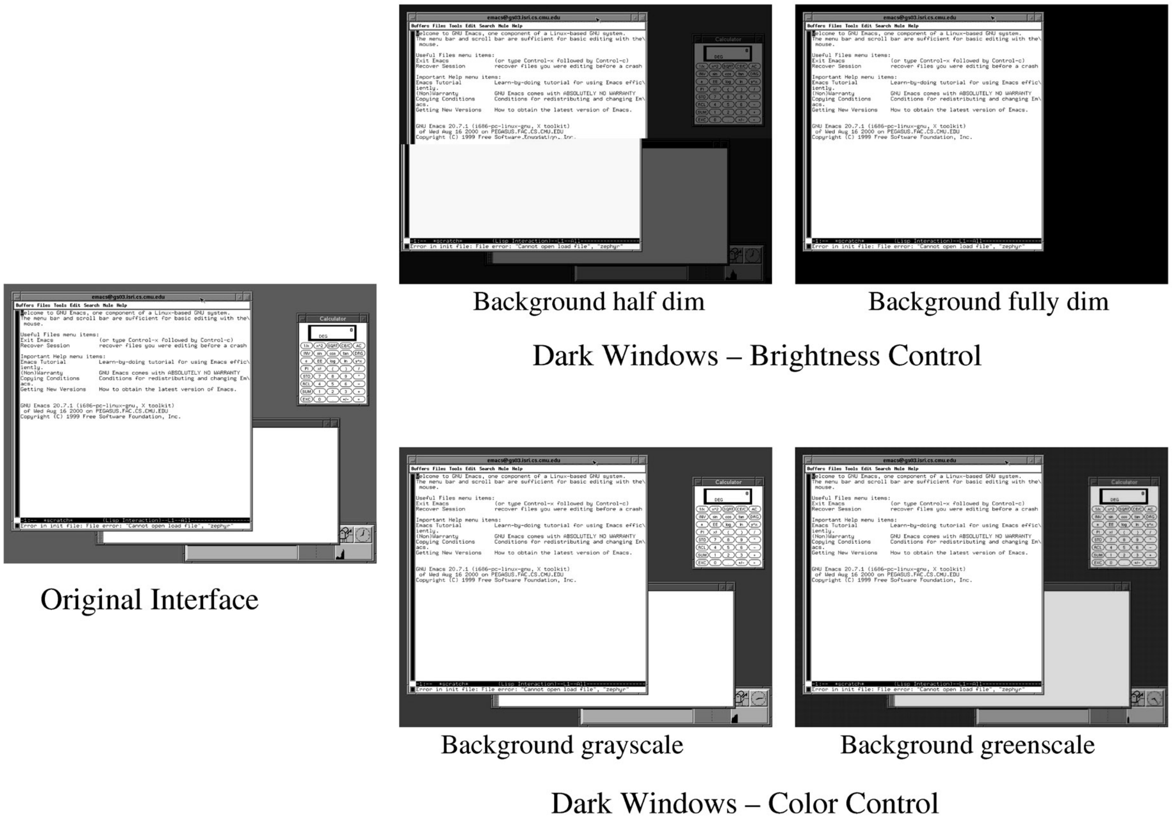

Figure 8: *Screen shots of the user interfaces with* dark windows *optimizations.*

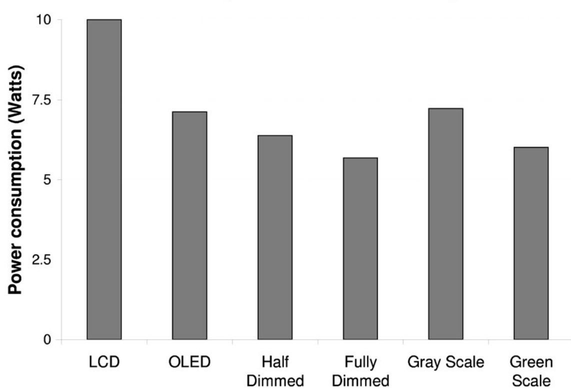

Figure 9: *Power benefits from energy-adaptive display designs.*

benefits with *no changes in the user interfaces.*

Our user interfaces could be improved in several ways (we discuss these in Section 4). However, our goal was not to perform an exhaustive exploration of the design space for user interfaces, but instead to establish that energy-adaptive display sub-systems can provide energy benefits with interfaces that users are likely to find acceptable, particularly in return for longer battery lifetimes.

Energy Measurements. Each of our interfaces modifies the pixels in a different manner, resulting in differing energy use. We measure the power consumed by our synthetic trace for each configuration on our modified VNC server. We study a default LCD configuration, a default OLED configuration, and four OLED configurations with the four dark windows optimizations discussed earlier. We use the OLED power model discussed earlier to compute the power for the five OLED configurations. The LCD power model is based on characteristics noted by Choi et al. [1]. For our synthetic trace, this leads to very little variance (less than 1%).

Figure 9 summarizes the results. Compared to the constant 10W power consumption of the LCD, the hardware support for energy-adaptive displays in the base *OLED* system achieves a 25% reduction in power. Most of these benefits are due to power reductions for the teal background color. (With our OLED display, a teal pixel - RGB: [0,131,131] consumes only 30% of the maximum power a pixel can consume - RGB: [255,255,255].) However, there are also some benefits from power reduction specific to the content of the windows (e.g., web browser). The software *dark windows* changes can provide additional power reductions. The *Fully-Dimmed* optimization provides an additional 20% over

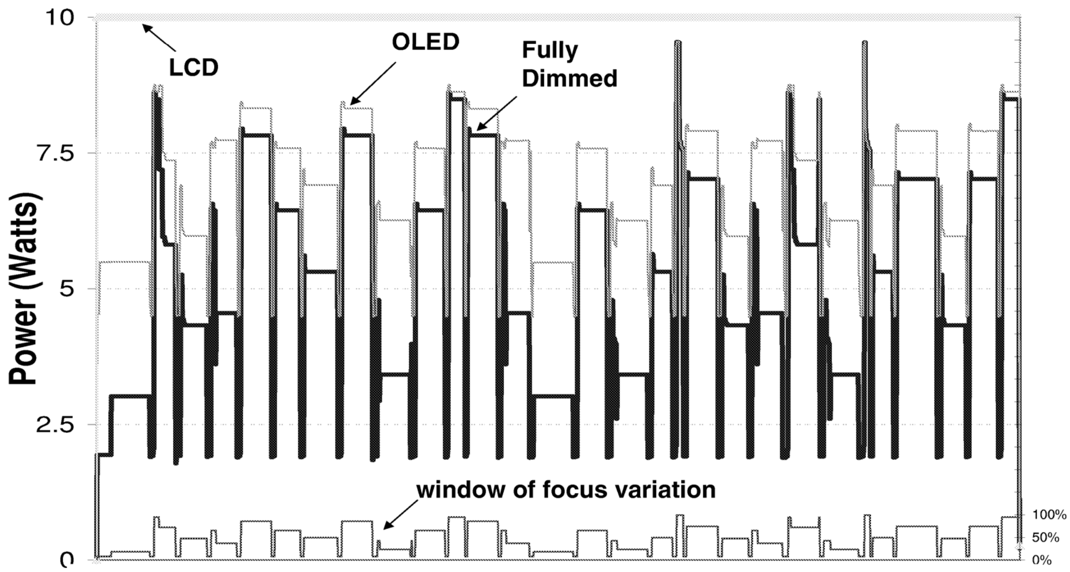

Figure 10: *Power variation over time for non-adaptive and energy-adaptive displays.*

the *OLED* case, and a total of 43% over the *LCD* case. These benefits are from dimming of the background windows (predominantly white) and the screen background (teal). *HalfDimmed* provides half these benefits while still allowing some of the information to be visible. The *GreenScale* optimization provides 40% energy benefits compared to *LCD* (and a 15% benefit compared to *OLED*). This optimization computes the average of the R, G, B values and includes it as the new value for the green pixel. (Recall that the green pixel takes the lowest energy compared to the other pixels.) The combination of using the most energy-efficient color and a reduction in brightness by 67% leads to the energy benefits. In contrast the *GrayScale* optimization averages the R, G, B values and uses this average as the new values for the R, G, and B pixels, turning them gray. However, our results indicate a 1% increase in energy with this configuration compared to *OLED* (though still 28% better than the non-adaptive *LCD* case). This is because converting our default background color (RGB: [0,131,131]) to gray scale ends up moving more bits to the higher-power R and B pixels (gray-RGB: [87,87,87]). An alternate background color such as pure blue or red may have benefited from having a grayscale background optimization.

Figure 10 shows the power variation over time for three cases - *LCD*, *OLED*, and *FullyDimmed*. The *LCD* case represents an non-energy-adaptive display and shows relatively constant power consumption, invariant to the size and content of what is displayed on the screen. The *OLED* case shows the benefits that can be obtained from using an energy-adaptive display technology. Since the benefits are mainly from the background screen color, the profile of the curve follows the profile of the percentage of the screen devoted to the background (after removing the window of focus and other background windows). The *FullyDimmed* shows the benefits from both hardware and software changes for energy adaptivity. The profile of this curve follows the profile of the window of focus curve.

As is evident from the above discussion, the benefits from energy-adaptive display designs are highly dependent on the choices of the background color and the window color. Our default trace modeled the windows default since that most closely represented a majority of the users in our test population. To better bracket the impact of other choices of background and window colors, Figure 11 shows the energy consumption of the various configurations for the extreme cases of pure white and pure black backgrounds and window colors. With black backgrounds and black windows, the base *OLED* design achieves most of the benefits from energy adaptivity – close to 80% reductions compared to *LCD*. The power consumed is mainly due to the controller and driver and minor elements of windows such as title bars, etc. Additional software optimizations get minimal improvements on top of this. In contrast, with white backgrounds and white windows, the base *OLED* configuration gets close to no improvements. The *FullyDimmed* system gets the maximum benefits (35%) by reducing the power spent on the background. The other two cases show intermediate points where both the hardware and software optimizations obtain good benefits. These results indicate

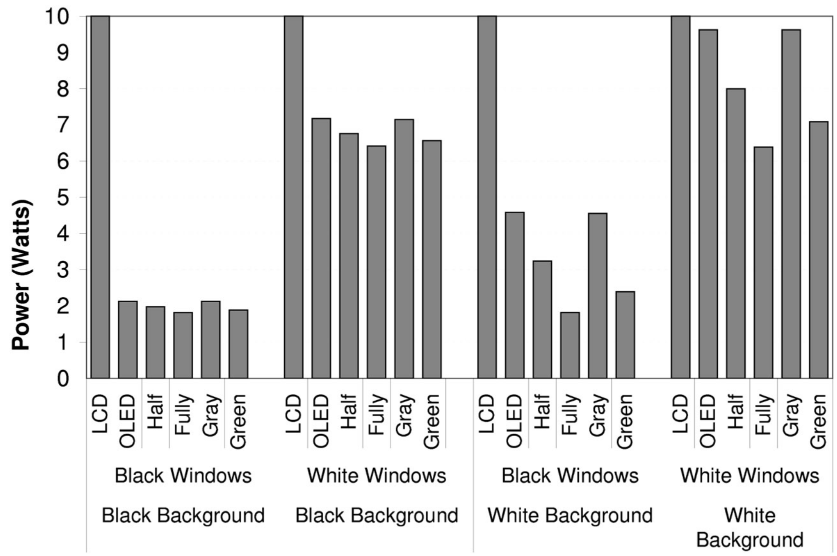

Figure 11: *Sensitivity of benefits from energy-adaptive designs to background and window colors.*

that different designs may be better for different usage scenarios. It is important for designers to understand typical user behavior before designing energy adaptive display sub systems. In some cases, it might be adequate to just include support for adaptivity at the display hardware level and choose color and size defaults with power in mind. In general, however, for systems that allow a great flexibility in user choices for background and window colors, it might be preferable to implement software optimizations in addition to the base hardware support for adaptivity.

3.4 Summary

Overall, our results indicate significant energy benefits from energy-adaptive display designs. The base OLED design achieves 30% reduction in energy compared to a base LCD non-energy-adaptive design – with no change in the user interface. The other dark windows optimizations change the user interface in different ways by dimming or changing the color of the background screen area and achieve significant, but user-specific, energy benefits. In particular, the choice of the background and window color can have a key impact on the power reductions. For the default windows background used by our users in the user study, the best optimizations, *Fully-Dimmed* and *GreenScale* achieve close to 40% energy benefits over the base non energy-adaptive design. An informal user study indicates reasonable acceptance of these user interfaces, particularly in the context of an awareness of the energy benefits from trading off the interface for longer battery lives.

4 Discussion

The configurations discussed in the previous section illustrate some example energy-adaptive designs. In this section, we discuss other possible energy-adaptive options for display design.

Display energy adaptivity in hardware. Rather than using OLEDs, other display technologies that enable energy-adaptivity can be used. These include other opto-electronic and emissive displays (Field-Emission Displays [FEDs] and even conventional Cathode-Ray-Tube [CRT] displays) as well as hybrid technologies like LCD displays with OLED backlights. With display technologies like LCDs that do not support energy variability, designs can still integrate a multi-modal "hierarchy of displays" configuration. For example, a mobile device could have two displays – one higher quality (high resolution, color, high refresh rate, larger size) and consequently higher energy use, and another lower quality and lower energy. While the adaptivity in this design is more coarse-grained than with the OLED systems considered in this paper, the insights from our study are still likely to be valid.

Display energy adaptivity in software. Rather than using the window of focus, other indications of user activity could be used. For example, the area around the

Figure 12: *Other energy-adaptive designs. The picture on the left illustrates a base system. The system in the middle uses two display panels to provide coarse-grain adaptivity for power. The right picture shows how display power for low-content messages can be reduced by using simpler visual or non-visual cues.*

cursor could be kept bright, while the rest of the screen is dimmed. Another design dimension is to include support for user-controlled dimming areas. For example, one possible user interface could include a *"sticky-lamp"* placed by the user to light up a specific portion of the screen. Much as we do in the physical world, the user could use multiple "sticky-lamps" to light up the work-area. An alternate implementation could include a "head-light" on the mouse pointer. The users could then point the light over regions that they are interested in and move down as they read along.

Still other dimming interfaces could be application specific. For instance, in a programming environment, there may be a concept of the current procedure and related variables. Portions of the screen related to these could be made bright – for example, all uses of the variables and all calls to the procedure. In an email application, perhaps only the current message needs to be bright. In a word processor, the line of text being edited could be bright, the surrounding couple of lines lightly dimmed, and the rest of the document greatly dimmed. Similarly, in the case of applications like Microsoft Powerpoint which use frames within an application, the notion of a frame-of-focus can be defined, similar to the window of focus. As another dimension to these user interfaces, these can all be made time-based. For example, areas of the screen that have recently changed could be bright, fading to a dim value as time progresses. When inactivity is detected, an email application could dim its screen area until new mail arrives.

Other user interfaces can be developed by combining the interfaces above. Additionally, other sorts of display mismatches could be exploited. In this paper, we have focused on identifying the mismatch between the total area of the display and the area of interest to the user. Other properties of the display, such as resolution and refresh rate could be exploited as well.

Support for output modes beyond displays. The notion of having multiple displays can be taken one step further to match output content to notification mechanisms beyond displays. For example, an email notification that says "You have mail" on the display could be replaced by an LED that blinks on the arrival of email or other similar notification mechanism such as speech output, vibrations, etc.(Figure 12). As discussed in Section 2, a lot of the smaller windows are typically low-content windows which may be amenable to other forms of non-visual communication. This combined with the large design space for alternatives for energy adaptiveness indicate the potential for an interesting future area of research – *energy-aware user interfaces*.

5 Related Work

Concurrent with our work, there are two other studies that have looked at adapting the output of the displays from an energy perspective. Choi et al. [1] perform a detailed characterization of the power consumption of the display sub-system of a handheld device (including the power of the panel, the panel bus, the backlight, the frame buffer and the data and address bus driving the frame buffer). They propose three optimizations that (1) vary the refresh rate to exploit the after-image caused by the time constant of the storage capacitors, (2) vary the color depth to be able to reduce the memory requirements and hence the memory power, and (3) vary the backlight luminance with a corresponding compensation of the brightness and contrast. Kamijoh et al [4] discuss the energy trade-offs in the IBM wristwatch computer. While they focus mainly on the hardware-level tradeoffs and kernel optimizations to use the various standby and idle configurations of hardware, they also discuss the implication of controlling the number of pixels turned on or off on the energy as well as reducing the duty factor of the display to control the brightness of the entire screen (e.g., at night). Additionally, while not focusing mainly on the display component of the power, Flinn and Satyanarayan [2] also evaluated in detail the energy benefit of reduced computation with lowered fidelity of images for web browsing and video playing. Their study also proposed a method called "zoned backlighting," to enable energy benefits in the display subsystem. While zoned backlighting could allow independent control of illumination level for different regions of the screen, no existing display supports such zoned backlighting yet.

Similar to these studies, our work also explores the possibility of adapting the output of the displays from an energy perspective. However, in contrast to the studies, we primarily focus on the *content* and *intent* of the output screen display. In particular, our work is the first to perform a detailed characterization of display usage patterns to identify and understand the common mismatches between workload/user needs and current display properties. Based on these insights, our work also explores several new user interfaces and hardware designs that allow energy-adaptive control on the portions of the screen that are not of immediate relevance to the user, while continuing to provide similar functionality on portions of the screen of relevance to the users.

Finally, this work focuses mainly on the display panel power. Complementary to our work, other studies have also focused on the power consumed by the display controller and driver (e.g., [8]).

6 Conclusions

As mobile systems, applications, and services become more pervasive, it becomes ever more important to identify design strategies to lower energy consumption and increase battery lifetimes. This paper focuses on the display sub-system and motivates and evaluates energy-adaptive system designs for future mobile environments.

The first part of our study performs a detailed analysis of display usage traces from 17 users, representing a few hundreds of hours of active usage. To the best of our knowledge, this study is the first to identify, quantify, and analyze potential mismatch opportunities in workload/user needs and current display properties. We find that on average, the window of focus – a good indication of the area of interest to the user – uses only about 60% of the total screen area. Additionally, in many cases, the screen usage is associated with content that could have been equivalently displayed, with no loss in visual quality, on much simpler lower power displays. Our analysis of the user traces indicates that many of these mismatches could be traced back to the typical content of the windows as opposed to specific user preferences.

Based on these insights, the second part of the study proposes energy-adaptive display systems that match energy use to the functionality required by the workload/user to obtain significant energy savings. To support the energy adaptivity in hardware, our designs leverage emerging display technologies like OLEDs that provide variability in the energy consumed based on the properties of the pixels. For energy adaptivity at the software level, we propose several *dark windows* optimizations that allow the windowing environment to change the brightness and color of portions of the screen that are not of interest to the user. We develop prototypes of the user interfaces and model the power benefits of such energy adaptive designs. Our results indicate significant energy reductions with acceptable tradeoffs in the user interface.

In addition to the designs that we evaluate, we also discuss other points in the design space including several other alternative hardware and software interfaces for energy-adaptivity. These designs are likely to achieve even further energy benefits. Our work leads to several interesting challenges including the design of energy-aware user interfaces as well as more intelligent heuristics to automatically identify mismatches between the workload/user intent and the display sub-system functionality.

As mobile systems continue to develop, the contribution of the display power to the total mobile system power is only likely to increase, particularly in larger mobile devices. Similarly, as mobile workloads continue to develop, mismatches between the display system required for the most aggressive application and the needs of the common-case workloads are only likely to get exacerbated. The combination of these trends indicate a huge potential for system designs that flexibly adapt their energy consumption based on workloads needs. We believe that energy-adaptive display designs like the ones that we discuss in this paper are an extremely promising approach to exploit this potential and that they will become an important part of future mobile system designs.

7 Acknowledgements

This work greatly benefited from discussions with Ramakrishna Anne, Bill Hamburgen, John Rudin, and Ken Nicholas. We would also like to thank Norm Jouppi, Dan Sieworek, Amin Vahdat, and the anonymous reviewers for their comments on the paper. We would like to thank John Sontag and Jim Rowson for their support of this work. Finally, we would like to thank the users who participated in our user study. Lu (Annie) Luo's work in this paper was primarily supported by an HP internship. Her research is also funded in part by NSF under award number CCR-0205266 and by an IBM PhD Fellowship.

References

[1] Inseok Choi, Hojun Shim, and Naehyuck Chang. Low-Power Color TFT LCD Display for Hand-Held Embedded Systems. In *Proceedings of the International Symposium on Low Power Electronics and Devices*, pages 112–117, August 2002.

[2] Jason Flinn and M. Satyanarayanan. Energy-Aware Adaptation for Mobile Applications. In *Proceedings of the 17th ACM Symposium on Operating Systems Principles*, December 1999.

[3] Stanford Resources Inc., editor. *Organic Light-Emitting Diode Displays: Annual Display Industry Report*. Second edition, 2001.

[4] N. Kamijoh, T. Inoue, C. M. Olsen, M. T. Raghunath, and C. Narayanaswami. Energy Trade-offs in the IBM Wristwatch Computer. In *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC-01)*, pages 133–140, October 2001.

[5] Trevor Pering, Tom Burd, and Robert Broderson. The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms. In *Proceedings of the International Symposium on Low Power Electronics and Design*, August 1998.

[6] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual Network Computing. In *IEEE Internet Computing, Vol.2 No.1*, pages 33–38, Jan/Feb 1998.

[7] Sanjay Udani and Jonathan Smith. The Power Broker: Intelligent Power Management for Mobile Computers. Technical Report MS-CIS-96-12, Distributed Systems Laboratory, Department of Computer Information Science, University of Pennslyvania, 1996.

[8] S. Xiong, W. Xie, Y. Zhao, J. Wang, E. Liu, and C. Wu. A Simple and Flexible Driver for OLED. In *Proceedings of the Asian Symposium on Information Display (ASID99)*, pages 147–150, 1999.