Two-Step Automatic Zooming – Further Tuning of Speed Dependent Automatic Zooming in order to Address Overshooting and Course-Correction Problems

Thomas Luan Dang
Computer Science Department,
University of British Columbia
Address here
+1-604-339-1319
dqluan@gmail.com

ABSTRACT

The two major problems that users face during usability studies of Speed-Dependent-Automatic-Zooming (SDAZ) are an ease to overshoot the target once approaching it, and the difficulty in course correcting to approach a target or to correct an overshoot. I hypothesize that the root causes of these two problems are i) Zoom-level still continuously changes as the user does fine course correcting near the target, ii) The user does not have any break during navigation, and iii) SDAZ requires considerable dexterity to be used effectively. I propose an SDAZ-inspired navigation technique and implement a document browsing application to demonstrate the technique. The new technique, Two-Step-Automatic-Zooming (TSAZ) also handles zoom-out and zoom-in automatically, but separates the two actions with a "rest period", which users need to explicitly dismiss manually. Also unlike SDAZ, TSAZ does not allow continuous changing of zoom-level during panning, but only samples a brief moment, the "intention", when the zooming out initially takes place. My contributions are to make hypotheses explaining the two major usability problems with SDAZ, and implement a solution to improve upon SDAZ. I also planned to carry out an informal usability study with the TSAZ document browsing application, but was unable to due to time constrains.

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

INTRODUCTION

Overview-detail, Zooming, and Focus-context are the three major classes of techniques that facilitate the access of vast information spaces (Cockburn et al., 2008). Two-dimensional Scrolling (or panning) are crucial in navigating any information space that is defined by both its information and its environment (Jul et al., 1998). However, for viewing the space from different scales, zooming is required to let the user navigate easily with a small screen space or rudimentary input devices. An overview-detail approach requires screen space, and a focus-context approach causes distortion in the data's visualization and requires enough screen space to be usable (Furnas, 1986). My goal in this research is to extend navigation techniques that does not cause distortions on the data and requires a minimal amount of screen real estate.

Zoomable interfaces have traditionally been implemented with manual zooming and panning. Examples in this category are numerous and ubiquitous. From map navigators such as Google Map and Map Quest, to Document Viewers such as Acrobat, and Image Viewers such as Photoshop, zooming has been a separate action from panning. Even more experimental interfaces such as Pad/Pad++ (Bederson et al., 1996, Perlin et al., 1993) handle zooming and panning separately. The disadvantages of this approach are

1. Manual zooming requires extra input (widgets for explicitly zoom in/out) and possibly extra screen space as well.
2. It may be difficult for users to coordinate zooming and panning separately

SDAZ attempts to identify and solved several problems with current scrolling/zooming interfaces. A traditional set of scroll bars require the user to move back and forth between the document and the scroll bars in order to scroll. Rate-based scrolling, where the information space scrolling speed is determined by the scrolling speed of the mouse cursor, is a potential improvement over scroll bars. However, when scrolling very fast at the detailed view of the information space, users become disoriented because the visual flow becomes excessive. Igarashi and Hinckley described this as a "motion blur" effect, and designed SDAZ to make the visual flow appear constant while keeping the high speed of information flow. SDAZ changes not only the scrolling speed, but also the zoom level, based on the mouse scrolling speed.

A related technique to SDAZ in 3D navigation is Depth Modulated Flying (Ware, 1997). Depth Modulated Flying automatically adjusts flying velocity based on depth information, moving the camera faster in a zoomed-out view. The time to reach the target is proportional to the perceptual distance to the target in the current view.

Several other works were proposed to improve upon the scroll bar paradigm such as OrthoZoom (Appert et al., 2006) and LensBar (Masui, 1998), which use displacement of the mouse cursor perpendicular to the scroll bars to control zooming. The OrthoZoom Scroller and the LensBar improved upon the scroll-bar paradigm, allowing the zooming dimension to be controlled by the mouse cursor displacement from the scrollbar. While OrthoZoom Scroller has yielded more promising results to SDAZ in usability tests, it is outside of the current scope of this paper to analyze and improve upon. By definition, OrthoZoom is a scroll bar, and is designed specifically for 1-D multiscale navigations. Also, OrthoZoom does not explicitly solve the screen space problem. I limit the scope of my research to techniques that are suitable for 2D multiscale navigation tasks.

SDAZ is a potentially more applicable and versatile solution for the zoom / pan problem, and I am fascinated by its potential as a interface for 2D-space navigation on devices with limited screen real-estate. I was also equally interested in how to improve it. In this paper, I do not aim to improve the computational or resource usage efficiency of the technique. I aim to improve the users' experience and the users' performance in common navigation tasks when using automatic zooming.

Since Igarashi and Hinckley's initial work in 2000, there have been several usability studies on automatic zooming. On SDAZ in particular, Igarashi et al. have conducted an

informal usability study of a map navigation application, and a large document viewer application. The initial results were mixed but promising. For the document viewer application, users' performance was comparable to scroll bars, despite the newness of SDAZ compared to scroll bars. For the map navigation task, the results were less promising. Cockburn, et al. also conducted a usability study on an SDAZ-based map navigation application, but using real satellite data instead of procedurally generated maps. The result was a great improvement from Igarashi and Hinckley's previous result.

The common problems identified by users during these usability studies are:

1. It is very easy to overshoot the target when approaching it
2. Course correcting near the target or after an overshoot is difficult, potentially causing more overshooting
3. The constantly moving information is disorienting

There have been several works of research on improving the performance and user experience of SDAZ-based applications. Several approaches taken involved refining the zooming speed and motion (Wijk and Nuij), and implementing an SDAZ-based map using real instead of simulated data (Cockburn et al.). Nevertheless, no research has attempted to analyze the cause of the course-correction / overshoot problem identified by the usability studies as the most major issues faced by users.

My contribution is to make hypotheses on the causes of the course-correction / overshoot problem, and to implement a solution for these problems. I hypothesize that the main causes of these problems are:

1. SDAZ continuously adjusts zoom-level in the entire navigation task, thus the fine panning to be done near the target, whether to acquire the target for the first time or to correct an overshoot, causes zoom level to change, confusing the user.
2. The user does not have any break during the navigation to stop, think, rest, or refine their action. Implicitly, SDAZ assumes that users have perfect knowledge of where they want to go, and can navigate there perfectly in one motion.
3. Since the current implementations of SDAZ utilizes a 2-D input device (a mouse) to control 3 dimensions (pan & zoom), it requires considerable dexterity to use effectively.

Two-step Automatic Zooming is a technique that builds upon SDAZ. It also unifies rate-based scrolling and zooming. However, TSAZ is only partially automatic. The zoom-out and zoom-in process is broken up by a "rest period" when the user has to explicitly advance from. This resting period allows the user time to think, course correct, and, possibly, regain some stamina from looking at constantly moving information. Also, unlike SDAZ, TSAZ only attempts to capture the initial intended zoom-level when zooming out, not constantly linking zooming to panning. This is so that the zoom-level does not change in the final phase (target acquisition) of a navigation task, so that the user feels free to perform slow, fine panning movements.

I make the assumption that automatic zooming interfaces refer to those that are controlled by a mouse with at least two buttons. However, I acknowledge that the choice of input

devices can drastically alter the feel and effectiveness of a navigation technique, and the relative performance and user experience of different techniques should be evaluate anew when a different input device is used. An experiment on 6 DOF input control showed that rate control is more effective with isometric or elastic devices, because of their self-centering nature (Zhai et al., 1993). It is also reported that an isometric rate-control joystick can surpass a traditional scroll bar and a mouse with a finger wheel (Zhai et al., 1997). Another possibility is to change the rate of scrolling or panning in response to tilt, as demonstrated by Rekimoto (1996) as well as Harrison et al. (1998).

COURSE CORRECTION AND OVERSHOOTING PROBLEMS IN SDAZ

In SDAZ, a user always starts by looking at the zoom-in, detailed view of the information space, for example, a large document. Zooming out is initiated by holding down the mouse button and start scrolling. Mouse scrolling speed is simulated by the displacement of the cursor from the initial point when the button was first held down.

After breaching a small minimum scroll speed, SDAZ will take over. Both zoom-level and panning speed and direction changes dynamically based on the displacement of the mouse-cursor. SDAZ handles panning similarly to rate-based scrolling, where a higher mouse displacement will cause a larger scroll speed in the information space. Large scroll speeds causes exceedingly high visual flow rate that overloads the users' perceptual capabilities. SDAZ mitigates this problem by zooming-out as the scroll speed increases and zoom-in as it decreases. Due to perspective foreshortening, fast scrolling seems slower from a high altitude.

MEDIUM TO LONG DISTANCE NAVIGATION PROBLEM

In a navigation task where the target is reasonably far away, SDAZ performs marvelously in the beginning, since it allows for the maximum rate of scroll while maintaining a tolerable visual flow. However, when close to the target, users inevitably have to slow down their scrolling in order to acquire the target. They also have to adjust their direction subtly as well. I believe that it requires considerable dexterity to control both the zoom level and the panning at once near the target. I hypothesize that this is one of the main causes for overshooting.

Also, when already overshooting the target, the user has to backtrack. This tends to cause a disconcerting "swelling" of the target, as the user moves their mouse in the reverse direction. Igarashi introduced a special-case delay reaction time to mitigate this problem, but the workaround only make the swelling less noticeable, not fixing the problem.

The changing zoom level and pan location at the same time is also a potential cause for error and discomfort in navigations. Graphical objects (and text) change shape and position via zooming due to perspective foreshortening. Semantic zooming can also make objects look completely different at different zoom levels. Since human visual working memory is very limited. I hypothesize that this constant changes in the display is very taxing and confusing for the user.

In previous usability studies of SDAZ, some users had adopted "coping" mechanisms in order to work around the difficulties of using SDAZ:

1. Zoom out, maintain the level of zoom toward the target, and then immediately drop straight down to minimize the amount of fine adjustments needed.
2. Zoom out just a little and scroll very slowly toward the target. This beats the purpose of SDAZ entirely.

The second coping mechanism beats the purpose of the technique entirely. The first mechanism, while still allowing SDAZ to be used effectively, assumes that the user knows relatively well where to find the target. This means that the performance of SDAZ will improve only after a period of initial browsing and familiarizing with the information space. Furthermore, it also exposes another minor problem with SDAZ.

VERY SHORT DISTANT NAVIGATION PROBLEM

When zooming in straight down. Perspective foreshortening causes the target to move away from the cursor during zoom-in. If the target ends up just outside of the screen area, a small pan has to be executed in order to acquire the target. Since SDAZ will also adjust zoom level as the user pans, the resulting change in zoom level can be visually confusing, and another cause for errors.

In a more specific application, a document viewer, very short distant panning is actually a very common operation. During the normal course of reading, a user will constantly have to move several lines to half a page at a time. He/she would still want to be able to scroll relatively fast, without any intention for zooming out. Using a technique that would rigidly assume an intention for zooming can cause unnecessary visual strain during the normal course of using applications where browsing is as common an operation as target acquisition.

TWO-STEP AUTOMATIC ZOOMING

This section describes the technique of Two-Step Automatic Zooming (TSAZ). TSAZ is my attempt to address the major problems identified above with SDAZ. I will describe the concepts and rationale behind the technique, and discuss the implementation issues. Finally, I will discuss my plan for conducting a usability test of the demo application, and suggestions for how the test could be done.

CONCEPTS AND DIFFERENCES FROM SDAZ

The design concepts of TSAZ is to take the concept of automatic zooming based on scrolling speed, which simplify the requirements for screen real-estate and input devices, then to break it up to two distinct phases, zoom-in and zoom-out.

In SDAZ, the user only initiates the zoom-out action explicitly. After that, the entire action from the initial mouse click to acquiring the target is none-stop. In TSAZ, the user also manually initiates the zoom-out process. However, a break is provided between the zoom-in process and the zoom-out process. The user will manually initiate the zoom-in process.

However, the processes themselves, both zoom-in and zoom-out, are still automatic. TSAZ still retain the major benefits of SDAZ over traditional, manual zooming. The goal that I set out to achieve with TSAZ is:

	Manual Zooming	SDAZ	TSAZ
zoom-in/out button on UI	2	None	None
zoom-in/out button on device	2	1	1
Automatic Zoom-out	No	Yes	Yes
Automatic Zoom-in	No	Yes	Yes
Manual initiation of zoom-out	Yes	Yes	Yes
Manual initiation of zoom-in	Yes	No	Yes

Figure 1: Feature Comparison between Manual Zooming, SDAZ, and TSAZ

Separating automatic zoom-in and zoom-out addresses one of the major problems identified with SDAZ. The break provides a resting period for users, which may improve the accuracy of fine course corrections near the target. In addition, the resting period may improves user experience and comfort.

The automatic-zoom-out and zoom-in algorithms in TSAZ are also different from SDAZ. When zooming-out in SDAZ, displacement (an analog for speed) is continuously sampled, and zoom-level is constantly adjusted accordingly. In TSAZ, displacement is only sampled for a brief initial period (the actual value of this constant will be discussed in the implementation). This represents the “intention of zooming”. I make the assumption that when the user wants to navigate a relatively long distance, he/she has a rough idea in mind of where the target is, and how fast he/she wants to scroll in order to navigate to the target. Not changing the zoom-level after the initial zoom-out simplify the navigation task back to panning only, and thus requires much less dexterity then controlling both panning and zooming at the same time. This simplification is beneficial when the user approaches the target and needs to adjust the panning subtly.

One of the initial motivations for SDAZ is to limit the visual flow by zooming out to a higher altitude. SDAZ keeps the visual flow constant by continuously zoom out as the user increases their panning speed. In TSAZ, visual flow is not guaranteed to be constant, because the maximum zoom-out level is fixed after the initial sampling time, and will not

change afterward until the user had zoomed in and restart the zoom-out anew. If the user made the wrong assumption about zoom-level, underestimating the necessary altitude then the visual flow will eventually increases, as the user increase panning speed after the zoom level has frozen. However, I believe that unless the zoom-level is grossly wrong, the increase in visual flow will no longer be enough to cause motion blur and confusion. Utilizing the 80-20 rule, I make another assumption that as long as the initial zoom-out (the "intention of zooming") brings the altitude within 80% of the "correct" altitude in relation to the panning speed, attempting to adjust zoom level further yields diminishing return in user experience and performance. If the user over-estimated the zoom-level, then he/she can simply decrease the panning speed. Over-estimating the zoom-level, and subsequently decreasing the panning speed, does not cause an increase in visual flow.

If the user is unfamiliar with the information space, then the probability for having an incorrect intention of zooming out increases. However, SDAZ also presents a challenge to the user when he/she is new to the information space, because adjusting altitude at the same time as panning speed and direction takes considerable dexterity. It is not clear without a usability study to prove whether the inability to change zooming level on the fly will be greatly detrimental to performance when the user is unfamiliar with the information space.

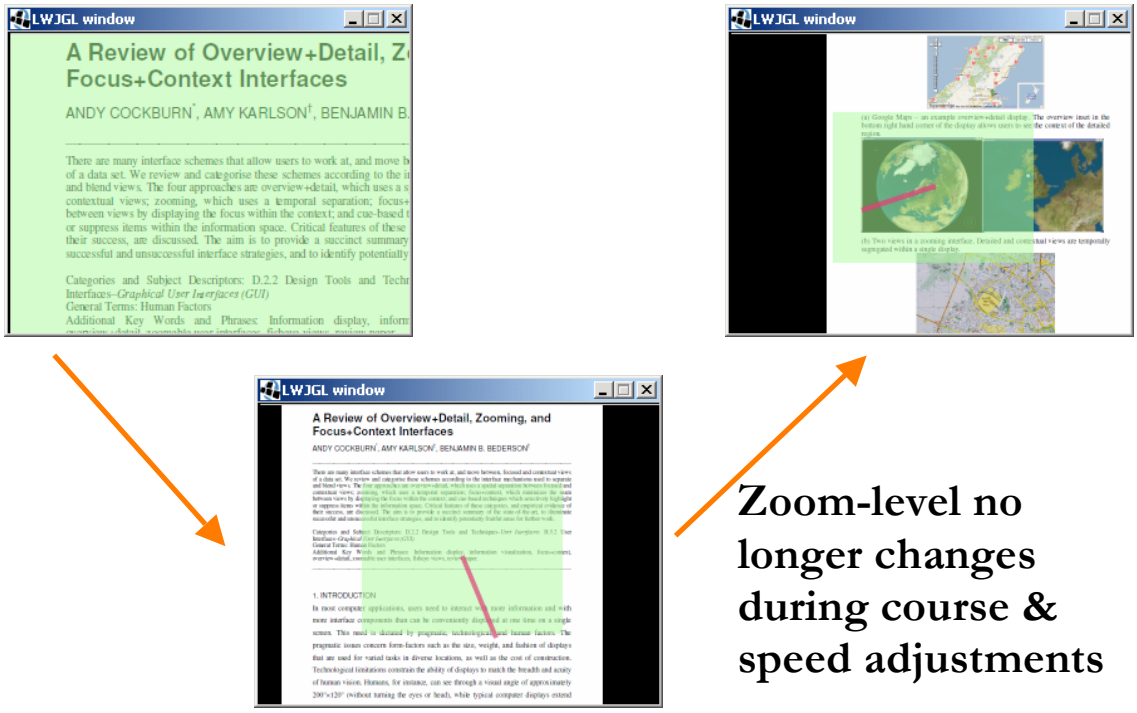


Figure 2: TSAZ only samples the "intention of zooming out", the mouse displacement over a brief moment after the user scrolls faster than the minimum zooming threshold. The zoom-level is kept constant afterward while rate-based scrolling is still in use. In SDAZ, the zoom-level is adjusted constantly based on the scrolling rate.

Zooming-in in TSAZ is initiated by the user, and is constant speed. I took the recommendation from Ware and Fleet (1997) to modulate the zoom-in speed so that the

time required to zoom-in completely is the same regardless of the starting altitude, and is roughly 1 second. I also took a convention from traditional animation, and from the work on "3-D Point of Interest" by MacKinlay et al. to slow down the zoom-in velocity in latter stage of the zoom-in process. The decision to make the zoom-in constant speed instead of speed-dependent is also to take some of the control away from the user. Perspective-foreshortening causes the target underneath the focus area (center of the viewing area) to move away, and possibly out of the viewing area. TSAZ allows the user to continue panning as the automatic zoom-in occurs, so that the user can "chase" after the target as it moves away. In similar vein, Cockburn et al. (2003) found that users' eyes naturally followed the cursor instead of the center of the screen, so he modified the zoom-in in SDAZ to center on the cursor instead. I decided to let the user chase after the target instead of automatically centering on the cursor because of an observation that I made in my personal experience with SDAZ. The cursor often moves near the edge of the viewing area, and sometimes outside of the viewing area, especially when the area is small.

In SDAZ, "chasing" after the target during zoom-in requires a lot of dexterity, because the panning speed is still being sampled during zoom-in. Moving the cursor toward the target as it moves away from the center of the viewing area can trigger a zoom-out, and subsequently, an overshoot.

The final difference between TSAZ and SDAZ is the minimum threshold required to start zooming-out. In addition to a minimum zooming threshold, I introduced another threshold for panning (rate-based scrolling) to start. I also pushed the minimum threshold for zooming-out much farther than the threshold for panning.

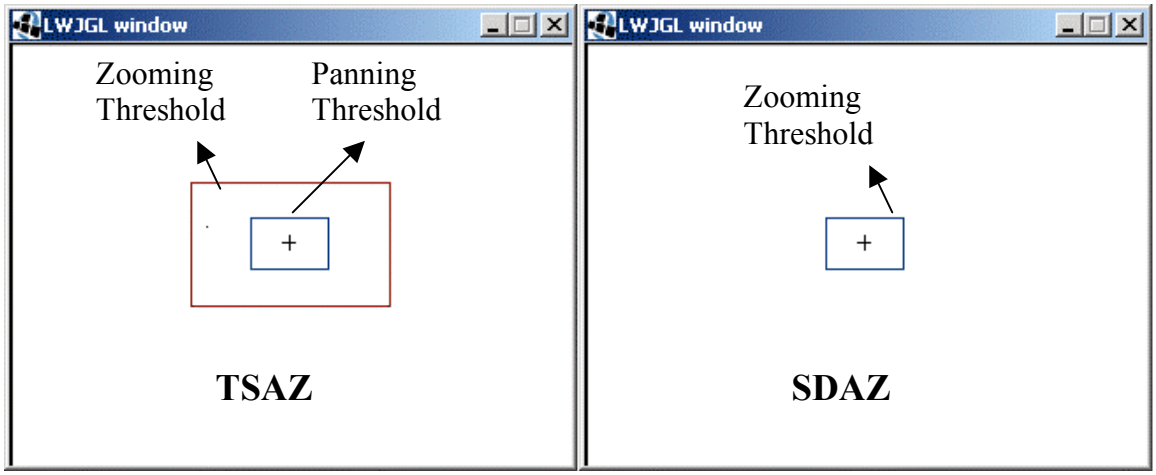


Figure 3: comparing minimum thresholds for zooming and panning between TSAZ and SDAZ.

The rationale for this modification is my realization that SDAZ is not very easy to use for very short distance navigation. When browsing a map or a document, most of the time, no zooming is required, because the distance to traverse is small (1 page of document, for example). Keeping with the theme of capturing only the intention for zooming, I decide to push the minimum displacement required to start zooming farther. Zooming is only necessary when the visual flow becomes high enough during rate based scrolling that it

becomes a problem. For short distances, rate-based-scrolling does not present a problem. In TSAZ, the minimum threshold is tuned so that the minimum level of zoom will allow the view of one entire page of document in the viewing area. The zoom-out action is still guaranteed to be smooth instead of a sudden catapult upward once the scrolling speed exceeds the minimum threshold for zooming out.

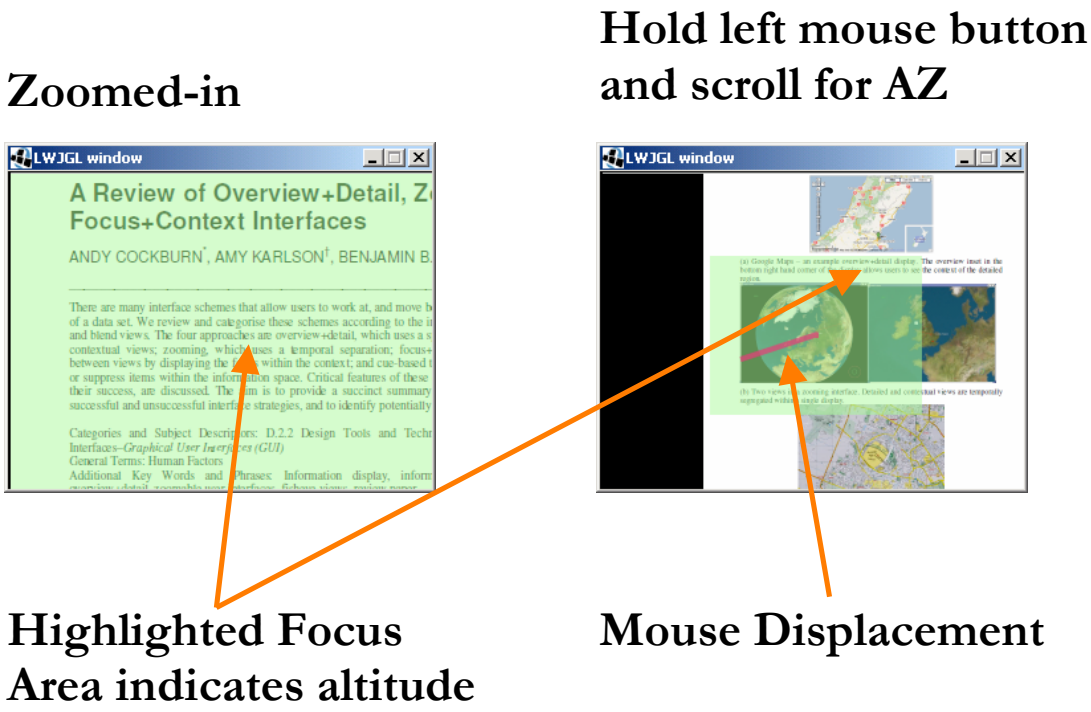
THE IMPLEMENTATION

For the purpose of discussion and demonstrating techniques, I have implemented a document viewer using TSAZ. The viewport of the document viewer is deliberately much smaller than the document, both in length and width. This is to simulate trying to read a full-sized document on the screen of a mobile device. When the viewport is smaller than the document both in height and width, most navigation tasks in browsing the document involves two dimensions, an analog to a map. The small viewport was deliberate, so that one application can serve to demonstrate two types of real world problems.

I also planned to implement a map viewer using TSAZ. However, I faced considerable difficulty trying to import the very large satellite image data, so I could not finish that demo application at the time of this paper's writing.

The demo is programmed in Java, with an OpenGL graphics engine. OpenGL is bridged into Java using the Lightweight Java Game Library (LWJGL). I did not have time to import and render actual document file formats (such as PDF and HTML), so I create .png images of the pages of a document. The demo can be run on Mac OS 10.4 and above and Windows 32, provided that the graphics hardware supports OpenGL; and Java VM v1.5 or higher is present. The demo renders smoothly at 60fps on my test hardware, with a 1.6 GHZ Intel Atom processor, and 2GB of RAM.

When the user starts to zoom out, the initial viewing area, where the zoom-in will be centered upon, is marked with a translucent rectangle. Mouse displacement is marked with a line from the center of the viewing area. These visual markers are provided to aid the navigation process by indicating the level of zoom and the scrolling speed, and are identical to those employed in Igarashi's SDAZ HTML viewer. I did not implement semantic zooming due to time constrains, but the page number was given to the side of each page. This page number is clearly visible even at the highest level of zoom.



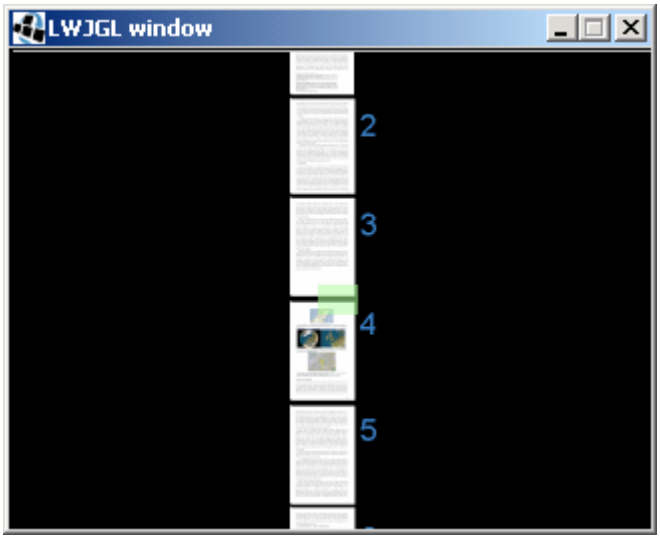


Figure 4: visual markers on the TSAZ document viewer

The demo can be downloaded at http://people.cs.ubc.ca/~luan/tsazdemo.rar

I faced several issues during the implementation

CHOOSING CONSTANTS

Capturing the "intention of zooming" is a concept that I haven't seen mentioned in my literature research, so I do not have any recommendation and convention to base my implementation upon.

1. Displacement Sampling Time: I use a constant of 0.6ms after tuning the demo application myself. This constant is chosen arbitrarily, and will need to be tuned further through usability studies.
2. Minimum Panning and Zooming Thresholds: I replaced the threshold to start zooming/panning in SDAZ with a threshold only to begin panning, then make the zooming threshold (bounding box) twice that. This constant is also chosen arbitrarily based on my own tuning of the demo application.

I believe that better, more grounded constants could have been chosen based on a closer examination of visual flow to determine the upper limit on scrolling speed before zooming becomes absolutely necessary. Some fraction (tunable) of this upper limit can then be used to determine the displacement sampling time and the minimum zooming threshold. The work by Eriksson et al. (2005) also suggests that the display area can also affect display effectiveness at high visual flow rate. The display area, therefore, is another possible factor to be accounted for in optimizing these constants. Unfortunately, I did not have time to incorporate a careful examination of all of these factors into the first version of TSAZ.

3. Maximum altitude: The maximum zoom-out level was also set arbitrarily at 8X. Due to time constrains, I have used a fixed document instead of loading arbitrary documents. The maximum altitude, and, as a consequent, panning speed, should be dependent on the size of the information space. However, for a document viewer, there is an upper limit on the useful altitude. For a world map, semantic information is rich and built into the shape of the country and continents, the terrain, and the names of location. For a document, zooming out too far without semantic zooming, and every page will look the same.

DISCUSSIONS

This section discusses the strengths and weaknesses of the Two-Step Automatic Zooming technique and its potential application.

STRENGTH AND WEAKNESSES

Two-Step Automatic Zooming addresses several major problems that users have found while using Speed-Dependent Automatic Zooming. It is similar in spirit to SDAZ, and was designed with the same benefits in mind. TSAZ uses only a two-dimension input device, such as a mouse, and one button, such as the mouse button, to control the entire zooming / panning process. TSAZ does not require any additional space for explicit zooming controls or for scrollbars. When using TSAZ, users don't have to scroll toward a widget, navigate, and return to the information space. Navigation can be initiated from anywhere on the information space.

TSAZ reduces the dexterity requirement of automatic zooming by limiting the control of zoom-level. Using a two-dimension input device, it is difficult to control panning on the X and Y axis, as well as zooming on the Z axis, at once. In TSAZ, heuristics are used in

order to capture the user's "intention" of zooming, and control of the Z axis is taken out of the user's hand as much as possible, while still retaining the simple controls of SDAZ.

TSAZ is particularly suited for applications where screen real estate is a limiting factor, and especially where input devices are simple and not very accurate. Map, image, and document viewers on mobile devices can all benefit from automatic zooming. Since the input devices on most mobile devices are inaccurate (due to price and physical space available), the lower dexterity requirement of TSAZ will be beneficial.

The most major weakness of TSAZ is the inability to change zoom-level on the fly after the initial zoom-out. While this is also the technique's strength, this limitation can challenge its performance when the user is not familiar with the information space. The user needs to complete a Zoom-Out – Pan – Zoom-In sequence before being able to repeat. If the user made the wrong assumption about the relative location of the target, he/she will waste some time restarting the navigation process.

Similar to SDAZ, TSAZ is designed to work on information spaces of intermediate size. TSAZ is designed to improve upon the useful range of SDAZ. While SDAZ is designed for the intermediate between very small and very large navigation tasks, TSAZ is suitable for both very small and intermediate navigation tasks. However, finding a particular paragraph or sentence in a collection of documents will be very difficult with a zoom / pan interface, but will be trivial with an index search.

Other limitations to this work are caused by the time constrains of this research. The document viewer does not have semantic zooming, so the maximum useful altitude is limited. The document used in the demo application is also fixed, as the capability to import and render standard document file formats such as HTML and PDF was also not implemented.

USABILITY STUDY

Finally, the greatest limitation to this work is a lack of a usability study. In the beginning, I planned to implement rich data entry and recording features in order to conduct a usability study. I planned to conduct an informal usability study comparing the performance of SDAZ, TSAZ, traditional scroll bars, and an OrthoZoom-like scrollbar in 2-D multiscale navigation. The information space would be a large document. The task would be to navigate to different parts of the document and answer "fill in the blank" questions. The data entry interface would be a multiple choice, to eliminate differences in typing speed. The control for this experiment would be the performance of traditional scroll bars. However, I did not have enough time to debug and release these feature in the demo application.

FUTURE WORK

I would like to improve the SDAZ demo in the future:

1. Finish the data entry and recording features for usability testing.

2. Implement traditional scrollbars and a version of the OrthoZoom scroll bars.
3. Implement loading of arbitrary documents.
4. Implement semantic zooming.
5. Conduct an informal usability study in order to validate the Two-Step Automatic Zooming technique.

I also would like to port the TSAZ demo to a mobile platform and conduct a usability test on this platform, where automatic zooming has the most potential. I am also particularly interested in implementing TSAZ with a touch screen interface, such as that on the Apple iPhone. Touch screen interfaces are similar to a mouse cursor, but potentially even less accurate, due to the widely varied thumb sizes of users. An automatic zooming technique with very low requirements for dexterity and accuracy can potentially be very useful.

The concept of capturing only the intention of zooming out based on the gesture of the user is a relatively novel concept. The first version of SDAZ uses an arbitrary heuristic of sampling only a brief moment of scrolling speed. However, I would like to conduct more research in devising better heuristics for capturing users intentions, so that more steps of the navigation can be done automatically.

LESSONS LEARNED

I have learned about the different factors to consider when designing zooming / panning interfaces, and the difficulties of translating a zooming technique into software. I also realized, from my own observation and my literature research, that limits to visual perception and dexterity are also important factors to consider when designing automatic zooming.

CONCLUSION

I have described a technique for automatic zooming that is an alternative to Speed Dependent Automatic Zooming. While the zoom-out action is still determined by scrolling speed, the Two-Step Automatic Zooming technique introduces the concept of capturing the intention of zooming out by sampling scrolling speed only for a brief moment. A break is introduced into the Zoom-Out – Pan – Zoom-In sequence in order to give users time to rest and adjust their navigation. Unlike SDAZ, the zoom-in in TSAZ is speed-independent, to encourage further fine panning toward acquiring the target.

The goal of capturing the intention of zooming-out and making the zoom-in speed-independent is to take away the control of the Z axis from the user as much as possible, since controlling three dimensions with a 2D input device requires considerable dexterity.

BIBLIOGRAPHY

Appert, C., Fekete, J., OrthoZoom Scroller: 1D Multi-Scale Navigation, Proceedings of SIGCHI ‘06, pp 21-30, 2006.

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.

Bourgeois, F., Guiard, Y., Multiscale Pointing: Facilitating Pan-Zoom Coordination, Extended Abstracts of CHI02: ACM Conference on Human Factors in Computer Systems, Minneapolis, Minnesota, USA, 758-759, 2002.

Cockburn, A., Karlson, A., Bederson, B., A review of overview+detail, zooming, and focus+context interfaces, ACM Computing Surveys 41(1), 2008.

Cockburn, A., Looser, J., Savage, J., Around the World in Seconds with Speed Dependent Automatic Zooming, 2003

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

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

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

Igarashi, T., Hinckley K., Speed-Dependent Automatic Zooming for Browsing Large Documents, Proceedings of UIST ‘00, pp. 139-148

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

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

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

Ware, C., Fleet, D., Context Sensitive Flying Interface, Proceedings of the 1997 Symposium on Interactive 3D Graphics, Providence, RI, 127-130, 1997.

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

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.