The Radial Scroll Tool: Scrolling Support for Stylus- or Touch-Based Document Navigation

G.M. Smith, m.c. schraefel
Electronics and Computer Science
University of Southampton
SO17 1BJ
United Kingdom
E-mail: [gms | mc]@ecs.soton.ac.uk

ABSTRACT
We present radial scroll, an interface widget to support scrolling particularly on either small or large scale touch displays. Instead of dragging a elevator in a scroll bar, or using repetitive key presses to page up or down, users gesture anywhere on the document surface such that clockwise gestures advance the document; counter clockwise gestures reverse the document. We describe our prototype implementation and discuss the results of an initial user study.

Categories and Subject Descriptors: H.5.2 [**User Interfaces**]: Input devices and strategies

Additional Keywords and Phrases: Scrolling, radial scroll, touch screens, large displays, stylus input

INTRODUCTION
Navigation within a document is an everyday computing task. There are three basic strategies for performing such navigation: (1) smooth or continuous scrolling; (2) paging forward or back; (3) going to a relative or (4) absolute position in the document. The last of these is typically accomplished by means of following a hyperlink, while the first three are typically accomplished by using components of the "scroll-bar:" clicking on the scroll arrows, or between the scroll arrows and the "elevator", or by dragging the elevator.

Document navigation can also be considered in terms of the number of dimensions, such as horizontal or vertical movement, allowing one to move up-down or left-right in the document, for example, or both, and if both, at the same time (such as in diagonal movement). Conventional GUI scroll-bars only support scrolling in one dimension. The "hand" drag tool found in many graphics programs, for example, is an example of a widget that supports two-dimensional smooth scrolling. Against this background, we presents a new tool that supports smooth variable speed scrolling in one dimension which is tailored for use with pen-based systems, such as electronic whiteboards and Tablet-PCs.

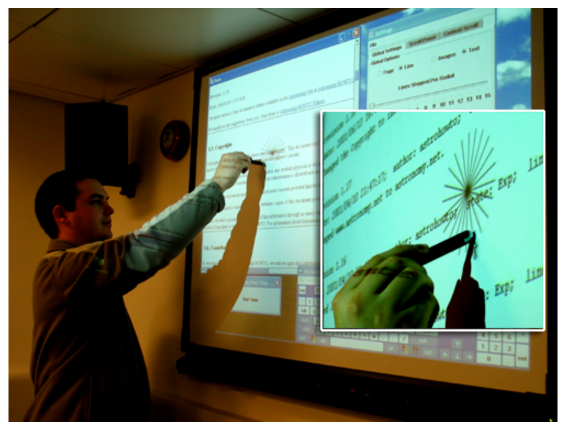

Figure 1 Radial scroll Tool being used on touch sensitive wall display. Inset shows the end of a pen is being used to operate the tool.

Motivation and Related Work
When navigating within a document, experienced users select the most appropriate strategy from the above set. For example, smooth scrolling is the primary strategy used when moving relatively small distances within a document. Since this represents one of the most common cases, alternatives to the traditional scroll-bar and keyboard "scroll arrows" have appeared: the mouse scroll-wheel, for instance, has become almost ubiquitous in desktop GUI systems. The scroll wheel differs from other methods of scrolling in that scroll rate is easy both to vary and control. Refining movements from relative to absolute positions of the scrollbar in a large document, however, are difficult and often result in several pages being skipped or the information becoming blurred [6]. For either line by line scrolling or paging, arrow or page keys can be pressed once or held for a fixed speed, automatic repeat. Repeatedly pressing these keys to focus on a particular location becomes tiresome quickly.

Beyond technical limitations, efficient scrolling can also be compromised by lack of document knowledge. Only if users are familiar with the document can they use the scrollbar to quickly locate the approximate position of interest and then use the mouse wheel to locate the exact point of interest; otherwise they are limited to scrolling through the document from top to bottom. In this case, mouse wheel speed is problematic: although in theory it allows the user to vary the rate of scroll, the upper scroll speed is quite low and is bounded by the rate at which the user can turn the wheel. Access speed has been addressed in work done by Hinckley *et al* [5] who show that that mouse wheel scrolling is improved by acceleration algorithms. Zhai *et al* [7] also investigated using isometric controls that vary scroll rate with the force applied. Blurring of the document due to scrolling at speed can also be overcome through the use of ‘speed-dependent automatic zooming’ (SDAZ) techniques which involve zooming out from the document as the rate of scroll increases [2].

Radial scroll is influenced by the SDAZ work to support speed in continuous scrolling, but also leverages the Vernier effect noted by Evans et al [3]. The FlowMenu work of Guimbretiére et al [4] also begins to explore the idea of using circular motion to enable continuous adjustment of a parameter – in this case zoom level – but does not make use of the Venier effect to allow for simple variable rate control. Evans uses circular gesturing as a means of controlling both a turntable and stirrer device for examining objects in three dimensions. The turntable is set in motion by having the user draw a small arc; the system records the rate the arc is drawn. The turntable then continues to rotate any object placed on it at a speed proportional to the recorded arc rate. Evans noted that the Vernier effect could be used to enable rapid rotation of the object; radial scroll makes more specific use of this effect, particularly to support stylus or touched based interaction.

Indeed, beyond problems explored with keyboard and mouse based scrolling, problems with scrolling are amplified in touch-based applications. A tablet PC, for instance, is designed to be operated with only a stylus when undocked and often lacks a device that is functionally equivalent to the mouse wheel. Combined with a lack of arrow keys this often means the only viable way to locate a document section is to use the stylus to drag the scrollbars. Tablets generally suffer from a small amount of parallax error due to the display being slightly below the touch sensitive surface that can make grabbing and dragging a small scrollbar handle a challenge. Likewise, large touch screens suffer from similar problems to the tablet, but increase physical demand and potentially further decrease usability. When the display becomes large, the problem becomes the magnitude of the movement required by the user [1]. The user may need to make a hand movement of several feet in order to scroll from the top to the bottom of the document. As such, techniques like SDAZ may also suffer from the same problems described above when deployed in a touch-based context.

RADIAL SCROLL

Radial scroll provides a scrolling control that is independent of the length of the document but that also allows the user to scroll as quickly and accurately as necessary using as much or as little of the display as they wish. As such, we hypothesized that the method is particularly suited to address the scrolling problems with stylus or touch based interfaces described above. Radial scroll is deployed as a transparent overlay on top of a document. The widget is activated when, for instance on a Tablet PC, the stylus is pressed to the display surface (see Figure 1 and Figure 2). The overlay displays a number of radial lines emerging from a central point (the stylus’s current position). As the user moves the cursor in a circular motion, stepping over the radial lines, the document is scrolled a user-definable fixed amount. The radial scroll makes extensive use of the Vernier effect: if users wish to scroll more slowly they move away from the center point; to scroll more quickly they move closer to the center point.

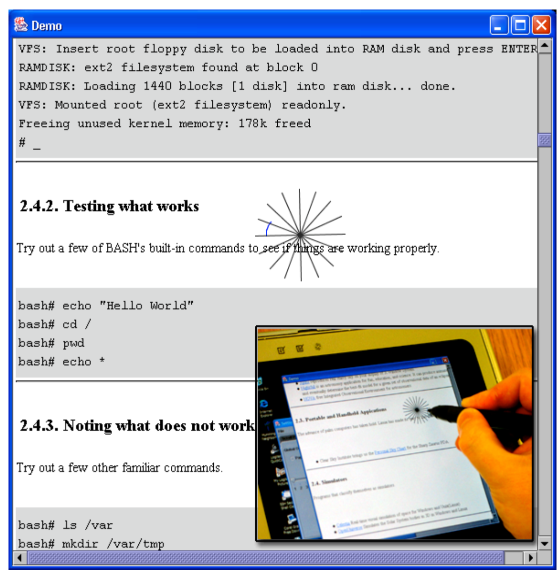

Figure 2 Radial scroll used in context mode. The short arc indicates where the user is beginning to cursor around the radial lines to scroll forward in the document. Inset shows tool used on tablet PC.

Two modes of operation have been developed. Mode one provides the user with a dedicated location for the radial scroll widget in the interface. The user moves the cursor to this palette in order to perform a scrolling action; this is analogous to moving to the scrollbar. The second mode of operation allows the radial scroll to be used above the active document and is activated by touching the screen or performing a gesture with the stylus such as drawing a circle on the screen. We refer to mode one as static mode and two as context mode.

Implementation

The radial scroll component has been developed as a Java™ Swing widget and can be used in any Swing based user interface in static mode with the same ease as any other widget. In order to use the radial scroll widget in context mode it is necessary to intercept all mouse events at the top layer of the interface stack (within Swing user interfaces this is the glass pane). The mouse events are then interpreted and re-dispatched to the relevant components like the radial scroll widget, drawn on top of the document.

As with most widgets the radial scroll allows many parameters to be configured to suit the exact deployment conditions. These have been exposed through a settings panel (see Figure 3) in the prototype and allow the user control over a number of settings including, the number of radial lines drawn, whether the radial lines are actually drawn and the size of the step per line crossed. The user can also optionally change how the radial scroll is activated and switch mouse gesture recognition on an off.

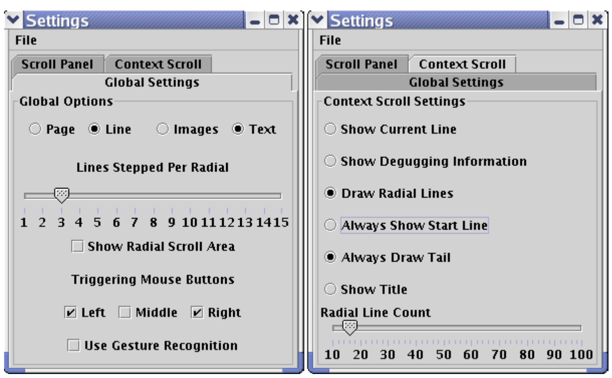

Figure 3 Settings options provided to the user.

While the user has the radial scroll activated stylus positioning information is sent to the radial scroll widget. This information is used, along with history information, to determine the direction of rotation around the center point. Initial implementations placed the center point of the radial scroll directly under the stylus pointer when the scroll tool was activated. This often resulted in an initial jump in document position as the user moved away from the center and accidentally crossed radial lines and was fixed by placing the initial center point a few (<20) pixels below the initial activation location. Rotational direction information is recovered using vector mathematics involving two vectors indicating the current and previous stylus positions. If the dot product of the previous and current vectors is positive the direction of rotation is clockwise. The angle of any vector can be recovered using the simple formula shown in Equation 1.

$$Cos\theta = A \bullet B \,/\, |A||B|$$

Equation 1 Determining the angle between two vectors.

In order to assist with testing the effectiveness of the radial scroll a logging system has been implemented which tracks movements within the document and allows playback for later analysis. When browsing images the loading of each image is recorded along with a timestamp.

USER STUDY

We have carried out a small user study in order to compare radial scroll in context mode with standard scrolling techniques available in touch-based systems. The prototype application we used allows us to display HTML documents and sets of images. Testing was carried out on a large, touch sensitive wall mounted display (125 by 95 cm running at a resolution of 1024x768 pixels) and a tablet PC (running in portrait mode at a resolution of 768x1024 pixels). Our testing protocol was a modified version of that used by Cockburn and Savage [2] for comparing speed-dependent automatic zooming against standard scroll/zoom methods. In our case we ran a within group, 2*2*2*2 repeated measures, factorial design for interface type (tablet and large screen touch displays), scroll distance (short and long scroll distances), technique (radial scroll and traditional) and task type (find a picture, find a heading). For the find a heading task, we used a 40 page manual with multiple headings. For the find a picture task, we used a collection of 110 computer renderings, picked because they displayed a single clear topic.

Participants were asked to find 3 short distance (within 5 pages/images) and three long distance targets (20 pages/images apart, backwards or forwards). Eight participants (four men and four women) took part. In each case, exposure to the radial scroll interface was interleaved and counter-balanced with use of the traditional scrolling method to minimize learning effects for each task. For the traditional tasks, participants were given the option either to drag the scroll bar in the document interface, or use a software keyboard's arrow or page keys. With the exception of one user all chose to use the scrollbars for the text document tasks. With the image task, the traditional method for the image viewer application was keyboard page up and down keys as there were no scroll bars. This meant users tapped the soft keyboard keys with their fingers on the wall display, or with the stylus on the tablet. The dependent measure was task completion time.

Observations and Analysis

For both the tablet and the wall mounted display, with both text and image selection, radial scroll worked better for navigating to short distance targets and traditional scroll worked better, for long distance targets as can be seen in Figure 4 ($F=4.96$, $p=0.027$). These are not unexpected results since simply dragging the scrollbar to locate a rough document area is a fast way to move through a document. Also, particularly in the wall mounted display, we noted during testing with the long scroll distances that when users had to radial scroll for longer searches, their circles tended to drift off center. This resulted in the user having to recenter the radial scroll (pick up their finger or stylus, put it down again) and then continue scrolling. As users became more accustomed to using the radial scroll the amount of drift diminished but did not disappear. Drift was not generally a problem with the tablet as users were able to rest their hand on the screen, and repeated the gesture from a consistent position.

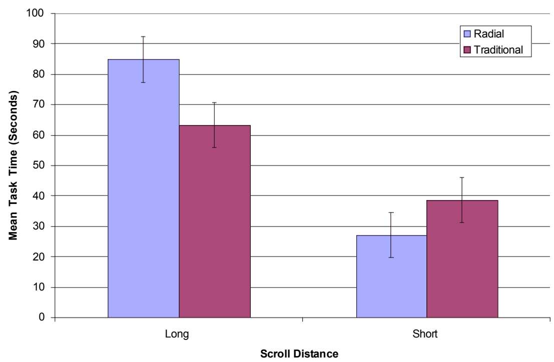

Figure 4 Heading location times dependent on scroll distance.

We also noticed a recurrent problem with the use of scrollbars: participants would inadvertently let go of the scrollbar elevator, and, when reselecting the elevator, accidentally skip sections of the document. Skipping was caused by users selecting an area just above or below the elevator on the scrollbar which is the equivalent of a page up or down keystroke. Skipping was not an issue with the radial scroll: it that is very forgiving regarding exact click locations. During the wall display heading location tasks we frequently observed users stopping to massage their shoulder after only one or two scrolls from the top of the document. Radial scroll however, allowed participants to initiate scrolling from anywhere on the document surface. Participants soon adapted to using lower screen positions for radial scrolling or in one case, the non-dominant hand.

Despite the closeness of the results between the techniques, radial scrolling was preferred by participants for the task for which it was designed: continuous, one dimensional variable speed scrolling. Participants responded positively to radial scroll on the Tablet PC, and described it as the strongly preferred method for scrolling on the wall mounted display.

CONCLUSIONS AND FUTURE WORK

Radial scroll is an effective technique for variable speed document scrolling for touch-based devices. We have found that for short scroll target acquisition tasks to which continuous scrolling is suited, radial scroll performs better than traditional techniques. To better support paging, plans include making radial scrolling faster when locating distant headings by giving the user the ability to change aspects of the scrolling mid-scroll, such as switching per line verses per page scrolling, while gesturing. Improvements to long distance scrolling speed may also be gained by modulating the document scroll step size with distance from the center of the widget. It may be advantageous to increase scroll step size, perhaps even to per page steps, when the user has moved beyond a threshold distance from the center. The most important improvement is the development of a method to detect when a user drifts from the original center of the radial scroll and then automatically move the center to the new location. By tracking the arc of the cursor path and predicting its future location a new center point can be computed and the radial scroll moved to this location, eliminating the need for the user to monitor the widget while they gesture. The goal here will be to support what we have been thinking of as "eyes-free" gesturing for document navigation in large displays.

ACKNOWLEDGMENTS

The authors wish to thank our study participants, and Maria Karam and Hugo Mills in particular. Thank you to Bill Buxton and our reviewers for their invaluable feedback. This work is part of the Smarttea Project, CombeChem (www.smarttea.org), EPSRC grant no: GR/R67729/01.

REFERENCES

1. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., and Zierlinger, A. Drag-and-Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch- and Pen-operated Systems. In *Proc. of Interact 2003* pp. 57-64.

2. Cockburn, A. and Savage, J. Comparing Speed-Dependent Automatic Zooming with Traditional Scroll, Pan and Zoom Methods. In *Proc. BCS HCI 2003* pp. 87-102.

3. Evans, K., Tanner, P. and Wein, M. Tablet-Based Valuators That Provide One, Two, or Three Degrees of Freedom. *Computer Graphics* 15(3), pp. 91-97.

4. Guimbretiére, F., Winograd, T., FlowMenu: combining command, text, and data entry. In *Proc. UIST 2000*, pp. 213-216.

5. Hinckley, K., Cutrell, E., Bathiche, S., and Muss, T. Quantitative Analysis of Scrolling Techniques. In *Proc. of CHI'2002 Conference on Human Factors in Computing Systems*, pp. 65-72.

6. Igarashi, T., Hinckley, K., Speed-dependent Automatic Zooming for Browsing Large Documents. In *Proc. of UIST 2000*, pp. 139-148.

7. Zhai, S., Smith, B., Selker, T., Improving Browsing Performance: A Study of Four Input Devices for Scrolling and Pointing Tasks. In *INTERACT'97: the Sixth IFIP Conference on Human Computer Interaction*, pp. 286-295.