OnScreenDualScribe: A Computer Operation Tool for Users with a Neuromuscular Disease

Torsten Felzer[1], I. Scott MacKenzie[2], and Stephan Rinderknecht[1]

[1] Institute for Mechatronic Systems, Technische Universität Darmstadt, Darmstadt, Germany
{felzer,rinderknecht}@ims.tu-darmstadt.de
[2] Department of Computer Science and Engineering, York University, Toronto, Canada M3J 1P3
mack@cse.yorku.ca

Abstract. We developed a tool based on a modified number pad aimed at empowering persons with neuromuscular diseases to efficiently operate a computer and enter text. As the keypad lies securely in both hands, the system is ideal for someone who has motor problems using a full-size keyboard but cannot use speech recognition as an alternative method, because of dysarthria. The software offers various assistive techniques; for example, text entry is facilitated with the help of word prediction. An ambiguous mode with word-level disambiguation allows text entry with six keys. Initial empirical results with the system – which is already in regular use – indicate that it indeed represents a viable alternative, since it decreases effort without increasing the time to operate a computer.

Keywords: human-computer interaction, keyboard replacement, mouse emulator, word prediction, ambiguous keyboards, dysarthria, neuromuscular diseases, Friedreich's Ataxia.

1 Introduction

Be it for work or leisure, it is almost impossible today to avoid the computer. Entering text to write a scientific article, to compose an e-mail, or even to control a video game, is often the most important activity in this context. The standard input tools for human-computer interaction – a full-size keyboard and a mouse – allow for fast and efficient computer operation, provided the user is able to operate them.

For persons with disabilities who are often unable to employ a standard keyboard, there is a large variety of alternatives available. The problem is that most alternatives are much less efficient. Many scanning solutions, for example, are prohibitively slow, though only requiring a single input in the form of a switch activation [14,1]. The other extreme is speech recognition. Text entry using speech can even be faster than with the standard keyboard, but entry requires the ability to articulate clearly.

What if someone has a neuromuscular disease and cannot employ speech recognition because of dysarthria? And what if that someone must invest considerable effort to work with a standard keyboard, but is forced to make that investment, since the alternatives are too slow? These questions were the starting point for the research reported in this paper.

With the original objective to help a particular Friedreich's Ataxia (FA) patient (let's call him John), we developed several alternative interaction systems over the last few years. The goal was to help not only one person, but anyone with similar conditions: FA is an inherited, progressive neuromuscular disease that affects the neural pathways between the brain, cerebellum, and muscles [9] and leads to impaired muscle coordination. There is currently no cure for the disease [12], but the individual symptoms can be treated. In a sense, our newest solution – called *OnScreenDualScribe* – is such a treatment since it is intended to reduce the effort required to interact with a computer. The input device the system is tailored to is shown in Fig. 1. The novel idea is to prepare the unconventionally used device – with a form factor that is ideal for someone with a neuromuscular disease, as the keypad lies securely in both hands – to become a comprehensive assistive input tool, capable of conveniently replacing both the keyboard and mouse.

The remainder of the paper is organized as follows. In the next section, we look at various approaches concerning keyboard replacements (albeit mostly restricted to text entry only) reported in the literature. This is followed by an introduction of our solution in two sections dealing with the chosen input device and the software implementation. The system has been evaluated in an initial pilot study, which is described before the paper is concluded with a discussion of the outcome and a reiteration of the main points.

2 Related Work

With the progress in computer technology, methods to help individuals with disabilities enter text by providing a suitable keyboard alternative become more and more sophisticated. Earlier systems [5] use basic techniques such as word prediction [8] to facilitate input. Although operating a plain word prediction utility in the simplest form still relies on keyboard input, such an alternative is, in some sense, a replacement of the standard device with a more enhanced variant. Growing computational power makes it possible to analyze spoken words online. So speech recognition [18,6] can be used as an alternative.

There are many approaches in between that use an input device different from the standard keyboard, but not requiring extensive computational power. Examples include *EdgeWrite*, a text entry system based on two-dimensional traces representing individual characters [17], *Dasher*, which also uses a pointing device for text entry [16], as well as numerous approaches dealing with small, ambiguous keyboards [13,11], or switch-activated scanning systems [15].

Unfortunately, for an FA patient, these approaches either demands too much or too little. In other words, they either require unimpaired motor (or vocal) skills, or they simply do not take full advantage of the true abilities of the target population, thus making computing tasks unnecessarily slow.

Our system is intended as a computer control tool for patients with FA and similar diseases. As will become clear below, an efficient solution involves combining keyboard and mouse replacement in one adequate manual device.

3 Hardware Specifics

We wanted the input device supported in *OnScreenDualScribe* to be inexpensive, readily available, and convenient to handle. It became clear early on that the best choice would be a small, off-the-shelf number pad, turned by 90 degrees counterclockwise, with stickers attached to the keys, remapping them. We finally decided[1] on the device depicted in Fig. 1a, which is a wireless model (for maximum convenience). The stickers and the corresponding key layout are schematically shown in Fig. 1b.

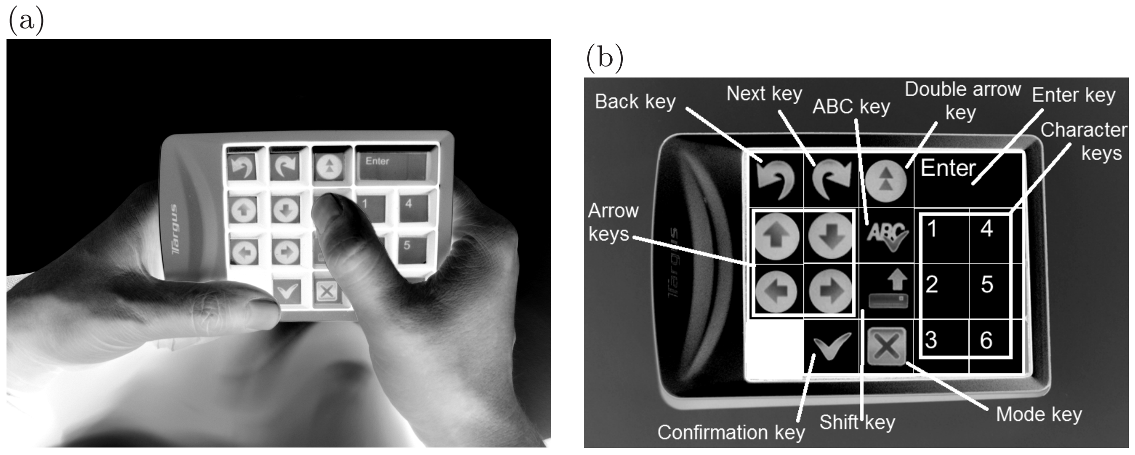

Fig. 1. *DualPad* 2.0: Inexpensive 18-key wireless number pad used for operating a computer; (a) usage: user can reach every key with either thumb without repositioning the hands; left and right edges serve as tactile guide when aiming at the keys; (b) stickers for the 18 keys with explanations.

The left two columns are operated with the left thumb and the remaining keys with the right thumb. The depicted layout is used in most contexts. *Ambiguous mode* is the only exception. Here, the location of some of the keys is slightly different. This is necessary since all character keys are in columns 4 and 5. Therefore, without change, the right thumb would do all the typing while the left thumb would be idle. The actually applied stickers were modified from Fig. 1b to also reflect the second layout.

[1] We tried about seven or eight different number pads until we found one that we considered acceptable.

4 Software Overview

OnScreenDualScribe is written in C++ under the Windows® operating system. It captures physical keystrokes on the number pad but does not pass them through to the active window (this is accomplished by defining all keys as “Hotkeys”). Instead, the software sends virtual input events to the active application. More details are given below.

4.1 General Architecture

The software operates in nine modes, each of which is responsible for different computing tasks. The 18 keys of the *DualPad* are re-mapped, depending on the currently active mode. The program window, which is shown as a narrow vertical stripe on top of all other windows (i.e., it is always visible) at the left or right edge of the screen (see Fig. 4a), displays several mode-specific indicators, for instance, which key is mapped to which function in the current mode.

To help novice users see the key associations, the key indicators resemble the physical layout of Fig. 1b. Two examples are illustrated in Fig. 2, which shows the indicators for *function mode* and *mouse mode* (with almost identical left hand keys).

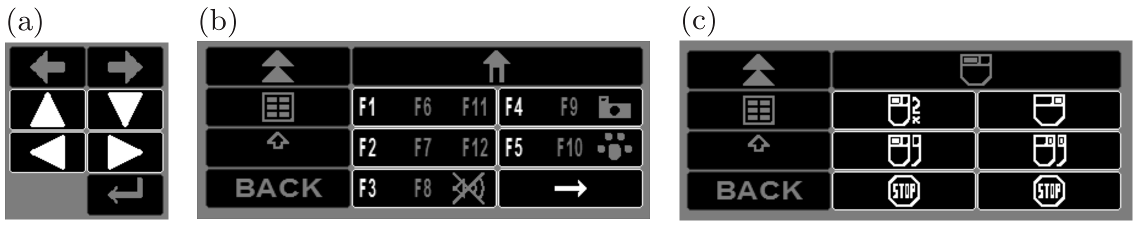

Fig. 2. Indicators for the keys in *function mode* and *mouse mode*: (a) left hand keys for scrolling (*function mode*) or moving the pointer (*mouse mode*) and navigating in a web browser; (b) right hand keys in *function mode* for evoking special virtual keystrokes; (c) right keys in *mouse mode* for generating clicks (left, right, single, double, drag).

Some of the *DualPad* keys directly lead to the generation of virtual keystrokes (or mouse events; see the subsection on pointing device operations). For example, pressing the “Confirmation key” on the *DualPad* in most modes generates an input event that corresponds to striking the Return key of a standard keyboard. Other keys either change the program state (e.g., in the course of selecting a word candidate; see below) or alter the currently active mode.

4.2 Text Entry

There are two different methods for entering text in *OnScreenDualScribe*. The basic method, also called *dual mode*, allows users to “write” the characters presented in the two-dimensional 8 × 6 virtual keyboard of Fig. 3a by selecting a row (with one or two keystrokes on the four arrow keys) and another keystroke on the character key that corresponds to its column.

Despite the labeling, the arrow keys are not used to move the highlight marking the selected row up or down. Rather, striking arrow key a_i once directly highlights row i, striking it twice highlights row $i + 4$ ($i = 1, ..., 4$). This type of selection scheme is demonstrated in Fig. 3b. Figure 3c shows the selected row on the character keys.

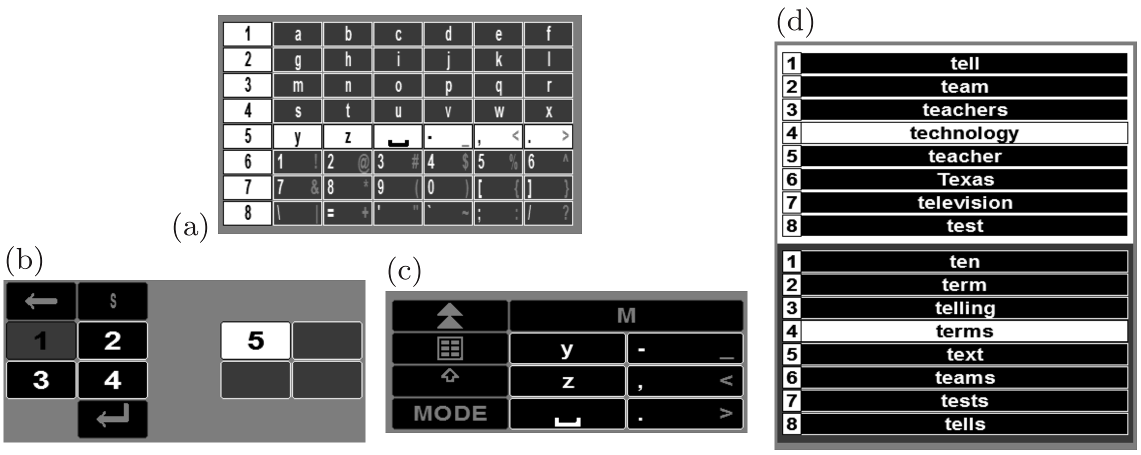

Fig. 3. Text entry in *dual mode*: (a) Virtual keyboard with eight rows and six columns showing the available characters (shifted versions for non-letter characters are shown in gray); (b) left hand keys – arrow keys (here labeled "1" – "4", cf. Fig. 2a) are used to highlight one of the eight rows of the virtual keyboard; (c) right hand keys – character keys correspond to the highlighted row; (d) word prediction list resulting from entering "te" (i.e., "$\rightarrow 2 \uparrow 5$")

With the exception of "y" and "z", all letter characters require exactly two keystrokes. Since the highlighted row resets to row five after choosing a character, the *KSPC* (keystrokes per character) for this input method is slightly less than 2.0. However, after each character, *OnScreenDualScribe* looks up the entered string in a dictionary (which is a text file containing 100,000 words, in the English version using the Corpus of Contemporary American English [2]) and presents a frequency-ordered list of extensions. Keystrokes are saved if the intended word is in the list. Selection of a suggested word is done analogous to the selection of a row of the virtual keyboard – the word list is divided into two sublists, and again, one of the eight rows/entries is highlighted in either sublist (see Fig. 3d, Fig. 5c, and Fig. 5d).

To some extent, word prediction brings this "open" text entry technique close to a full-size QWERTY keyboard (where *KSPC* is 1.0). Applying

$$KSPC = \frac{\sum_{i=1}^{n} (K_{w_i} \cdot F_{w_i})}{\sum_{i=1}^{n} (L_{w_i} \cdot F_{w_i})}, \tag{1}$$

to the newest edition of the dictionary file – where K_{w_i} is the number of keystrokes (at least) required to enter the word w_i, F_{w_i} is the frequency of w_i in the corpus, and L_{w_i} is the number of characters of w_i [10] – yields a *KSPC* of 1.1169 for *dual mode.*

The second text entry method implemented in *OnScreenDualScribe* consists of an ambiguous keyboard with six character keys and dictionary-based disambiguation. In *ambiguous mode*, character keys are used to compose a sequence of “code characters” represented by the digits 1 to 6. Each code character represents four or five characters of the alphabet (see Fig. 4b), and as the user types code characters, the software matches the code sequence among the words in the same dictionary file as above. The resulting frequency-ordered list of matching candidates contains at most 16 entries (selection as before), including extensions – if there are less than 16 candidates of equal length as the code sequence (see Fig. 4d).

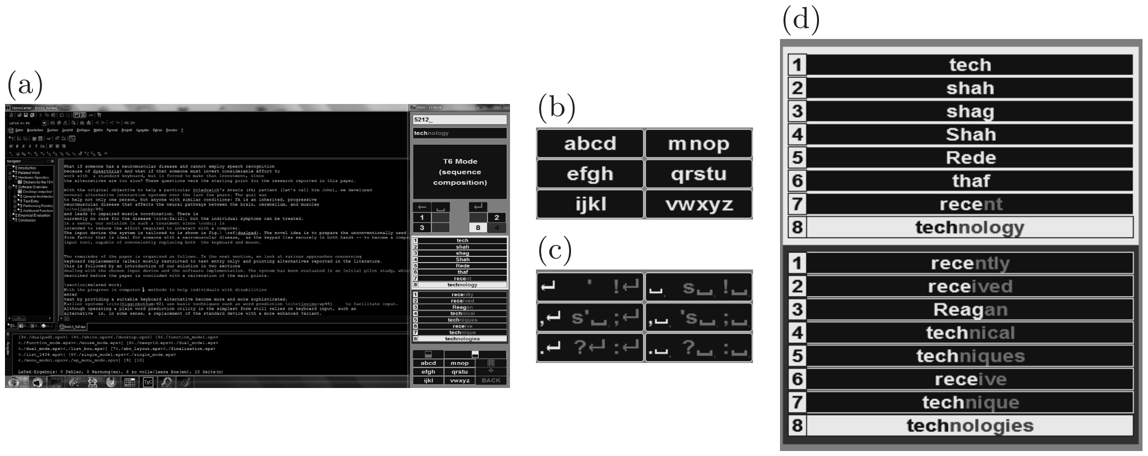

Fig. 4. *Ambiguous mode*: (a) desktop snapshot with program window at the right; (b) default arrangement where 26 letters are distributed over six keys in alphabetic order; (c) options to follow a selected candidate; (d) candidate list belonging to the sequence “5212”

This mode is more efficient than *dual mode* (*KSPC* is only 0.8678), but entering out-of-dictionary words is not possible. After selecting a candidate, the user chooses among several options for finalizing the selection (see Fig. 4c) by adding a space or basic punctuation (which reduces the need to switch to the basic technique between words).

4.3 Performing Pointing Device Operations

In addition to a keyboard, certain computing tasks (such as switching applications or clicking on a URL) require a pointing device or are hard to perform otherwise. Since the *DualPad* is held with both hands, using a manual device for pointing (like a mouse or a trackball) requires constant repositioning of the hands, which is very uneconomical.

Therefore, it was considered necessary to include a *mouse mode* (see Fig. 2c). The challenge was to make it possible to control the mouse pointer precisely, while allowing fast movement across the screen at the same time.

To do this, *OnScreenDualScribe* continually moves the mouse in one direction after the corresponding arrow key is pressed once. The initial speed is very low (ten pixels per second) and is reset when a different arrow key is pressed. However, pressing the same arrow key again accelerates the movement. When the pointer has reached the desired destination, the user generates a click or stops the movement with the right hand keys.

4.4 Additional Functionality

Operating a computer requires keyboard functionality that is beyond plain text entry. *OnScreenDualScribe* provides several ways to use special keys, like Shift, Ctrl, or Alt. An example is *single keystroke mode*. In this mode, a single keystroke directly triggers a virtual keystroke (as opposed to *dual mode*). It gives access to frequently used keys, like the (regular) arrow keys, Escape, Delete, or Paste (Figs. 5a and 5b).

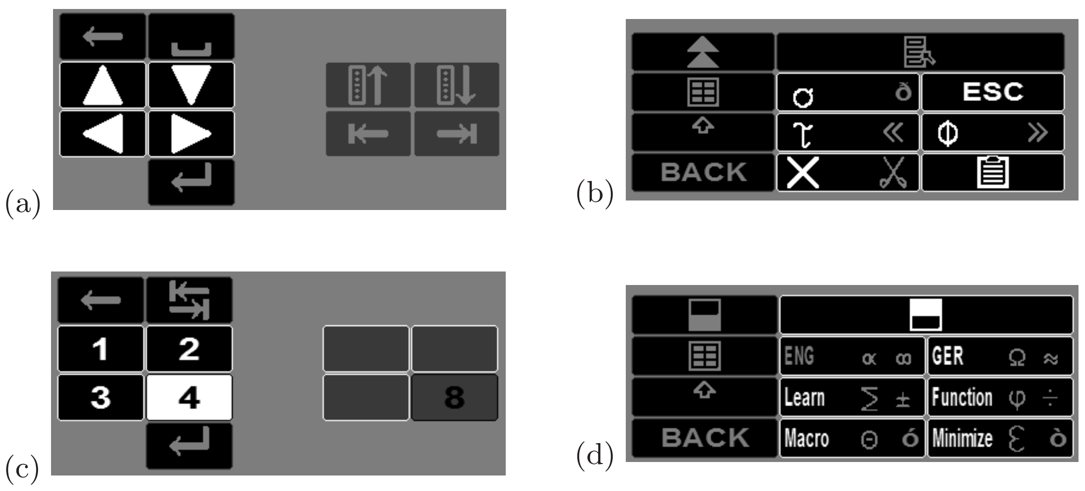

Fig. 5. Key indicators in *single keystroke mode*: (a) arrow keys to relocate text cursor; (b) right hand keys to access common keyboard functions; and *menu mode*: (c) arrow keys to control the highlight in word prediction sublists; (d) right hand keys to select desired sublist or trigger other modes.

Finally, there is a *macro mode* for sending arbitrary keystroke sequences, a *spell check mode* (to check the spelling of an entered word), and a *learning mode* to build and update a personal user-dependent dictionary supplement. The *menu mode* – which also allows users to select word prediction candidates or to toggle between English and German language versions (at runtime!) – is used to activate these miscellaneous modes (Figs. 5c and 5d). The German dictionary is based on [7]) – however, statistical data given in our paper (i.e., *KSPC*) refers to the English version only.

5 Empirical Evaluation

The system was originally built for John, a 41-year-old FA patient, who regularly uses *DualScribe* [3], a predecessor system which is limited to a proprietary editor window and is thus unable to control arbitrary applications. John helped evaluate a simpler variant of the tool, which was based on a game controller [4]. A pilot study involved entering portions of a 2,100-character text over the course of five days for at least two hours per day.

He repeated the experiment with the new system, only this time entering the entire text each day (which almost always took less than 120 minutes). On the old system, his entry rate was around 2 wpm; now it is over 3.5 wpm. With the standard keyboard, John is able to achieve entry rates between 2 wpm and 4 wpm, but this demands much more effort than *OnScreenDualScribe* due to frequent typing errors and the need to operate a full-size and cumbersome keyboard.

Furthermore, the first author included a very unconventional form of empirical evaluation: About 90% of this paper was "written" with *OnScreenDualScribe*. The fact that the first author did not switch to the standard keyboard after one or two days certainly speaks to the quality of the interface.

6 Discussion

As *OnScreenDualScribe* allows its users to enter text with fewer keys (often with fewer keystrokes) than usually needed, it reduces the physical load posed on the user. Always knowing which key to press next, when to look at the candidate lists, or deciding if the intended word is present (and where) certainly increases the cognitive load. However, judging this limitation requires considering the target population.

John switched to the tool and its predecessor as soon as they became available – at first, just for composing emails, later, for all computer interaction needs. By now, the tool has become an indispensable assistant that made his life a lot easier. He says: "of course, I have to concentrate, and when I am tired, my writing speed drops, but taking away the effort is much more important for me". In addition, his entry rate is now, more than half a year after the above mentioned experiment, regularly between 3 wpm and 5 wpm. As a reduced vigilance also resulted in lower entry rates when he was still using the standard keyboard, he is faster with the alternative method at any given point in time.

Besides, John has become so proficient in using the program that he can anticipate the number of letters required before a word appears near the top of the candidate list in *ambiguous mode*. He also developed strategies to easily circumvent problems like out-of-dictionary words (e.g., the word "neuromuscular" is not in the dictionary, but "neuro" and "muscular" are; as John knows that, he can quickly start a new word after having entered "neuro" when trying to enter the compound word in *ambiguous mode* – instead of entering the entire sequence, just to see that the word is *still* not in the list).

This is also the reason why conducting a usability study with a larger number of participants is particularly challenging. Non-expert users should practice with the software for a longer time (ideally several weeks or even months) to be able to make full use of the tool's assistive power. Finding someone who is willing to do that is not an easy task.

7 Conclusion

This paper introduced an assistive computer application, called *OnScreenDual-Scribe*, which is designed to replace the usual input method for interaction with a computer – involving a full-size keyboard and a mouse – by having the user press keys on an inexpensive, commercially available number pad. The system was developed with the immediate objective to devise a practical and efficient solution for John, a particular FA patient who participated in earlier studies evaluating text entry alternatives, and who also took part in a pilot study testing *OnScreenDualScribe*.

The broader idea is to help not only one person, but anyone with similar conditions. Since it is based on a small, compact manual device, it represents a viable alternative for persons with neuromuscular diseases, who are not able to use speech recognition, but who are (within limits) able to use both hands. Furthermore, able-bodied users looking for a small-size keyboard (e.g., for browsing the Internet on certain TV's) should also be interested in using the system.

After the pilot study, John immediately decided to completely switch to the new tool. The most eye-catching novelty in his workplace is the new 33" monitor he now uses – in his own words: "not a necessity, but a nice addition to my computer equipment, making it easier to scan the word lists". A standard keyboard and mouse are still on the table, "but only as a fallback option, in case Windows hangs".

The most important task for the future involves a larger usability study. In addition, debugging and improving the software – as well as designing a proprietary input device – are the next steps.

Acknowledgments. This work is partially supported by DFG grant FE 936/6-1 "EFFENDI – EFficient and Fast text ENtry for persons with motor Disabilities of neuromuscular orIgin".

References

1. Baljko, M., Tam, A.: Motor input assistance: Indirect text entry using one or two keys. In: Proc. ASSETS 2006, pp. 18–25. ACM Press (2006)
2. Davies, M.: Word frequency data from the Corpus of Contemporary American English (COCA), Downloaded from `http://www.wordfrequency.info` (January 27, 2011)

3. Felzer, T., MacKenzie, I.S., Rinderknecht, S.: DualScribe: A keyboard replacement for those with Friedreich's Ataxia and related diseases. In: Miesenberger, K., Karshmer, A., Penaz, P., Zagler, W. (eds.) ICCHP 2012, Part II. LNCS, vol. 7383, pp. 431–438. Springer, Heidelberg (2012)
4. Felzer, T., Rinderknecht, S.: Using a game controller for text entry to address abilities and disabilities specific to persons with neuromuscular diseases. In: Proc. ASSETS 2011, pp. 299–300. ACM Press (2011)
5. Higginbotham, D.J.: Evaluation of keystroke savings across five assistive communication technologies. Augmentative & Alternative Communication 8, 258–272 (1992)
6. Hirsimäki, T., Kurimo, M.: Analysing recognition errors in unlimited-vocabulary speech recognition. In: Proc. NAACL 2009, pp. 193–196. Association for Computational Linguistics (2009)
7. Institut für Deutsche Sprache: Korpusbasierte Wortformenliste DeReWo, v-100000t-2009-04-30-0.1, mit Benutzerdokumentation. Programmbereich Korpuslinguistik, Mannheim, Germany (2009), http://www.ids-mannheim.de/kl/derewo/
8. Koester, H.H., Levine, S.P.: Modeling the speed of text entry with a word prediction interface. IEEE Trans. Rehab. Eng. 2(3), 177–187 (1994)
9. Lecky, B.R.F.: Neuromuscular disorders: Clinical and molecular genetics. Brain 122(4), 1–790 (1999)
10. MacKenzie, I.S.: KSPC (keystrokes per character) as a characteristic of text entry techniques. In: Paternó, F. (ed.) Mobile HCI 2002. LNCS, vol. 2411, pp. 195–210. Springer, Heidelberg (2002)
11. MacKenzie, I.S., Felzer, T.: SAK: Scanning Ambiguous Keyboard for Efficient One-Key Text Entry. ACM Transactions on Computer-Human Interaction (TOCHI) 17(3), 11:1–11:39 (2010)
12. National Institute of Neurological Disorders and Stroke: Friedreich's Ataxia Fact Sheet, http://www.ninds.nih.gov/disorders/ friedreichs_ataxia/ detail_friedreichs_ataxia.htm (accessed on October 5, 2012)
13. Nesbat, S.B.: A system for fast, full-text entry for small electronic devices. In: Proc. ICMI 2003, pp. 4–11. ACM Press (2003)
14. Simpson, R.C., Koester, H.H.: Adaptive one-switch row-column scanning. IEEE Trans. Rehab. Eng. 7(4), 464–473 (1999)
15. Wandmacher, T., Antoine, J.Y., Poirier, F., Départe, J.P.: Sibylle, an assistive communication system adapting to the context and its user. ACM Trans. Access. Comput. 1(1), 1–30 (2008)
16. Ward, D.J., Blackwell, A.F., MacKay, D.J.C.: Dasher - a data entry interface using continuous gestures and language models. In: Proc. UIST 2000, pp. 129–137. ACM Press (2000)
17. Wobbrock, J.O., Myers, B.A., Kembel, J.A.: EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. In: Proc. UIST 2003, pp. 61–70. ACM (2003)
18. Zhang, W., Duffy, V.G., Linn, R., Luximon, A.: Voice recognition based human-computer interface design. Computers & Industrial Engineering 37, 305–308 (1999)