Supporting Blind Users in Selecting from Very Long Lists of Items on Mobile Phones

Luca Chittaro and Alessandro Marassi
Human-Computer Interaction Lab
University of Udine
via delle Scienze 206, 33100 Udine, ITALY
+39 0432 558450
http://hcilab.uniud.it

ABSTRACT
Searching for an item in a long ordered list that does not fit the screen is a frequent task when using mobile devices. This paper explores four different interfaces to support blind users in carrying out this task. Two of them are based on the idea of tree-augmentation of a list, proposed by Furnas [3], and differ in their depth versus breadth ratio. The other two interfaces adopt the more traditional technique of list scrolling based respectively on standard multitap and T9 keyboard entry. The paper reports also on the results of a pilot study of the four interfaces conducted on two blind users.

Categories and Subject Descriptors
H.5.2 **[Information Interfaces and Presentation**]: *UserInterfaces – Interaction styles, User-centered Design, Voice I/O,* H.4.2 **[Computers and Society**]: *Social Issues – Assistive Technologies for persons with disabilities*

General Terms
Design, Experimentation, Human Factors.

Keywords
Blind users, Mobile Devices, Menu Selection, Long Lists.

1. INTRODUCTION
Mobile devices such as PDAs, smart phones and navigation systems are increasingly used by blind persons. Mobile devices offer both sighted and blind users the opportunity to access large amounts of information coming from local or remote applications. Accessing this information often requires to select the desired item in long ordered lists, e.g. contact information in address books, alphanumeric index entries in music catalogs or databases, destinations in navigation systems, etc.

Various mechanisms have been proposed for sighted users to facilitate list scrolling and subsequent item choice e.g., buttons for moving up and down, thumbwheels on the side of devices, scrollbars or gesture-based techniques on touch-sensitive screens. Many interfaces also support list scrolling based on keyboard entry: pressing any key of a (virtual or physical) keyboard will automatically scroll the list up to the first item whose first character matches the pressed key, and any subsequent keypress will further refine the search considering also the second character, then the third and so on. The alternative idea of tree-augmentation of lists has been proposed by Furnas [3] based on his Effective View Navigation (EVN) theory and partially explored on desktop systems [4] as well as mobile devices [2].

MobileHCI 2010 September 7 - 10, 2010, Lisboa, Portugal.
ACM 978-1-60558-835-3.

At present, blind and visually impaired users select items from long lists on mobile devices by using: (i) a screen reader that converts the text in the display into speech (sensory substitution), and (ii) the device keyboard, in multitap or in T9 mode. In this research, we focus on exploring with blind users the above mentioned traditional techniques as well as those [2][4][5] based on EVN theory [3]. We first introduce these techniques in Section 2, then we present the four specific interfaces we implemented in Section 3. Section 4 illustrates and discusses the pilot study we carried out on blind users, while Section 5 concludes the paper.

2. RELATED WORK
In [3], Furnas shows, through a worst case characterization, how the search of an item in a long list could be improved by means of tree-augmentation. Informally, this means adding a hierarchical organization to the original long list of items for supporting its exploration in such a way that the user is at any moment presented with a very small number of items in a single screen together with an easy way for coming closer to her target when it is not available in the current screen. In the simplest case, the tree is binary (degree k=2, see Figure 1: the two sons of the root split the list in two subranges and the process is repeated recursively) and the starting screen presented to the user will contain the first, the middle, and the last item of the original list (e.g. "Antelope", "Moose", "Zebra"). If one of the three items is the target, the user can directly choose it, otherwise she can indicate the subrange in which the target should be (e.g., she is looking for "Cat" and indicates that it is between "Antelope" and "Moose" or she is looking for "Parrot" and indicates that it is between "Moose" and "Zebra"), thus moving to a new 3-items screen that refers to the chosen subrange (showing its first, middle and last item), and so on.

A tertiary tree (k=3) organization will support screens with 4 directly selectable items and 3 possible subranges to choose from (see Figure 2b for an example), a quaternary tree (k=4) will support screens with 5 items and 4 possible subranges, and so on.

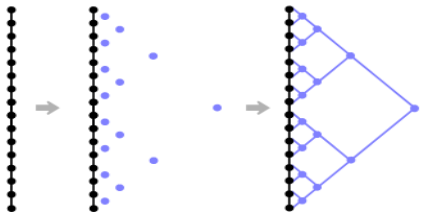

Figure 1. Tree augmentation [3]

Landauer & Nachbar [4] were the first to study tree-augmentation techniques for item search tasks in long lists, focusing on the tradeoff between depth and breadth. Eight sighted users were asked to find a word among 4096 dictionary words by subsequent selections on a touch screen, carried out by finger pointing. The screen was organized into 5 to 33 horizontal stripes (k=2 to k=16) alternating alphabetically ordered list items and unlabeled subrange-buttons. The average search time varied from 23.4 down to 12.5 seconds as the number of subranges available on the screen increased from 2 to 16, indicating an advantage in using a broad and shallow tree structure over a more narrow and deeper one.

In a more recent study [5], a binary tree-augmentation method (BinScroll) was tested on small screens with 24 sighted users. BinScroll is meant for devices with limited input capabilities and is operated by using only four arrow keys. Participants were asked to find a target among 10000 movie titles. The display showed 3 movie titles in a single column, implicitly identifying the two subranges (no subrange buttons were shown). The right arrow was used to select the middle item when it was the target; the up (down) arrow was used to select the upper (lower) subrange; the left arrow was used to go back to previous screens one by one. Average search time was 14 seconds, close to the best result obtained in [4]. Only the interface with 16 subranges in [4] offered a better performance, but the list was smaller. However, it must be pointed out that Binscroll might have improved performance, because users did not need to move their hand on the screen to select the displayed items, but just keep it on the 4 arrow keys.

In [2], tree-augmentation has been studied on mobile devices, using a list of about 14000 items. Three different interfaces have been studied on sighted users: in "Six-Tree" (k=6), the screen is organized into 13 horizontal stripes alternating 7 list items and 6 subrange-buttons (directly selectable). In "Bin-Tree" (k=2), there are 3 list items and 2 subrange-buttons. Selections are carried out on the screen of the mobile device using a stylus. The last interface ("Keyb") implements instead list scrolling based on keyboard entry, described in the introduction of this paper. Experimental results show that with respect to "Keyb" (6.06 s), the two interfaces based on tree-augmentation produce significantly longer task completion times (approximately three and four times greater) and more errors. Users reported difficulties in dealing with alphabetical ordering. These results suggest that, with sighted users, list scrolling based on keyboard entry should be preferred to tree-augmented lists when it is possible to have a virtual (or physical) keyboard available. Nevertheless, techniques based on tree-augmentation could still be useful when no full keyboard (either virtual or physical) can be provided (e.g., in some wearable computer scenarios as those envisaged in [5]). The study confirmed also the breadth versus depth trade-off in tree-augmentation techniques pointed out by [4], but in the different context of mobile devices.

Selection using traditional scrollbars (as well as Alphasliders) by sighted users was explored in [1]: subjects had to find a target movie among 10000 movie titles. A CRT display was employed and selections were carried out using a mouse. The average selection time with the fastest of the tested Alphaslider versions was 24 seconds, while the traditional scrollbar took on average 25 seconds. The tree-augmented lists proposed in [4] and [5] seem to offer an efficiency advantage over scrollbars and Alphasliders.

3. THE INTERFACES CONSIDERED

In our research, we have developed interfaces to support blind users of mobile devices in selecting items from very long lists. Two of the interfaces we discuss in the following are based on tree-augmentation and differ in their depth versus breadth ratio. They are similar to the interfaces tested in [5] and [2], but augmented with text-to-speech (TTS) to make them accessible to blind users.

The other two interfaces implement list scrolling based on standard multitap and T9 keyboard entry, respectively. We have exploited multitap and T9 because they are already familiar to blind users from SMS writing. However, from preliminary interviews we conducted with local associations of blind persons, multitap list scrolling seems to be the only technique currently used by blind users in selecting items from very long lists, while T9 (which exploits a pre-packaged dictionary) is mainly used in text writing. Techniques based on voice recognition still appear to suffer performance problems with very long lists, especially in noisy environments.

All interfaces have been implemented in C# and tested on a HTC i-mate SP5 phone equipped with a commercial TTS engine (Acapela). For brevity, in the following we will refer to the 5 possible actions the user can perform on the phone micro-joystick as UP, DOWN, LEFT, RIGHT, and ENTER. The use scenario we implemented concerns the choice of a geographical destination in a list of about 14000 locations.

3.1 Binary tree-augmented list ("BinScroll")

The first interface we implemented is inspired by the previously described BinScroll [5]: screens contain 3 list items (see Fig. 2a), and the RIGHT, UP, DOWN, LEFT actions have the same effects described in the related work section. The TTS reads the middle item when a screen is presented. We made this choice because preliminary tests showed that (i) reading all three items increased search time considerably, and (ii) having only the middle item read is enough for the user to decide to choose it or to proceed with a subrange, since the middle item either follows (in the alphabetic order) the target item (located in the upper subrange) or precede the target item (located in the lower subrange).

When the user selects the target (RIGHT), the TTS says 'destination' followed by the target item and 'do you confirm?'. If the user confirms the choice (another RIGHT), she hears 'destination confirmed' followed by the target item.

When the user goes back (LEFT), the TTS says 'back', the former screen is presented again and its middle item is read. If the user reaches the initial screen, speech feedback is 'you have reached the initial screen'. If the search is not successful (i.e. the user reaches the bottom of the search tree without having found any target), speech feedback is 'target item not found'.

(a) **(b)**

Figure 2. Screenshots of "Binscroll" (a) and "Three-Tree" (b).

3.2 "Three-Tree" interface

The "Three-Tree" interface exploits a tree-augmented structure with k=3: the screen shows 4 list items and 3 subrange-buttons (see Fig. 2b). As we did with BinScroll, we minimized the number of items read by TTS: at every screen, only the 2 middle items are read. This information is sufficient for the user to proceed: if the target precedes the first middle item, the user chooses the upper subrange; if the target is between the two middle items, she chooses the middle subrange; if it follows the second middle item, she chooses the lower subrange.

Input is based on UP and DOWN to select respectively the upper and the lower subrange-button (as in BinScroll), while ENTER selects the middle subrange-button; LEFT goes back as in BinScroll, while RIGHT enters an 'item selection' mode in which UP and DOWN select between the 2 middle items read in the current screen, eventually confirming the choice with a second RIGHT. The sentences pronounced by TTS in response to user actions are the same described in BinScroll.

3.3 T9 list-scroll interface

In the third interface, as the user presses the numeric keys on the keypad, a list of all items that match the composed T9 string is updated on the screen. The TTS starts to read the list, but can be restarted by new keypresses. To delete the last character, users press LEFT, consistently with the back action in preceding interfaces. If the search is not successful (i.e. the string does not match any item), the system gives an audio feedback ('target item not found').

If the target is in the displayed list, RIGHT allows the user to explore the list. Each RIGHT reads one list item, starting from the first, while ENTER selects the last read item (and TTS pronounces the word 'destination' together with the target item itself).

3.4 Multitap list-scroll interface

The fourth interface is analogous to the third. The only difference is that keypresses of the numeric keys are handled according to the standard multitap mode: for example, pressing once key 2 will select 'a', after a timeout of 1 second, while pressing twice key 2 will select 'b', and so on.

4. PILOT STUDY ON BLIND USERS

We carried out a pilot study on two male blind users, aged respectively 36 and 39. We deliberately recruited two persons who are interested in assistive technologies, have advanced computer knowledge, and regularly use phones and PCs equipped with screen readers, because we wanted to get very detailed feedback to find possible major problems before proceeding with an experimental evaluation on a larger sample of blind users.

In the first trials, we employed the list of streets of the city where the blind users live (1334 items). Later, to be consistent with the size of lists in [2][5], we used the list of 13926 locations of the country (Italy) where the two blind users live.

Users were asked to try all the four different interfaces to find specific targets. Before trying an interface, they received an explanation about how to use it, and were trained by finding a few targets until they felt confident with the interaction method. Then, they carried out the task of finding a sample of 20 locations.

The following data was collected to characterize efficiency and usability: (i) search time to find a given target; (ii) N_ERR: number of errors in terms of wrong subrange selections (for Three-Tree and BinScroll), wrong characters entered (for T9 and Multitap), and wrong list item selections (for all interfaces); (iii) N_CLICKS: number of clicks needed to find and select the searched target. Table 1 summarizes the data collected from the two users (denoted as U1 and U2 in the Table). Performance results are consistent between the two users for all interfaces.

Interface	Mean N_CLICKS		Mean N_ERR		Time [s]	
	U1	U2	U1	U2	U1	U2
BinScroll	14.1	13.7	7.2	5.5	52.8	52.2
Three-Tree	7.4	8.9	0.9	1.2	63	70.4
Multitap	16.8	19.5	0.1	0.1	20.5	22.8
T9	6.4	6.6	0	0	10.2	12

Table 1. Results of the tests for the two pilot users (U1, U2)

Unlike prior experiments [2][4] on sighted users, which showed that broader tree-augmented lists were to be preferred to deeper ones, the results we obtained with blind users favor deeper rather than broader trees: BinScroll gave better results than Three-Tree. One reason for this is the sequential nature of the speech modality: for Three-Tree, the TTS must read 2 items for each

screen, while for Binscroll one item is enough. To further explore this, we proposed users a tree-augmented structure with k=5 (6 list items and 5 subrange-buttons): they immediately complained about its very high cognitive load, which made it useless, further remarking the difference from studies conducted on sighted users.

The T9 list-scroll interface was more efficient than Multitap and much more efficient than tree-augmented methods with the two blind users. Since the considerable previous experience of the users with T9 might have widened the gap between T9 and the other interfaces, it would be interesting to repeat the tests with blind users who are not familiar with T9.

The mean number of errors (Table 1) has been calculated considering all the 20 different test trials. Users made no errors with T9 and only a few with Multitap. More errors were made with the interfaces based on tree-augmentation, with BinScroll producing much more errors than Three-Tree. This can be explained by the fact that both users reported they sometimes did not correctly understand unfamiliar names pronounced by TTS (in Italian, this happens especially with names beginning with “b” and ”p” or “c” and ”g”). From this point of view, Three-Tree (with 2 pronounced names per screen) provided users with more information to decide the next choice than BinScroll (with 1 pronounced name). It also must be noted that this issue did not occur in the initial trials carried out with the list of the city streets which were familiar to users.

In the subjective preferences of the two blind users, T9 list-scroll interface was considered to be the best, followed by Multitap. Both users remarked that BinScroll and Three-Tree require a higher level of attention. One of them preferred BinScroll to Three-Tree. Although both users were very satisfied with the quality of the TTS, they complained about the above mentioned ambiguity issue resulting from the pronunciation of some unknown names.

5. CONCLUSIONS AND FUTURE WORK

In the pilot study on two blind users presented in this paper, list scrolling based on T9 had a better performance than list scrolling based on multitap and both had a better performance than tree-augmented techniques. A clear difference from studies of sighted users emerged: a deeper tree-augmented list gave better results than broader ones. Consistently with studies of sighted users, interfaces based on tree-augmentation imposed a higher cognitive load and appear to be more error-prone.

On the other hand, it must be stressed that the tree-augmented interfaces we implemented can be used with just a micro-joystick and just one finger. They can thus become very attractive when a device has no keyboard or can have only a limited number of hardware keys (this applies not only to phones, but also to other portable devices such as mp3 players, health monitors, or domestic appliances).

As we have discussed, sequential speech synthesis was one of the factors that slowed down performance with respect to the visual versions of the same interfaces. Time required by speech output has to be improved as well as the correct understanding of unfamiliar names uttered by the TTS which influenced the overall error rate. We are currently planning to investigate in more detail these issues by giving users the means to get additional information about the uttered nouns, which could be spelled on demand, as well as the possibility to personalize reading speed of the TTS.

6. ACKNOWLEDGMENTS

We are grateful to Andrea Roveretto (Biblioteca Italiana per i Ciechi - Centro di Consulenza Tiflodidattica di Trieste) and Vincenzo Zoccano (Unione Italiana dei Ciechi e degli Ipovedenti di Trieste) who generously devoted their time to evaluate the interfaces.

This research has been partially supported by the Italian Ministry of Education, University and Research (MIUR) under project FIRB RBIN04M8S8.

7. REFERENCES

[1] Ahlberg, C. and Shneiderman, B. The Alphaslider: A Compact and Rapid Selector. In *Proc. Conf. on Human Factors in Computing Systems (CHI 94),* 365-371. ACM Press, 1994.

[2] Chittaro, L. and De Marco, L. Evaluating the Effectiveness of “Effective View Navigation” for Very Long Ordered Lists on Mobile Devices. In *10th IFIP International Conference on Human-Computer Interaction (INTERACT 2005)*, Springer Verlag, 482-495, 2005.

[3] Furnas, G. W. Effective View Navigation. In *Proc. Conf. on Human Factors in Computing Systems (CHI '97)*, ACM Press, 367-374, 1997.

[4] Landauer, T. and Nachbar, D. Selection from alphabetic and numeric menu trees using a touch screen: breadth, depth, and width. In *Proc. Conf. on Human Factors in Computing Systems (CHI ‘85)*, ACM Press, 73-78, 1985.

[5] Lehikoinen, J. and Salminen, I. An Empirical and Theoretical Evaluation of BinScroll: A Rapid Selection Technique for Alphanumeric Lists, *Personal and Ubiquitous Computing* 6(2), 141-150, 2002.