***Reinvisioning the Keyboard as a Spatial User Interface***

***Abstract***

We explored whether blind participants were able to use a keyboard surface interaction system named Fingers to access a shopping website at similar proficiency to using a traditional screen reader. Each participant completed four tasks with Fingers and again with VoiceOver. We logged time spent and interactions made per task, and found no clear indication that Fingers or VoiceOver was faster than the other. However, users consistently made more interactions with Fingers than with VoiceOver, but it is unclear whether this positively or negatively impacted their time. In post-study surveys, many participants said they felt learning a new mental model of a keyboard was challenging. We suggest further research into the applicability of keyboard surface interaction as an alternative form of web accessibility, but that future studies take a longitudinal nature in order to help participants learn the mental model required and overcome the novelty effect.

***Introduction***

In an ideal world, the web would be equally accessible to those with and without disabilities. Websites rely heavily on the visual presentation of content, but in an equally accessible world, different interaction paradigms would exist to make websites equally usable to those who can and cannot see them. Today, however, those with visual impairments cannot take advantage of the affordances provided by visuals on websites, and instead must rely on the underlying structure of websites’ HTML, as well as adherence to accessibility standards by developers. Current screen reader technology presents all of the information on websites as a one-dimensional stream of data, allowing users to navigate through it via the DOM tree or linearly. But the information conveyed by websites isn’t only contained within its HTML; rather, the spatial organization of content on a page and the visual relationships between different elements convey information to sighted users that the blind are unable to pick up on. In this paper we describe a finger-tracking system that mixes haptic feedback over a traditional keyboard with software that maps web content to keyboard keys. This way, blind users may be able to develop a mental model of how a page “looks” by moving their fingers across the keyboard. Information is thus presented in two dimensions instead of just one.

***Related Work***

Although the most common assistive technology for the visually impaired today is the screen reader, efforts have been made to both improve upon screen reader design and involve other interaction techniques such as haptic feedback to better communicate information to users. The creators of the Mercator system realized early on that screen reader interactions must be efficient, intuitive, and similar to those engaged in by sighted users. Just as graphical user interfaces (GUIs) are typically designed to support sighted users’ mental models, a screen reader must be designed to support the mental models of unsighted users [Edwards, Mynatt, Stockton, 1994].

Mercator is a nonvisual interaction system designed to help visually impaired users navigate UNIX workstations. Although UNIX is rarely directly interacted with anymore, much of the research into nonvisual interactions explored during Mercator’s development is in use by screen readers today. The system dynamically generates a nonvisual interface as the GUI itself populates, providing both auditory and non-auditory output. Information is grouped as nodes on a tree that is navigable via keyboard arrow keys and key combinations [Edwards, Mynatt, Stockton, 1994]. Users are able to navigate between groupings of elements as a way to scan an application page, and can delve more deeply into particular child nodes of a certain grouping. Different tones are used to indicate the presence and function of certain page elements, such as loud tones for available form fields, and muffled ones for grayed-out portions [Mynatt, Edwards, 1992].

Whereas Mercator provides a cheap, software-only way for visually impaired users to interact with computers, GUIB (Textual and Graphical User Interfaces for Blind People) consists of a comprehensive software and hardware suite. GUIB involves software, two speakers to provide spatial sound output in 2D, and an interactive Braille display that contains a representation of the screen. As a user’s screen changes, the Braille display updates. In this way, the team behind GUIB is repurposing the haptic feedback provided by Braille to create an understandable spatial context for users to work with screen elements. Users can interact with their computers by tapping different parts of the GUIB display, and are able to read certain icons descriptions as actual Braille. Other forms of information are read through the speakers. Challenges facing the success of GUIB are its high cost due to the extra hardware, as well as difficulty in communicating a screen change to users [Mynatt, Weber, 1994].

Although mice provide a difficult interaction paradigm for blind users as it is difficult to track cursor position, touch interfaces show promise. One such interface repurposes the trackpad of a MacBook computer to activate the VoiceOver screen reader upon mouse move. Such a system is a promising alternative to normal mouse interaction since users’ movements are constrained to the dimensions of the trackpad. Researchers found users typically suffer from “finger fatigue” and “fat fingers” after extended use. To remedy these complaints, the two algorithmic techniques were developed. One which implements gesture recognition for swiping between page elements, and the other to impose a drag length bound to reduce overall swipe distance. Blind users found the two types of interactions useful based on qualitative surveys, but only two participants took part in the study [Ahmed, Islam, Borodin, et al., 2010].

Microsoft Research developed a far more sophisticated system, although it was not created with the specific intention of improving accessibility for disabled users. The team built a custom mechanical keyboard that supports traditional keyboard use, but also includes a 16x4 array of IR proximity sensors. They then trained a random forest classifier on a set of hand movements over the keyboard. The keyboard and software combined to create a 3D interface used to investigate the use of gestures on or over a keyboard. The supported gestures allowed users to quickly pan and zoom documents, open up a task switcher menu, open a color wheel and select a color, and engage in whole page or by-paragraph scrolling [Taylor, Keskin, Hilliges, et al., 2014].

***Fingers Technology***

Ramos et al. developed a simpler and cheaper alternative to Microsoft Research’s mechanical keyboard in the form of a glove with infrared light emitter on the pointer finger and two Wiimotes to track its movement across a keyboard [Ramos, Li, Rosas, et al., 2015]. Their proof-of-concept finger-tracking system, named Fingers, was designed to allow users to keep their hands on the keyboard at all times, instead of moving a hand away to use a trackpad or mouse. They found users considered Fingers faster, more accurate and less uncomfortable than using a trackpad, and more accurate and less uncomfortable, but slower than using a mouse [Ramos, Li, Rosas, et al., 2015]. Similar to the mechanical keyboard, Fingers was not intended for use as an accessibility system. We repurposed the setup and code to explore whether this kind of keyboard surface interaction and haptic feedback would lend itself to blind users trying to navigate a web page.

***Non-visual Interaction Techniques Using Fingers***

**Shopping Application**

[insert screenshot of Shopping from Asus here]

We developed a simple Node.js shopping application named Shopping to test the efficacy of different interaction techniques using Fingers in a normal context. Shopping followed common eCommerce website design patterns including sidebar menu containing categories, search, grid product view spanning multiple pages and individual product pages. We did not spend a significant amount of time working on the visual design of Shopping since participants would be unable to see it. A MongoDB database contained pre-selected products from the Etsy API for each category, but searches made fresh requests to the service.

As categories and products were loaded into the client side, JavaScript worked to both populate the main user interface (UI) (one which would be visible to sighted users), as well as an interaction layer (IL). The interaction layer was comprised of boxes whose size and location on the screen were based on the screen resolution of the laptop used, and the physical size of the keyboard keys. Each UI element and its corresponding IL box contained information about a category or product. The information was presented as text within the UI element, and saved as a data attribute within the IL box. Every box corresponded to a particular key and boxes were arranged in rows similar to the keyboard.

**Hover-Press Interaction Techniques**

[insert photoshopped image of Fingers use here]

The two interaction techniques supported were hover and press of keys. Since Fingers tracks pointer finger movement as if it were the mouse, we were able to add hover handlers to each interface layer box that activated a simple screen reader for that particular item. Our “screen reader” used with Fingers fed boxes’ content through the Web Speech API. Pressing a key would activate it and allow the participant to navigate to a different part of Shopping. Although the side menu and product grid were presented in a vertical fashion visually, our virtual representation rotated them 90 degrees. This way, participants could hover over the number keys to listen to product categories listed on the menu, and press a number key to filter products by that category. Similarly, hovering over an alphabetical key would cause Shopping to read the title and price of products, and pressing an alphabetical key would direct the user to the product’s unique page.

One of our interaction design goals was to provide participants with a high degree of mobility on Shopping without sacrificing their understanding of where their hand lay on the keyboard. To that end, we restricted the number of hover-press keys in each number and alphabetical key row to the first six. In other words, only keys 1-6, Q-Y, A-H, and Z-N accessed category or product information.

In order to accelerate prototype development, we decided to program Shopping entirely in the client-side as a single-page application. As such, it did not support normal backwards navigation, so participants must hover over the grave accent (`) key to hear the word “Back”, and press that key to go to the previous page. In the interest of logging participant interaction data, we later built a proper Node.js application with a MongoDB database behind the client-side app.

**Other Interaction Techniques**

Exceptions to the hover-press interaction rule included viewing additional pages of product grids, accessing product page information, and search. Participants could access additional product grid pages by pressing U, J, or M on the keyboard. We chose those keys in order to convey a sense of “going past” the current selection of products. Hovering over each of these keys caused Shopping to inform the user of its purpose.

Since product pages only included detailed information about that particular product, hovering over certain alphabetical keys would cause Shopping to read that information to a participant, but pressing a key did not activate anything. However, product pages retained the side menu, so hovering and pressing of number keys remained the same.

Pressing the Shift key accessed Shopping’s search bar. Once a participant focused on the search bar, all hover and press interactions were disabled to prevent confusion while typing; otherwise, the application would constantly read product information and visit product pages as a participant typed. Instead, Shopping read each key typed by the participant. Participants could press Shift again to exit the search bar, or Enter to filter products by that particular query.

***Experiment Method***

**Participants and Apparatus**

We ran studies with five female participants aged 24 to 61 with varying degrees of blindness. Four of the five have congenital blindness; the other has *retinitis pigmentosa*, a hereditary disorder. Instructions for the study were read aloud. Participants wore the right-handed Fingers glove and sat approximately two feet from a laptop, with a hand resting on the keyboard. The Fingers tracking system was placed behind the laptop with the Wiimotes facing downward in order to sense the glove’s infrared light emitter.

**Procedure and Design**

The study required participants to complete a series of four tasks twice; once using Fingers, and once using their screen reader of choice. A video camera capturing participants’ hands and the keyboard was set up. Before beginning any tasks, participants were given the opportunity to navigate around the shopping application in order to develop some familiarity with Fingers while the researchers took notes on their style of use.

The four tasks were designed to both emulate normal shopping website use cases as well as fully explore the range of interactions supported by Shopping.

*Task 1*

Search for the word “apple”, listen to three products, and go to the page of the third product.

*Task 2*

Type “dog” into the search bar, but instead of searching, go the “Accessories” category and then go back a page.

*Task 3*

Listen to two products, go to the page of the second one, and find its category or categories.

*Task 4*

Got to the “Bath and Beauty” category, listen to the first product, go to the next page, and listen to that page’s first product.

After each participant completed the tasks using Fingers, she opened a copy of the same shopping application sans Fingers interaction layer, and performed the tasks again using VoiceOver on a provided laptop. Since most participants were unfamiliar with VoiceOver, we walked them through an introduction to basic key commands. Participants were still required to press the grave accent key to return to a previous page.

***Results and Analysis***

**General Results**

**Time**

We were interested to see how quickly participants were able to complete tasks using Fingers versus VoiceOver, as well as how many hover or key press interactions they made per task. The table below shows average time per task in seconds across all participants. Participants averaged faster times on the first and third tasks with Fingers, and faster times on the second and fourth with VoiceOver. Average speeds for the first task with both Fingers and VoiceOver were significantly slower than for proceeding tasks, and analysis of our videos shows participants felt more hesitation in the beginning, but tended to gain a better understanding of the interactions as time went on.

Every participant’s screen reader of choice was JAWS, so although the mental model for VoiceOver was similar, they still needed to learn how to use a new screen reader. Application development decisions made early on to speed prototyping and iteration later made it impossible to have participants complete the tasks with their own laptops. It is highly likely that participants would have completed each of the four tasks far faster with their own setup than with VoiceOver, but we were unable to collect that data.

Average time per task (in seconds)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Task 1 | Task 2 | Task 3 | Task 4 |
| Fingers | 157.258 | 131.774 | 81.683 | 146.019 |
| VoiceOver | 196.705 | 95.368 | 119.493 | 126.652 |

The chart above shows that average time may not be the most accurate measure of time for tasks 1 and 4 due to significant outliers by participants 2 and 4. Participant 2 had a tendency to become distracted during each task and take her hands off of the keyboard. Each time she did so, she required an additional reorientation period upon returning to the task, and we estimate this added approximately 35 seconds to her first task with Fingers, 10 seconds to her third task, and 40 seconds to her fourth task. With VoiceOver, the distractions added roughly 30 seconds to her first task and 15 seconds to her third.

The chart also shows data is missing for many participants. Participant 1 did not complete her fourth Fingers task or any of the VoiceOver tasks due to technical issues. Some participants were unable to complete their fourth VoiceOver task due either to time constraints or data loss caused by a finicky Internet connection.

Despite the setbacks caused by technical issues, we found that participants were significantly faster using either Fingers or VoiceOver for task 1, but that pattern didn’t hold true for the proceeding tasks. Additionally, participants using Fingers tended to complete tasks 2 and 3 more quickly and with less variation than for other tasks. These lower times also held true for completing tasks 2 and 3 with VoiceOver, although there was far more variation between participants.

Participant 3 completed most tasks with Fingers and VoiceOver faster than any other participant. We believe this is because although legally blind, she could still make out shapes on the page (but not text) and this gave her a better mental model of the page layout than the other participants were able to develop.

**Interactions**

Screen reader use is typically marked by rapid pressing of various shortcuts and key commands. As such, we expected participants to engage in fewer overall hover and press interactions with Fingers than press interactions with VoiceOver. However, upon submission of each task, participants required time to reorient their hands on the keyboard. During this reorientation, multiple hover interactions were activated as participants’ fingers brushed along the keys. Notwithstanding the reorientation process, we believe Fingers required more interactions on average per task than did VoiceOver.

Average number of interactions per task

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Task 1 | Task 2 | Task 3 | Task 4 |
| Fingers | 43.2 | 40.2 | 29.0 | 70.8 |
| VoiceOver | 21.3 | 7.0 | 15.5 | 28.5 |

The chart above makes it clear VoiceOver required far fewer interactions than Fingers. It’s difficult to identify a clear relationship between time and number of interactions as each participant worked through the tasks in her own way. When stuck, participants tended to either navigate around the page or to stay still. The best example of this is comparing the performance of participants 1 and 2 on Fingers task 1. Participant 1 completed this task approximately twice as fast as participant 2, but logged almost twice as many interactions. Similarly for VoiceOver, participant 2 took far longer on task 1 than did participant 5, but logged approximately one third as many interactions.

**Participant Results**

Although generally there was no discernible correlation between time and interactions, analysis of individual participants’ results show interesting relationships on a case-by-case basis. We are not including charts for participant 1 since she did not complete the VoiceOver portion, and thus there is nothing to compare her Fingers time and interactions against.

We don’t have data for participant 2’s performance on the fourth VoiceOver task. For tasks 1 through 3, the participant consistently logged fewer interactions with VoiceOver than with Fingers, despite taking longer to complete the tasks with VoiceOver in two cases. She worked through tasks 1 and 3 faster with Fingers than with VoiceOver, with close times for task 2.

Participant 3 displayed a much closer alignment of time and interactions for both Fingers and VoiceOver. For tasks 3 and 4 especially, the bars comparing time for Fingers versus VoiceOver with those comparing number of interactions map closely together. With the exception of the first task, she tended to take longer with Fingers than with VoiceOver.

Similar to participant 2, participant 4 logged consistently higher numbers of interactions per task with Fingers than with VoiceOver, and those relationships stayed constant regardless of which system she completed the tasks more quickly with. She took longer to complete the first two tasks with Fingers, but then completed the last two faster than she did with VoiceOver.

Comparing participant 5’s time per task with number of interactions per task doesn’t show any clear patterns. Whereas participant 4 had no instances in which she completed a task with Fingers in a comparable amount of time as with VoiceOver, and participants 2 and 3 each had one instance, participant 5 had two (tasks 2 and 3). As with participant 2, we don’t have data for participant 5’s VoiceOver performance on task 4.

**Qualitative Results and Post-Survey**

We rotated the product grid 90 degrees such that Q corresponded with the first product, A the second, Z the third, W the fourth, and so on. Despite our instructions indicating this ordering of products, every participant fell into a pattern of tracing their finger across a row of keys instead of down a column to access sequential products. It was clear that our workaround to fit more products within a keyboard by modifying the page layout was unintuitive.

After each participant completed the tasks using Fingers and VoiceOver, we asked her to rate her experience of using Fingers and VoiceOver on a Likert scale of 1 to 7. Participants rated their experience using Fingers at an average of 3.8, and their experience using VoiceOver at an average of 4.8.

We then asked participants what they liked and did not like about the glove, and suggestions for improving this kind of interaction. Many participants appreciated Fingers as novel, but didn’t necessarily consider it an ideal way to interact with the web. Some appreciated the shortcuts to go back and focus on search, as well as the haptic feedback provided by the shape of the keys. However, participants had negative comments regarding the accuracy of the finger tracking, how the glove’s material muffled the haptic feedback, and how the need to use their other hand to orient the gloved one felt like a waste of time. Participants’ suggestions for improvements closely mirrored their negative feedback.

Some participants informed us they experienced difficulty learning how to use Fingers because they already had an idea or understanding of what purposes a keyboard and its keys serve. One participant became visibly flustered because her mental model of how a keyboard works was so ingrained that she found herself unable to remember our directions and expressed confusion at how each Fingers interaction worked.

***Discussion and Conclusion***

We were curious going into this study if the benefit provided by creating an accurate two-dimensional representation of website content would outweigh the effort required to adopt a new mental model. Participants across age groups showed a range of adaptability, but due to the ingrained nature of their keyboard mental models, it’s impossible to tell from a first impression whether a keyboard surface interaction system such as Fingers will improve web usability for the visually impaired.

We suggest future web accessibility studies involving Fingers are of a longitudinal nature. This way, participants will overcome the novelty effect of the technology, and researchers will begin to determine whether the affordances of a haptic interface modeled after actual websites outweigh the existing benefits provided by screen readers.

The Fingers apparatus itself could use much improvement. Though it is already low-cost, its finger-tracking accuracy could be greatly improved via construction of a purpose-built stand. Additionally, replacing the glove, infrared emitter and Wiimotes with a camera using machine vision to track a user’s (gloveless) fingers would improve haptic feedback and reduce the hassle associated with putting on a wearable to interact with one’s computer.

**References**

Ahmed, F., Islam, M. A., Borodin, Y., & Ramakrishnan, I. V. (2010). Assistive Web Browsing with Touch Interfaces. *ASSETS '10 Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, 235-236. <http://dx.doi.org/10.1145/1878803.1878848>

Edwards, W. K., Mynatt, E. D., & Stockton, K. (1994). Providing access to graphical user interfaces—not graphical screens. *Assets '94 Proceedings of the first annual ACM conference on Assistive technologies*, pp. 47-54. <http://dx.doi.org/10.1145/191028.191041>

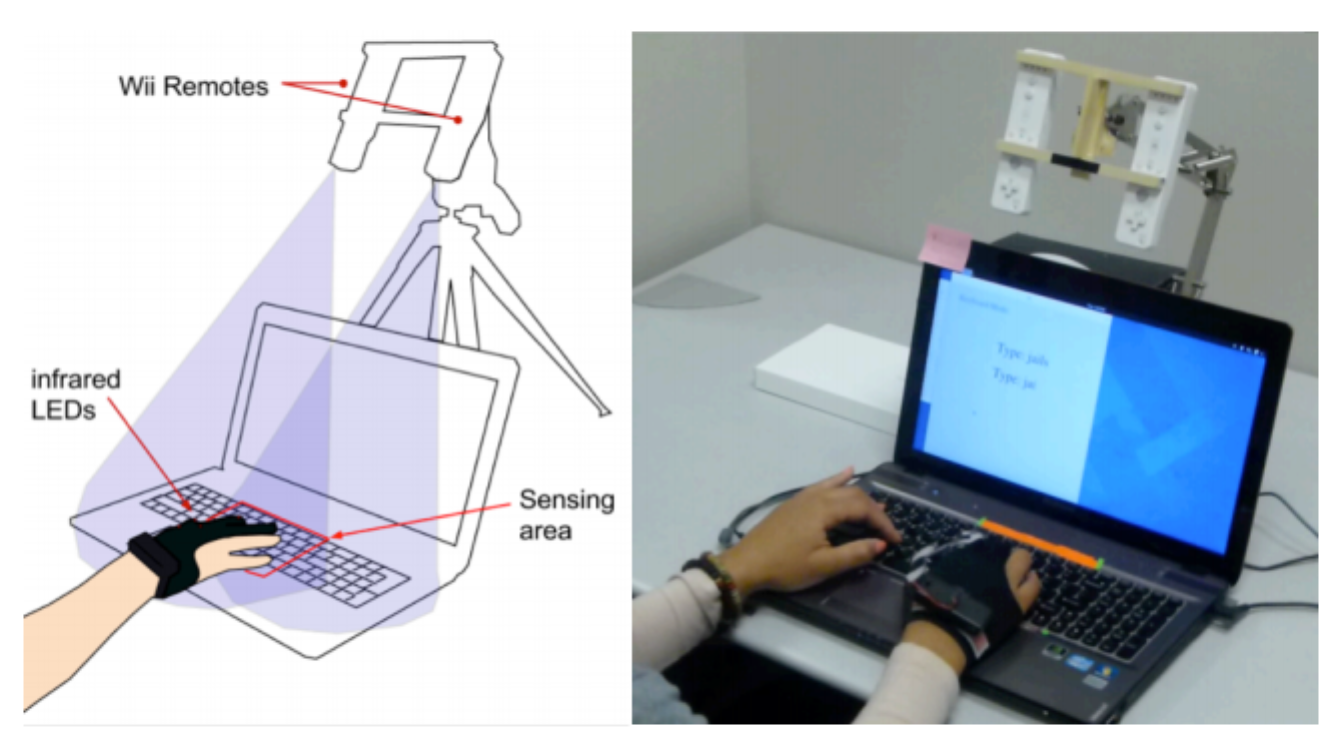
Mynatt, E. D., & Edwards, W. K. (1992). Mapping GUIS to Auditory Interfaces. *UIST '92 Proceedings of the 5th annual ACM symposium on User interface software and technology*, pp. 61-70. <http://dx.doi.org/10.1145/142621.142629>

Mynatt, E. D., & Weber, G. (1994). Nonvisual presentation of graphical user interfaces: contrasting two approaches. *CHI '94 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 166-172. <http://dx.doi.org/10.1145/191666.191732>

Ramos, J., Li, Z., Rosas, J., Banovic, N., Mankoff, J., & Dey, A. (2015). Keyboard Surface Interaction: Making the keyboard into a pointing device. *arXiv*. arXiv:1601.04029

Taylor, S., Keskin, C., Hilliges, O., Izadi, S., & Helmes, J. (2014). Type-hover-swipe in 96 bytes: a motion sensing mechanical keyboard. *CHI '14 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1695-1704.

Anything else?

* In general, I need to improve the transitions and flow
* Give concrete examples of how using fingers works
  + Ask Ben or Josh if they’re willing to make me a diagram of it a la (right)
  + On the other hand, why can’t I just take a picture (aerial) of someone using it and then draw arrows near their finger?
  + 
  + (Maybe some motion included, so arrows indicating back/forth motion)
* Take aerial photo of someone using Fingers on a mac or my asus and add some arrows indicating movement (a la left)
* When I edit, make it clear that a faster task time is ideal, but it’s less clear whether more or fewer interactions is better.
* Also, make sure my language supports the fact that each participant completed the same tasks with both Fingers and VoiceOver. I explain this in my Experiment Design section, but my phrasing in results doesn’t really support it