An Evaluation Of Integrated Zooming and Scrolling On Small-Screens

Steve Jones **Matt Jones** **Gary Marsden** **Dynal Patel** **Andy Cockburn**

Department of Computer Science
University of Waikato
Private Bag 3105
Hamilton, New Zealand
{stevej, mattj}@cs.waikato.ac.nz

Department of Computer Science
University of Cape Town
Private Bag Rondebosch 7701
Cape Town, South Africa
{gaz, dpatel}@cs.uct.ac.nz

Dept of Computer Science
University of Canterbury
Christchurch
New Zealand
andy@cosc.canterbury.ac.nz

Abstract

Speed-dependent automatic zooming (SDAZ) has been proposed for standard desktop displays as a means of overcoming problems associated with the navigation of large information spaces. SDAZ combines zooming and panning facilities into a single operation, with the magnitude of both factors dependent on simple user interaction. Previous research indicated dramatic user performance improvements when using the technique for document and map tasks. In this paper we propose algorithmic extensions to the technique for application on small-screen devices and present a comparative experimental evaluation of user performance with the system and a normative scroll-zoom-pan interface. Users responded positively to the system, particularly in relation to reduced physical navigational workload. However, the reduced screen space reduced the impact of SDAZ in comparison to that reported in previous studies. In fact, for one-dimensional navigation (vertical document navigation) the normative interface out-performed SDAZ. For navigation in two dimensions (map browsing) SDAZ supports more accurate target location, but also produces longer task completion times. Some SDAZ users became lost within the information space and were unable to recover navigational context. We discuss the reasons for these observations and suggest ways in which limitations of SDAZ in the small-screen context may be overcome.

1. Introduction

Users regularly deal with information spaces that are too large to be fully displayed within their available window, or even within a single screen. Documents, web pages, pictures, spreadsheets and filestore folders are types of information space that commonly suffer from this problem. A well-established technique for allowing navigation around large spaces is to provide a ‘viewport’ within which a subset of the space is

displayed. The subset shown within the viewport can be controlled by the user, who conceptually moves either the viewport around on top of the space, or the space around under the viewport. Scrollbars are a common control mechanism for this interaction, supporting both continuous and discrete (often in terms of pages) navigation actions, with one scrollbar providing vertical viewport control and another providing horizontal control. A further mechanism allows the user to drag the space within the viewport (often termed panning), manipulating its location directly in any direction without constraint to either horizontal or vertical movements. Another approach increases or decreases the size of the information space subset visible in the viewport via a zoom function.

Although many systems provide all three of these scrolling, panning and zooming operations, there are numerous limitations to the navigation support that they provide. Igarashi and Hinckley (2000) note the attentional overhead incurred in changing focus between document content and scrollbars. Users must initially focus attention on a scrollbar to situate the cursor within the appropriate control item—usually one of two directional arrows, a scroll handle or either side of the scroll handle. During a scroll operation the user must then consider both the effect of the operation on the document and further possible actions in the scrollbar. Igarashi and Hinckley (2000) suggest that this can increase operational time. They also observe that small scrollbar movements can result in large movements of the viewport for long documents, causing disorientation and confusion for the user.

Cockburn and Savage (2003) note that because zooming changes the visible proportion of the information space, more scrolling is required when zoomed in, and less when zoomed out to achieve the same transformation of the viewport. Given that scrolling operations are dependent upon the current zoom-level, scrollbars can have varying effects in response to the small set of available user actions. To predict, or interpret scrollbar interactions, users must therefore understand the relationship between scroll-distance and zoom-level, adding further overhead to scrolling operations.

Users may be able to manipulate the zoom-level prior to scrolling or panning. For example, they may zoom out from the document (displaying more of it in the viewport), reducing the amount of subsequent scrolling activity because more of the document passes within the viewport with each scrolling action. The zoom-level can then be reset. However, this requires a number of interface actions, introducing further overhead for what is a very common activity. It is likely then that users incur higher scrolling costs as a trade-off for immediacy in manipulating the document.

When panning is not supported by an application, users are restricted to independent vertical and horizontal manipulation of the viewport. Consequently, navigation in other directions (such as diagonally) requires at least two scrolling operations to render a target location visible within the workspace.

The limitations of conventional scrolling techniques are of greater concern in the context of small-screen devices, such as Personal Digital Assistants (PDAs), Pocket PCs, mobile telephones and some laptop computers (Jones, et al., 1999b). These devices provide severely limited display areas in comparison to standard desktop displays devices, possibly restricting the visible portion of an information space to a few

percent of its overall area. Therefore, more scrolling is required to navigate within information spaces, with a likely commensurate negative effect on user interaction. Also, the presence of the scrollbars themselves requires valuable display area, further reducing the subset of the information space that can be displayed. Previous work has considered a range of alternatives to overcome the limitations in screen real-estate, including overview-filter-detail style approaches for small screen tasks such as Web browsing (Jones, et al., 1999a; Buyukkokton, et al., 2000b) and searching (Buyukkokton, et al., 2000a; Jones, et al., 2002); and gesture-based panning and zooming for viewing a wide variety of information including maps (Yee, 2003).

Insert Table 1 near here

Table 1 shows the comparative display sizes of a selection of current handheld, portable and desktop devices. Device display areas are compared by reading across a device row to a device column. For example, a Compaq iPAQ 5400 provides 7% of the display area provided by a Titanium Powerbook 15". The distinction between the functionality and applications provided on handheld devices and their larger desktop counterparts is becoming increasingly blurred. Device B, for example, provides a Microsoft Windows environment with standard productivity tools such as Microsoft Word, Excel and PowerPoint, yet has only 4% of the display area of a 21-inch (diagonal) desktop screen (device F).

Some information objects, such as electronic mail messages, may wrap to the available display width, increasing scrolling in the vertical dimension only. Such re-formatting may not be appropriate or possible for other objects such as word-processed documents, spreadsheets or images. There is clearly a need then for a navigation interface mechanism that overcomes current scrolling limitations.

In this paper we investigate the utility of an alternative navigation mechanism—speed-dependent automatic zooming—in the small screen context. This technique has been shown to improve user performance in navigation tasks, but has been evaluated on standard desktop displays only (Cockburn and Savage, 2003). We have extended the underlying algorithm for this technique for application on displays of any size, even allowing for dynamic resizing of the application window by the user. We have carried out an empirical investigation of the utility of the resulting interaction. We carried out our experiments using small-screen simulations on a standard desktop computer, allowing us to avoid the processor and memory limitations of a handheld device. At the time of the study these limitations meant that it was not possible to deploy an SDAZ application on a handheld device such that it would run at a sufficiently responsive speed. Further, handheld devices employ a wide variety of input mechanisms such as styli, joysticks, direction pads, scroll wheels, rocker switches and keypads. By providing mouse-based control we have established baseline performance levels using a control device that is familiar to most users. Such simulations are familiar from prior work (such as Khella and Bederson, 2004).

In the following section of this paper we describe alternatives to standard scrolling mechanisms, focussing on the speed-dependent automatic zooming technique, and report on prior evaluations of its utility. We then describe our algorithmic extensions and our implementation that can be deployed on a range of devices. In

the following two sections we present and report on a user study that investigated how well this technique supported users in navigation tasks on a small-screen device, and compare its efficacy to a standard navigation interface. Finally, we discuss the observed results and offer conclusions regarding the application of this technique to small-screen user interfaces.

2. Related work

One approach to easing the limitations of conventional scrolling is to provide alternative input mechanisms, particularly via redesign of a pointing device such as a mouse. The Microsoft IntelliMouse exemplifies pointing devices that contain a 'scroll wheel'. The wheel is rotated forward or backward by the user to control upward or downward scrolling, and is free-moving. The relationship between scroll distance and the magnitude of wheel movement can be adjusted to increase or decrease scroll speed, but scrolling only occurs in response to rotations of the wheel. However, the IntelliMouse is also a rate control device. When the wheel is depressed the degree of rotation controls scroll speed—the further that the wheel is rotated from its original position, the faster the speed of scrolling. When the wheel is released scrolling stops.

The IBM ScrollPoint mouse replaces the scroll wheel with an isometric joystick. The degree of pressure exerted on the joystick is mapped to scroll-speed. Zhai and Smith (1999) carried out an evaluation of scrolling techniques, comparing these two styles of device, standard mouse-controlled scrollbars, and two-handed interaction via a standard two-button mouse and associated keyboard-located joystick. For navigation-based tasks, they found no significant difference between completion times for the standard mouse and mouse wheel, or between the joystick mouse and two-handed input. Under both the joystick mouse and two-handed input conditions subjects were significantly faster than with the standard or mouse wheel devices. Hinckley et al (2002) provide further insight into the performance of these devices. They observed a crossover effect, with a mouse wheel supporting the best performance for short scroll distances, and a mouse-based joystick performing best for long distances. They also found that the mouse wheel performance could be significantly improved when an acceleration algorithm was applied.

Some small screen devices have isotonic or isometric wheel controls or joysticks that can be used for scrolling, and therefore go some way towards addressing scrolling issues. Although this may alleviate some effort for the users of these devices, rate-based scrolling has an attendant problem, noted by both Igarashi and Hinckley (2000) and Cockburn and Savage (2003). As the scroll-rate increases, the rapidity with which the scrolled object moves causes visual blurring and consequent user disorientation.

Speed-dependent automatic zooming (SDAZ) was proposed by Igarashi and Hinckley (Igarashi and Hinckley, 2000) to alleviate blurring and other problems with conventional scrolling techniques. SDAZ links scroll-speed and zooming of the information space. As scroll-speed increases, the zoom-level decreases in parallel; that is, more of the information space is visible within the display area. For slow scroll-speeds or when no scrolling is occurring, the zoom level is set to its normal value. No scrollbars are required, and scrolling may occur in any direction, although it may be constrained to suit particular

applications or types of information space. SDAZ, then, is a graphically-based means of providing access both to overviews and details; earlier work has demonstrated the effectiveness of dynamic, hierarchical, textual-presentation of information spaces on small displays (e.g.,(Jones, et al., 1999a; Buyukkokton, et al., 2000b)).

SDAZ is initiated when the user presses the mouse button. Scrolling begins when the user drags the mouse away from the location where the mouse button was pressed, and the direction of the mouse in relation to its original position determines the direction of scrolling. The information space scrolls in the opposite direction to that of the mouse—the user is indicating the direction in which they wish to navigate. The scroll speed is proportional to the distance of the mouse from its original position, which is controlled by the user. The zoom-level is adjusted automatically during scrolling to reduce the effect of high visual flow and the consequent blurring that occurs when the document is scrolled quickly at normal scale.

Igarashi and Hinckley (2000) initially suggest the following equations to compute speed and scale:

$$speed = C * dy$$

Equation 1

$$scale = s0^{(dy-d0)/(d1-d0)}$$

Equation 2

In Equation 1 *dy* is the current distance that the mouse has been dragged (for vertical scrolling only). *C* is a constant that can be used to modify scroll speed, although suitable values are not reported. In Equation 2 *s0* specifies the minimum scale to which the scrolled object can be zoomed. *d0* provides a distance threshold below which zooming does not take place. This allows the user to scroll slowly within the distance specified by *d0* without any scaling. *d1* specifies the mouse distance beyond which no further zooming takes place. However, to achieve a more natural interaction they amend Equation 1 as follows

$$speed = v0 / scale$$

Equation 3

where *v0* is a constant defining the speed reached at d0, i.e. the speed up to which scroll speed in linearly related to distance. This amendment conflicts with the notion that zooming (or scale) is dependent upon scroll speed. In fact, the zoom level is dependent upon the distance that the user moves the mouse, and speed is dependent upon the resulting zoom level. Figure 1 replicates Igarashi and Hinckley's graph of the behaviours of Equation 2 and Equation 3. Although the graph reflects the desired scrolling and scaling behaviour it does not accurately reflect the effect of the equations. Although the scale value is 1 when the mouse distance equals *d0*, for values of *dy* greater than zero and less than 1, the scale value is greater than

1. Also, neither scale nor speed become constant at *d1*. Both of these effects must be produced programmatically.

Igarashi and Hinckley further modified their equations during implementation by introducing a delay in scaling. This was necessary to avoid instantaneous zooming to full size when the mouse button was released, and undesirable 'swelling' effects when the user reversed scrolling direction. This implementation was then evaluated in an informal usability study of two interfaces. The first interface was a simple web-browser, although the study tasks did not require users to navigate between web pages; in effect it was a vertically scrolling document browser. Subjects were required to locate target images within a web page, using standard vertical scrollbars and the SDAZ implementation. Task completion times were approximately equal, although six of the seven subjects preferred using the SDAZ interface. The second interface was a map browser, for which the maps were artificially generated. Subjects used a joystick to navigate to a marked location by panning across the map. One condition provided zooming via a joystick button, and the other via the SDAZ implementation. The task completion times were too diverse to provide evidence regarding the interaction modes, and subjective preferences were split four to three for the two modes (the majority preferring the SDAZ mode).

Cockburn and Savage (2003) carried out a more substantial evaluation of their own implementations of SDAZ document and map viewing application. Igarashi and Hinckley made some implementation compromises to achieve rapid interaction in their Java implementation, including rendering of small text as horizontal line placeholders. Cockburn and Savage created C/OpenGL implementations providing smooth animation at high frame rates. They computed zoom level and scroll speed in a different manner from that originally proposed by Igarashi and Hinckley, employing linear transformations of scroll rate and zoom level. However, their mechanism truly bases zooming on scrolling speed. Unfortunately, they did not report suitable values for some key variables in their SDAZ interaction model. However, the minimum zoom level value must be controlled programmatically, with a value of zero providing the maximum level of detail, which is presented when either the mouse is stationary or within the threshold value. The implementation also constrained zooming functions so that the display did not immediately change from fully zoomed-out to fully zoomed-in when the user inverted the mouse direction.

Cockburn and Savage carried out an evaluation of their implementation for both document and map navigation tasks. For both tasks, the SDAZ systems were compared to standard desktop applications: Adobe Acrobat Reader for document navigation and Paint Shop Pro for map navigation. Navigation was required over either short or long distances and was controlled by a mouse for all tasks. Document navigation task completion times using SDAZ were 22% shorter on average in comparison to those for standard scrolling/panning/zooming techniques. In subjective responses participants reported lighter task loads when using SDAZ, and 11 of the 12 subjects expressed a preference for the SDAZ controls. On average, map navigation tasks were completed 43% faster using SDAZ. All subjects preferred the SDAZ interface and reported lower workloads for it.

These results are noticeably better than those reported for SDAZ than Igarashi and Hinckley, which is likely explained by a more responsive implementation, differing task types and participants, and different input devices for the map navigation task. We believe that the rigour of Cockburn and Savage's study provides concrete evidence of the potential performance gains that can be provided by replacing or augmenting current scrolling techniques with SDAZ.

Neither of the two investigations reported above evaluated this technique in a small screen context, or considered the application of SDAZ on small screen devices. Cockburn and Savage suggest that SDAZ is most suited to use for information spaces of intermediate size, that traditional scrolling is sufficient for small spaces, and that searching methods are required for large spaces (thousands of screens). Their study was carried out on a 19-inch desktop display, with a resolution of 1024x768 pixels, although there is no indication of the dimensions of the windows in which the documents or maps were displayed. Igarashi and Hinckley do not report the attributes of either the display or windows used in their study. Recently, Eslambolchilar & Murray-Smith (2004) have described an implementation of SDAZ on a PDA using tilting as a means of controlling the navigation; however, they only report small (n=5) informal user evaluations.

Clearly, display and window dimensions are a critical factor for small screen devices. The definitions of small, intermediate and large information spaces must be amended for application on small screens. For example, 25% of an information space with dimensions of 16x12 inches can be displayed within a window of 8x6 inches on a device such as F in Table 1. By comparison, only 3.6% of the space can be shown on a device such as B. In the next section we discuss our revision of the equations above to explicitly take account of the available display area.

3. A revised implementation of SDAZ

Our implementation addresses a number of factors not discussed in the prior work described above:

1. the proportion of the information space that is currently visible within the window in which it is displayed. This factor supports application on a range of display devices, including small screens, and allows for dynamic reconfiguration of behaviour when the user amends the window size during interaction;
2. how scroll speed maps to horizontal and vertical translations of the information space display;
3. scrolling and zooming behaviour when the bounds of the information space are reached either via scrolling or zooming actions;
4. generalising the algorithms such that they can be used to support navigation both horizontally and vertically, or constrained to one of the two directions.

Further, we amend the implementation so that the zoom level is only amended as the user drags the mouse away from its initial location. This avoids the issue of rapid scaling observed in prior implementations when users reversed their scrolling direction.

3.1 Algorithms

We follow Igarashi and Hinckley's revised equations in which scroll speed is linearly dependent upon the current zoom level. In fact we replace the notion of scroll speed with two factors: the required translation of the information space in the x and y dimensions. For both document navigation and map navigation we determine scroll speed via the following algorithm:

```
sdX <— 0
sdY <— 0
if (no horizontal constraint) sdX <— C * visibleArea * scaleValue * dx
if (no vertical constraint) sdY <— C * visibleArea * scaleValue * dy
if (will not scroll space out of viewport) scroll horizontally by sdX
if (will not scroll space out of viewport) scroll vertically by sdY
```

Algorithm 1

Where *sdX* and *sdY* are the required scroll distances in the x and y dimensions. *visibleArea* is a real value where 0 <= *visibleArea* <= 1 indicating the proportion of the information space currently visible within its enclosing window. *dx* and *dy* represent the distance that the mouse has been dragged in the x and y dimensions. *C* is a constant (with *C*>0) that can be placed under user control to modify scroll speed. A value of *C*=10 provides a responsive and natural interaction using the system configuration described later in the paper. Under user control, this can compensate for variable processing and graphics capabilities of different devices. *scaleValue* is a real value where 0 < *scaleValue* <= 1. When *scaleValue* is 1, the information space is displayed at its initial scale; it is at its maximum level of detail. A *scaleValue* of 0.25 would display the space at a quarter of its original size. Scrolling does not take place if the information space will be moved outside of the viewport by the scrolling operation.

For document and map navigation *scaleValue* is computed via the following algorithm:

```
If (dragDist has increased AND
        Full width of information space not visible AND
        Full height of information space not visible AND
        minThreshold <= dragDist <= maxThreshold AND
        scaleValue > minScaleValue)
        a <— abs( (dragDist-minThreshold)/(maxThreshold-minThreshold) )
        scaleValue <— minScaleValue + ( (1-a) * (1-minScaleValue) )
```

Algorithm 2

dragDist is the distance that the mouse has been dragged from its initial location during the current scroll/zoom operation. *minThreshold* is the mouse drag distance below which only scrolling is operational and no zooming takes place. *maxThreshold* is the mouse drag distance above which no further zooming takes place. *dragDist*, *minThreshold* and *maxThreshold* are integer values expressed in terms of the coordinate system used in the implementation. *minScaleValue* is real with 0 < *minScaleValue* <= 1, and indicates the maximum extent to which the information space can be scaled to reduce its size. For example

a value of 0.1 would mean that no further zooming out takes place one the space is displayed at 10% of its original size.

The value *a* reflects the proportion of the distance between *minThreshold* and *maxThreshold* at which the mouse is currently located, and therefore *scaleValue* is inversely proportional to *a*.

The five conditions in Algorithm 2 provide the following behaviour (illustrated in Figure 1):

- zooming only occurs as the user drags the mouse away from the location of the start of the zoom/scroll operation
- no further zooming out occurs once either the full width or height of the information space is visible in the window;
- zooming occurs only when the mouse drag distance is within the allowable bounds;
- no further zooming out occurs once the minimum scaling factor has been reached.

Insert Figure 1 near here

The two algorithms are executed only during a pointing action (for example, while a button is depressed on a mouse, or while a stylus is in contact with the screen). When the action ends (the mouse button is released) the information space is transformed in two ways:

- the scale value returns to 1, to display the space at its normal magnification level. This is smoothly animated to minimise visual disruption;
- the information space is repositioned within the viewport to place the location under the pointing device at the end of the operation at the centre of the viewport. Again, this is smoothly animated.

The two transformations occur in parallel.

3.2 Implementation

Our application is implemented in Java 1.4 using the Piccolo[1] 1.0 Zooming User Interface (ZUI) toolkit (Bederson, et al., 2004). It has been run successfully on Microsoft Windows, Linux and Macintosh OS X operating systems. The SDAZ behaviour of the application is provided by user interface event handler classes. One event handler class (*SpeedCoupledPanEventHandler)* deals with scrolling operations, and the other (*SpeedCoupledZoomEventHandler*) with zooming operations. As a result, minimal programming is required to add this behaviour to information space viewing applications.

The application manipulates images of the documents and maps to be navigated. The display image can be a command line argument, as can the minimum and maximum zoom thresholds, the minimum zoom value and navigation direction constraints. The initial layout strategy can also be specified. For textual

[1] http://www.cs.umd.edu/hcil/jazz/index.shtml (December 2003)

documents, the viewport of the window can be placed at the start of the document, which is centred horizontally. For images such as maps, the viewport can be placed so that the centre of the map appears at the centre of the window.

Insert Figure 2 near here

Figure 2 illustrates the software in use for viewing a city map. All of the available display area is used to present the information space. Four feedback mechanisms are provided to the user, and are heavily emphasised in the Figure for clarity. During use of the software these items are displayed as unobtrusive but visible lines that are a single pixel wide. They are provided during a navigation action, and removed on completion of the action. At the start of an action, the two concentric circles are placed so that the location of the action on the display is at their centre. The smaller circle indicates the minimum threshold value, and the larger indicates the maximum threshold value, as described above. As the user drags the pointing device, a direction line is drawn between the starting position and its current location, indicating the direction of travel. If the pointer remains within the smaller circle, no scaling of the information space occurs, only scrolling.

The user is free to navigate in any direction. As the pointer moves further away from the starting position, the scroll rate increases. When the pointer moves beyond the inner circle, both scaling and scrolling operations take place. Scaling is progressive until the pointer reaches the outer circle, at which point the minimum scale value is reached and no further scaling occurs, although scrolling is still active.

The rectangle indicates to the user the subset of the information space that will be visible on completion of the operation. It dynamically changes size proportional to the current scale value. To begin with it bounds the entire viewport, and does so until scaling is applied to the information space. It is always centred on the location of the pointing device, and consequently the location under the pointer will be located at the centre of the viewport on completion of the operation.

4. Experimental evaluation

We carried out an experiment to compare the efficiency, user perceptions and usability issues of our SDAZ implementation with those of traditional scrolling techniques in the context of small screens.

Our hypotheses where that with SDAZ:

- *the number of actions required to complete tasks will be lower than in the conventional case*. Our expectation was that users would take advantage of the ability to exert continuous control over SDAZ navigation actions and thus carry out fewer actions. Failure of this hypothesis to hold would suggest that users are unwilling or unable to deviate from using multiple discrete actions as engendered by a conventional interface;

- *the time to complete tasks will be lower than in the conventional case*. We expected that the reduction in excessive visual flow afforded by SDAZ, combined control over scrolling and panning, and having the point of control located on the information space would lead to improved target acquisition times; and,
- *the accuracy in locating targets will be greater than in the conventional case*. Our expectation was that users would 'point at' or 'touch' targets due to having navigation control on the actual information space (which is particularly desirable if information items are densely packed).

4.1 Conditions

The variables that were manipulated in the evaluation:

- *interface type*: either the SDAZ interface or a traditional scrolling/panning interface (henceforth termed 'standard');
- *information space type*: either a map or document;

4.2 Subjects

Twelve subjects took part in the experiment. Each was an undergraduate or postgraduate university student. and three of the subjects were female and nine male. The first language of ten of the subjects was English, and all subjects had written and spoken English-language proficiency suitable for study at an English-speaking university. All subjects used computers on a daily basis. None had previously used SDAZ interfaces, and four had used map-viewing software, although mainly via a Web-based interface.

Subjects were recruited through poster and e-mail advertisements, and each received a book token in return for their voluntary participation.

4.3 Tasks

Each participant completed 48 experimental tasks in two sets. A set of 24 tasks was carried out with each of the two interface types. Half of the subjects carried out SDAZ tasks first, and half standard scrolling interface tasks first. For each interface type, twelve document navigation and twelve map navigation tasks were undertaken.

Examples of document navigation tasks are: "Find the next non-colour image down from your starting location" (picture location), and "Find the "Summary" section of the document down from your starting location." (heading location). Headings were all emphasised within the document text.

Examples of map navigation tasks are: "You are currently located at Wairau Inter School in Coatesville. To the north-west is a school called Coatesville School. Locate it." (locate by direction), and "You are currently located at St Francis School. Follow highway 16 west until you reach Royal Road School."

(locate by path). All navigation start and end points were the locations of schools, which were coloured yellow on the map, and therefore visually distinct from other types of location.

Figure 3 shows samples of the two information spaces used in the experiment: a road map of a city and surrounding area, and a research paper. Both exhibit structural and/or presentational aspects to support navigation. The map uses colour coding for identification of highways, schools, hospitals and other types of location, and further semantic encoding such as matching text size to the size of geographic areas. The research paper was conventionally structured with sections and subsections, the headings of which were visually emphasised in the text. Figures and tables provided additional location landmarks. For both map and document, major semantic markers such as suburb names and section headings were visible and readable at all zoom levels although finer detail such as road names and paragraph text were not. Clearly the ability to interpret the information space at low levels of detail is important whether navigating by SDAZ or conventional methods. Novels and small scale maps, for example, may not support this and so we would expect the previously reported advantages of SDAZ to be somewhat dependent on the nature of the information space. In this study we have adopted the approach of others by employing a map and a document with clear structure and semantic markers.

Insert Figure 3 near here

4.4 Materials

The experiment was conducted using a standard desktop computer with a 1.8Ghz AMD Athlon processor, 240Mb of RAM, and running the Microsoft Windows XP operating system. Display characteristics (resolution and colours) were set to simulate those of a representative target device—a Compaq iPAQ 3870 Pocket PC. Both the SDAZ and standard interface were presented in windows corresponding to the display dimensions of the iPAQ device, namely 2.26 inches wide by 3.02 inches high. Input was via a standard two-button mouse.

Insert Figure 4 near here

The SDAZ interface was as described above. The settings for default scroll speed, zoom thresholds and minimum zoom level were set after a pilot study in which a range of users were observed in unconstrained use of the software. Hence, in the absence of any evidence regarding optimal calibration, the settings were informally established to match those that were comfortable for a range of users.

The standard interface simulates a tool such as Adobe Acrobat Reader and is illustrated in Figure 4. It provides the following navigation functions:

- *panning*: when the 'hand' tool is selected the user can drag the information space using the pointing device. The extent of the drag distance is constrained by the bounds of the viewport window, and the direction is constrained by the information type. For the map to the left of the figure, panning can occur in any direction, but only vertically for the document to the right;
- *scrolling*: vertical and horizontal scrollbars (vertical only for the document to the right) behave in a normative manner. The arrows at either end allow for small scroll increments which can either be discrete or continuous. The 'elevators' within the scrollbars can be dragged backwards and forwards by the user with corresponding scrolling of the viewport. The scrollbar channels can be clicked for a scroll distance equal to the size of the viewport (horizontal or vertical, depending on the scroll direction), and the increments can be either discrete or continuous;
- *zooming*: a zoom level menu (to the top right of the interface) allows selection of discrete zoom levels (25, 50, 75 and 100 percent), and when a value is selected the information space is immediately scaled to the corresponding level. Additionally, zoom-in and zoom-out tools can be used to move between the discrete zoom-levels by clicking on the viewport.

A printed questionnaire was developed and issued to subjects to record their backgrounds and experiences. Printed task sheets were provided to each subject.

4.5 Procedure

On arrival subjects were presented with a printed Bill of Participants rights, and a consent form describing the nature of the experiment and tasks that they would be requested to perform. Once consent was provided the study proper began.

To begin, a subject was provided with a training workbook for the first class of task to be presented (either document or map navigation). This was divided into two sections, each describing one of the two interaction styles to be used (SDAZ and standard) and associated training tasks. The subject was asked to read the description of the first interaction style, and it was then demonstrated to them by the experimental supervisor. They then worked through four training tasks corresponding to the four combinations of distance (short, long) and task type (locate image or heading for document, locate by direction or path for maps).

The subject was then asked to read the description of the second interaction style, which was then demonstrated to them, followed again by four training tasks. Subjects could ask questions of the supervisor throughout the training period.

Subjects were then provided with an experimental workbook that presented 24 tasks for the first task class (document or map navigation). They were instructed to begin working through the first set of 12 tasks

using the first interaction style. Once these were completed, the application software was changed as appropriate and the second set of 12 tasks undertaken.

The subject was offered a short break if required. The same training and experimental task process was then repeated for the second task class.

Upon finishing each task set a subject completed a NASA Task Load Index worksheet (Hart and Staveland, 1988), and was prompted to comment on their experiences with the systems they used.

All tasks (including demonstration and training tasks) were presented via an on-screen dialogue window, shown in Figure 6. Each individual task commenced with the subject clicking the "Read Task" button, which resulted in the display of the task instructions. Once the subject wished to begin the task proper, they clicked the "Start Task" button. This resulted in the display of the required navigation window (SDAZ or scrolling) immediately below the task dialogue, with the view of the document or map placed at the start location of the task. On completing the task to their satisfaction, the subject clicked the "Task Completed" button. The navigation window and the task instructions were then removed. The subject proceeded to the next task by once again clicking the "Read Task" button.

4.6 Data captured

The software was instrumented to automatically record performance data for each task undertaken by each subject. For the SDAZ interface the data recorded for each task was:

- *user interface actions*. The number of distinct user interface actions for the task, with an action defined as a mouse press and mouse release event pairing;
- *action timings*. The start time, end time, and duration of each user interface action recorded in milliseconds;
- *task duration*. The time taken on the task, in milliseconds, defined by the time between the user clicking the "Start Task" and "Task Completed" buttons;
- *accuracy*. The distance, in pixels, of the target location from the centre of the viewport when the subject indicated task completion.

For the standard interface the data recorded for each task was:

- *user interface actions*. The number of distinct user interface actions for the task, with an action defined as any one of: pan, zoom in by clicking, zoom out by clicking, vertical scroll action, horizontal scroll action, pan tool selection, zoom out tool selection, zoom in tool selection, zoom level combo-box selection;
- *action timings*. The start time, end time, and duration of each user interface action recorded in milliseconds;

- *task duration*. The time taken on the task, in milliseconds, defined by the time between the user clicking the "Start Task" and "Task Completed" buttons;
- *accuracy*. The distance, in pixels, of the target location from the centre of the viewport when the subject indicated task completion.

For each set of tasks for an interaction style-document type pairing (four sets in total) a subject completed a modified NASA Task Load Index questionnaire. The questions are shown Figure 7. Each response was a numeric value of 1, 2, 3, 4 or 5, with the extreme values given indicative textual labels.

The experimental administrator also made notes of critical events and comments made by the subjects.

5. Results

The observed quantitative data sets were subjected to the Kolmogorov-Smirnov test of normality prior to further analysis (Siegel and Castellan, 1988). This test revealed a high degree of non-normality across the data sets, leading to the use of the non-parametric techniques reported in this section.

5.1 User interface workload

Table 2 shows the mean number of user interface actions per task for the four combinations of interface and document types. Outlier values of more than 3 standard deviations from the mean have been removed. For the map stimulus, the mean number of actions using SDAZ was 1.97, compared to 17.66 using the standard interface. For the document stimulus, the mean number of actions using SDAZ was 2.76, compared to 6.92 using the standard interface. For both the map and textual document types, the differences are significant (Mann-Whitney, $U=1436.5$, $p<0.0001$).

Insert Table 2 near here

Figure 5 shows the percentage of map tasks completed with a given number of actions. 55% of the SDAZ-based tasks were completed with only 1 user interface action, and all SDAZ-based tasks were completed with 11 or fewer actions. Only 6% of the map tasks with the standard interface were completed with a single action, and half of them required up to 14 actions to be completed.

Insert Figure 5 near here

Figure 6 shows the corresponding data for document-based tasks. 41% of the SDAZ-based tasks were completed with only 1 interface action, compared to 23% for the standard interface. All of the SDAZ-based tasks were completed with 13 or fewer interface actions, compared to up to 50 actions for the standard interface.

Insert Figure 6 near here

5.2 Task completion times

Table 3 shows the mean task completion time (in seconds) for the four combinations of interface and document types. Outlier values of more than 3 standard deviations from the mean have been removed. Mean completion times are longer for the SDAZ interface for both map and textual documents. The difference is not significant (Mann-Whitney, U=9684, p=0.6353) for the map stimulus, but is significant for the document stimulus (Mann-Whitney, U=8183.5, p<0.002).

Insert Table 3 near here

Figure 7 shows the percentage of map tasks completed in a given number of actions. For both interface types, approximately 50% of all tasks were completed within 30 seconds, and 80% within 60 seconds. Completion times are more divergent after the 60 second threshold—all standard interface tasks were completed within 145 seconds, yet 5% of the SDAZ tasks took longer than this to be completed.

Insert Figure 7 near here

Figure 8 shows the corresponding data for document-based tasks. Divergence between the performance of the interface types is more pronounced than for the map stimulus. 73% of all tasks were completed within 30 seconds with the standard interface, compared to 52% with SDAZ. The corresponding values at the 60 seconds threshold are 89% and 76% respectively.

Insert Figure 8 near here

5.3 Accuracy

Accuracy is measured by the distance (in pixels) of the target location from the centre of the viewport on task completion.

Table 4 shows the mean accuracy (in pixels) for the four combinations of interface and document types. SDAZ supported a significantly higher level of accuracy for the map stimulus than the standard interface (Mann-Whitney, U=5988.5, p<0.0001). The opposite is true for the document stimulus, with the standard interface supporting a significantly higher level of accuracy (Mann-Whitney, U=8430.5, p=0.0061).

Insert Table 4 near here

For the map stimulus, a distance of less than 140 pixels indicates that the target was definitely visible within the viewport on task completion, and we characterise this as a successful result. The corresponding distance for the textual stimulus is 170 pixels.

Insert Figure 9 near here

For the map stimulus, 92% of all SDAZ tasks and 89% of all standard interface tasks were successful. The frequency distribution of accuracy (in 10 pixels bins) is shown in Figure 9. Although the percentage of successful tasks is very similar for both interfaces, the graph reveals that subjects were more accurate with SDAZ than the standard interface on maps tasks. For the document stimulus, accuracy levels were 78% and 76% for the SDAZ and standard interfaces respectively—again very similar, but noticeably lower than for maps. Figure 10 shows the frequency distribution of accuracy (in 10 pixel bins) for document tasks. Using either interface, subjects tended to place the target location within the viewport, but not centred within it.

Insert Figure 10 near here

The descriptive measures of accuracy are somewhat skewed by outlier values. When values greater than one standard deviation from the mean are removed (5% of all values) we see the outcomes in Table 5.

Insert Table 5 near here

Again, the difference is significant between the SDAZ and standard interfaces for both the map stimulus (Mann-Whitney, $U=5294$, $p<0.0001$) and the textual stimulus (Mann-Whitney, $U=7019.5$, $p=0.0015$).

5.4 Use of navigation tools in the standard interface

Subjects' use of the navigation tools in the standard interface was recorded. These operations can be characterised as navigation, zooming or tool selection, and the frequency of their occurrence is shown in Figure 11. Navigation operations occurred when subjects dragged the viewport contents whilst the 'hand' tool was active (panning), or used the scrollbars (vertical only in the case of the document stimulus). There were a total of 2938 actions for map tasks, and 1172 actions for document tasks.

Insert Figure 11 near here

For map-based tasks, almost 85% of operations were for navigation, with 82.3% dedicated to panning. For document-based tasks 75% were for navigation (11.5% for panning, and 63.7% for vertical scrolling). Subjects changed the zoom-level at which the map or document was displayed by clicking on the viewport whilst either of the magnifying glass tools was active (increase or decrease zoom-level), or selecting a zoom-level from a menu. Relatively few operations modified the zoom-level—6.4% and 13.5% for the map and document stimuli respectively—and almost all involved clicking on the viewport.

The remainder of the operations involved tool selection: 8.4% for maps and 10.2% for documents.

5.8 Subjective workload measures

A task load questionnaire, based on the NASA Task Load Index, was administered to each subject after each set of tasks for a document type-interface type pairing. Responses were on a scale of 1 to 5, and were normalised so that lower values reflect lower task loads. Table 6 shows mean ratings for each interface for both maps and textual documents, with the lower of each pair of values emphasised. The overall mean

ratings are almost identical, with values of 2.70 and 2.73 for the SDAZ and standard interfaces respectively for map tasks. The corresponding values for document are 2.60 and 2.61.

Insert Table 6 near here

Mental workload was judged to be very similar for both interfaces and type of stimulus, with the standard interface requiring marginally less for document tasks. SDAZ was considered more forgiving than the standard interface for map tasks, and vice versa for document tasks.

There is a marked difference between physical workload requirements (as judged by the subjects) of the two interfaces. Irrespective of stimulus type, SDAZ was considered to require less physical effort than the standard interface.

There is no significant difference between the overall task load responses for the two interface types for maps (Wilcoxon matched-pairs signed ranks test, $p=0.8279$), or for documents (Wilcoxon matched-pairs, signed ranks test, $p=0.8593$).

6. Discussion

For the SDAZ interface, the integration of scrolling and zooming into a single interface action resulted in a low number of user interface actions overall, with almost half of all tasks completed with a single action. SDAZ also supported rapid task completion in most cases, with half of the tasks completed within 30 seconds and almost 80% within 60 seconds. Reduced interface actions and low task times are complemented by a high level of success in task completion, with more than 80% of the tasks having a successful outcome. We conclude then that, in general, the subjects could use the SDAZ system effectively and efficiently, despite no prior experience and limited training with such an interface.

The comparison of interface actions between SDAZ and the standard interface matched our expectations. The number is significantly lower for SDAZ, by factors of 9 and 2.5 respectively for map-based and document-based tasks. The difference is so pronounced for map tasks because of the problems with standard support for scrolling and panning in two dimensions. In the best possible case the standard interface requires a single action to locate a target. This could be either a horizontal or vertical scrolling action when navigation in a single dimension will bring the target into view. Alternatively it could be a panning action in any direction when the target is within one screen of the current viewport location (because the pan tool does not allow dragging beyond the bounds of the viewport). Both of these situations are unlikely.

Insert Figure 12 near here

For other situations, panning requires multiple actions because each action only moves the view by the size of the viewport, and scrolling requires congruent but discrete manipulations of the horizontal and vertical

scrollbars. Manipulating the zoom level can ameliorate these problems, but subjects did so infrequently with the standard interface, overwhelmingly preferring to navigate via multiple panning actions.

Whilst it is a coarse measure, the comparative number of actions is interesting because it shows that most subjects were immediately comfortable with few, but continuous dynamic control actions with SDAZ. Whereas, for example, with the standard UI users far preferred to use multiple discrete panning actions to navigate maps. It establishes then that users are likely to quickly adopt the novel interaction mechanism of SDAZ, rather than attempt to simulate the mechanisms with which they are familiar from pan-scroll-zoom interfaces. A continuous SDAZ action is very tolerant to a user's velocity and direction errors because they can be immediately and responsively reversed whilst maintaining focus on the navigation goal. This is not the case with the standard scroll-pan-zoom interface (with the exception of dragging a scrollbar 'elevator').

Clearly this measure does not reflect the complexity of what users do during an SDAZ navigation action. Figure 12 illustrates the actions of two users on the same map navigation task. This data comes from a recently completed study in which all but one of factors were consistent with the study reported here, the difference being that users carried out tasks on a handheld computer and used a stylus for control. Figure 12(a) and Figure 12(b) show each user's actions in relation to the screen of the device. Each line represents a user action initiated by a stylus down event, followed by stylus motion, and completed by raising the stylus. The starting point is marked by a small circle, the end point by an arrowhead, and each line has a numerical label that indicates the ordering of the actions (with 1 being the first action). The two approaches appear strikingly different (they characterise the variation in user strategies that we observed). The first user achieved most of the task through a single complex exploratory action, and then completed it with a short action to focus in on the target. The second user took an incremental approach, using more short actions to converge on the target, and consequently experiencing multiple zoom-out and zoom-in transformations. Figure 12(c) and Figure 12(d) present the same actions over the duration of the tasks, which are very similar: 55 and 65 seconds respectively. These figures reveal some similarity between the strategies of the two users, with interleaved bursts of activity and inactivity, the difference being that the first user did not raise the stylus. This suggests that this user did not wish to lose the current context (i.e. wished to maintain zoom level) and used it to determine their next action after some reflection. A closer consideration of Figure 12(b) reveals that the second user began the majority of actions by moving the stylus a sufficient distance to effect a change to the zoom level, such that it was similar to that in place at the end of the previous action. Perhaps then, SDAZ should allow users to maintain zoom context even when the stylus is lifted (perhaps for some finite time). This is likely to be helpful in a truly mobile context where continuous stylus contact is difficult to achieve.

SDAZ tasks took longer to complete, on average, than the standard interface tasks. However, there was no significant difference between the interfaces for map-task completion times. For some map-based tasks, subjects became lost and unable to quickly reorient themselves in relation to either the start or target locations. One explanation for this is that the standard interface allowed subjects to zoom out from the map

(thus seeing more context) without any further navigation. With SDAZ, zooming is not possible independent of further scrolling (see Figure XXX), which is likely to exacerbate the subject's orientation difficulties. We believe that for such tasks SDAZ would benefit from contextual cues that, at the very least, allow users to reorient themselves with respect to, or revisit, their starting location.

Completion times are significantly better with the standard interface than SDAZ for document-based tasks. One reason for this is that the number of interface actions with SDAZ reduced slightly from map-based to document-based tasks, whereas they halved with the standard interface. The actions required by SDAZ are exactly the same (although constrained to a single dimension), whereas with the standard interface the tasks could be completed with vertical scrolling only. Additionally, subjects were likely very practiced with vertical document scrolling and target acquisition (for example, in word-processing applications or web browsers), yet had very limited experience of SDAZ. This is somewhat at odds with the findings of Patel *et al.* (2004) in their study of vertical scrolling SDAZ and standard interfaces for photograph browsing, suggesting that characteristics of the information space (such as structure and visual cues) are important factors.

There was little difference between levels of successful task completion between the two interfaces, with SDAZ marginally higher. SDAZ supported significantly higher accuracy for map-based tasks. This meets our expectation because SDAZ interaction supports the user pointing at the target at the end of the navigation action, placing it automatically at the centre of the viewport as a result. However, this expectation is confounded by the fact that the standard interface supported significantly higher accuracy for document-based tasks. Again, this may be partially attributed to the subjects' expertise in vertical scrolling tasks, yet accuracy was noticeably lower for document-based tasks in comparison to those involving maps. Once explanation is that subjects positioned targets as if preparing for ongoing navigation from that target. For map tasks this may mean placing the target close to the centre of the viewport to provide maximum context in all directions. For document tasks this may mean placing the target at the top of the viewport to maximise the amount of the document that could be read sequentially following the target. This issue requires further investigation.

All study participants used SDAZ with default settings for scroll speed, zoom thresholds and minimum zoom level. These were established through a pilot study to suit a range of users, and at the time there was no empirical evidence regarding optimal calibration of SDAZ systems. Varying the values of these settings may clearly impact upon user performance with SDAZ. More recently Savage (2004) has carried out a detailed study of SDAZ calibration issues. He considered theoretical maximum visual flow rates taking into account screen size, human visual processing capabilities, zoom levels and a user's field of view. Following an empirical study Savage suggested default SDAZ settings, and concluded that the speed at which users could comfortably scroll was inversely proportional to magnification level for vertically scrolled documents. For maps, comfortable scroll rates remained quite constant across a range of zoom levels. This emphasises the need to calibrate SDAZ systems differently dependent on the information space

characteristics, which we did for the systems used in our study. Although helpful, Savage's findings only partly account for display size, and the question as to whether his recommendations hold for small displays remains open.

The use of a mouse-controlled small-screen simulation on a desktop computer does not address the issues of form factor, input mechanism and environment that will likely affect user performance in a 'real-world' context. However, a strength of our study has been to remove these variables, focussing on the effect of limited display space on the efficacy of SDAZ and enabling direct comparison to earlier work utilising the full real estate of a large display. Additionally we have some evidence from the recent work of Lau (2004) and Patel *et al*. (2004) that, in general, our findings hold for user experience with SDAZ on handheld devices such as mobile handsets and PDAs.

None of the subjects commented upon the efficacy of the control feedback illustrated in Figure XXX. This suggests that it was not intrusive, but does not indicate whether or not it was useful. Alley (2004) reports on a study of this feedback. He observed that it neither significantly improved nor degraded user performance, and concluded that SDAZ provides enough implicit feedback through transformations of the information space to nullify the effect of such explicit feedback.

Overall, subjects perceived no significant difference between the workloads imposed by the two interfaces. This reflects well on SDAZ given the minimal experience (through training) that the subjects had with this novel interface, in comparison to a great deal of experience with the standard interface.

The physical actions to control the SDAZ interface are simple, yet map to quite a complex relationship between drag-distance, scroll speed and zoom level. This is reflected in subjects' judgement that SDAZ was more complex than the standard interface. Despite this, for map-based tasks they also considered there to be no difference between the two interfaces in terms of mental workload. For document-based tasks the standard interface was seen to impose less mental workload than SDAZ. One likely reason for this is that the subjects had more experience with vertical scrolling tasks using a standard interface than navigation in two dimensions, whereas their experience with SDAZ for the two types of workspace was the same.

The integration of scrolling and zooming controls in SDAZ clearly provided a perceived reduction in physical workload in comparison to the standard interface for both map and document tasks. This matches the performance data, as does the subjects' perception of their success in completing the tasks, which is almost identical across interfaces for both maps and documents. At odds with this is the fact that the subjects were less satisfied with their performance with SDAZ than the standard interface. We believe this to be because, from prior experience, they were clear as to what constituted optimal performance with the standard interface. With limited experience of SDAZ, they considered there to be room for improvement in

their use of the technique. This is likely to also underlie the higher level of frustration that they experienced with SDAZ.

In essence, the workload measures confirm the subjects' acceptance of SDAZ. In light of the fact that they had not used the technique before, this is very positive, and we believe that after more practice users will favour SDAZ based on perceived workload.

7. Current Developments

Outlying values in the SDAZ data (particularly with respect to time and accuracy), and observations during the study suggest that when subjects failed at tasks, they failed very badly. Through observation, the main cause of these extreme failures is the fact that the subjects became 'lost'—they did not know how to reach the target, or more importantly, how to return to their starting location to reorient themselves and begin the task again. On large displays, with surrounding context for the user's focus, this problem is likely to be less evident. On small displays, minimal navigational activity can remove the user's starting point from the viewport.

A common solution to support the user in understanding how the area visible in the viewport relates to the whole information space, is to provide an overview map. Such an overview would condense the entire space into a small display area, with the viewport location emphasised. Although perhaps feasible on a desktop display, small displays have limited space to present an overview without obscuring the detailed view of the information space.

Our solution is to apply the 'halo' technique (Baudisch and Rosenholtz, 2003) which we have already implemented and will evaluate in the near future. Figure 16 illustrates the use of halos. The image to the left of the figure shows the user in the course of a navigation action on a London Underground map. To the bottom right a halo indicates both the direction and distance to be travelled to return to the start location of the navigation action. The halo is always situated at the edge of the viewport, and its position is dynamically updated during navigation. By moving in the direction of the halo the user moves towards the start location. The width of the halo reflects the distance that the user needs to travel to return to the start location—the wider the halo, the greater the distance. By making halos semi-transparent, and peripheral to the user's focus within the viewport no additional display space is required, and no detail of the map is obscured.

8. Conclusions

One of the motivations for doing this work were the dramatic improvements reported by Cockburn and Savage (2003) when SDAZ was employed in a large-screen context. While our implementation provides the sort of responsive user experience seen in Cockburn and Savage's system, as well as closely following

their experimental design, the results are clearly not as impressive as those on the large screen. In that context, document tasks were completed 22% quicker and map tasks 43% faster, on average; and task workloads were perceived as lower. The small screen size, then, reduces the effectiveness of scheme.

The benefits of SDAZ on a large screen come from being able to use contextual cues to aid navigation even when the speed of navigation is high and the magnification level low. However, on a small screen, when the system zooms-out, as the user increases the speed of scrolling, ability to look ahead is much reduced in comparison to a similar level of magnification on a larger display.

This degradation of contextual information might also explain the better performance of the small screen map tasks compared to the document ones. With the former, even on a small screen, it is possible to make out some useful context as the user flies higher over the image; in contrast, section heading and other document landmarks, can become less distinct. Documents with more striking structures – for example Web pages with distinct regions for navigation and content – may fare better.

The 'lost in the information space' problem is well established in a variety of application domains. Some users of our SDAZ system were unable to regain the context of their current location in relation to the full information space and target location after one or more navigation actions. That is, users did not know in which direction to travel to either find the target, or to return to the starting point and begin again. This occurred in five percent of all SDAZ tasks, although there is no evident pattern across subjects, tasks or information space (map or document). We have suggested halos as a potential solution to this problem, by at least providing a visual cue to enable a user to reacquire their start position.

However, in general the results, particularly for the map tasks, were encouraging. The SDAZ technique is unfamiliar as is the handheld environment to many users. However, the performances witnessed were comparable to those where a well-know form of interaction was employed. In order to improve on the standard technique, though, ways of compensating for the reduced resolution of zoomed-out views are needed.

For both document and maps one can imagine a series of semantic highlights that are introduced as the magnification-level decreases: e.g., in maps, key landmarks could be emphasised while lower-level detail is removed; and, in documents, major section headings could be presented in a larger font-size beside the thumbnail image of the page. We are exploring such orientation features in the mapping application, by adding waypoints and other markers; and, in a photo browsing system, by presenting temporal information to help users assess whether groups of photos comprise an event.

Finally our study provides some insight into the behaviour that users exhibit with a normative pan-scroll-zoom interface in the small screen context. Overwhelmingly they preferred to use multiple panning operations for two dimensional navigation, and vertical scroll operations to navigate a document. Therefore, it seems clear that scrollbars are of little utility when users wish to move in directions that are offset from the vertical and horizontal, yet are the preferred navigation tool for single dimension

information spaces. SDAZ may be a technique that is generally effective for both one- and two-dimensional navigation.

References

Alley, B., 2004. A Comparison of Speed Dependent Automated Zooming Fedback Mechanisms. Research report, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Baudisch, P. and Rosenholtz, R., 2003. Halo: a Technique for Visualizing Off-Screen Objects, Proceedings of CHI'03: Human Factors in Computing Systems, ACM Press, 481-488.

Bederson, B.B., Grosjean, J. and Meyer, J., 2004. Toolkit Design for Interactive Structured Graphics. IEEE Transactions on Software Engineering. 30(8), 535-546.

Buyukkokton, O., Garcia-Molina, H. and Paepcke, A., 2000a. Focused Web Searching with PDAs. Computer Networks (Proceedings of the 9th International World Wide Web Conference). 33(1--6), 213-230.

Buyukkokton, O., Garcia-Molina, H., Paepcke, A. and Winograd, T., 2000b. Power Browser: Efficient Web Browsing for PDAs, Proceedings of CHI'00: Human Factors in Computing Systems, ACM Press, 430-437.

Cockburn, A. and Savage, J., 2003. Comparing Speed-Dependent Automatic Zooming with Traditional Scroll, Pan, and Zoom Methods, People and Computers XVII: British Computer Society Conference on Human Computer Interaction, 87-102.

Eslambolchilar, P. and Murray-Smith, R., 2004. Tilt-based Automatic Zooming and Scaling in Mobile Devices - a State-space Implementation, Proceedings of Mobile HCI2004: 6th International Conference on Human Computer Interaction with Mobile Devices, Springer, in press.

Hart, S.G. and Staveland, L.E., 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research, In: Hancock, P. and Meshkati, N. (Ed.), Human Mental Workload, Elsevier Science, 139-183.

Hinckley, K., Cutrell, E., Bathiche, S. and Muss, T., 2002. Quantitative Analysis of Scrolling Techniques, CHI'02: Proceedings of the ACM Conference on Human Factors in Computing Systems. CHI Letters 4(1). ACM Press, 65-72.

Igarashi, T. and Hinckley, K., 2000. Speed-dependent Automatic Zooming for Browsing Large Documents, UIST 2000: Proceedings of the 2000 ACM Conference on User Interface Software and Technology (San Diego, CA), ACM Press, 139-148.

Jones, M., Buchanan, G. and Mohd-Nasir, N., 1999a. Evaluation of WebTwig — a Site Outliner for Handheld Web Access. Proceedings of the International Symposium on Handheld and Ubiquitous Computing. Lecture Notes in Computer Science. 1707(343-345.

Jones, M., Marsden, G., Mohd-Nasir, N. and Boone, K., 1999b. Improving Web Interaction on Small Displays, Proceedings of the 8th World Wide Web Conference,

Jones, M., Buchanan, G. and Thimbleby, H., 2002. Sorting Out Searching on Small Screen Devices, Proceedings of the 4th International Symposium on Mobile HCI, Springer, 81-94.

Khella, A. and Bederson, B.B., 2004. Pocket PhotoMesa: a Zooming Image Browser for PDAs, Proceedings of Mobile and Ubiquitous Multimedia (MUM 2004), ACM Press, 19-24.

Lau, C.-C., 2004. Speed Dependent Automatic Zooming for Browsing Photographic Collections on a Smartphone. Research report, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Savage, J., 2004. The Calibration and Evaluation of Speed-dependent Automatic Zooming Interfaces. MSc thesis, Department of Computer Science, University of Canterbury, Christchurch, New Zealand.

Siegel, S. and Castellan, N.J., 1988. Nonparametric Statistics for the Behavioral Sciences (2nd edition), McGraw Hill College Div.

Yee, K.-P., 2003. Peephole Displays: Pen Interaction on Spatially Aware Handheld Computers, Proceedings of ACM CHI'03: Human Factors in Computing Systems, ACM Press, 1-8.

Zhai, S. and Smith, B.A., 1999. Multistream input: An Experimental Study of Document Scrolling Methods. IBM Systems Journal. 38(4), 642-651.