CENTAUR: A Two-Panel User Interface for Mobile Document Access

Greg Schohn and Adam Berger

Nokia Inc
503 Martindale Street
Pittsburgh PA 15212

Abstract. This paper introduces a novel user interface designed to mitigate some of the usability problems in mobile web access. The interface consists of two side-by-side panels for representing a richly-formatted document (e.g. a web page). In one panel is a "wide angle" view of the entire document, partitioned into regions. In the other panel is a "zoomed in" view of the currently-active region. We describe a prototype system, called CENTAUR, which automatically and in real-time reformats electronic documents into this side-by-side view.

1. Introduction

Accessing richly-formatted content on mobile browsers—PDAs and handheld phones, for instance—can be a trying experience. The causes are multifold: limited memory and processing power of the devices, high latency of cellular data networks, and input devices (such as the Bell keypad) optimized for numeric dialling rather than text entry. Most fundamental, however, is that web content is almost always composed for eventual viewing on a PC display. The difference in display capabilities between a PC and a mobile device is typically significant.

This paper introduces CENTAUR, a prototype system whose purpose is to bridge this gap through the use of a novel user interface. Designed for the emerging class of higher-resolution mobile devices, CENTAUR renders documents using a technique that is a hybrid between two popular approaches:

1. High-fidelity reproduction of the original page format
2. Content reformatted to accommodate the target device's geometric constraints

The recent trend in mobile devices is towards larger and higher-resolution displays. One recently introduced device, the Nokia 7700, is on par with the VGA standard of 15 years ago: 640x320 pixels and 65,536 colors. We can expect this trend to continue, but only up to a point: form factor constraints demand that mobile devices and their displays remain small. Assuming the resolving power of the human eye doesn't improve, there always will exist a biologically-imposed limit on the amount of information presentable on a fixed-size mobile device display. In other words, the gap between PC and mobile display capabilities won't disappear anytime soon.

1.1 "Narrow-Screen" and "Full-Scale" Layout

Figure 1 depicts, conceptually, the two display techniques in common use in mobile browsers.

Figure 1: *Conceptual views of two commonly used small-screen viewing modes.* **Left***: Narrow-screen layout of a web page. The original document has been reformatted into a linear or "ticker-tape" form, which requires only up-down scrolling to view the content.* **Right***: Full-scale view. This mode maintains complete fidelity to the original layout, but requires both up-down scrolling and left-right panning.*

Systems implementing *narrow-screen layout* (NSL) modify the layout of the original content to fit the width of the device display: adapting or removing tables, frames, multicolumn text, shrinking images to fit the display, and so on. The end result is a document which may be viewed in its entirety using vertical scrolling alone; no horizontal panning is required. There exist several commercial systems implementing NSL. For instance, the Opera Smartphone mobile browser can be configured to render HTML in NSL [7].

NSL's obvious benefit is the ease of navigation through the reformatted page. On the other hand, NSL suffers from what one might call *ill-defined serialization*. That is, the optimal or most intuitive serialization of a two-dimensional web page is not always obvious[1]. In some cases, a meaningful projection onto one dimension does not even exist. For example, should cells from a two-dimensional table be presented in row-major order or column-major order? How should the serialization procedure handle elements which have been assigned an absolute (x,y) position on the page using CSS?

The right half of Figure 1 displays what we will call *full-scale layout*. Here the content isn't reformatted, but instead rendered as-is in the mobile browser. The user must pan north-south and east-west to view the page in its entirety. Full-scale layout avoids the serialization-related problems of NSL, but does require of the user a burdensome amount of panning and scrolling, and also requires of the target browser a significant amount of CPU and memory to compute a rendering of the document.

Most importantly, user studies have shown that both rendering modes lead to disorientation; that is, people often lose track of their location in the document while navigating through it [3][8].

[1] In fact the problem is even more severe. Using CSS and dynamic HTML, web authors can provide a third dimension to their web content: pop-up windows, hovering text, layers, and so on.

2. A hybrid solution

CENTAUR renders content in two side-by-side panels, as shown in Figure 1. The two panels can be displayed using a frameset. Other techniques such as CSS absolute positioning or HTML tables may be used when the browser lacks support for frames.

The left panel of the display is a raster image of the entire document. The document image is partitioned into visually discrete regions. Each region is individually hyperlinked. Regions are visually demarcated; for example, with different background colors or with uniquely-colored borders. Typically, the entire thumbnail view is rendered as an imagemap; however, in the case that the target browser lacks imagemap support, the thumbnail may be rendered using the HTML `<table>` element, with hyperlinked images comprising the table's cells.

The right panel contains a NSL view of the currently-active region of the document. This panel is represented as XHTML markup. Vertical scrolling may, as usual, be required to access the full contents of the region.

Figure 2: *CENTAUR view of a web page. When a user selects a region from the left panel, an NSL view of that region appears in the right panel. We call this operation* ***region selection****. When a user clicks on a hyperlink in the right panel—thus loading a URL—a thumbnail image of the new document appears in the left panel and an NSL view of the default region appears in the right panel. We call this operation* ***document selection****.*

2.1 Finding regions

A key functional component of CENTAUR is partitioning a web page into regions. Although the "quality" of a partitioning is a subjective notion, we can set forth some guiding principles:

- Individual regions should be clearly visible in the thumbnail image. Regions that are extremely small (just a few pixels) in one or more dimensions are undesirable.
- Each region should be contiguous and convex.
- Partitions should follow the principle of least astonishment: paragraphs should not be separated across regions, nor images from their captions, and so on.

- Regions should not exceed a prespecified size.

"Size" may be defined many different ways. A simple but effective definition is this: the *size* $|x|$ of a region x is the number of pixels occupied by the content (text, images, form controls, etc.) in that region.

It has been noted that most published content, print and electronic, follows a grid-based layout, where content streams through rectangular regions on the canvas [2]. Web browsers represent the grid layout of an HTML document using a hierarchical rectangular structure known as a frame tree [9]. Not to be confused with the HTML `<frame>` element, a frame is a visual pane that contains content and the location of that content on the canvas. Frames may contain other frames, in a hierarchical way. Together, this frame tree corresponds to the hierarchy of the underlying XML-based Document Object Model (DOM). For example, a 2x2 HTML table has a frame that corresponds to the table, and four children frames, one for each `<td>`.

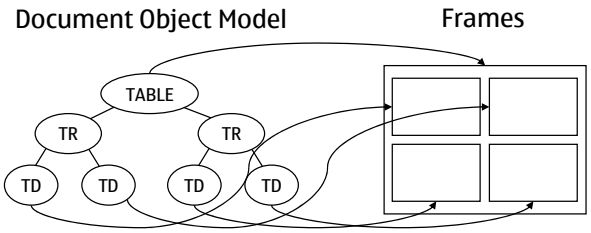

Figure 3: *DOM and corresponding frame tree representation of an HTML excerpt representing a table.*

Frames are a good starting point for region-finding, since they capture the hierarchical visual structure of a web page. To narrow the search space for regions, CENTAUR considers only regions that correspond to individual nodes in the frame tree, or to a sequence of sibling nodes. But even with this restriction, the search space is too large; after all, a frame with n children has 2^{n-1} different partitions into regions. CENTAUR therefore employs a greedy search strategy. Working from the root frame node, CENTAUR recursively divides the frame tree, selecting divisions that minimize the variance in size of the resulting regions.

More concretely, imagine that a frame node x has children $\{x_1, x_2, x_3, \ldots x_n\}$. Inserting a seam somewhere in this sequence, the expected size of the resulting regions is $E=(|x_1|+|x_2|+\ldots+|x_n|)/2$. The score S_k of a seam before child k is the inverse of the variance in size of the resulting halves:

$$S_k = \frac{1}{(|x_1|+|x_2|+...+|x_{k-1}|-E)^2 + (|x_k|+|x_{k+1}|+...+|x_n|-E)^2}$$

In other words, a split that results in a more even division of content receives a higher score.

Notice that this region-finding method takes no account of the actual content of a frame node. Ideally, a recursive region-finding algorithm as above would notice that a frame node consists of two child frame nodes with very similar (or dissimilar) characteristics; for example, both consist of a list of hyperlinks, or both are pure text, or both contain only images.

After generating the thumbnail image for the left panel, CENTAUR analyzes the page and select as active the first region which appears to contain "main content." Recent work has demonstrated automatic techniques for skipping past navigational, branding, and search forms at the top of a web page, and identifying the start of main content [5].

To illustrate, notice that in Figure 2 the right panel contains content from the "main" portion of the web page: a region containing the day's headlines, as opposed to logos, advertisements, or navigation links.

3. Previous work

Like CENTAUR, the Thunderhawk HTML browser [1] offers a two-panel viewing mode: one panel is a high-level image of the page, and the other panel contains a zoomed-in view of the active region. The key difference is that Thunderhawk does not itself generate a partitioning of the page; instead, it requires the user to specify a rectangular region of interest within the overview image. It is this region which Thunderhawk renders, zoomed-in, in the second panel.

Also related is Frayling's SmartView system [6]. Within a single panel, SmartView offers either a thumbnail view of the entire page or a NSL view of one region of the page. SmartView contains a region-finding subsystem which relies on a few heuristics involving table and form information. CENTAUR extends the concept of SmartView to a side-by-side view and offers several important enhancements, including region labeling and document analysis to locate the main content region(s).

References

[1] BitStream Thunderhawk Browser: http://www.bitstream.com.

[2] A. Hurlburt (1977) *Layout: The Design of the Printed Page.* Watson-Guptill Publications. New York.

[3] M. Jones, G. Marsden. N. Mohd-Nasir, K. Boone and G. Buchanan (1999) "A site-based outliner for small screen web access." In *Proceedings of the 8th International World Wide Web Conference*. Toronto, CA.

[4] HTML 4.01 Specification (1999). Available at http://www.w3.org/TR/html4/.

[5] A. Lee and V. Hanson (2003) Enhancing web accessibility. In *Proceedings of the 11th Annual ACM International Conference on Multimedia*.

[6] N. Milic-Frayling and R. Sommerer (2002) SmartView: Enhanced document viewer for mobile devices. *Microsoft Technical Report* MSR-TR-2002-114.

[7] Opera's Small-Screen Rendering: http://www.opera.com/products/smartphone/smallscreen/.

[8] V. Roto and A. Kaikkonen (2003) Perception of Narrow Web Pages on a Mobile Phone. *Proceedings of the 19th International Symposium on Human Factors in Telecommunications.* Berlin, Germany.

[9] C. Waterson (2002) Introduction to Layout in Mozilla. Slides available at http://www.mozilla.org/newlayout/doc.