Behavior Research Methods
2009, 41 (3), 951-956
doi:10.3758/BRM.41.3.951

Selection-based virtual keyboard prototypes and data collection application

Barbara Millet
Texas Tech University, Lubbock, Texas

Shihab Asfour
University of Miami, Coral Gables, Florida

and

James R. Lewis
IBM Software Group, Boca Raton, Florida

An emerging area of research in engineering psychology is the evaluation of text entry for mobile devices using a small number of keys for the control of cursor direction and character selection from a matrix of characters (i.e., selection-based data entry). The present article describes a software tool designed to reduce time and effort in the development of prototypes of alternative selection-based text-entry schemes and their empirical evaluation. The tool, available for distribution to researchers, educators, and students, uses Action Script code compiled into an executable file that has an embedded Adobe Flash Player and is compatible with most operating systems (including Microsoft Windows, Apple OSX, and Linux).

There is a strong demand for mobile computing devices. Indeed, the mobile domain is overwhelmed by consumer demands for microdevices that offer full functionality, including the ability to perform some tasks that require text entry. By definition, microdevices have limited physical space for text input and output, and there is still a need for efficient text-entry methods that work within these constraints (Lewis, Commarford, Kennedy, & Sadowski, 2008). To support text entry on smaller devices, some researchers have investigated text entry using selection-based methods (as in Bellman & MacKenzie, 1998; MacKenzie, 2002; Millet, Asfour, & Lewis, 2008; Sandnes, Thorkildssen, Arvei, & Buverud, 2004; Wobbrock, Myers, & Rothrock, 2006).

Selection-based data-entry methods use a virtual keyboard that has a moveable selector (or cursor) over a matrix of characters, with the cursor operated using as few as two keys (one direction key and the selection key, with wraparound enabled). An advantage of this method is that it eliminates the need for screen areas large enough to accept stylus or finger entry, and it does not require the expense of a touchscreen user interface.

For any new text-entry technique, it is important to evaluate its effectiveness. Generally, there are two approaches to evaluation: predictive modeling and empirical evaluation. Predictive models use metrics of human performance to explore the effectiveness of text-entry designs without the need to implement the design (MacKenzie, 2003). A limitation of these models, however, is that they predict expert, error-free performance. Empirical evaluations, on the other hand, require the collection of user data, and are thus able to assess user satisfaction and novice performance to attain absolute performance measures and to explore interface learnability (MacKenzie & Soukoreff, 2002). The empirical evaluation of a new text-entry technique is demanding, involves detailed planning, and requires a working version of the technique.

When designing an empirical evaluation, it is important that one consider the test equipment. Ideally, the devices used in the evaluations should be high-fidelity prototypes of those planned for use in the real world. In practice, a working prototype might not exist, and the development resources needed to create one might not be available (Sirisena, 2002). An alternative approach is to use software simulators (Bellman & MacKenzie, 1998; Butts & Cockburn, 2002; Dunlop & Crossan, 2000; MacKenzie, 2003; MacKenzie, Kober, Smith, Jones, & Skepner, 2001; MacKenzie & Zhang, 1999).

In the present article, we present a tool that facilitates the design and evaluation of selection-based virtual keyboards. The tool supports the use of five keys (up, down, left, and right arrows, and "Enter"), so that experimenters can use them to study five-key techniques. The tool can display virtual keyboards with different layouts, facilitate comparative evaluations, and automate the test protocol and data collection. The overall goals in publishing this

S. Asfour, sasfour@miami.edu

tool are to facilitate the ability of others to replicate experiments, permit others to extend selection-based text-entry research to other keyboard layouts and conditions, and contribute to the human–computer interaction (HCI) community by making this tool available to researchers, educators, and students.

SYSTEM DESCRIPTION

This tool supports the development and evaluation of selection-based virtual keyboards. It takes as input .xml files that define test phrases and keyboard layouts. A session control interface presents the evaluation environment to the test participant and allows the moderator to control specific keyboard attributes. A data log captures session data in a portable comma-separated value (.csv) format. This log contains a list of all typing key events that occurred throughout the test session.

The source code is in the Action Script (AS) language, the coding language for Adobe Flash. Flash is a programming environment that supports complex vector graphics and animations, and it is compatible with most operating systems, including Microsoft Windows, Apple OSX, and Linux. The compiled executable (.exe, also compatible with most operating systems) file has an embedded Adobe Flash Player, so there is no need to download a player to run the tool, which starts with a simple click on the main .exe file.

SYSTEM FEATURES

There are many conditions that can influence the design and evaluation of new or refined text-entry methods. This tool provides numerous design features, allows for comparative evaluations, and automates the test protocol and data collection. Overall, this tool can facilitate the design and evaluation of text-entry methods using virtual keyboards and indirect selection-based input. Notable features of the tool are flexible character layouts, the use of an extended character set, the automation of test protocols and data collection, and design enhancements that include error handling, capitalization, different cursor modes, and typematic keying.

Flexible Character Layouts

The tool supports any character layout, allowing for the evaluation of standard (QWERTY and Alphabetic) and nonstandard layouts, both static and dynamic. The layouts are stored in an .xml file, and it is possible to change the layout as a function of the last character typed (as in Bellman & MacKenzie, 1998). This feature permits the implementation of predictive keyboards based on digraph frequencies (Bellman & MacKenzie, 1998; MacKenzie, 2002; Millet et al., 2008). The tool also supports the display of keyboard layouts across *n* rows, where *n* is a user-specified attribute. Furthermore, adjustments to the application window can reduce or increase the size of the onscreen keyboard.

Use of an Extended Character Set

Much of the previous research in selection-based virtual keyboards has used test texts composed only of words and spaces (Bellman & MacKenzie, 1998; MacKenzie, 2002; Sandnes et al., 2004). Rather than restricting future studies to alphabetic characters and spaces, this tool allows researchers control over the inclusion or omission of alphabetic and numeric characters, punctuation, symbols, edit functions, and modifier functions.

Automation of Test Protocol and Data Collection

Typically, text-entry experiments require participants to transcribe text using the input method of interest. To support transcription, this tool reads the test phrases from a file and presents them to the user for input. As participants enter text, the tool records a timestamp and key code for each keystroke, saving these in a .csv file for follow-up analyses.

Design Enhancements

Error correction, capitalization, typematic keying, and specification of cursor home positioning and cursor movement are supported. Researchers can enable or disable most of these features.

Error handling. Errors and error handling are important dimensions to consider when designing and evaluating text-entry methods. Previous studies of selection-based entry (Bellman & MacKenzie, 1998; MacKenzie, 2002; Sandnes et al., 2004) have not included error correction (only measuring words per minute, sometimes with a correction for errors), but the only way to measure true text entry throughput is with correct words per minute (Lewis et al., 2008). Consequently, the tool provides a delete key to allow participants to correct input errors. At this time, it is not possible to disable error correction, but it is likely that future versions of this tool will allow researchers to disable this feature.

Capitalization. For some real-world data entry, it is important for users to be able to produce both upper- and lowercase letters. To support these tasks, the keyboard .xml file permits the optional presentation of an on-screen "Caps" key.

Cursor modes (positioning and movement). Different cursor positioning and movement techniques can affect the ease and efficiency of selection-based text entry (Millet et al., 2008) so that researchers can enable or disable these features. Cursor positioning can be persistent or snap-to-home. A persistent cursor remains on the selected position after character selection. A snap-to-home cursor jumps back to an assigned key (the designated cursor home position) immediately after each character entry. Additionally, the tool optionally allows cursor wraparound within or between keyboard rows.

Typematic keying. When enabled, the typematic (auto repeat) feature allows users to continuously produce a character by pressing and holding its key beyond a specified delay threshold.

KEYBOARD LAYOUTS

The keyboard layouts to be tested are stored in plain text (.xml) files and are easily created and edited using Microsoft Notepad or an .xml editor. The application supports

XML Script

```
<key>
 <layout1>
  <order order="Space">Space|a|b|c|d|e|f|g|h|i|j|k|l|m|Caps|n|o|p|q|r|s|t|u|v|w|x|y|z>
  </order>
 </layout1>
</key>
```

Resulting Layout

Figure 1. Sample script and resulting keyboard layout.

only one key file at a time, but the file can store multiple keyboard layouts. This file also specifies the characters to display and the order in which they should appear. The tool's extended character set includes:

- •All letters: a–z.
- •All numbers: 0–9.
- •Punctuations: period (.), comma (,), question mark (?), exclamation mark (!), single quote (‘), double quote (“), semicolon (;), and the colon (:).
  Other: tilde (~), at sign (@), number sign (#), dollar sign ($), percent sign (%), caret (^), ampersand (&), asterisk (*), underscore (_), dash (-), forward slash (/), backward slash (\), plus sign (+), equal sign (=), round brackets (( )), curly brackets ({ }), square brackets ([ ]), and angle brackets (< >).
- •Edit and modifier keys: “Space,” “Caps,” and “Enter.”

See Figure 1 for a sample of .xml script that generates a static alphabetic layout with “Space” and “Caps” keys.

Alternatively, this tool can also implement dynamic layouts (e.g., the fluctuating layout of Bellman & MacKenzie, 1998), in which the layout changes as a function of the most recently entered character. To generate a dynamic keyboard with *n* characters, researchers must specify *n* order rows in the keyboard file, with each row defining the character layout to use for each character.

TEST FILES

In a typical text-entry evaluation, the experimental task is to input text using the text-entry method(s) under study. Generally, there are two types of tasks: text creation and text copy (transcription). In a text-creation task, participants memorize or generate the source text (MacKenzie & Soukoreff, 2002). This differs greatly from a transcription task, in which the participant reads the text to be entered (Butts & Cockburn, 2002; Curran, Woods, & Riordan, 2006; Dunlop & Crossan, 2000; James & Reischel, 2001; MacKenzie & Zhang, 1999, 2001).

Although text creation more closely mimics real-world usage, transcription tasks are more common in empirical evaluations, mainly because it is difficult to measure the effects of nonbehavioral aspects (such as “pondering”) when participants compose text (MacKenzie & Soukoreff, 2002). Additionally, in text-creation tasks, it is difficult to ascertain whether or not a participant committed an error because it is sometimes not possible to know what the participant intended to enter. Consequently, transcription is the favored approach when evaluating text entry in laboratory settings (Wobbrock, 2007). For these reasons, this tool allows the evaluation of text entry when performing transcription tasks.

To automate the test protocol, the test file contains all of the test phrases for a given participant and specifies the order of presentation of the keyboard layouts to use. Upon launch, the application reads the .xml file containing the text phrases. During execution, the tool presents these phrases to the participant for input, with the test phrases presented to the user one at a time in the specified order.

Sample Test Text Script

```
<test>
 <set1>
  <keyboard_type>1</keyboard_type>
  <keyboard_row>2</keyboard_row>
  <keys_per_row>14</keys_per_row>
  <keyboard_home_location>1</keyboard_home_location>
    <phrase>Hello World</phrase>
 </set1>
</test>
```

Figure 2. Sample script for a single test phrase.

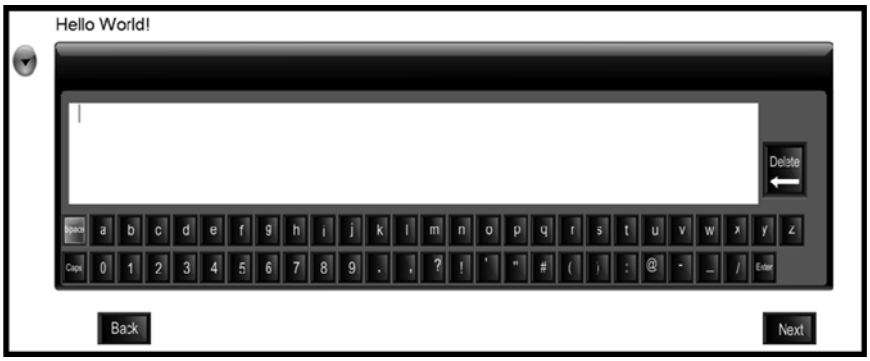

Figure 3. User interface for text-entry evaluations.

Figure 4. The main menu.

The test file uses an .xml format for creation and editing using Microsoft Notepad or an .xml editor. Figure 2 shows sample .xml script for the display of one test phrase ("Hello World") with one keyboard layout (keyboard_type = 1, as is defined in an associated key.xml file). The script specifies a two-row layout (keyboard_row = 2) with 14 keys in the upper row (keys_per_row = 14). The cursor home position is the character in the first position (keyboard_home_location = 1), which in this example maps to the "Space" character (see Figure 1).

USER INTERFACE

The user interface consists of a large text field at the top of the screen at which the stimulus phrase appears, an output field displaying the characters typed by the participant, and the character set of the input method under evaluation. The "Delete" key appears above the keyboard (in line with the user output).

The cursor position appears as a gray box around a white character. The input keys control the cursor on the screen, allowing users to navigate the character set and the "Delete" key. The input keys are the up-, down-, left-, and right-arrow keys (used for navigation) and the "Enter" key (used for selection), as is shown in Figure 3.

All other controls are accessed using a mouse. Typically, the experimenter (rather than the participants) will use these controls. Clicking the "Next" button signals the end of entering the presented test phrase and advances to the next test phrase. If the current test phrase is the last string in the test file, the "Next" button is disabled. Likewise, clicking the "Back" button returns to the previous test string. If the current string is the first phrase in the test file, then the "Back" button is disabled.

Pressing the down-arrow button at the top-left corner of the application displays the normally hidden main menu, which provides functions for uploading a specific test, starting the test, adjusting settings (via the "Options" item), ending the test (which automatically saves the associated data log file), and exiting the application (as is shown in Figure 4).

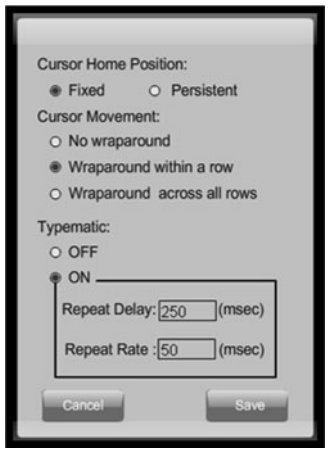

Figure 5. Options screen.

As is shown in Figure 5, the "Options" menu item allows the setup of several parameters that control important characteristics of the test keyboards: cursor home position, cursor wraparound, and typematic keying. By default, the cursor home position is fixed, the cursor wraparound is enabled, and typematic keying has its repeat delay (initiation) set to 250 msec, with a repeat rate (continuation) of 50 msec (as specified in Lewis, Potosnak, & Magyar, 1997).

DATA COLLECTION

The tool's data logging of participant keypresses (including arrow and selection keys) allows fine-grained analysis of subsecond events and low-level actions, measured to the nearest millisecond. Data collection begins upon initiating the test. Thereafter, the software records all participant interactions with the keyboards. The header of the data file captures the presented and the transcribed text, keyboard configurations, and the settings for each test session. Every subsequent line in the log file records the sequence of actions taken by the participant to generate the transcribed string, beginning with a timestamp and followed by event codes and one or more identifiable handles. Specifically, for each keypress, the tool collects the following information:

- The key pressed (e.g., "Left," "Right," "Up," "Down," or "Enter").
- User-entered character, modifier, or edit key.
- Time elapsed between keypresses, or, for the last key, the elapsed time to clicking "Next."
- An indication of whether the user employed typematic keying.

See Figure 6 for a sample portion of a data file.

The tool gathers the data needed for the statistics and performance measures. To save the data, however, the test moderator must select "Done" from the main menu. The tool then saves the data as a .csv file in the same location as that of the .xml and .exe files, appending a date and timestamp to the file name to distinguish among multiple test sessions and to prevent the accidental overwriting of data files. Researchers can retrieve data and use them for subsequent analysis, parsing and combining them as needed for the performance measures under investigation.

Input:	Hello World!		
Output:	Hello World!		
keyboard Type:3			
Cursor Home Position:Fixed			
Cursor Movement:Wraparound with a row			
Typematic:ON			
Repeat Delay:250ms			
Repeat Rate:50ms			
Time (msec)	Keystroke	Notes	
2500	down		
3344	enter=Caps		
4047	right		
4453	right	typematic	
4531	right	typematic	
4609	right	typematic	
4688	right	typematic	
4875	right		
5031	right		
5156	right		
5859	enter=h		
6500	right		
6922	right	typematic	
7000	right	typematic	

Figure 6. Sample data file.

CONCLUSION

In the present article, we presented a tool for the evaluation of text-entry methods for mobile devices. The main objective of this tool is to facilitate the design and implementation of prototypes of selection-based text-entry methods. The savings of time and resources in the empirical evaluation process is another one of its benefits. The current version of this tool supports the flexible evaluation of selection-based virtual keyboard layouts that use five keys for text entry. Although it is not designed to do so, it may be possible to use this tool for the design and evaluation of certain types (e.g., with no error correction) of three- and two-key selection-based methods.

AUTHOR NOTE

This tool was designed by the authors and developed by contractors. Special thanks to Daixtech Inc. for providing partial funding and for managing the tool development effort. The application executable file for this program may be obtained by e-mailing a request to S.A. The executable file will be available for distribution in September 2009. Correspondence concerning this article should be addressed to S. Asfour, Department of Industrial Engineering, University of Miami, 1251 Memorial Drive, 268 McArthur Engineering Bldg., Coral Gables, FL 33146 (e-mail: sasfour@miami.edu).

REFERENCES

BELLMAN, T., & MACKENZIE, I. S. (1998). A probabilistic character layout strategy for mobile text entry. In W. A. Davis, K. S. Booth, & A. Fournick (Eds.), *Proceedings of Graphics Interface '98* (pp. 168-176). Vancouver: Canadian Human–Computer Communications Society.

BUTTS, L., & COCKBURN, A. (2002). An evaluation of mobile phone text input methods. *Australian Computer Science Communications*, **24**, 55-59.

CURRAN, K., WOODS, D., & RIORDAN, B. O. (2006). Investigating text input methods for mobile phones. *Telematics & Informatics*, **23**, 1-21.

DUNLOP, M. D., & CROSSAN, A. (2000). Predictive text entry methods for mobile phones. *Personal Technologies*, **4**, 134-143.

JAMES, C. L., & REISCHEL, K. M. (2001). Text input for mobile devices: Comparing model prediction to actual performance. In M. Beaudouin-Lafon & R. J. K. Jacob (Eds.), *Proceedings of the ACM CHI 2001 Human Factors in Computing Systems* (pp. 365-371). New York: ACM Press.

LEWIS, J. R., COMMARFORD, P. M., KENNEDY, P. J., & SADOWSKI, W. J. (2008). Handheld electronic devices. In C. Melody Carswell (Ed.), *Reviews of human factors and ergonomics* (Vol. 4, pp. 105-148). Santa Monica, CA: Human Factors and Ergonomics Society.

LEWIS, J. R., POTOSNAK, K. M., & MAGYAR, R. (1997). Keys and keyboards. In M. Helander, T. K. Landauer, & P. V. Prabhu (Eds.), *Handbook of human–computer interaction* (pp. 1285-1315). Amsterdam: North-Holland.

MACKENZIE, I. S. (2002). Mobile text entry using three keys. In O. W. Bertelsen (Ed.), *Proceedings of the Second Nordic Conference on Human–Computer Interaction* (pp. 27-34). New York: ACM Press.

MACKENZIE, I. S. (2003). Motor behavior models for human–computer interaction. In J. M. Carroll (Ed.), *HCI models, theories, and frameworks: Toward a multidisciplinary science* (pp. 27-54). San Francisco: Morgan Kaufmann.

MACKENZIE, I. S., KOBER, H., SMITH, D., JONES, T., & SKEPNER, E. (2001). Letterwise: Prefix-based disambiguation for mobile text input. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology UIST 2001* (pp. 111-120). Orlando: Association for Computing Machinery.

MACKENZIE, I. S., & SOUKOREFF, R. W. (2002). Text entry for mobile computing: Models and methods, theory and practice. *Human–Computer Interaction*, **17**, 147-198.

MACKENZIE, I. S., & ZHANG, S. X. (1999). The design and evaluation of a high-performance soft keyboard. In *Proceedings of CHI 1999* (pp. 25-31). Pittsburgh: Association for Computing Machinery.

MACKENZIE, I. S., & ZHANG, S. X. (2001). An empirical investigation of the novice experience with soft keyboards. *Behaviour & Information Technology*, **20**, 411-418.

MILLET, B., ASFOUR, S., & LEWIS, J. R. (2008). Designing a hybrid layout for a five-key text entry technique. In T. B. Leamon, A. R. Kumar, & J. E. Fernandez (Eds.), *Proceedings of the International Society for Occupational Ergonomics and Safety* (pp. 156-162). Chicago: ISOES.

SANDNES, F. E., THORKILDSSEN, H. W., ARVEI, A., & BUVERUD, J. O. (2004). Techniques for fast and easy mobile text-entry with three-keys. In *Proceedings of the 37th Hawaii International Conference on System Sciences* (pp. 1-10). Los Alamitos, CA: IEEE.

SIRISENA, A. (2002). *Mobile text entry*. Unpublished master's thesis, University of Canterbury.

WOBBROCK, J. O. (2007). Measures of text entry performance. In I. S. MacKenzie & K. Tanaka-Ishii (Eds.), *Text entry systems: Mobility, accessibility, universality* (pp. 47-74). San Francisco: Morgan Kaufmann.

WOBBROCK, J. O., MYERS, B. A., & ROTHROCK, B. (2006). Few-key text entry revisited: Mnemonic gestures on four keys. In *Proceedings of the ACM Conference on Human Factors in Computing Systems '06* (pp. 489-492). New York: ACM Press.

(Manuscript received February 12, 2009;
revision accepted for publication March 24, 2009.)