Applying Small-Keyboard Computer Control to the Real World

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] Dept. of Electrical Engineering and Computer Science,
York University, Toronto, Canada M3J 1P3
mack@cse.yorku.ca

Abstract. This paper presents a usability study for text entry with a new version of the assistive keyboard replacement *OnScreenDualScribe*. Over five sessions (approximately 1 hr/session), three able-bodied novice participants achieved an entry rate of 13.9 wpm. In a case study, one disabled expert achieved an entry rate of 6.6 wpm. The main aspects of the software are described and differences to the ancestor *DualScribe* are highlighted. Finally, the potential impact of the system for persons with neuromuscular diseases – a user group it particularly accommodates – is elaborated.

Keywords: Human-computer Interaction, Assistive Technology, Word Prediction, Ambiguous Keyboards, Neuromuscular Diseases, Keyboard Replacement, Mouse Alternative, Combined Input Device.

1 Introduction

OSDS (or *OnScreenDualScribe*) is a tool that replaces the standard PC input devices (i.e., a full-size keyboard and a mouse) with a single, compact device, while allowing efficient interaction. It consists of a numeric keypad and software translating physical keystrokes into virtual events directed at the currently active window.

Prospective users of *OSDS* are persons who are either unable or unwilling to employ standard input devices for controlling a computer. The former group particularly includes users with a neuromuscular disease, while the latter refers to mobile users or users operating an entertainment-centered PC. In both situations, it is important to replace the large keyboard with a small, albeit usable, alternative. For able-bodied users, this admittedly often involves touch-based devices, but those are unsuitable for many disabled users due to the absence of haptic feedback.

The input device for which *OSDS* is designed, called *DualPad*, has been subject to considerable development. It started as a game controller, evolved into a special-purpose keyboard, and ended up as an off-the-shelf numeric keypad with

K. Miesenberger et al. (Eds.): ICCHP 2014, Part II, LNCS 8548, pp. 180–187, 2014.

stickers attached to the keys. It is ideal for persons with a neuromuscular disease who often have specific problems using a full-size keyboard without holding on to anything. The *DualPad* is gripped firmly with every key reachable using the thumbs. Repositioning the hands is never necessary.

After considering the different research areas in the development of *OSDS*, the tool is briefly described, focusing on the new features. The section that follows introduces a usability study evaluating the text entry capabilities of the newest software version. Results are presented for three able-bodied participants and for the first author (as a disabled, but experienced supplement) transcribing more than 500 phrases (each between 20 and 40 characters long). A summary and a look to future work conclude the paper.

2 Related Work

Related work in association with the *OSDS* input technique involves several areas of human-computer interaction. One is *two-thumb text entry*, which refers to the way an input device is operated. Various realizations exist, for example split keyboards for touch-based applications [1] or mini-QWERTY keyboards, realized either as physical devices or as a soft keyboard on a touchscreen [2,3]. As mini-QWERTY keyboards generally contain a similar number of keys as a full-size keyboard, except with smaller keys, even physical realizations are not suitable for users with reduced fine-motor control.

Since *OSDS* is not limited to text entry, but offers emulation of pointing operations as well, a second related area is *combined input devices*: Every smart phone is an example. However, smart phones (and tablets) are touch-oriented and not suitable for every user. Of course, there are examples that add haptic feedback to a touchscreen using transparent tangible objects [4], but whether those solutions can effectively mimic a physical keyboard is unclear.

Another related area deals with the question of how to implement a *keyboard-driven mouse replacement*. A simple way is to assign keys to moving a mouse pointer in cardinal directions and other keys to the emulation of clicks at the current pointer position [5]. It will become clear below that this is not the only solution.

In addition, *OSDS*, is designed for *real-world applications*. For example, editing is not restricted to deletion of the character entered last (as for most proof-of-concept implementations, e.g., [6]), but cursor operations as well as copy, cut, and paste are supported. *OSDS* is not only a helpful assistant for text entry, it also offers mouse control. Single left clicks, double clicks, right clicks, and dragging operations are all supported (see also [7]).

3 System Description

The first author has the progressive neuromuscular disease Friedreich's Ataxia [8]. He developed *OSDS*, as an extension to *DualScribe* [9], with the objective to regain productivity lost due to the progression of his disease. The main extension

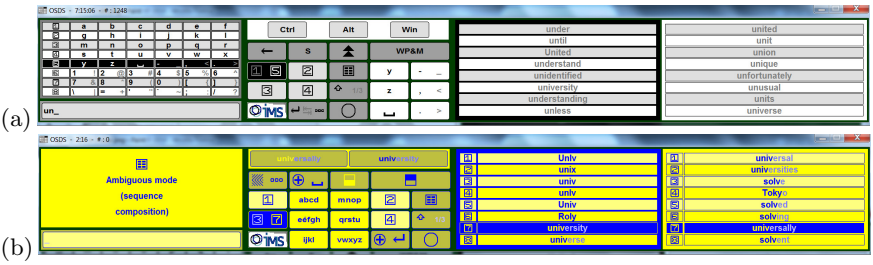

Fig. 1. Text entry in *OSDS*: (a) *dual mode* with (from left to right) character matrix, *DualPad* avatar, top-ranked completion candidates (#1 – #8), candidates ranked #9 – #16; (b) in *ambiguous mode*, the *DualPad* avatar illustrates a slightly modified key layout

is that *OSDS* behaves similarly to an onscreen keyboard, working as a mediator between the user and the active application.

Unlike its ancestor, *OSDS* does not rely on a dedicated editor window: It does not even require input focus. Instead, it intercepts the input signals before they reach the window in focus (which may belong to, for example, an email client) and generates new, virtual input events that are sent to the window. In doing so, the tool seamlessly interfaces with any existing application on the user's computer. This means that users of *OSDS* have full access to the editing functionality offered by other programs (without being restricted to a single proprietary editor).

All computing tasks a user may encounter are grouped into a dozen modes, all offered by the current software. The *DualPad* key triggering an action depends on the current mode. However, the program window shows an avatar of the *DualPad* which reveals the valid key associations.

The major goal in devising the tool was to replace the regular keyboard with a smaller alternative while optimizing for text entry. The text entry component, which shares basic ideas with the older version, comprises two different input methods.

In *dual mode* (fig. 1a), printable characters are arranged in a rectangular matrix, with corresponding keys emulated by selecting the coordinates of the characters. While entering a prefix, the software looks for completions in an internal dictionary and suggests selectable candidates.

Ambiguous mode works like T9 (known from the numeric keypad of mobile phones) with six ambiguous keys. The candidate lists suggest matches of the entered key sequence using the same dictionary (fig. 1b depicts the situation after entry of four letters).

Even though *DualScribe* caters to a proprietary editor window, the first author began to use it in practice, copying the entered texts and pasting them into other edit controls. However, this was almost as cumbersome as the regular keyboard, since using a pointing device for starting or activating other programs still required constant repositioning of the hands. Therefore, it was decided to include an internal mouse mode in the next version [10].

Initially [11], mouse control was realized in the form of a keyboard mouse, but since keyboard mice require pressing or releasing keys at exact points in time, this solution was far from optimal: Many persons with neuromuscular diseases have difficulties with temporal synchronization because they have a longer reaction time.

As an alternative, the newest version of *OSDS* implements a mouse mode that is not time-critical. Target acquisition is done in a stepwise fashion by recursively selecting tiles and sub-tiles on the screen, terminating with a click (of any type) at the target position.

In addition, *OSDS* offers a huge number of small (and some not-so-small) features, for example (to name a few), the operating system's task switcher can be called quickly, the program window may be hidden while the tool works in the background, the language environment can be switched at runtime, and typematic delay and repeat rate can be configured for ease of use. It is this kind of miscellaneous functionality that turns a promising idea into a practical input device replacement.

4 Evaluation

Our initial evaluation of *OnScreenDualScribe (OSDS)* 2.0 is now described. The evaluation focused on the *dual mode* text entry method depicted in fig. 1a. The evaluation involved 15 hours of testing with able-bodied users. Instead of testing 15 participants, one hour each, 3 participants were tested over 5 one-hour sessions each. Thus, patterns of learning should emerge. After presenting the initial evaluation, a case study with a member of the target user community is described.

4.1 Participants

The participants were members of the local community at the second author's university. There were two males, one female, with a mean age of 21.0 years (*SD* = 1.4). All are regular users of computers reporting usage of 4.3 hours per day. None had prior experience with *OSDS*.

4.2 Apparatus

The *DualPad* hardware consisted of a Perrix numeric keypad (`www.perix.com`) with keytop labels affixed for operation with the *OSDS* software (see fig. 2a). The keypad is connected via a USB cable to a host desktop computer running *Windows 7*.

There were two components to the software. The *OSDS* software captured the input signals and preprocessed them according to the current mode (i.e., *dual mode* text entry). *OSDS* presents a UI showing the current mode (see fig. 1a). Most characters require two keystrokes. A keystroke with the left thumb selects a group of characters. This is followed by a keystroke with the right

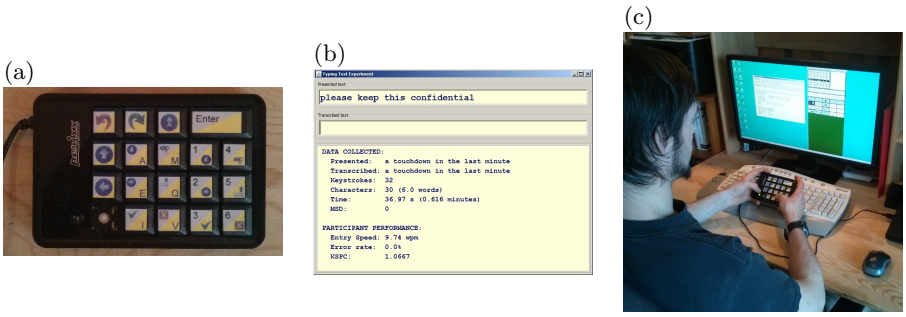

Fig. 2. Usability study: (a) Perrix numeric keypad with keytop labels for *OSDS*; (b) *Typing Test Experiment* evaluation software; (c) a participant entering text using *OSDS* and *Typing Test Experiment*

thumb to select the character within the group and transfer the character to the application in focus. As entry proceeds, up to 16 candidate words are presented in the bottom half of the UI (see right two images in fig. 1a). If the desired word appears, early word selection is possible with two key presses: "WP+M" on the right thumb (see second image of fig. 1a) followed by a left-thumb key press to select the word.

The evaluation software was *Typing Test Experiment*, a general-purpose text entry evaluation tool written in Java[1] ([12], p. 317). The tool randomly selects a phrase of text from a set and presents it in a text field. The user enters the phrase using the current text entry method (in this case, *OSDS* in *dual mode*). The transcribed phrase appears in a separate text field. When the user presses ENTER, the results appear in a static textbox and a new phrase is presented for entry. An example is shown in fig. 2b. Keystroke data, timestamp data, and summary statistics are also written to a disk file for follow-up analyses.

4.3 Procedure and Design

After a brief introduction and demonstration, testing began. Each session (aka "day") consisted of 3 practice blocks following by 10 data-collection blocks. Each block consisted of three phrases of entry. There were five sessions scheduled on consecutive days (or sometimes with one or two intervening days). Fig. 2c shows a participant entering phrases.

4.4 Results and Discussion

The grand mean for entry speed was 10.9 wpm, ranging from 7.1 wpm in session 1 to 13.9 wpm in session 5 (see fig. 3a). The grand mean for character-level error rates was 3.2% (see fig. 3b).

[1] The software is freely available as a download at `http://www.yorku.ca/mack/HCIbook/`.

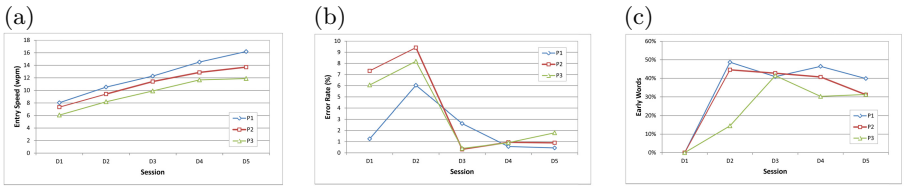

Fig. 3. Results for (a) entry speed, (b) error rates, and (c) early word selections

The unusual pattern for error rates is likely due to basic learning in session 1, learning to use early word selection in session 2, and the acquisition of skill in sessions 3, 4, and 5.

As noted, participants did not use early word selection until the second session. The percentage of words selected early is shown in fig. 3c. With practice, participants settled in to a routine whereby 30–40% of the words are selected early.

The results of the initial evaluation provide a general validation for text entry using *OSDS* in *dual mode*. We now describe a case study with a member of the target user community.

4.5 Case Study

The first author (who has Friedreich's Ataxia) entered text using *OSDS* in *ambiguous mode* along with *Typing Test Experiment*. Entry was performed in one day with a total of 75 phrases, presented in 15 blocks with 5 phrases each. The overall mean entry speed was 6.6 wpm (fig. 4) The mean error rate (not shown) was 0.5%.

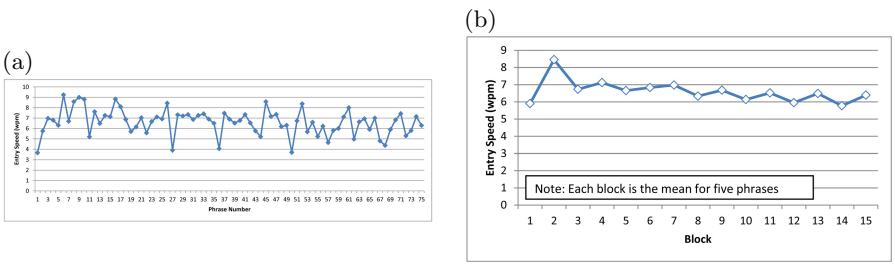

Fig. 4. Entry speed by the participant in the case study: (a) all 75 phrases; (b) 15 blocks with 5 phrases each

Fig. 4a might give the impression that the entry rate is very jittery, erratically jumping between a low of 3.7 wpm and a high of 9.2 wpm. However, as seen in fig. 4b, the average entry rate is rather constant – only block 2 is a (statistical)

outlier. This shows that, in contrast to the novice users, there is little or no learning taking place.

Interestingly, the experiment was nearly identical to the one conducted with *DualScribe* about two years earlier which yielded an entry rate of slightly over 4 wpm for the first author. Both tools share the same *ambiguous mode* and the input devices are very similar. Since both experiments also used the same pool of phrases, the only cause for the >50% increase in entry rate can be the experience gained in the intervening time.

Furthermore, the preparation of this paper represents an intrinsic demonstration of the usability of the system, since the first author has written large parts by himself: As a consequence of generally using little else to control a computer, he employed *OSDS* for nearly all tasks during composition. The only exception were two keystrokes on the snapshot key[2] to produce figs. 1a and 1b. Therefore, it is clear that the tool can effectively be used in practice – the first author can use it, so others should be able to as well.

5 Conclusion

This paper presented a computer control method to replace standard input devices with a single device and new software to achieve a compact and powerful input system that is well-suited for persons with certain disabilities. The most important property of the software, called *OSDS*, is its suitability for practical broad-based use instead of being restricted to particular computing tasks.

The system is in daily use by the first author who has Friedreich's Ataxia and growing motor problems since late childhood. His experience shows that it is truly life-changing. Use of a regular keyboard (which had required ever-increasing physical effort) for general tasks, and text entry in particular, became slower and slower. For entry speed, *OSDS* has turned back the clock by about ten years; the effort is gone almost completely.

However as the tool is quite complex, novice users require considerable practice (up to several months) before they are able to make full use of its power. The foremost challenge for the future is to enlist test participants and to encourage persons with similar diseases as the first author that the time invested is worth it.

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. Oulasvirta, A., Reichel, A., Li, W., Zhang, Y., Bachynskyi, M., Vertanen, K., Kristensson, P.O.: Improving two-thumb text entry on touchscreen devices. In: Proc. CHI 2013, pp. 2765–2774. ACM (2013)

[2] Of course, *OSDS* offers a mode to emulate the snapshot key, but when the tool is in other modes, striking that key on a regular keyboard cannot be avoided.

2. MacKenzie, I.S., Soukoreff, R.W.: A model of two-thumb text entry. In: Proc. Graphics Interface 2002, pp. 117–124. Canadian Information Processing Society (2002)
3. Clarkson, E., Lyons, K., Clawson, J., Starner, T.: Revisiting and validating a model of two-thumb text entry. In: Proc. CHI 2007, pp. 163–166. ACM (2007)
4. Weiss, M., Wagner, J., Jansen, Y., Jennings, R., Khoshabeh, R., Hollan, J.D., Borchers, J.: SLAP widgets: Bridging the gap between virtual and physical controls on tabletops. In: Proc. CHI 2009, pp. 481–490. ACM (2009)
5. RH Designs: Mouse Emulator, `http://rhdesigns.browseto.org/mouseemulator.html` (accessed on January 17, 2014)
6. Felzer, T., Strah, B., Nordmann, R.: Automatic and self-paced scanning for alternative text entry. In: Proc. IASTED Telehealth/AT 2008, pp. 1–6 (2008)
7. Jansen, A., Findlater, L., Wobbrock, J.O.: From the lab to the world: Lessons from extending a pointing technique for real-world use. In: Ext. Abstracts CHI, pp. 1867–1872. ACM Press (2011)
8. Delatycki, M., Williamson, R., Forrest, S.: Friedreich Ataxia: An overview. Journal of Medical Genetics 37(1), 1–8 (2000)
9. 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)
10. Felzer, T., Rinderknecht, S.: Mouse mode of OnScreenDualScribe: Three types of keyboard-driven mouse replacement. In: CHI EA 2013, pp. 1593–1598. ACM Press (2013)
11. Felzer, T., MacKenzie, I.S., Rinderknecht, S.: OnScreenDualScribe: A computer operation tool for users with a neuromuscular disease. In: Stephanidis, C., Antona, M. (eds.) UAHCI/HCII 2013, Part I. LNCS, vol. 8009, pp. 474–483. Springer, Heidelberg (2013)
12. MacKenzie, I.S.: Human-computer interaction: An empirical research perspective. Elsevier (2013)