Efficiency or Quality of Experience: A Laboratory Study of Three Eyes-Free Touchscreen Menu Browsing User Interfaces for Mobile Phones

Vladimir Kulyukin*,[1], William Crandall[2] and Daniel Coster[3]

[1]*Department of Computer Science, Utah State University, Logan, UT, USA*

[2]*Smith-Kettlewell Eye Research Institute, San Francisco, CA, USA*

[3]*Department of Mathematics and Statistics, Utah State University, Logan, UT, USA*

Abstract: A growing number of individuals who are blind or visually impaired is using smartphones in their daily activities. The touchscreen is a standard component of smartphones. While benefitting people with low vision by enhancing control of the text style and color and the size of images and text, the touchscreen has the downside for visually impaired users in that physical buttons for input of command selection and text entry are replaced with the touchscreen's soft buttons. To overcome this limitation, we are investigating eyes-free approaches to using the smartphone's touchscreen for information browsing. In this article, we present a laboratory study of three eyes-free touchscreen user interfaces for browsing menu hierarchies. Our findings indicate that quality of experience and familiarity may be as important as the time efficiency of completing tasks.

Keywords: Mobile computing, eyes-free computing, touchscreen gesture interfaces, blindness, visual impairment.

1. INTRODUCTION

A growing number of individuals who are blind or visually impaired (VI) are using smartphones in their daily activities [1]. Smartphones provide an impressive compliment of features in a compact, portable form factor suitable for executing or contributing to real-time wayfinding tasks. GPS with Geographical Information System (GIS) map data is included with most, if not all, smart phones. The "open" architecture of the Android operating system allows inexpensive, special purpose applications ("apps") to be readily designed and distributed. In addition to two-way voice and data communication, many provide effective text-to-speech (TTS) and moderately effective speech-to-text (STT) functionality as user interface (UI) options.

The touchscreen is a standard component of smartphones. Manufacturers prefer "soft" controls (for such functions as keyboards and pushbuttons) to hardware ones as the touchscreen replaces expensive and less reliable mechanical parts and reduces the device footprint. Consumers, for the most part, find touchscreens efficient. Therefore, one should expect to see relatively more touchscreens on phones with the possibility that mechanical controls may disappear altogether. While benefitting people with low vision by enhancing control of the text style and color and the size of images and text, the touchscreen has the downside for VI users in that physical buttons for input of command selection and text entry have been replaced with the touchscreen's soft buttons. Although replacing the mouse by voicing the area of finger contact on the screen is available and shortcuts in typing through word completion are quite effective in reducing the number of keystrokes necessary to perform a text entry task, real time manipulation of these interface options are inefficient in the context of using the smart phone for wayfinding when on the street. Furthermore, the current convention in web access is to drill down the hypertext-linked, hierarchical menus where text entry is minimized and rapid scanning of text at each level is desirable.

Toward this end, we are investigating eyes-free approaches to using the smartphone's touchscreen and trackball for gesture control over information navigation, thus giving blind and VI users access to information that is either stored on the smartphone or brought into the smartphone through its wireless connection. Such systems are useful in a variety of contexts ranging from accessible shopping [2] to remote infrared audio signage (RIAS) [3] to indoor and outdoor navigation [4].

Our approach complements and draws on previous and current research on touchscreen accessibility. The *Slide Rule* interface [5] provides several accessible multi-touch interaction techniques for touchscreen interfaces for browsing lists, selecting items, and browsing hierarchical information. The *EarPod* system [6] provides access to hierarchical audio menus through a circular touchpad. The *Talking Fingertip* technique [7] allows blind and VI users to scan touchscreens and hear the descriptions of the items on the screen. In the *Talking Tactile Tablet* [8], a stylus can be used to explore two-dimensional space and receive feedback through speech and a tactile overlay. The *Touch 'n Talk* system uses speech and tactile overlays to enable the users to edit text documents. As an alternative to the touchscreen, the

*Address correspondence to this author at the Department of Computer Science, Utah State University, Logan, UT, USA;
Tel: (435) 797-8163; Fax: (435) 797-3265;
E-mails: vladimir.kulyukin@usu.edu, vladimir.kulyukin@gmail.com

BlindSight system [9] uses the phone keypad to access speech menus. We endorse the research objectives of these approaches and contribute to them by first focusing on the efficacy of simple, single finger methods appropriate to browsing information while 'on the go'.

We took the measure of the complexity of a browsing task to be the minimum number of UI actions required to transverse a tree in order to reach the answer to a particular query. By evaluating subjects' behavior in solving various eyes-free search navigation problems using three modes of finger gestures (UIs), we provide a useful model for evaluating other information browsing methodologies. The project was implemented on the Google *Nexus One* smartphone running Android 2.2. The Android OS was selected to maximize potential public impact. Unlike the proprietary iPhone OS, Android is open source, can be programmed with standard programming languages (Java, C/C++), has emulators and integrated development environments (IDEs) for Windows, Linux, and Mac OS, requires no mandatory developer fees, which is fundamental to scientific knowledge sharing, and is gaining momentum in the mobile phone accessibility literature [10]. We hope that the end product will be an open source solution for all Android platforms such as Motorola BackFlip, Sprint's HTC EVO, Google Nexus One, Google Droid, and Google Droid Incredible).

This remainder of the article is organized as follows. In Section 3, each UI is described in detail. The browsing tasks and methodology of measuring their complexity are presented in Section 4. The design of our experiments is described in Section 5. Section 6 gives our results. In Section 7, the results are discussed. A summary and conclusions are presented in Section 8.

2. EYES-FREE USER INTERFACES

To evaluate the suitability of touchscreen interfaces for eyes-free information browsing on smartphones, we have designed and implemented three UIs. All three UIs are based on the common DPAD metaphor (*up*, *down*, *left*, *right*, *select* gestures) familiar to many smartphone users. The UIs are designed for browsing single-inheritance hierarchies, i.e. trees where each node, except the root, has exactly one parent and the root node has no parent (See Fig. **1**). This type of information organization is very common and can encode many semantic relationships from standard menu hierarchies to sophisticated XML ontologies. We make no claims about the generalizability of our interfaces to other knowledge encoding structures. A standard menu layout of the tree structure is assumed, with the root being the leftmost node and the leaves being the rightmost nodes, as shown in Fig. (**1**). When the focus moves to a specific node, the node is said to be *activated*. An activated node may be rendered through audio, vibration or, for low-vision users, even graphically.

2.1. User Interface 1

The first UI (UI1) is a joystick-type interface with five user operations: up, down, left, right, select. UI1 was implemented with the *Google Nexus One* trackball (small round ball at the bottom of the phone). At each activated node in the tree, the user can execute five actions:

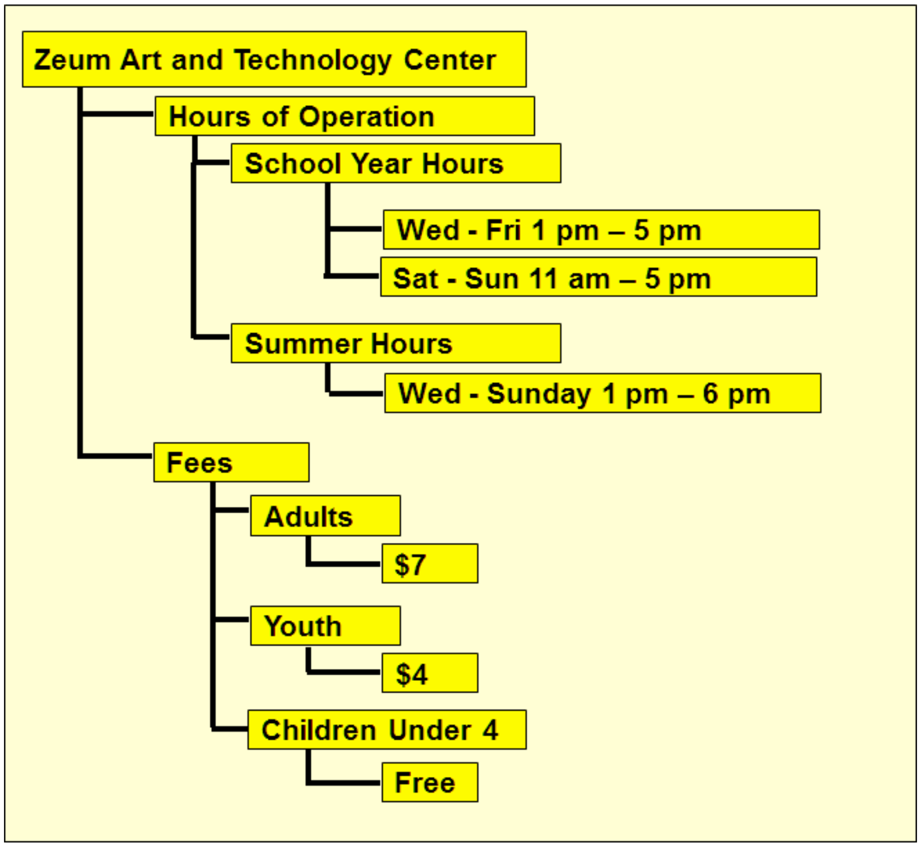

Fig. (1). Sample Menu Hierarchy.

1. **trackball-left** (move left to the parent of the current node and hear result);
2. **trackball-right** (move right to the first child of the current node and hear result);
3. **trackball-up** (move up to the previous node on the same level and hear result);
4. **trackball-down** (move down to the next node on the same level and hear result);
5. **trackball-tap** (select current node).

For an example of how UI1 works, consider the sample hierarchy in Fig. (**1**). Suppose the user starts at the root node *Zeum Art and Technology Center* and hears the spoken message "Zeum Art and Technology Center" through the TTS engine. If the user executes **trackball-left**, the user hears the spoken message "No parent node," because the root node has no parent. If the user executes **trackball-right**, the focus moves right to the node *Hours of Operation*, which becomes activated, and the user hears the spoken message "Hours of Operation." If the user executes **trackball-up**, the user hears "No previous node," because there is no node upward of *Zeum Art and Technology Center* on the same level of the hierarchy. If the user executes **trackball-down**, the user hears "No next node" as there is no node below the root node. If the user executes **trackball-tap**, a specific action associated with *Zeum Art and Technology* is executed by the system. In our current implementation, the user hears the spoken message consisting of the text associated with the node in a database. These UI1 actions and their effects are summarized in Table **1**.

Table 1. UI1 Actions and Effects at Zeum Art and Technology Center Node in Fig. (1)

UI1 Actions	Effects
trackball-left	Message "No parent node."
trackball-right	*Hours of Operation* is activated; Message "Hours of Operation."
trackball-up	Message "No previous node."
trackball-down	Message "No next node."
trackball-tap	Action associated with node *Zeum Art and Technology Center* is executed.

Suppose, to continue with our example, the user is at the root node and executes **trackball-right** to move to *Hours of Operation*. The UI1 actions at *Hours of Operation* and their effects are summarized in Table **2**.

2.2. User Interface 2

The second UI (UI2) is a one finger touch gesture interface with five gestures:

6. **finger-left** (user moves finger left on touch screen);
7. **finger-right** (user moves finger right on touch screen);
8. **finger-up** (user moves finger up on touch screen);
9. **finger-down** (user moves finger down on touch screen);
10. **finger-tap** (user taps finger on touch screen).

Suppose the user again starts at the root and hears "Zeum Art and Technology Center." The UI2 actions and their effects are summarized in Table **3**.

Suppose the user executes **finger-right** to move to *Hours of Operation*. The UI2 actions and their effects when *Hours of Operation* is activated are given in Table **4**.

To implement these gestures, we divided the Nexus One touch screen into four quadrants: 1, 2, 3, and 4, starting with 1 as the top left quadrant and moving clockwise. A touch gesture is classified as finger-left if the gesture starts in quadrants 2 or 3, ends in 1 or 4, and the line passing through the start and end points has the absolute slope of no more than 45 degrees. A touch screen is classified as finger-right if it starts in 1 or 4, ends in 2 or 3, and the absolute slope of the line through the start and end points does not exceed 45 degrees. A gesture is classified as finger-down, if it starts in 1 or 2, ends in 3 or 4, and the line's absolute slope is no more than 45 degrees. A gesture is classified as finger-up, if it starts in 3 or 4, ends in 1 or 2, and the line's absolute slope is no more than 45 degrees. Finally, a gesture is classified as finger-tap if the size of the segment that connects the start and end point of the gesture is below a specific threshold.

Table 2. UI1 Actions and their Effects at Hours of Operation in Fig. (1)

UI1 Actions	Effects
trackball-left	*Zeum Art and Technology Center* is activated; Message "Zeum Art and Technology Center."
trackball-right	*School Year Hours* is activated; Message "School Year Hours."
trackball-up	Message "No previous node."
trackball-down	*Fees* is activated; Message "Fees."
trackball-tap	Action associated with *Hours of Operation* is executed.

Table 3. UI2 Actions and their Effects at Zeum Art and Technology Center in Fig. (1)

UI2 Actions	Effects
finger-left	Message "No parent node."
finger-right	*Hours of Operation* is activated; Message "Hours of Operation."
finger-up	Message "No previous node."
finger-down	Message "No next node."
finger-tap	Action associated with node *Zeum Art and Technology Center* is executed.

Table 4. UI2 Actions and their Effects at Hours of Operation in Fig. (1)

UI2 Operations	Action Effects
finger-left	*Zeum Art and Technology Center* is activated; Message "Zeum Art and Technology Center."
finger-right	*School Year Hours* is activated; Message "School Year Hours."
finger-up	Message "No previous node."
finger-down	*Fees* is activated; Message "Fees."
finger-tap	Action associated with *Hours of Operation* is executed.

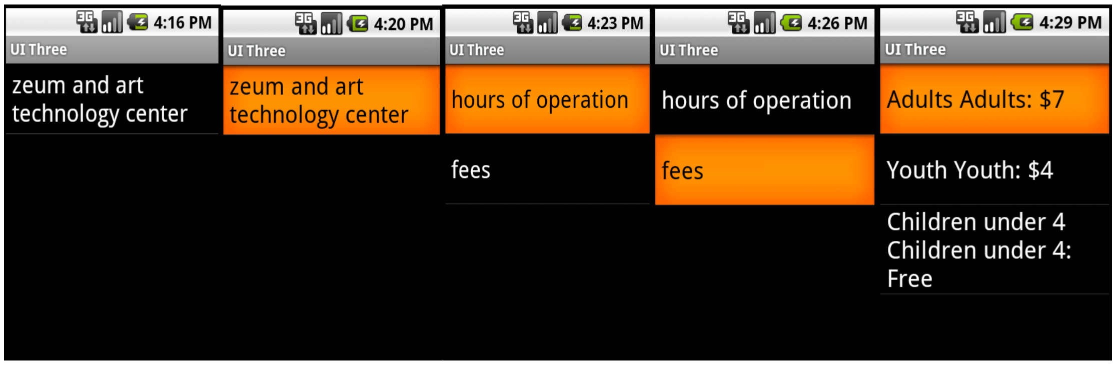

Fig. (2). A sequence of UI3 screen shots.

2.3. User Interface 3

User Interface 3 (UI3) is based on three accessibility features natively available on the Android OS: *TalkBack*, *SoundBack*, and *KickBack* [11]. This UI may be viewed as a reasonable benchmark against which other methods can be compared. TalkBack reads currently highlighted items on the screen; SoundBack beeps when the user does something with the screen; KickBack is similar to SoundBack but uses gentle vibrations instead of beeps. In UI3, all three features were turned on. The user is given the choice of using either the trackball or a finger for navigation and selection.

Fig. (**2**) gives an example of how UI3 works. Suppose that the user wants to find the admission fee for adults for the Zeum and Art Technology. Moving from left to right, the first picture in Fig. (**3**) is the first screen shot. The user uses the trackball to select the only item on the screen. As soon as this item is selected (screen shot 2 in Fig. **2**), the user hears the name of the item through TalkBack, hears a beep through SoundBack, and feels a light vibration through KickBack. The phone screen now displays the menu shown in the third screen shot in Fig. (**2**). When this menu appears on the screen, the user hears "Hours of Operation" spoken through TalkBack. The user uses the trackball to go up and down the list. When the user goes down the list (screen shot 4 in Fig. **2**), the user hears "Fees" accompanied by a beep and a light vibration. When the menu item Fees is selected, the screen displays the menu shown in screen shot 5 and hears the message "Adults seven dollars."

3. BROWSING TASKS

We define a browsing task as an information query that the user must answer by browsing a specific menu. For example, in the menu in Fig. (**2**), the user may want to know the fee for the children under four. The complexity of a browsing task can be measured by the number of available UI actions taken by the user to find an answer to the query. Since we are browsing single-inheritance hierarchies, the complexity of a browsing task can be measured in terms of five abstract user actions: up, down, left, right, and select. Each of the three UIs offers a specific implementation of each abstract action. For example, in UI1, the up action is implemented with **trackball-up**, the down action is implemented with **trackball-down**, etc.

To standardize task complexity measurement, we assign a number to each browsing task that measures the *smallest* number of actions required to complete it. This number approximates the number of actions that a knowledgable user (the one who knows where the answer is in the menu) will execute to find the answer. Suppose again that the user wants to find an answer to the query is "What is the fee for the children under four?" in the menu in Fig. (**3**). The smallest number of abstract actions from the root node *Zeum Art and Technology Center* to the node *Free* is six: 1) right (to *Hours of Operations*), 2) down (to *Fees*), 3) right (to *Adults*), 4) down (to *Youth*), 5) down (to *Children Under Four*), and 6) right (to *Free*). Thus, the complexity of this browsing task in this menu is six.

Under this measurement scheme, browsing tasks can be meaningfully compared with each other to determine if one is harder than the other. For example, if we compare the browsing task "What are the Sunday hours of operation during the school year?" with the browsing task "What is the fee for the children under four?", our complexity measurement method ranks the former task as easier than the latter, because the former task's complexity of four (right, right, down, down) is smaller than the latter task's complexity of six.

4. EXPERIMENTS

4.1. Training and Experimental Menus

We have designed five menus for our experiments: two for user training and three for the actual experiments. The first training menu consisted of thirteen menu items that varied from letters to numbers to color names and had three levels. The second training menu consisted of twenty five nodes and had four levels. The second training menu coded a taxonomy of plants (trees, bushes, and flowers) in various geographic regions of the U.S.

The three menus used in our tests contained information on the Yerba Buena Gardens, a public recreational facility in San Francisco, CA. We chose to represent information about this facility, because it already has several Talking Signs transmitters installed on its premises. Therefore, in the future, when the hardware component of our system is ready, we hope to test it on the premises. The first experimental menu encoded information about the Yerba Buena Skating rink. It had twenty three nodes and four levels. The second experimental menu encoded information about the Yerba Buena Rocco's Restaurant. It contained fifty nodes and had four levels. The third experimental menu encoded information about the Yerba Buena Zeum and Art Technology Center. It contained eleven nodes and four levels.

4.2. Easy and Hard Queries

For each test hierarchy, we designed two queries: one easy and one hard. Given a hierarchy, an easy query is a query with the smallest browsing task complexity, as measured by the method described above, whereas a hard query is a query with the largest complexity. In other words, a query is easy for a given hierarchy if there is no other query for that hierarchy that can be answered in a smaller number of actions. Similarly, a query is hard for a given hierarchy if there is no other query for that hierarchy that can be answered in a larger number of abstract actions.

To illustrate these definitions, consider again the menu hierarchy given in Fig. (**2**). The query "What are the school year hours of operation Wednesday through Friday?" is easy, because it can be answered in four actions, which is the smallest for the hierarchy. On the other hand, the query "What is the fee for the children under four?" is hard, because it can be answered in six actions, which is the largest number of actions for the hierarchy. It should be noted that, under these definitions, a hierarchy may have several easy and hard queries.

4.3. Design and Method

Participants

Seven visually impaired participants were recruited for the experiments. The age ranged from thirty two to sixty. Five were completely blind; one had some peripheral vision. None of the subjects was able to read the mobile phone display. There were two guide dog users and four cane users. Two subjects never used a mobile phone and never had one; the other five had mobile phones and used them for phone calls.

Participant Training

Each session was broken into three parts. The first part was a tutorial. Each participant was instructed on how to use each of the three user interfaces. The participant was given a Google Nexus One smartphone with the software installed on it and was shown how to use each interface on two training hierarchies. When the participant was comfortable with a specific interface, he or she was given two training browsing tasks to complete. The training was finished when the participant successfully completed both browsing tasks. While completing the training tasks the participant was allowed to ask the experimenter for assistance.

Experimental Methods

Once trained, each participant was asked to use each of the three user interfaces to complete one hard browsing task on each of the three experimental menus. Subsequently, three participants, randomly chosen, were asked to complete an easy browsing task (in addition to the hard one) on each of the three experimental menus. Availability of participants and time and budget constraints on the study dictated that only three participants completed this phase of the study. In the third phase, the participants were given an opportunity to provide feedback on each interface.

To estimate the potential of speech-based information browsing, five participants were asked to speak a series of twelve one or two word commands to the speech-to-text engine of the Android Incredible smartphone. The commands are listed in Table **9**. For each command the number of repetitions was recorded before the command was recognized by the engine. If a command was not recognized after five repetitions, the command was considered unrecognized and scored as a 5. Thus, for each command, the number of repetitions ranged from 1 to 5.

A qualitative survey was taken of each participant after the formal experiments. Each participant was asked to specify which UI the participant preferred and why. The participants were also given the opportunity to offer free-form comments on touch gesture interfaces and their experience with them. The participants' responses were logged into text files.

Statistical Design

To minimize bias due to a potential learning effect, the order in which each participant used each of the three user interfaces was randomized for each participant, as was the assignment of a particular menu query (easy or hard, as appropriate) for each participant using each interface. Since each participant used each interface, the experiment was formally a repeated measures (specifically, a cross-over) experiment with user interface the "within subjects" factor of primary interest. The effects of gender and any previous mobile-phone experience were controlled for as "between subjects" factors when analyzing the results for all seven participants. For the three participants who answered both easy and hard queries with each interface, a second factor, "EasyHard" with two levels, was included along with its interaction with user interface.

Variables

For the experimental data, the measured dependent variable was completion time. The log of the completion time was analyzed to reduce the adverse impact of non-normality due to observed right skewness of the raw completion time data. The independent variable was factor (UIID, user interface ID at three levels: UI1, UI2, and UI3, each corresponding to the specific UI. Since one hard query was randomly assigned to each of the three UIs and each of the three experimental menus, there were a total of twenty one observations in the primary phase of the study with each of the seven participants completing three runs, each run consisting of a user interface coupled with a randomly assigned query. Other independent factors were gender (2 levels) and any previous cell-phone experience (2 levels).

Statistical Model

After transformation to the log-scale, residual analysis indicated approximate normality (Shapiro-Wilk test, $P > 0.95$), no more than modest non-constant variance, and no indication of auto-correlation within each participant's three measurements (i.e., no significant autocorrelation within participants was found using SAS PROC MIXED correlation structures). Additionally, no major outliers were observed. Consequently, a standard parametric repeated measures ANOVA was used to assess differences in mean log completion time among the three user interfaces, with adjustment for gender or previous mobile phone experience as appropriate. All analyses were conducted using the SAS statistical software [13].

Statistical Power and Effect Sizes

Observed effect sizes (Cohen's d, mean difference divided by estimated standard deviation) are provided in Tables **6** and **8**. The small n-size of this study, 7 participants for the primary phase of data collection, dictated that only large effect sizes greater than (about) 1 could be easily detected. The approximate power to detect an effect size of 1 (at least two means differing by 1 standard deviation) was ~87%. Consequently, only large effects were found to be statistically significant in this study. Because of the exploratory nature of this study, no adjustment was made to P-values for multiple comparisons.

Data Collection Protocols

As the participants used the software, all their actions were automatically logged and timestamped by a program running on the smartphone. When a participant forgot what he or she was looking for or became completely lost in the menu, the participant was allowed to stop the experiment and ask for clarification or ask to start over. All participants were encouraged to do the best they could**.**

5. Results

5.1. Hard Queries

The first repeated measures ANOVA analysis modeled the data collected on the seven participants who were asked to use each of the three UIs to answer a hard query on each of the three experimental menus.

Overall, adjusting for gender and any previous mobile-phone experience, mean log-completion time differed significantly among the three user interfaces, ($F(2,16) = 6.04$, $P = 0.0111$). Table **5** below summarizes the analysis and shows that UI3 had the lowest observed mean log-completion time with all seven participants using one randomly assigned hard query for each of the experimental menus. The last column of Table **5** converts means back to the measured time scale (seconds) for ease of comparison.

Table **6** provides the t-statistics, P-values, and observed effect sizes for each user interface comparison. Table **6** shows that the mean log completion time for UI3 was significantly lower than the mean log completion times for both UI1 ($t = 2.391$, $P = 0.030$) and UI2 ($t = 3.375$, $P = 0.004$). The mean log completion times for UI1 and UI2 did not differ significantly. Thus, UI3 is the clear winner in this analysis. UI2 was the least efficient, on average, but not significantly worse than UI1. The third entry in each cell of Table **6** is the observed effect size: mean difference divided by the (estimated) standard deviation.

5.2. Easy *vs* Hard Queries

Three of the seven participants were randomly selected to answer both easy and hard queries on the assigned menus. Thus, in addition to the hard queries, these individuals were

Table 5. Mean Log Completion Time (seconds) for Three UIs

UUID	Mean Completion Time (Log Scale)	Standard Error (Log scale)	Mean Completion Time (sec)
UI1	5.461	0.241	235
UI2	5.794	0.241	328
UI3	4.654	0.241	105

given one easy query each and therefore this part of our experiment produced eighteen observations (each participant completing six runs). The independent factors were again User Interface ID (UIID) and EasyHard (coded 1 for Easy and 2 for Hard), with Gender as an adjustment effect. The dependent variable was the log of completion time.

Table 6. t-Statistics and P-Values and Observed Effect Sizes when Comparing Mean Log Completion Time

t-Statistic/ P-Value/ Effect Size Comparing Mean of UIi to UIj		
	UI2	**UI3**
UI1	-0.989 0.337 0.530	2.391 0.030 1.281
UI2		3.375 0.004 1.812

The overall ANOVA model was significant, F(6,11) = 19.14, P < 0.0001. F-tests for each factor's effects are presented below in Table **7**. All effects were significant at the 0.05 level, including the interaction of UIID with EasyHard, adjusting for Gender.

An interaction line-plot for the mean log completion time for UUID and EasyHard queries at each user interface is in Fig. (**3**) below. The lower blue line plots the mean log completion times of the easy queries against the three user interfaces. The upper dashed curve plots the mean log completion times of the hard queries for the three user interfaces. The significant interaction between UIID and EasyHard is visually evident in that the blue and red lines are non-parallel. Specifically, the magnitude of the difference between the mean log completion times for an easy versus hard query depends of the user interface. The difference is large for UI2, moderate for UI1 and small for UI3. It is also visually apparent that mean log completion time was effectively the same for Easy queries for all UIs (blue line is nearly flat), but this was not the case for Hard queries, where UI2 was less efficient than either UI1 or UI3.

Table **8** provides the detailed assessment of the differences in mean log completion time among the six combinations of UIID and EasyHard. As suggested by the blue line in Fig. (**3**), there was no significant difference in mean log completion time among the three User Interfaces for easy queries: UI1 versus UI2, P = 0.4594; UI1 versus UI3, P = 0.4347; and UI2 versus UI3, P = 0.2317. In contrast, for hard queries, UI2 was significantly less efficient than both UI1 (P = 0.0060) and UI3 (P = 0.0002). UI1 was also significantly less efficient than UI3, although only marginally so (P = 0.0475). Finally, for UI3, there was no significant difference in mean log completion time (P = 0.2169), but hard queries took significantly longer than easy queries for both the other interfaces: P = 0.0112 for UI1 and P = 0.0001 for UI2. Mean log completion time was significantly greater for hard queries with UI2 than any other interface or query difficulty (see significant P-values in column label UI2 Hard in Table **8**).

For hard queries, UI3 is the clear winner, while UI2 the worst. For easy queries, the choice of interface does not matter, although UI3 had the lowest observed mean completion time. Thus, for time efficiency without knowing the difficulty of the query, UI3 is the UI to use.

After the experiments were completed, we asked each subject to decide on their preferred interface. Five subjects chose UI2, two subjects chose UI1. None chose UI3. In other words, most subjects preferred the least time-efficient user interface over the two more time-efficient counterparts.

As a pilot for future work, we looked briefly at speech recognition of simple commands relevant to the browsing task under consideration. Seven out of twelve speech commands were recognized in fewer than two repetitions (Table **9**).

6. DISCUSSION

Our experiments show that, for our sample, UI3 was the clear winner in terms of time-efficiency of task completion, while UI2 was the worst. When the participants needed to complete easy queries, the choice of interface did not matter, although UI3 had the lowest observed mean completion time. The qualitative survey of the participants showed that most participants (five out of seven) preferred the UI2 interface, with the two other subjects opting for UI1. None chose UI3. In other words, the participants preferred the two less time-efficient user interfaces over the most time-efficient interface.

This incongruence between quantity and quality suggests that time-efficiency was not the only factor that determines the user preference. Since six out of seven participants in our sample used mobile phones on a regular basis, they quickly became comfortable with the five gesture DPAD metaphor and learned to browse the menus with the five touch screen gestures. The two users who preferred the trackball interface, UI2, said that they preferred keys to gestures, because they had phones with a physical keypad. The touchscreen was completely new to them and they said that they needed more time to become comfortable with it.

If the participants preferred UI2, why did they fail to complete the queries with UI2 as fast or faster than with UI1

Table 7. F-Tests for Effects of UIID and EasyHard

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
UIID	2	3.139	1.569	12.03	0.0017
EasyHard	1	4.363	4.363	33.45	0.0001
UIID*EasyHard	2	1.255	0.627	4.81	0.0315
Gender	1	6.221	6.221	47.70	<.0001

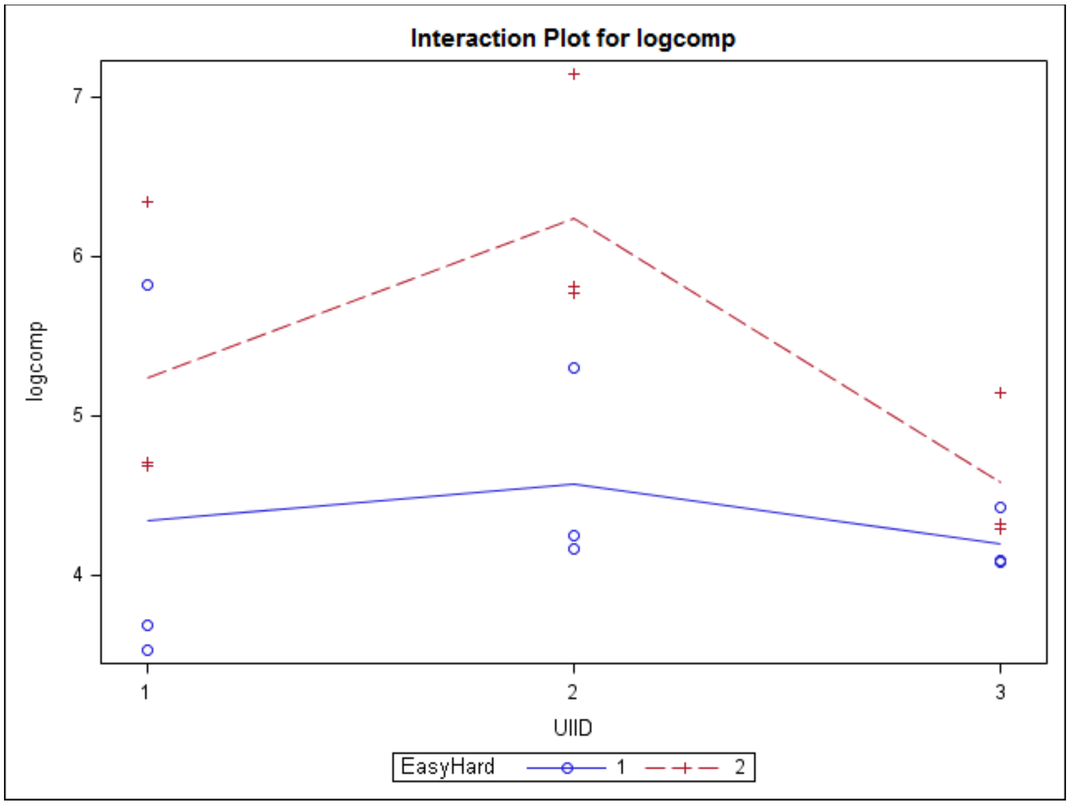

Fig. (3). Interaction Plot of UUID and EasyHard.

Table 8. t-Statistics, P-Values and Effect Sizes for Combinations of UIID and EasyHard

t – Statistic for Comparing Each UI and EasyHard Mean/P_Value/Observed Effect Size					
i/j	**UI1 Hard**	**UI2 Easy**	**UI2 Hard**	**UI3 Easy**	**UI3 Hard**
UI1 Easy	-3.04 0.011 2.59	-0.77 0.459 0.64	-6.43 <.001 5.36	0.50 0.628 0.42	-0.811 0.435 0.66
UI1 Hard		2.27 0.044 1.88	-3.39 0.006 2.77	3.54 0.005 2.91	2.23 0.048 1.83
UI2 Easy			-5.67 <0.001 4.63	1.27 0.232 1.02	-0.04 0.965 0.03
UI2 Hard				6.93 <.001 5.63	5.62 <0.001 5.69
UI3 Easy					-1.31 0.217 0.60

or UI3? One conjecture, based upon our observations of the training sessions, is that there may simply be a semantic dissonance between the UI2 gestures and how many people understand the term "hierarchy." Many people understand this term as a genealogical tree where a child node resides below its parent node. Thus, one has to go "down" to get from a parent to a child and go "up" to get from a child to a parent. In UI2, on the other hand, one must go right to go from a parent to a child and to go left from a child to a parent. This cognitive mismatch might have caused the slower browsing times for UI2. Although the time to navigate these test menus are in excess of what is practical for "on the go" use, further research will help clarify the above conjecture about semantics and possibly lead to an overall increase in the time-efficiency for these methods. In future work, we will also test other accessible information browsing systems such as the IDEAL Web Reader, a gesture-based open source web browser (www.apps4android.org).

Table 9. Speech Commands and Average Number Repetitions for Recognition Required

	Commands	Num. Repetitions
1.	left	1.2
2.	right	1
3.	up	1.8
4.	down	1
5.	press	1.4
6.	select	1.6
7.	move left	2.8
8.	move right	2.6
9.	move up	1.8
10.	move down	2.6
11.	item press	4.2
12.	item select	2.8

Regarding the pilot for speech input, in addition to the small number of participants - all of whom were native English speakers - it should be noted that the speech recognition trials were done under laboratory conditions with no extraneous sounds present. Since real world environments have substantial background noise, the performance of the present speech-based browsing task would certainly be degraded. Speech input for browsing control may play a bigger role as reliable operation in noise increases and as *natural language processing* develops.

9. SUMMARY AND CONCLUSIONS

We designed and implemented three eyes-free UIs for browsing single-inheritance hierarchies. We took the measure of the complexity of a browsing task to be the minimum number of UI actions required to transverse a tree in order to reach the answer to a particular query. Seven VI participants were each given a Google Nexus One smartphone (with the UI software installed) for a hands-on demonstration of each of the three interfaces on two training hierarchies. Three participants, randomly chosen, were asked to complete an easy browsing task (in addition to the hard one) on each of the three experimental menus. For each test hierarchy, we designed two queries: one easy and one hard. The participants were given an opportunity to provide feedback on each UI. The experiments showed that UI3 (the interface that relied on the Google's accessibility tools) was the most time efficient for our sample of participants while UI2 (the touchscreen interface) was the least time efficient. The qualitative user feedback showed that five out of seven participants preferred UI2. We conjectured that this incongruence may be explained by a semantic dissonance between our UI2 gestures based on horizontally structured hierarchies and the common understanding of the term "hierarchy" which implies a vertical structure.

Our semantic dissonance conjecture indicates that touch gestures should, at least to some extent, reflect the semantic understanding of the task prevalent in the target population. That semantic understanding may simply not exist if the computer literacy levels of the target population vary significantly. For example, two of our participants who use touch screen phones on a regular basis had no problem understanding the menu and DPAD metaphors on which UI2 is built.

No far reaching conclusions can be drawn from our, rather informal, evaluation of speech commands. A high recognition rate across the speakers is testimony to the quality of Google's web-based speech recognition service accessible through an application on the

Android Incredible smarthphone. It is reasonable to expect that that the quality of mobile speech recognition will improve even more in the future and will handle a greater variety of accents. Thus, speech may become a viable option for controlling eyes-free information browsing. One caveat, however, is that, even if speech recognition becomes completely accurate, some people may still have privacy reservations of speaking commands to the system in public spaces [12].

The efficiency of UI3, based on the default accessibility options available on the Google Nexus One phone, testify to the fact that the accessibility of the Android platform is improving [10]. The fact that none of the participants chose UI3 as the interface that they would use indicates that quality of experience and familiarity may be as important as the time efficiency of completing tasks.

ACKNOWLEDGMENTS

We are grateful to the participants for their productive and constructive feedback. This research has been supported, in part, by The Smith-Kettlewell Eye Research Institute and the National Institute on Disability and Rehabilitation Research to the Smith-Kettlewell Rehabilitation Engineering Research Center on Low vision and Blindness, San Francisco, CA.

REFERENCES

[1] Kientz J, Patel S, Tyebkhan A, Gane B, Wiley J, Abowd G. Where's My Stuff? Design and evaluation of a mobile system for locating lost items for the visually impaired. In: Proceedings of the 8th ACM Conference on Computers and Accessibility (ASSETS 2006), Portland, Oregon, USA, October 22-5, 2006.

[2] Kulyukin V, Kutiyanawala A. From ShopTalk to ShopMobile: vision-based barcode scanning with mobile phones for independent blind grocery shopping. In: Proceedings of the 2010 Rehabilitation Engineering and Assistive Technology Society of North America Conference (RESNA 2010), Las Vegas, NV 2010.

[3] Crandall WF, Brabyn JA, Bentzen B, Myers L. Remote infrared signage evaluation for transit stations and intersections. J Rehabil Res Dev 1999; 36(4): 341-55.

[4] Kutiyanawala A, Kulyukin V, Bryce D. On Self-sufficiency of verbal route directions for blind navigation with prior exposure. In: Proceedings of the 25th Annual International Technology and Persons with Disabilities Conference (CSUN 2010), San Diego, CA 2010.

[5] Kane S, Bigham J, Wobbrock J. Slide Rule. Making mobile touch screens accessible to blind people using multi-touch interaction techniques. In: Proceedings of 10th Conference on Computers and Accessibility (ASSETS 2008), October, Halifax, Nova Scotia, Canada 2008; pp. 73-80.

[6] Zhao S, Dragicevic P, Chignell M, Balakrishnan R, Baudisch P. Earpod. Eyes-free menu selection using touch input and reactive audio feedback. In: Proceedings of the Computer-Human Interaction (CHI 2007). ACM Press 2007; pp. 1395-404.

[7] Vanderheiden GC. Use of audio-haptic interface techniques to allow nonvisual access to touchscreen appliances (Poster). In: Proceedings of the Annual Meeting of the Human Factors and Ergonomics Society Annual 1996; 40: pp. 1266.
[8] Landau S, Wells L. Merging tactile sensory input and audio data by means of the Talking Tactile Tablet. Proceedings of EuroHaptics 2003, IEEE Computer Society; 2003: pp. 414-8.
[9] Li KA, Baudisch P, Hinckley K. Blindsight. Eyes-free access to mobile phones. in: proceedings of the computer-human interaction (CHI 2008), ACM Press 2008; pp. 1389-98.
[10] D. Burton. Can an Android Make Your Mobile Phone Accessible? (2010). AFB AccessWorld, Available from: http://www.afb.org/afb press/pub.asp? DocID=aw110202.
[11] Speech Enabled Eyes-Free Android Applications. Available from: http://code.googl e.com/p/eyes-free.
[12] Kulyukin V, Kutiyanawala A. Accessible Shopping Systems for Blind and Visually Impaired Individuals: Design Requirements and the State of the Art. J Rehabili 2010; 2: 158-68.
[13] SAS Software 9.2 (TS2M2), SAS Institute Inc., Cary, NC, USA.

Received: March 21, 2011 Revised: May 10, 2011 Accepted: May 10, 2011