Speed-dependent Automatic Zooming for Browsing Large Documents

Takeo Igarashi
Computer Science Department,
Brown University
119 Waterman Street,
Providence, RI 02912, USA
+1-401-863-7651,
takeo@cs.brown.edu

Ken Hinckley
Microsoft Research
One Microsoft Way,
Redmond,
WA 98052-6399, USA
+1-425-703-9065
kenh@microsoft.com

ABSTRACT

We propose a navigation technique for browsing large documents that integrates rate-based scrolling with automatic zooming. The view automatically zooms out when the user scrolls rapidly so that the perceptual scrolling speed in screen space remains constant. As a result, the user can efficiently and smoothly navigate through a large document without becoming disoriented by extremely fast visual flow. By incorporating semantic zooming techniques, the user can smoothly access a global overview of the document during rate-based scrolling. We implemented several prototype systems, including a web browser, map viewer, image browser, and dictionary viewer. An informal usability study suggests that for a document browsing task, most subjects prefer automatic zooming and the technique exhibits approximately equal performance time to scroll bars, suggesting that automatic zooming is a helpful alternative totraditional scrolling when the zoomed out view provides appropriate visual cues.

KEYWORDS: Navigation, zooming, scrolling, rate control, web browser.

INTRODUCTION

Navigation techniques provide a way to access vast information spaces through limited screen space. Scrolling (or panning) and zooming are fundamental techniques for freely moving around two-dimensional continuous space. Scrolling allows the user to move to different locations, while zooming allows the user to view a target at different scales. Scrolling and zooming are commonly used in computing systems, but current interfaces still have some fundamental limitations.

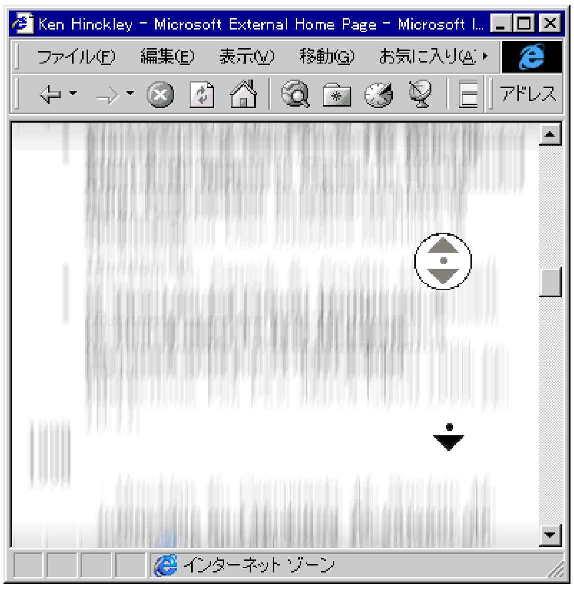

Ken Hinckley - Microsoft External Home Page - Microsoft I...
ファイル(F) 編集(E) 表示(V) 移動(G) お気に入り(A)
アドレス
インターネット ゾーン

Figure 1: The problem of rate-based scrolling. The user becomes disoriented when the document scrolls too fast. (This mocked up blur was generated using Photoshop™)

With typical scrolling interfaces, it is difficult to browse a large document efficiently. Using the traditional scroll bar, the user must move back and forth between the document and the scroll bar. This can increase the operational time and may cause significant attentional overhead. In addition, in a long document, small movement of the handle can cause a sudden jump to a distant location, resulting in disorientation and confusion. An alternative approach is a rate-based scrolling interface [22] that maps displacement of the input device to the velocity of scrolling. The Microsoft IntelliMouse™ provides a wheel for scrolling but can also map the mouse position to velocity, and the IBM ScrollPoint II™ mouse [2] maps the force exerted on a small joystick to velocity. An advantage of these devices is that the user does not have to move the mouse cursor to the scroll bar. A problem with rate mappings is that there is an upper limit for usable scrolling speed, because exceedingly fast scrolling causes disorientation (Figure 1). As a result, the user is forced to wait until the document slowly scrolls to a distant location.

Speed-dependent automatic zooming is a new navigation technique that unifies rate-based scrolling and zooming to overcome these limitations. The user controls the scrolling speed only, and the system automatically adjusts the zoom level so that the speed of visual flow across the screen remains constant. Using this technique, the user can smoothly locate a distant target in a large document without having to manually interweave zooming and scrolling, and without becoming disoriented by extreme visual flow.

We tested the idea on several prototype applications, including a web browser, a map viewer, an image browser, a dictionary viewer, and a sound editor. The Web browser with semantic zooming (Figure 6) was particularly appealing to test users . An informal usability study with the web browser and map viewer showed that for the web browsing task, automatic zooming was preferred by most subjects, and it exhibited approximately equal performance time to scroll bars, even though the test users were much more familiar with scroll bars. The map navigation interface was not as successful. Overall, we feel that automatic zooming shows promise and represents a novel approach that ties together zooming user interfaces with rate-based scrolling techniques.

RELATED WORK

Several techniques have been proposed to improve the manipulation of scroll bars [1][14]. They allow the user to control scrolling speed while dragging the knob, enabling fine positioning in large documents. LensBar [13] combines these techniques with interactive filtering and semantic zooming, and also provides explicit control of zooming via horizontal motion of the mouse cursor.

Zoomable user interfaces, such as Pad [16] and Pad++ [3], use continuous zooming as a central navigation tool. The objects are spatially organized in an infinite two-dimensional information space, and the user accesses a target object using panning and zooming operations. A notable problem with the original zoomable interfaces is that they require explicit control of both panning and zooming, and it is sometimes difficult for the user to coordinate them. The user can get lost in the infinite information space [10]. Our automatic zooming interface is an attempt to smoothly integrate continuous zooming with traditional scrolling interfaces by introducing constraints between scale and speed.

Information visualization techniques, such as Fisheye Views [6], Perspective Wall [12], and the Document Lens [18] address the problem of information overload by distorting the view of documents. The focused area is magnified, while the non-focused areas are squashed but remain in spatial context. The user specifies the next focal point by clicking or panning. Our goal is to improve accessibility to large information by extending navigational techniques which use distortion-free layout.

For three-dimensional navigation, Depth Modulated Flying [20] improves traditional flying techniques by automatically adjusting flying velocity based on depth information. The system sets the velocity proportional to the distance to visible objects.The camera moves fast in a zoomed-out view, ensuring that the time to reach the target is proportional to the perceptual distance to the target in the current view.With point of view navigation [11] the user navigates by clicking on a target position on the screen, which causes the camera to fly to the target. The time to reach the target is proportional to the logarithm of the distance to the target, so that navigation speed becomes scale independent: for example, it always takes the same amount of time to halve the distance to the target, regardless of the current scale.

The particular input device used can also influence the effectiveness of rate control. An experiment on 6 DOF input control [21] showed that rate control is more effective with isometric or elastic devices, because of their self-centering nature. It is also reported that an isometric rate-control joystick [2] can surpass a traditional scroll bar and a mouse with a finger wheel [22]. Another possibility is to change the rate of scrolling or panning in response to tilt, as demonstrated by Rekimoto [17] as well as Harrison et al. [9].

SPEED-DEPENDENT AUTOMATIC ZOOMING

In this section we introduce the speed dependent automatic zooming interface. First we describe the concept, then we give a detailed description of the interactive behavior.

Concept

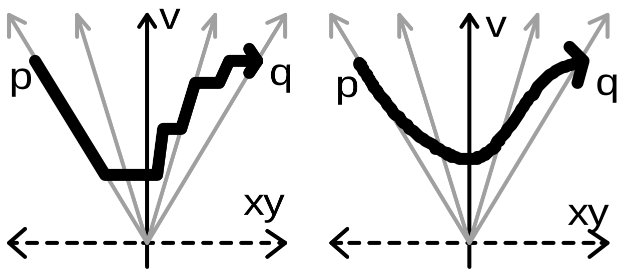
The design concept of automatic zooming is to adjust zoom level automatically to prevent extreme visual flow during rate-based scrolling. That is, the system automatically zooms out when the scrolling speed increases, and zooms back in when the scrolling speed decreases. This is consistent with our observation that users move slowly when focusing on details, but quickly when focusing on the global overview. The corresponding mathematical concept is to adjust the scale based on the following equation:

$$\texttt{scale = constant / speed} \quad (1)$$

However, if the speed is smaller than a predefined threshold, then `scale=1`. This relationship ensures that perceived scrolling speed on the screen (the visual flow of the document across the screen) remains constant regardless of the actual scrolling speed in the information space.

The goal is to provide automatic zooming so that the user can move to a target position quickly without becoming annoyed or disoriented by extreme visual flow. In addition, it should also provide a smooth transition between the magnified local view and a global overview during typical navigation tasks. The user zooms out to a global overview, identifies the target, and then can zoom in on it efficiently without having to manually change the document magnification factor.

The efficiency of the navigation using automatic zooming can be explained by the smooth curve-shaped pan-zoom trajectory in Figure 2, which is a space-scale diagram [8]. In traditional manual zooming interfaces, the user has to interleave zooming and scrolling (or panning); thus the resulting pan-zoom trajectory forms a zigzag line. In the automatic zooming interface, the zoom level changes smoothly according to the scrolling speed, and thus results in a smooth curve in the space-scale diagram.



a) Manual zooming/panning. b) Automatic zooming.

Figure 2: Space-scale diagram [8] of the pan-zoom trajectory. An efficient pan-zoom trajectory results from speed dependent automatic zooming. (v=scale, xy=space, p=initial position, q=target position)

Implementation Issues

The concept presented above is a simple design intuition, but we have found that a straightforward implementation of the idea causes several problems in actual operation. This section describes some implementation issues of the interactive behavior that are necessary for an effective realization of the concept.

The first problem we observed in our initial implementation was that the change in zoom level caused by mouse movement appeared unintuitive. We initially set the scrolling speed proportional to mouse movement, and then calculated the scale based on equation (1). However, as the user increases the speed, this formula causes a sudden drop in scale at first, and then slow convergence afterwards. The problem is illustrated in Figure 3. To achieve perceptually constant scale change, we set the scale exponential to the mouse movement based on the following equation.

$$\texttt{scale} = \texttt{s0}^{\texttt{(dy-d0)(d1-d0)}} \qquad (2)$$

(`dy` indicates the mouse movement. `s0`, `d0`, and `d1` are predefined constants, representing the minimum scale, mouse movement when the zooming starts, and the maximum mouse movement, respectively.)

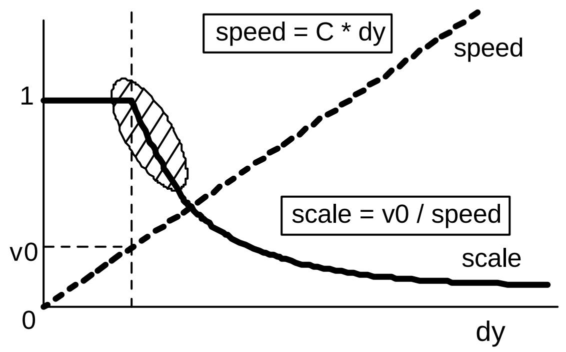


Figure 3: The original mapping from mouse position to speed and scale. A sudden drop in scale (shaded area) occurs when the user first starts moving the mouse. dy is mouse movement. v0 and C are predefined constants.

After calculating the scale based on this equation, we then calculate the scrolling speed based on equation (1) (Figure 4). Although this approach breaks the straightforward relation between the speed and the mouse position, it results in more natural interaction than the initial implementation.

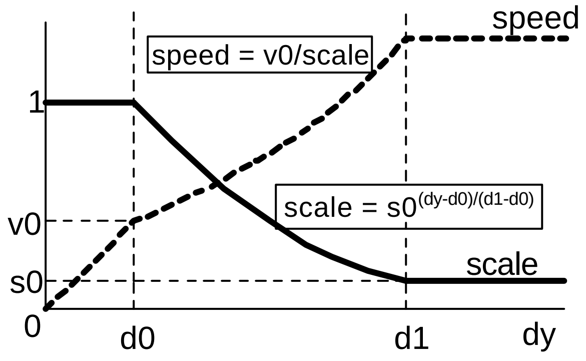


Figure 4: Revised mapping from mouse position to speed and scale. Scale changes at constant rate. `d0`,`d1`,`v0`, and `s0` are predefined constants.

Another significant implementation problem is that the document appears to "swell" suddenly if the user reverses the scrolling direction because the rate necessarily crosses zero (hence zooming in) when the rate changes signs. A similar problem occurs when the user stops scrolling by releasing the mouse button: the rate drops to zero and causes the document to instantaneously zoom in to full size.

To prevent these problems, we introduced delay to the zooming-in process. The document zooms in slowly when the user reverses the mouse direction, temporarily breaking the basic equation (1). That is, a maximum limit is imposed on the rate of change of the scale (Figure 5). Note that no delay is needed for the zooming-out process, nor would it be desired: high speed scrolling with slow zooming-out could cause extreme visual flow across the screen, which is precisely what the auto-zooming technique is intended to eliminate. Therefore, delay is only needed during special cases (reversing directions and cessation of scrolling) when zooming in may be an undesired side-effect of the `scale=constant/speed` relationship.

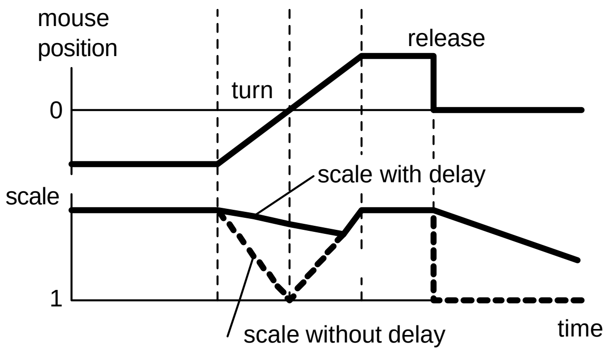


Figure 5: In a straightforward implementation, the zoom can change very suddenly (dotted line, bottom figure) in response to the mouse position (top figure). With zoom-in delay, the zoom level changes slowly (solid line, bottom figure) in response to the mouse position during the zoom-in process to prevent sudden, undesired zooming of the document.

EXAMPLE APPLICATIONS

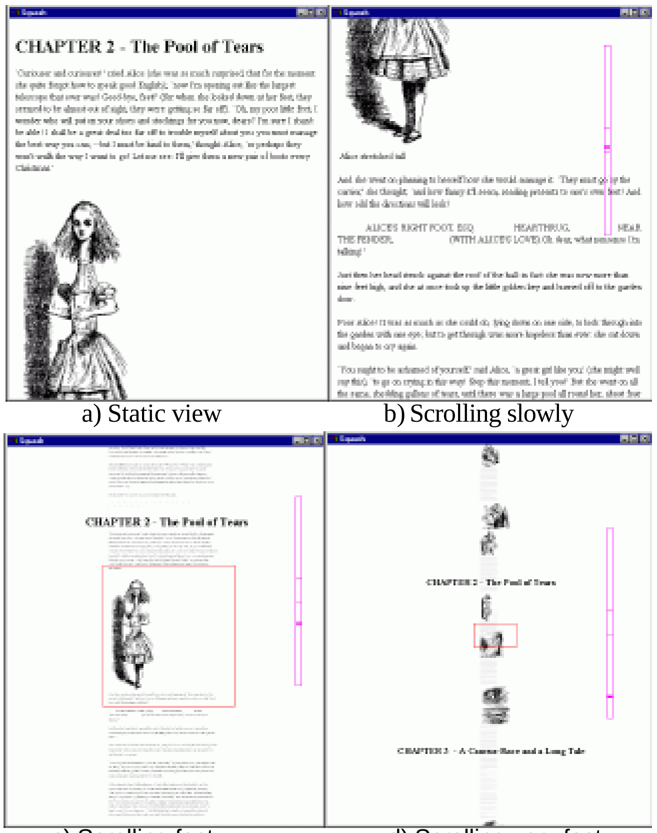
We implemented several example application systems to explore the automatic zooming technique and to clarify its strengths and limitations.

Web Browser

Existing scrolling techniques do not work well for browsing long documents with 1000 or more lines. When using scroll bars, the handle becomes too small to grab, and a small movement of the handle causes a sudden jump to a distant location. When using rate-based scrolling, the user must patiently wait until the document slowly scrolls to a distant location because fast scrolling causes visual disorientation.

We implemented a prototype web browser incorporating automatic zooming to address this problem. Rate-based scrolling with automatic zooming allows users to scroll very fast without causing disorientation, and it also provides a smooth transition to a high-level overview of the document. The user can zoom out to see the overview, and zoom in to a target sentence by controlling only the scrolling speed. Our preliminary implementation experience and usability testing of this approach suggests that it can significantly enhance the browsing experience for such documents.

In our prototype automatic zooming browser (Figure 6a), when the user presses the mouse button, a pink slider appears. The document starts to scroll when the user moves the mouse while holding the button (Figure 6b). The distance between the initial position and the current mouse position specifies the scroll speed. As the speed increases, headings of the document become more salient (Figure 6c, 6d) to give a better overview of the document structure (semantic zooming [16]). When the user releases the mouse button, an animated transition gradually returns the document to the original base scale.


CHAPTER 2 - The Pool of Tears
CHAPTER 3 - A Caucus-Race and a Long Tale

a) Static view b) Scrolling slowly

c) Scrolling fast d) Scrolling very fast

Figure 6. Scrolling a long document using speed dependent automatic zooming. The document automatically zooms out when the user scrolls fast. The speed of visual flow across the screen is held constant. Section headings and images become salient in the zoomed-out view to guide navigation.

The key to success of automatic zooming in this application is the semantic zooming feature, which provides context information during the scrolling operation. Various browsers provide zooming options, but scale typically changes only discretely, and it requires tedious manual operation. In addition, zooming typically scales the entire document uniformly, and it is difficult to locate a target in the minimized view. The semantic overview of our technique is similar to the "outline" view of word processing programs, but we provide a *smooth transition* among the different views for efficient navigation, which we believe is crucial to its effectiveness.

Our prototype browser is written in Java™. It contains a basic HTML parser, and the system uses the section headings and images detected by the parser as landmarks for semantic zooming. To improve performance, plain texts are rendered as simple horizontal lines inthe zoomed-out view. A limitation of the current implementation is that it uses the fonts (of discrete sizes) available on the system. We tested advanced zoomable UI toolkits [3,4], but the performance was unsatisfactory. System-level support for continuously scalable texts is desired for optimal use of the automatic zooming interface.

We believe this technique would work well for other applications that typically require scrolling through long documents, such as a word processor or source code editor, but we have not yet implemented these.

Map Viewer

Map viewing is a good example of an application that requires multi-scale interaction. A map typically covers a much wider area than is visible on a single screen. The user has to pan and zoom repeatedly to reach the target view.

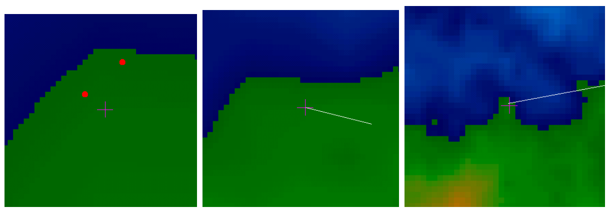


Figure 7: Map navigation using automatic zooming. The original view is on the left. When the user starts moving, the view starts to zoom out (center). The right image shows the user moving at top speed, with the view fully zoomed-out. The speed of visual flow across the screen remains constant.

In our prototype system, the user navigates through the space by dragging the mouse. The relative position between the point where the dragging operation started and the current mouse position specifies the direction and the speed of camera motion. As the user moves faster, the view automatically zooms out. The view returns to the original scale when the user releases the mouse button.

We also tested a joystick for this example. The more the user tilts the stick, the faster he moves, and the smaller the view gets. Joysticks may be more suitable for rate-based scrolling because of their self-centering effect [22]. One problem we observed with joystick input is that first-time users tend to tilt the stick as far as it will go, which causes sudden speed-up and zoom-out. Users had to learn that subtle control of the stick is required for successful navigation.

The current prototype implementation uses an artificially synthesized map based on Perlin's noise function [15] to test the idea with minimum implementation effort, as well as to achieve high performance. Although this prototype allows the user to experience zooming and panning in a multi-scale environment, an implementation using real map data would be necessary to obtain further insights. High frame-rate interaction is possible, however, as shown in [3]. It may be possible to use the constrained relationship between scale and speed for performance tuning.

Similar techniques may be applicable to other applications, such as car navigation systems, CAD systems, image editors, and spreadsheets. In a car navigation system, for example, the scale (detail level) of the map could be set based on the actual speed at which the car is moving – a high-level overview for expressway driving, or a detailed map for city street driving. Spreadsheets also seem well suited to automatic zooming because the sheet is usually larger than the screen and the user tends to visit specific cells repeatedly.

Image Browser

We implemented automatic zooming for browsing a collection of images, such as a collection of personal digital photographs taken using a digital camera [5]. The images are aligned horizontally, and the user scrolls the list of images to browse them. The user controls the scrolling speed, and the view automatically zooms out when scrolling fast (Figure 8).

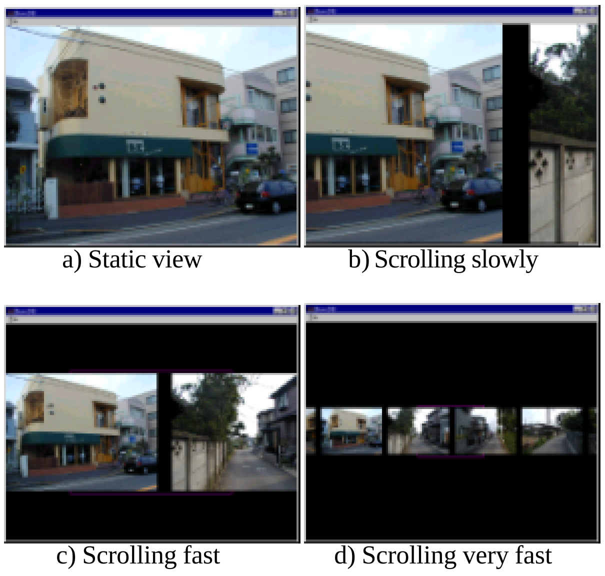


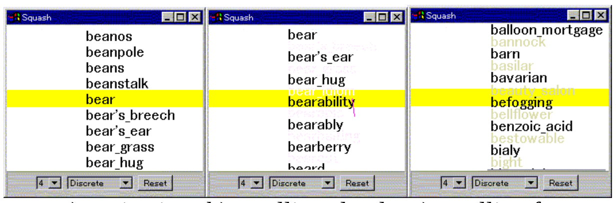
Figure 8. Image browsing with automatic zooming. The speed of visual flow across the screen remains constant.

Although this implementation is much better than simple scrolling, we felt that automatic zooming was much less effective for this application than for the Web Browser or Map Navigation examples described above. The Image Browser is different from the previous examples because abstraction may not be available here. In the Web Browser example, individual lines of text disappear, and the title headings serve as landmarks. In Map Navigation example, narrow streets fade away in the zoomed-out view, and highways and the coastline appear as landmarks. But in the Image Browser, it is typically useless to represent a set of images by a representative single image, and each image must be distinguishable. In other words, spatial abstraction is difficult to apply because the order of images is not important. As the result of this difference, with our current implementation the view cannot zoom out too much and thus the maximum scrolling speed is limited.

Automatic zooming does improve the simple scrolling interface, but a static array of thumbnails seems superior for browsing many independent images and for locating a target image among them. Automatic zooming might become more appropriate if the screen resolution and the screen refreshing rate could be significantly improved in the future, but with present systems we cannot recommend it for browsing a collection of images.

Dictionary

Zooming is a natural operation for spatial information, but it is also applicable to non-spatial, symbolic information [6]. In these cases, the zooming-out effect is achieved by thinning out less-important items. As an example of non-spatial information, we tested a dictionary viewer with automatic zooming. Words are listed in alphabetical order, and they can be scrolled vertically across the screen. As the user scrolls faster, the list starts to skip words (Figure 9).



a) static view b) scrolling slowly c) scrolling fast

Figure 9. Searching for a word in a dictionary using automatic zooming. The words start to be skipped as the user increases the scrolling speed. The scrolling speed of visible words is always constant.

The result was not very promising. It is confusing to see the words appear and disappear during scrolling. It is very difficult to locate the target word in the zoom-out view because the user has to constantly figure out the alphabetical order between the visible words and the target word. For example, when searching for "bear", the user has to steer between "bavarian" and "befogging" in the zoomed-out view, which causes significant cognitive overhead.

Sound Editor

We also tested a sound editor with automatic zooming. We expected that automatic zooming would be useful because editing an audio stream involves frequent zooming and panning operations. However, the continuously transforming waveform was just confusing. The lack of appropriate visual landmarks makes it difficult to use automatic zooming for this application. It might be useful to add visual labels to the audio stream using simple voice recognition techniques.

USABILITY STUDY

We performed a preliminary usability study to clarify the strengths and limitations of automatic zooming for the Web Browser and Map Navigation tasks. The main goal of this informal study was to obtain insights about the new interface by observing users' reactions. One of the authors sat next to each subject throughout the experiment and had brief conversations to discuss issues which arose during the study. Although our study is not intended as a formal experiment, we did take quantitative measures to obtain some initial insight about user performance.

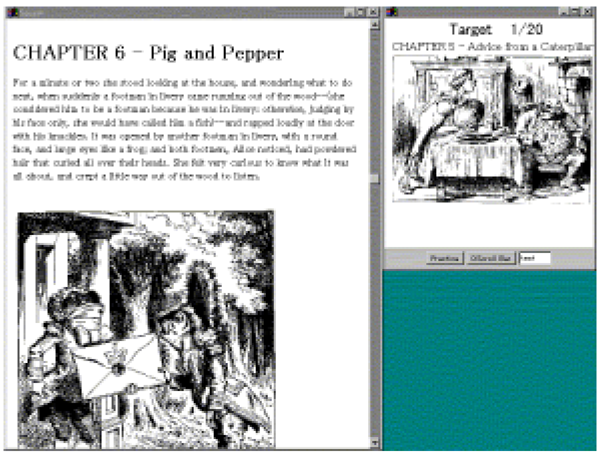
Seven test users participated in the study. All had moderate experience with computers. Table 1 shows a profile of the subjects. For each task (Web Browser and Map Navigation), users were first shown the interface, and then performed a set of practice tasks.

Table 1: Subject profile. "Computer skill" indicates the subjects' own evaluation on their computer skill. "Game play" indicates subjects' answer to the question of "how often do you play video games?".

#	Sex	Age	Computer skill	Game play
1	f	Middle	Average	Sometimes
2	m	Middle	Good	Seldom
3	m	Senior	Average	Seldom
4	f	Senior	Average	Not at all
5	f	Young	Good	Almost everyday
6	m	Middle	Average	Sometimes
7	f	Young	Good	Frequently

Web Browser

The first task requires the user to find specific images in a long web document (Alice's Adventures in Wonderland), using either the standard scroll bar interface or our automatic zooming technique (using a standard mouse for both interfaces). Order of presentation was counterbalanced. A target image is presented with the corresponding section title, and the next image appears when the user clicks the target image in the document. Figure 10 shows a snapshot of the screen. A predefined sequence of 20 images appears for all subjects. We used the same sequence in both conditions, but with reversed order to minimize learning effects (the order was balanced across the subjects).



CHAPTER 6 – Pig and Pepper

For a minute or two she stood looking at the house, and wondering what to do next, when suddenly a footman in livery came running out of the wood—(she considered him to be a footman because he was in livery: otherwise, judging by his face only, she would have called him a fish)—and rapped loudly at the door with his knuckles. It was opened by another footman in livery, with a round face, and large eyes like a frog; and both footmen, Alice noticed, had powdered hair that curled all over their heads. She felt very curious to know what it was all about, and crept a little way out of the wood to listen.

Figure 10: Snapshot from the user study (Web browser). Target image and section name is presented at the right. The user must scroll the document (left) and then click on the target image.

The resulting task completion times were approximately equal (Figure 11). This result is quite striking considering the significant difference between the two interfaces and the years of prior experience that participants had with scroll bars. In the scroll bar condition, it was difficult to find the target because an overview was not provided, but the user could jump to the target instantly if he successfully guessed the approximate target location. In contrast, users were forced to gradually approach to the target in the automatic zooming interface.

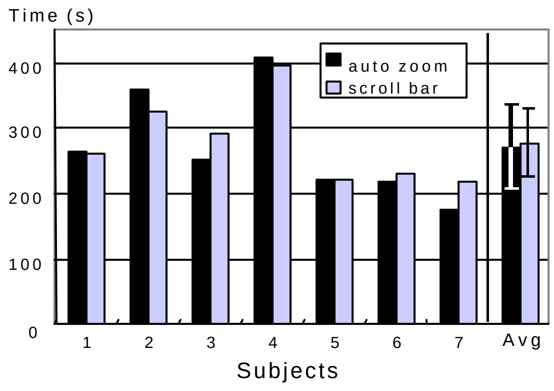


Figure 11: Task completion time (Web Browser). The performance was approximately equal.

Our main concern prior to the experiment was that automatic zooming might be too difficult to control for general users, but the subjects in our study appeared to control the automatic zooming interface fluently. The more dexterous subjects, especially frequent video game players, exhibited better performance using automatic zooming. In a subjective questionnaire, six out of seven subjects indicated preference for automatic zooming, but subject #2 preferred the standard scroll bar (Figure 12). Some subjects reported that with automatic zooming, the constant flow of the text made them dizzy.

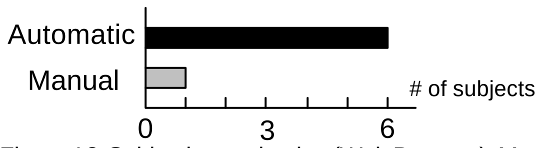


Figure 12: Subjective evaluation (Web Browser). Most users preferred automatic zooming.

This user study focused on finding visually distinctive targets. The results might have differed if different targets, such as particular sentences of text, were used as targets since they may not be recognizable when zoomed out. Simple text-based search might be preferred to find a specific sentence in an unfamiliar document. We designed our study based on the assumption that, in familiar documents, users gradually establish visual keys around known targets, and use the visual keys for navigation even when the target itself is not visually distinctive.

As a final note, the literature strongly suggests that rate-based scrolling can benefit from an isometric input [18][22], such as the miniature joystick found on the IBM ScrollPoint II mouse, as opposed to the mouse position input which we are currently using. As such, we expect that implementing automatic zooming with an isometric input device may help to improve its overall performance, although we have not yet tried this.

Map Navigation

The second task required subjects to visit targets in a two-dimensional map application using a joystick. Subjects navigated through the map using a traditional panning/zooming interface in one condition, and our automatic zooming interface in the other. In the traditional panning/zooming condition, the user used a zoom-in button and a zoom-out button on the joystick. The joystick buttons were not used in the automatic zooming condition.

A screen snapshot of the Map Navigation task is shown in Figure 13. A global radar is provided at the right-bottom corner. It indicates the location of the next target as a white dot and the current view as a red rectangle. As the user zooms out, the rectangle grows in the global radar. As soon as the user visits the target (that is, brings the target to the center of the screen), the next target appears. A predefined sequence of 20 targets was used for all subjects. The same sequence was used in both conditions, but with reversed order (the order was balanced across the subjects).

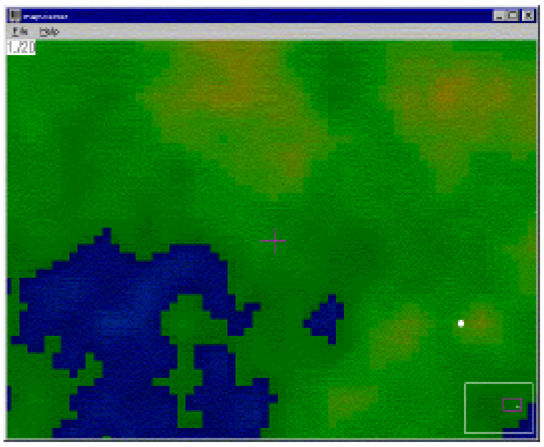


Figure 13: Snapshot from the user study (map viewer). The user navigates though the map using a joystick. The white circle indicates the target position. The global radar is presented at the right-bottom corner.

Figure 14 shows the task completion time for the map navigation task. The results were mixed compared with the previous Web Browser task. The Map Navigation task is more difficult than browsing a document, causing a range of strategies that differed widely among subjects. An efficient strategy is to zoom out until the target appears on the screen, move to the target, and then zoom in. However, some subjects slowly moved to the target without zooming out, which took a very long time. This same tendency was observed in both the traditional pan/zoom interface and the automatic zooming interface.

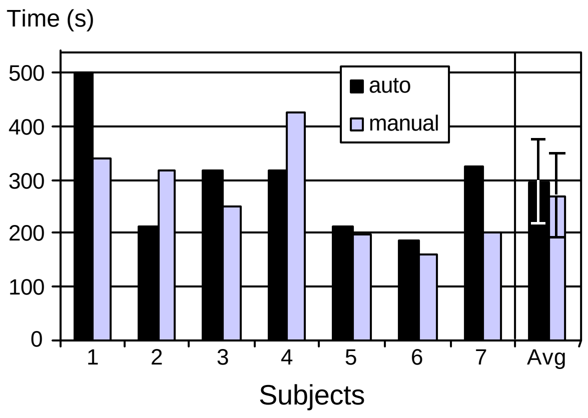


Figure 14: Task completion time (Map Navigation). The result was diverse.

The subjects' qualitative evaluations were also mixed. Four subjects (#2,4,5,6) preferred automatic zooming, while the other three preferred manual zooming (Figure 15). One subject disliked manual zooming because he had to control two different input streams, while another subject liked manual zooming because of the separate control. Several users confessed that they found automatic zooming more challenging and thus more fun to use. One user disliked manual zooming because she kept confusing the zoom in and zoom out buttons. These results suggest that for the Map Navigation task, automatic zooming may not be an optimal solution for everyone, but it can attract some users.

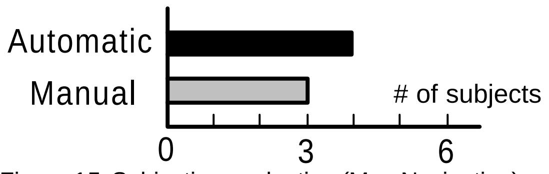


Figure 15: Subjective evaluation (Map Navigation). Users' preference was divided.

We observed that some subjects often found it difficult to stop at the target position correctly when using automatic zooming. They started to circle around the target. This can be explained as follows. When the user passes the target, he tries to go back by pulling the joystick backwards. This "going back" operation decreases the flying speed temporarily, and thus the view zooms in. As the view zooms in, the target appears to move away in screen space. This makes the user speed up too much, and he again passes the target. We had already implemented delay in the zoom-in process as described previously, but it was not enough. One solution to this problem would be to decrease the scroll and zoom speed in general to make it easier for users to control. However, as other users liked the high-speed setting, we feel that the automatic zooming interface must either adapt to the user's skill, or provide a customizable setting for the speed.

DISCUSSIONS

This section discusses potential target domains for speed-dependent automatic zooming interfaces and discusses the tradeoffs associated with the technique. Table 2, at the end of this section, summarizes these tradeoffs. What is interesting about automatic zooming is that it provides a different set of design trade-offs than traditional scrolling, while exhibiting similar performance -- in essence offering new design options for appropriate applications which require navigation of large information spaces.

The automatic zooming technique is designed for an information space of intermediate size. If the size is small enough, standard scroll bars or a set of thumbnails listed on the screen works well. On the other hand, a significantly large information space can only be navigated using search or indexing. Our technique covers the intermediate size between the two extremes. For example, documents of a few pages can be efficiently browsed by standard scrolling, books of one hundred pages can be effectively navigated with automatic zooming, and finding a particular sentence in a collection of books would require search or an index.

From our prototype implementations, we observed that the automatic zooming interface seems to works well for spatially organized information, such as a map. Web pages are also spatial in that the order of sentences, titles, and figures is essential. Landmarks in these documents provide a cue to find the desired target location in the zoomed-out view. On the other hand, automatic zooming seems difficult to apply to symbolic information such as a dictionary. Here, the spatial arrangement is not essential, and landmarks provided by the words themselves do not seem sufficient to help the user locate the target.

Our expectation is that users could benefit from the automatic zooming interface the most when they move among specific targets repeatedly. Frequent visits allow the user to memorize the spatial relationship among targets and landmarks, and the user can jump to a target without wandering around. We observed that the user's hand gradually learns an efficient speed control pattern to move to a specific target. However, the dynamic interaction of the automatic zooming can confuse first time users. Standard scrolling, zooming, and search might be the best solution for them.

We also expect that our automatic zooming interface will work best with self-centering, absolute input devices as opposed to spatial, relative pointing devices such as a standard mouse; the mechanical status of a self-centering device is directly associated with the scrolling speed. Controlling speed using a mouse can be difficult because the user has to rely on the visual feedback on the display. In previous work, self-centering joysticks have been found effective for rate-based scrolling [22].

As we observed in our user study, the automatic zooming interface is preferred by expert users with good hand-eye coordination. They liked the efficient control enabled by it, but its dynamic behavior may intimidate more novice or less dexterous users. Although the basic mechanism is easy to understand, it takes a while to become fluent with the interaction. Automatic zooming is useful for expert users because it can yield high performance with a certain amount of initial practice.

Table 2: Target domains for the speed dependent automatic zooming interface.

	Appropriate Domain	Less Appropriate Domain
Size	Intermediate	Small or huge
Type	Spatial	Symbolic, abstract
Frequency	Repetitive visit	One-time visit
Input device	Self-centering, absolute devices	Relative pointing devices
User	Experts	Novices

FUTURE DIRECTIONS

We plan to test automatic zooming in some other application domains, such as program code editors, spreadsheets, video browsing, and 3D navigation. A challenge is to design appropriate transitions from the static view to a global overview without confusing users.

Further research is required to improve the interactive behavior of the technique. We especially need to find a way to incorporate adaptation or customization mechanism to adjust various parameters to individual users.

Simple scaling causes a blank area to appear around the document in the zoomed-out view. We are considering the use of some distortion-oriented presentations in combination with automatic zooming for more efficient use of screen real estate.

We are also interested in testing automatic zooming on handheld devices, as the limited screen real estate makes standard scroll bars less effective, and many of these devices also include self-centering scrolling mechanisms which would provide an appropriate input.

CONCLUSION

We have described a new spatial navigation technique for browsing large documents that combines rate-based scrolling with continuous zooming. The basic idea is to automatically shrink the document when the user scrolls fast, thus maintaining constant perceptual scrolling speed and presentation of the global overview of the document. We also discussed various implementation issues which are essential to the interactive behavior. We implemented several prototype applications, including web browsing, map navigation, image browsing, a dictionary, and audio browsing. Our informal usability study showed that for the document browsing task, most users preferred automatic zooming to the traditional scroll bar. Dexterous users especially preferred and benefited from automatic zooming.

In general, our technique seems to work best for visually distinct data where a zoomed out view can provide appropriate scrolling cues. We believe that the idea of speed-dependent automatic zooming not only improves current rate-based scrolling interfaces, but also presents a novel interaction technique which may find application in multi-scale and 3D navigation tasks of future interactive systems.

REFERENCES

1. Ahlberg, C. Shneiderman, B., The alphaslider: a compact and rapid selector, CHI'94 conference proceedings, pp.365-371, 1994.

2. Barrett, R.C., Sleker, E.J., Rutledge, J.D., Olyha, R.S. The Negative Inertia: A dynamic pointing function, CHI'95 conference companion, pp. 316-317, 1995.

3. Bederson, B., Hollan, J., Perlin, K., Meyer, J., Bacon, D., and Furnas, G., Pad++: A Zoomable Graphical Sketchpad for Exploring Alternate Interface Physics. Journal of Visual Languages and Computing, 7, pp. 3-31, 1996.

4. Bederson, B. B., & McAlister, B., Jazz: An Extensible 2D+Zooming Graphics Toolkit in Java. Tech Report HCIL-99-07, CS-TR-4015, UMIACS-TR-99-24, University of Maryland, 1999.

5. Combs, T., Bederson, B. Does Zooming Improve Image Browsing? Proceedings of Digital Library (DL 99), pp. 130 – 137, 1999.

6. Furnas, G.W. Generalized Fisheye Views, Proceedings of CHI'86, pp. 16-23, 1986.

7. Furnas, G.W. Effective View Navigation. Proceedings of CHI'97, pp. 367-374, 1997.

8. Furnas, G.W., Bederson, B.B. Space-Scale Diagrams: Understanding Multiscale Interfaces. Proceedings of CHI'95, pp. 234-241, 1995.

9. Harrison, B., et al., Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces, Proceedings of CHI'98, pp. 17-24, 1998.

10. Jul, S., Furnas, G., Critical Zones in Desert Fog: Aids to Multiscale Navigation, Proceedings of UIST '98, pp. 97-106, 1998.

11. Mackinlay, J.D., Card, C.K., Robertson, G.G., Rapid Controlled Movement Through a Virtual 3D Workspace, SIGGRAPH 90, pp. 171-176, 1990.

12. Mackinlay, J.D., Robertson, G.G., Card, C.K. The Perspective Wall: Detail and Context Smoothly Integrated. Proceedings of CHI'91, pp. 173-179, 1991.

13. Masui,T. LensBar - Visualization for Browsing and Filtering Large Lists of Data. In Proceedings of InfoVis'98, pp.113-120, 1998.

14. Masui,T., Kashiwagi,K., Borden,G.R., Elastic graphical interfaces for precise data manipulation. CHI'95 Conference Companion, pp. 143-144, 1995

15. Perlin, K., An Image Synthesizer, Computer Graphics, Vol. 19 No. 3, 1985.

16. Perlin, K., Fox,D. Pad: An Alternative Approach to the Computer Interface, SIGGRAPH 93, pp. 57-64, 1993.

17. Rekimoto, J., Tilting Operations for Small Screen Interfaces, Proceedings of UIST'96, pp. 167-168, 1996.

18. Robertson, G.G., Mackinlay, J.D., The Document Lens, Proceedings of UIST'93, pp. 101-108, 1993.

19. Robinett, W., Holloway, R., Implementation of Flying, Scaling and Grabbing in Virtual Worlds, 1992 Symposium on Interactive 3D Graphics, 1992.

20. Ware, C., Fleet, D., Context Sensitive Flying Interface, 1997 Symposium on Interactive 3D Graphics, pp. 127-130, 1997.

21. Zhai, S., Milgram, P., Drascic, D., An Evaluation of four 6 degree-of-freedom input techniques, Proceeding of INTERACHI'93, pp. 155-161, 1993.

22. Zhai, S., Smith, B.A., Selker, T., Improving Browsing Performance: A Study of Four Input Devices for Scrolling and Pointing Tasks, INTERACT'97, pp. 286-292, 1997.