Evaluation of Progressive Image Loading Schemes

Chris Harrison       Anind K. Dey       Scott E. Hudson
Human-Computer Interaction Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
{chris.harrison, anind, scott.hudson}@cs.cmu.edu

ABSTRACT
Although network bandwidth has increased dramatically, high-resolution images often take several seconds to load, and considerably longer on mobile devices over wireless connections. Progressive image loading techniques allow for some visual content to be displayed prior to the whole file being downloaded. In this note, we present an empirical evaluation of popular progressive image loading methods, and derive one novel technique from our findings. Results suggest a spiral variation of bilinear interlacing can yield an improvement in content recognition time.

Author Keywords
Progressive image loading, downloading, web browsing.

ACM Classification Keywords
H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General Terms: Human Factors

INTRODUCTION
Images are a fundamental component of content on the World Wide Web. They are used in a multitude of ways, including design (e.g., logos), data visualization (e.g., graphs), and as a form of expression (e.g., art, photographs). However, images files are typically several orders of magnitude larger than text files, causing their transmission times over networks to be substantially longer.

This may not be a problem for those with high-speed, broadband connections, where images are often loaded in under a second. However, for users on slower connections, particularly prevalent in developing nations, images may take several seconds to more than a minute to fully load. This period is frustrating to users, and may preempt work until complete [5,6]. Mobile devices, such as cell phones, are also affected, as wireless speeds are generally much slower than that of hard-wired connections. Even on state-of-the-art mobile hardware, such as Apple’s recently released iPhone 3GS, a 5MB image takes roughly 13 seconds to download at a generous 3Mbps (UTMS turbo HSDPA has a theoretical maximum transfer speed of 7.2Mbps, but real world performance is considerably lower due to latency, congestion, packet loss, and overhead). On EDGE networks, still prevalent in the United States, the same image would take several minutes to fully download.

It should be noted that improvements in wired and wireless bandwidth will alleviate these effects over time. However, there are three forces competing against this progress. Foremost, multitouch interfaces found in the latest mobile devices are enabling users to seamlessly access and navigate high resolution content on small screens. This modality is certain to grow, and even five-second load times will not be acceptable to users, especially if once downloaded, the content was not of interest. Secondly, there has been a tremendous surge in the volume of photographic content online (due to, e.g., Flickr, Facebook). Lastly, the resolution of consumer digital cameras has increased from roughly 1 megapixel to more than 20 in just the past decade [1] (although this trend looks to be slowing).

IMAGE LOADING
Image loading in web browsers typically manifests itself in two ways. One behavior is for the browser to withhold the image until it is fully downloaded. Typically, an empty space is displayed in the interim period. The second method that web browsers employ is to progressively render the image as data is received.

The most frequently encountered progressive loading scheme renders an image’s pixels from top to bottom. The latter has the benefit of showing some visual content to the user as the image loads, and otherwise assures the user the image is indeed downloading. More importantly, however, is that it allows for quicker recognition - imperative for efficient traversal of web content.

Unfortunately, this top-to-bottom loading method may not render the pixels necessary for recognition until more than half of the image is loaded, wasting considerable time, bandwidth and energy [5]. Perhaps a simple change, such as left-to-right or bottom-up, would prove more effective. Also worth considering are methods that do not load pixels in a predefined order. For example, an importance metric could be calculated for pixels or arbitrary regions in an image. When the image is being transmitted to a user, the “most important” pixels could be sent first.

In general, it appears current image loading schemes are not aligned with how people create, capture, and view still content. This note provides what may be the first quantitative assessment of how different loading schemes perform from a human-centric perspective.
To assess the effectiveness of alternative static order loading schemes, a multi-stage experiment was devised. Roughly 4200 images were gathered randomly from the online photo sharing website Flickr.com. These had to have a resolution of no less than 1024x768 and a dimension ratio of approximately 4:3 (± 15% tolerance). Once downloaded, images were scaled down and/or cropped (equal removal from sides in one dimension) to a standard resolution of 1024x768 to facilitate viewing in the experiment interface.

We recruited 10 participants (4 female) with a mean age of 38.7. Each participant was shown 150 unique images from the larger image set. An additional 20 images were included that were seen by all participants, for a total of 170 images. These common images were included to assess the level of agreement between participants. The task required participants to select the most important (i.e., significant or visually salient) objects in each picture. They used a mouse to click-and-drag rectangular regions over desired portions of an image. Several or no regions could be selected.

The 1500 images processed by the 10 participants yielded 1924 rectangular ROIs. Participants tended to mark the twenty common images in a systematic fashion – suggesting people annotate images in similar ways. The average size of a selected region was 308x295 pixels, or approximately 11.6% of the image. The data was also used to calculate the probability of being selected as important for every pixel. Surprisingly, there were some pixels that had a 50% or greater chance of being selected as important, although these made up only 0.9% of the image. The average pixel had a roughly one in five chance of being selected as important (mean=19.2%, SD=13.9%). Figure 1 illustrates the clearly bi-normal probabilistic distribution.

**Figure 1.** The probability distribution of how likely pixels will be selected as important. A linear green to red scale is used, which represents a range of probabilities from 0% (never) to 52% (maximum likelihood). Right: Pixels with importance probabilities greater than or equal to one standard deviation above the mean are shown on a grey background (>33.1%).

---

**Table 1. Descriptions of the nine static-order loading methods.**

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top-to-Bottom</td>
<td>Rows of pixels from top to bottom.</td>
</tr>
<tr>
<td>Left-to-Right</td>
<td>Columns of pixels from left to right.</td>
</tr>
<tr>
<td>Outside-In Spiral</td>
<td>Spiral from the outside edge to the center. Opposite of inside-out spiral.</td>
</tr>
<tr>
<td>Inside-Out Spiral</td>
<td>Spiral from the center to the outside edge. Opposite of outside-in spiral.</td>
</tr>
<tr>
<td>Center-Out Horizontal</td>
<td>Start from the center and load columns of pixels, alternating sides.</td>
</tr>
<tr>
<td>Center-Out Vertical</td>
<td>Start from the center and load rows of pixels, alternating between top and bottom.</td>
</tr>
<tr>
<td>Outside-In Horizontal</td>
<td>Start from the left and right edges, and load columns of pixels, alternating sides.</td>
</tr>
<tr>
<td>Outside-In Vertical</td>
<td>Start from the top and bottom edges, and load rows of pixels, alternating sides.</td>
</tr>
<tr>
<td>Random</td>
<td>Load pixels in a pre-specified pseudo-random order. Looks like “static” that gets progressively denser.</td>
</tr>
</tbody>
</table>
In order to evaluate how efficient different static-order loading schemes were at rendering important regions of images, it was necessary to identify where the most important pixels were likely to be. In response, important pixels were defined to be those with importance probabilities greater than or equal to one standard deviation above the mean (>33.1% likelihood of importance). These pixels occupied a roughly elliptical region near the center of the image (Figure 1, right); a result supported by findings in [9]. This data was then used to run an image loading simulation. In addition to top-to-bottom, eight alternative static-order loading schemes were tested. Table 1 and Figure 2 describe these methods. Over the course of the simulating loading process, the number of important pixels that had loaded (i.e., rendered on screen) was tracked. The results are presented in Figure 3.

Inside-out spiral, as one would expect given the distribution of important pixels, does very well, loading all of the important pixels in about 35% of the total loading time. Its inverse, Outside-in spiral is the worst performing, not even beginning to load in the important pixels until almost 70% of the image is downloaded. Interestingly, left-to-right is about 4% slower than top-to-bottom. However, all of the functions that start from the center and work their way outwards handily outperform top-to-bottom.

**EVALUATION**

Our previous simulation provided initial insights into the performance characteristics of static-order loading schemes. However, it only served as a crude proxy to true human recognition. Additionally, static-order loading schemes, although most common, do not represent state-of-the-art loading techniques. In response, we devised a second study that built on the previous results and sought to more directly measure human recognition time.

**Conditions**

We selected six methods for user evaluation. The first was top-to-bottom, included as a baseline. Inside-out spiral, the best performing static-order loading from the previous experiment, was also included (as a performance upper bound for static-order schemes, referred to as spiral henceforth). Additionally, three methods were included that represent advanced loading features available in the GIF, JPG, PNG and JPEG2000 image formats [2,5,11]. Finally, the remarkable difference in performance between top-to-bottom and spiral loading, led us to create a new variant of bilinear interlacing that loads in a spiral fashion. Table 2 and Figure 4 describe these six methods.

**Setup and Procedure**

Sixty new images were randomly selected from the pool of 4200 collected for the static-order experiment. These images were shown to three, independent coders, who were asked to name up to three salient features contained in each image (e.g., “boat”, “sunset”, “cup”, “party”). They were also asked to provide up to five alternative words for each feature. For example, other words for “boy” might be “child” or “kid.” Additionally, the coders annotated the images with rectangular regions (click-and-drag interface), one for each salient feature.

Data from the three coders was combined by taking the intersection of the provided terms. If there was an agreement, a union of alternative words was taken (e.g., if a coder had “boy, baby, child” and another had “kid, boy, infant”, the feature labels would be “boy, baby, child, kid, infant”). This formulation offered the highest number of possible terms for the features that all three coders agreed were salient. The rectangular areas selected by the coders (each bound to one feature) were used to simulate JPEG2000’s rectangular ROI functionality. This simultaneously acted as state-of-the-art automatic ROI extraction.

Coders were also asked to rate the features in each image as “not obvious,” “somewhat obvious,” and “very obvious.” Four images were discarded: one because terms provided by the coders had no overlap, and three more because coders had marked them as “not obvious.” This left 56 images.

Seven participants (6 female) with a mean age of 36.0 were recruited. Each participant was shown all 56 images, although how the images were paired with loading schemes was unique for each participant, as was the presentation...
sequence (to control for order effects). However, this was arranged in such a way that participants saw every loading method exactly seven times, providing balanced numbers. Participants were told they would see a series of images load onto the screen in different ways. The objective was to name the items in these pictures as quickly as possible—a fun task that kept participants engaged. They were encouraged to speak out loud, using as many names for items as possible. When a participant correctly guessed one of the noted salient features, the experimenter would hit a button on a keyboard, which would record the elapsed time. This process would continue until participants had guessed all of the salient features (min=1, max=3), at which point the experimenter would say, “done”.

The data rate was set to simulate an average EDGE connection (130Mbps) downloading a 4.2MB image. This slower, although not totally unrealistic download speed, was purposely used to extend the experience, providing a clearer result of when users recognized content, as opposed to time spent iterating through alternative terms.

Results
Figure 5 summarizes the results from the user study. Top-to-bottom was clearly the worst performer, with participants taking on average 107 seconds to recognize image content. A Student’s t-test revealed that all other schemes were significantly faster than top-to-bottom (p<.001). Spiral, rectangular regions, and linear interlace all performed roughly the same, taking approximately 30 seconds before all the content was correctly identified. Finally, the best performing loading schemes were those based on bilinear interlacing. Participants required only 12 seconds on average to identify content using these methods, and were significantly faster than with any of the other methods (p<.001), although there was no significant difference between them.

Discussion
The most surprising result was that spiral performed as well as linear interlacing (GIF) and rectangular regions (JPEG2000). The latter is of particular interest as these regions were human annotated. The fact a naive method can perform just as well on average suggests expensive human annotation may offer only limited gain. Furthermore, it is clear ROI-based techniques could take advantage of spiral loading’s superior performance (for filling the regions).

The performance recognition of the two bilinear methods (top-to-bottom and spiral) was so superior—an order of magnitude better than top-to-bottom— that results tended to be overwhelmed by variations in participants’ reaction time (and ordering of guesses). This prevents a more detailed comparison of the two methods. However, we suspect they mirror the performance difference seen in static-order experiments, as they share the same loading mechanics.

CONCLUSION
We presented results from two image-loading experiments. Results offer quantitative comparisons between common loading methods. They also suggest a simple spiral variant of bilinear interlacing can improve the user experience.

ACKNOWLEDGMENTS
This work was supported in part by grants IIS-0713509 and IIS-0840766 from the National Science Foundation.

REFERENCES