Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 4 / ISSUE 1 / APRIL 2009 / 101-112
© 2009 Universidad de Talca - Chile

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762009000100009

Text Entry in the E-Commerce Age: Two Proposals for the Severely Handicapped

Julio Miró-Borrás[1] and Pablo Bernabeu-Soler [2]

Polytechnic University of Valencia, Department of Communications,
[1] jmirobo@dcom.upv.es, [2] pbernabe@dcom.upv.es

Received 4 September 2008; received in revised form 5 January 2009; accepted 5 March 2009

Abstract

People with severe motor disabilities have extreme access difficulties with all kinds of web services especially when they want not only to surf the web, but also write some text, e.g., to participate in an e-activity. Several problems arise when using traditional scanning systems, such as the low text entry rate, the time consuming task of learning the scan matrix layout, or simply, the poor visibility of the web page due to the large surface needed to display the complete scan matrix on the screen. We propose a reduced virtual keyboard based on scanning with only one switch as input device. The scan matrix consists of only three cells, so ambiguity is present due to the assignment of 26 characters to the three keys. Word-level and character-level disambiguation modes are explored using a mathematical model, and the text entry rates for an expert user were 15.9 and 10.3 words per minute respectively, using a scan period of 0.5 seconds. This keyboard could be embedded into a web page using a Java applet, JavaScript code or a Flash application, or be programmed as an independent application.

Key words: Handicapped, Text Entry, Scanning, Disambiguation, Ambiguous Keyboard, E-Commerce

1 Introduction

The World Wide Web in general and e-commerce in particular play an important role in contemporary society. Nevertheless there are special groups of people who cannot take advantage of the recent technological advances, creating greater social differences between them and able bodied society. This is what is known as the digital divide. Following the publication of the web content accessibility guidelines (WCAG) release 1.0 by the World Wide Web Consortium in 1999 [4], there is a growing recognition that users with disabilities have the same rights as others to access Information Technology with opportunities and risks [1]. According to World Bank estimates, 10% of the world's population (over 600 million) suffers from some kind of disability [30], [20]. Taking into consideration this figure, two main points of view arise: social and economic. The first refers to the inclusion rights of everybody; the second refers to the potential business that the disabled represent.

The WCAG help disabled people to access the web. Nevertheless not all groups have the same opportunities. These guidelines help the group of severely handicapped people, especially those with severe motor disabilities, to navigate through the web pages of the site, for example, permitting them to jump from one hyperlink to another simply by pressing the “tab” key in a standard keyboard, but, what can users do if they cannot use a keyboard?. How can they enter text to participate in what is called e-democracy, for example to express their opinion on a particular blog, wiki or mailing list? People with motor disabilities use a wide variety of creative solutions to control both their environment and computers. In general, a motor disabled person does not have the ability to efficiently use a multi-dimensional input device such as a keyboard, a joystick or a mouse. Users with a severe motor disability such as quadriplegia have the capacity to produce “single bit” signals [37], [29].

Communication aids are devices developed or specially adapted for people with severe communication impairments. There is a wide variety of communication aids because these people have a large variety of skills, needs, and problems. Some people with severe motor disabilities can use their hands; others cannot, and have to use alternatives, such as mouth-sticks, head-sticks, switches, or eye-pointing devices. In general, most communication aids for people with severe motor disabilities are designed to work with or to emulate a keyboard. Switches can be operated using their head, hands, arms, knees, feet, legs, shoulders or any body part over which they have muscular control. Other kinds of switches work by detecting movement such as a tilting arm or head, making a sound or breaking a beam of light. It is possible to find a special kind of switch called “Sip-Puff device” which works with breath.

Text entry is done using virtual keyboards on a screen and a scanning technique. Scanning interfaces move the focus of control in a grid, sequentially and automatically from item to item, with a standard timeout between moves. The user needs only to press the switch in order to select the item, which usually contains a letter. This is known as automatic scanning [40]. Another possibility is inverse scanning. In this case, when the switch is activated and held on, the scan starts and moves along as with automatic scanning. When the switch is released the scan stops and the item is highlighted. The selection is done using a second switch or by waiting for a dwell selection time without pressing the switch again.

Several problems arise when using traditional scanning systems such as the low text entry speed rate, the time consuming task of learning the layout of a scan matrix, or simply, the poor visibility of the application or web page due to the large surface needed to display the complete scan matrix on the screen. Efforts to increase the speed of text entry fall into several categories [37]: (1) rearrangement of the layout of the scan matrix; (2) grouped access to cells, typically RC (Row-Column) scanning; (3) adjustment of parameters such as scanning delay or dwell selection time; and (4) prediction of next block (mainly characters, words and sentences). These proposals to increase the speed do not address the other drawbacks.

Small devices such as mobile phones use reduced keyboards with fewer keys than letters so they are ambiguous keyboards. In order to disambiguate, mobile phones typically use two disambiguation modes. One of them is multitap, where characters are disambiguated one after another. The other is based on a dictionary, and the disambiguation is done at the end, when the user has entered all the characters of a word (one keypress for each character).

This paper describes our research into switch based text entry systems using a reduced scan matrix with the purpose of obtaining “interesting” text entry rates with minimal training and cognitive load. The 26 letters of the English alphabet are mapped onto only three keys so it is necessary to use a disambiguation process in order to tell the system the intended selection. This paper compares two modes of disambiguation over the same virtual keyboard. Both disambiguation modes are based on mobile phone text entry techniques, but considering scanning instead of pressing directly the keys. Moreover, the number of cells is only three instead of 9 or more. The first proposal is the word-level disambiguation mode: for each letter of a word, the user selects the cell that contains this letter, and at the end, disambiguates the full word. The second proposal is the character-level disambiguation mode: each letter of a word is disambiguated after selecting the cell that contains that letter. Based on a mathematical model, text entry speeds are obtained for expert users. Although the results presented here were obtained using a computer program, the system might be implemented on a web site and be downloaded in the form of a Java applet, JavaScript code or even a Flash application. This would permit severely physically handicapped people to participate in e-commerce or e-democracy activities or even if they were using a traditional scanning system, improve the text entry rate by using this proposal. As an input device, only a switch is necessary, which could be connected to a USB port of any computer.

The purpose of this paper is to present a novel text entry system based on scanning that permits users to write text faster than with the previous scanning systems. The learning period, although not specifically studied yet, is supposed to be shorter because of the alphabetical ordering of letters over the three cells. Finally, the surface needed to display it is smaller due to the use of only three cells instead of a typical scan matrix such as the 7x8 one used in the Tufts Interactive Communicator (TIC) [3]. Moreover, we present a mathematical model that can be used to evaluate and compare different reduced layouts using both proposed disambiguation modes.

The remainder of this paper is organized as follows. We review the related literature in next section. We present in section 3 the proposals, specifically the keyboard layout, the disambiguation modes and the mathematical model. We present the results of applying the model to the proposals and discuss their relevance in section 4. Finally, section 5 concludes this paper.

2 Related Literature

Most research in this area was done in the 80’s and 90’s when typical rates for one-switch systems were 0.5 to 5 words per minute (wpm) as Darragh & Witten summarize in [7]. They compiled data about proposals of different researchers and presented it in a way that allows us to compare them. Nevertheless, data available from scanning systems are quite inaccurate and it’s difficult to make comparisons between keyboards because of: (a) differences in the character set and input interface; (b) different scan periods, not always reported by the researchers; (c) number and type of switches and (d) different prediction systems and dictionary sizes. Results for some systems are reported in wpm, and for others in switch savings with respect to another system, typically TIC. The best systems with their rates expressed in wpm collected by Darragh & Witten were: TIC (5 wpm), ANTIC (or Anticipatory TIC, 6.8 to 7.8 wpm), MCCS (or Micro-computer Communication and Control System, 6.0 to 7.5 wpm, but two buttons) and CDC (or Communication and Device Control system, 10 wpm, but two buttons).

Damper [6], presented a mathematical model in order to compare the text entry speed of scanning systems, and applied it to a scan matrix similar to TIC with a scan period of 0.5 seconds. He predicted a text entry rate for this system of about 6.5 wpm. Lesher *et al*. [23] compared several layouts, with different sets of items, with/without character or word prediction, in terms of keystroke savings. Their best result was an average switch saving of 43.3% over the baseline RC layout, using an optimized configuration with a seven element character prediction list and a 7x7 elements scan matrix. Nevertheless, as the author recognizes, one cannot expect these gains to translate directly into communication rate improvements due to the dynamic updates of the display after the selection of each character.

All the previous references describe systems based on a scan matrix (with one character per cell). Next we present some references describing systems based on scanning that use ambiguous matrices (more than one character per cell). Kühn & Garbe presented a system called UKO (abbreviation for German "Unbekanntes Kommunikationsobjekt" or "unidentified communication object") with the letters mapped onto four keys plus two auxiliary keys [21]. They reported a text input speed of 6 wpm, using two switches by a 15-year old girl with cerebral palsy. Recently, Harbusch & Kühn [14] presented a study about five virtual keyboards, some of them ambiguous, with the results in scan steps per word, concluding the convenience of using ambiguous keyboards to obtain better results, and subsequently described an application called UKO II with the letters mapped onto three keys plus an auxiliary key [15]. No speed was reported. Predictions in UKO and UKO-II are shown in a separate list, so users need to see that list while introducing text. In a previous work [31], we presented a two cell keyboard with character level disambiguation, predicting a text input speed for expert user of 10.1 wpm. The present paper extends [32] with the word-level disambiguation mode.

In order to increase the text entry rate, most systems use prediction of the next block of text (mainly letters, words and sentences). Garay-Vitoria and Abascal summarize in [9] prediction applications and related features, such as block size, dictionary structure, prediction method, user interface and how the different languages affect prediction, for example for highly inflected languages. A more specific application of these techniques on augmentative and alternative communication (AAC) systems can be found in [33], [8], [2]. Some empirical research has been done with scanning systems augmented with text prediction [17], [18], [19], [36].

There are several papers related to ambiguous keyboards. A good introduction in text entry using an ambiguous keyboard and its disambiguation modes can be found in [27], [10], [13], [16], and empirical research is found in [5]. Mackenzie *et al*. [25] systematizes the use of KSPC (keystrokes per character) as a tool for a prior analysis, supporting the characterization and comparison of text entry methods before labour-intensive implementations. In the literature we found several contributions on physical and virtual keyboards with different number of keys and several layouts, where letters are assigned to keys in an alphabetical order, qwerty or even optimized, in order to maximize text entry speed, [24], [28], [38], [39]. Mackenzie *et al*. [26] presented a new technique to enter text using a prefix-based disambiguation method. This technique, called LetterWise, uses probabilities of letter sequences (prefixes) to guess the intended letter, so a dictionary is not necessary.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 4 / ISSUE 1 / APRIL 2009 / 101-112
© 2009 Universidad de Talca - Chile

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762009000100009

3 Proposal and Methodology

The typical scan matrix is replaced by a smaller one with only three cells, (Figure 1). The characters are arranged in alphabetical order unlike other common proposals such as qwerty or even optimised ones. Usually, this last kind of arrangement is the most difficult to learn and is more efficient than other layouts.

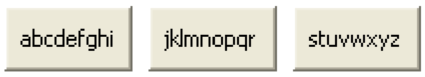

Figure 1: Layout of the three cell keyboard

As the number of characters is larger than the number of cells, the keyboard is ambiguous, so the presence of a disambiguation process to tell the system the intended characters when cells are selected is necessary. In this paper we propose two disambiguation modes: word-level and character-level disambiguation modes. The proposals are similar to the ones used in mobile phones, but taking into consideration that the user only uses a single switch, so scanning is used to highlight the different cells. In this case, instead of trying to minimize the movement of fingers following Fitt's Law [11], it is very important to minimize the number of scan cycles because it is the most time consuming action in scanning systems through the linguistic model as we describe later. In order to explain how a word is entered using both disambiguation modes, we number the keys as follows: key 1 is on the left and contains characters "a-i", key 2 is the one in the middle with characters "j-r", and on the right is key 3, with the character set "s-z". This way, word "help" is typed as 1122 because character "h" is on key 1, "e" on key 1, "l" on key 2 and "p" on key 2.

Since scanning systems are extremely slow, we have decided to use a linguistic model to optimize the number of scan cycles when entering text. A linguistic model is constructed using a corpus, which is an enormous collection of text of a particular language. In our work, we used the British National Corpus, which holds about 90 million English words (Site 1). Although the corpus is the same, we particularize the linguistic model for each disambiguation mode, as is explained later. From this corpus we generated several summary files with different number of the most frequent English words and their frequencies, used in the disambiguation process. The following sections describe the specific files for each mode.

3.1 Word-Level Disambiguation Mode

This proposal is an adaptation from the method "one key with disambiguation" from mobile phones. It uses only 3 cells instead of 9 keys in the standard mobile phone keyboard. It uses neither a separate key for space nor another one for the next function. Both of them are replaced with a combination of scanning modes: automatic and inverse scanning. The operation of this proposal is composed of two stages. The first is equivalent to the action of pressing the keys on a "physical" keyboard, but using automatic scanning. In the second, the user selects the desired word from a list of words, displayed one after another. The operation of the system is as follows: Initially the cursor is located on the most probable cell, and if it is not the desired one, the cursor advances to the next most probable cell, using automatic scanning. When the user presses the switch, the input is accepted, generating the code for that cell. All letters in the word except the last one are entered the same way. When introducing the last character, users must keep the switch pressed. This action reports the system that the word is over, and from this moment the second phase begins. Words that share the same code are displayed one after another using inverse scanning. When the desired word appears, the user releases the switch and this word is accepted. Of course, most probable words are shown first, in order to minimize the number of scan cycles. Figure 2 shows the finite state machine for this operation mode.

$$\begin{tikzcd}
\text{Edition} \arrow[loop left, "C_i"] \arrow[rr, bend left, "L_j"] & & \text{Next} \arrow[ll, bend left, "\overline{L}_k"]
\end{tikzcd}$$

Figure 2: Finite state machine for the word-level disambiguation mode

The basic states are "Edition" and "Next". The initial state is "Edition" and here, the scanning mode is "automatic". In the "Next" state, the system works in inverse scanning mode. The actions are:

C_i	Click the switch with cell *i* highlighted
L_j	Keep the switch pressed with cell *j* highlighted when last letter is reached
$\overline{L}_k$	Release the switch to select the desired word

A user that wants to enter the word "quick" should do the following actions:

1. For the four first letters, he or she clicks the switch when the corresponding cell is highlighted, producing the code 2311
2. As the last letter is "k", the user should keep the switch pressed when key 2 is highlighted, in order to inform the system this is the last character of the word
3. The system presents one after another, a list of words matching the code 23112. For example the list could be: "other", "queen", "quick"
4. The user releases the switch when the third word appears, and the word "quick" plus a space is entered into the system, for example, into a form field in an e-commerce web page

In relation to the linguistic model, using the corpus we generated a summary file with about 10,000 words (to be exact 10,911 words). This file contains all the words whose frequencies in the corpus are equal to or greater than 500. The reason for using such a reduced number of words is due to the impracticality of having enormous lists of words sharing the same code in the second stage. This file is used in different ways depending on the text entry stage. So, for the first stage, we obtained the probabilities of all characters in each position of a word, considering all the words in this file and their frequencies. Then, for each position, we obtained the cell probabilities simply adding the probabilities of each letter belonging to the cell. This way, cells are highlighted, in a probability order without considering the previous characters of the same word. The main objective is to permit the scan ordering of the cells to be fairly constant, at least in the same character positions. For the second stage, using the 10,000 words file we create a new file with an extra column: the code of each word considering the layout of the scan matrix. A word code is done changing each letter of a word by the cell number. So, the code of word “keyboard” for the layout in Figure 1 is 21312121. Note that all letters in the same cell share the same number. When second stage starts, only the words with the code just entered are shown one after another, the most probable first.

3.2 Character-Level Disambiguation Mode

This proposal is an adaptation from the “multitap” method used in mobile phones, and is inspired by the LetterWise proposal. It uses the same layout and number of cells as the former proposal. In this mode, disambiguation is done letter after letter, so once a cell is selected, it is necessary to start a disambiguation process in order to choose the right letter included in the cell. For this reason, the entry of each letter is composed of two stages. The first one is tantamount to pressing the key on a “physical” keyboard, but using automatic scanning. After clicking the switch the second stage starts. Now, letters in the cell are shown one after another using again automatic scanning. The user clicks the switch again when the desired letter appears, and a new cycle begins for next character in the word. All letters of a word are treated the same way except the last one. Keeping the switch pressed in the first stage when entering last letter tells the system than a space is required after that letter. The scanning mode in the second phase is different because the switch is still pressed. Now inverse scanning is used, and the selection is done simply by releasing the switch. The finite state machine of this mode is shown in Figure 3.

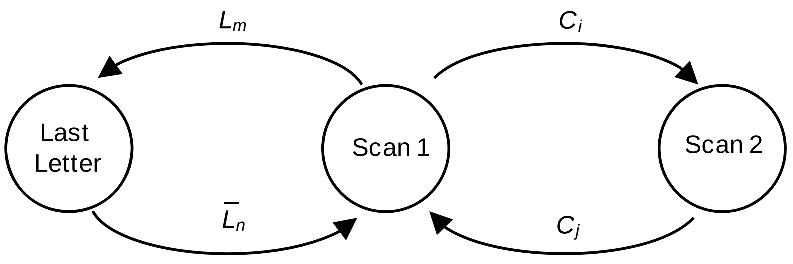

Figure 3: Finite state machine for the character-level disambiguation mode

The basic states are “Scan 1”, “Scan 2” and “Last Letter”. The initial state is “Scan 1”, which corresponds to the cell selection process. In “Scan 2” the character disambiguation is done and, in “Last Letter” state, the disambiguation of the last character is done, adding a space at the end. The actions are:

C_i	Click the switch to select highlighted cell *i*
C_j	Click the switch to select highlighted character *j*

L_m	Keep the switch pressed with cell *m* highlighted when last letter is reached
$\overline{L}_n$	Release the switch to select the last character and add a space at the end

A user that wants to enter the word "quick" should do the following actions:

1. For the four first letters, he does the following:
   - He or she clicks the switch when the corresponding cell is highlighted. For the first letter, he or she selects the cell 2
   - Then, the system shows one after another, all the characters in that cell, ordered by probability (see the linguistic model). For example, the list could be "o-p-m-r-l-n-j-k-q"
   - The user clicks when the desired letter appears, in this case, the last one
   - Repeat these actions for the following 3 letters in word quick, i.e., "u", "i" and "c"
2. As the last letter is "k", the user should keep the switch pressed when key 2 is highlighted, in order to inform the system this is the last character of the word
3. The system shows one after another, all the characters in that cell, ordered by probability (see the linguistic model). For example, the list could be "k-o-p-m-r-l-n-j-q"
4. The user releases the switch when the letter "k" appears, and the word "quick" plus a space is entered into the system

With regard to the linguistic model, using the corpus we generated a summary file with about 160,000 words (to be exact 160,258 words). This file contains all the words whose frequency in the corpus is greater than 2. This file is used in different ways depending on the text entry stage. So, for the first stage, we obtained the probabilities of all characters in each position of a word, considering all the words in this file and their frequencies. Then, for each position, we obtained the cell probabilities simply adding the probabilities of each letter belonging to the cell. This way, cells are highlighted, in a probability order without considering the previous characters of the same word. The main objective is to permit the scan ordering of the cells to be pretty constant, at least in the same character positions. For the second stage, using the 160,000 words file, we created the linguistic model based on k-grams and n-grams of fourth order (4-grams). These statistics were used to predict the next character in the disambiguation stage of scanning, using up to the three previous characters, and letters will be proposed the most probable one first and so on. We combined k-grams and n-grams in the following way: for the first 4 characters of each word the predictions are obtained from the k-gram model while for the rest of characters, the systems uses the fourth order n-gram model for the prediction of the next character. The difference between k-gram and n-gram is simple. In k-grams, prediction of one character is based on the previous ones, but always starting at the beginning of a word. Nevertheless, in n-gram model, the previous characters can start in any part of a word, even in a character belonging to the previous word, although this was not used in our model. Thus, the prediction of the first letter of a word uses only the character frequencies for word beginnings or 2-grams with first letter being a space. The prediction of 2nd, 3rd and 4th letters uses the k-gram model (just as 3-gram 4-gram and 5-gram respectively, with first character being a space). For the fifth character and above, the model used is 4-gram.

3.3 Mathematical Model

In order to analyze both proposals, we calculate the parameters described below. First of all, KSPW or number of keystrokes per word allows us to characterize ambiguous keyboards [39]:

$$KSPW = \sum_{w} K(w) \cdot P(w) \tag{1}$$

where *K(w)* is the number of keystrokes to enter word *w*, and *P(w)* is the probability of that word.

P(w) depends on the corpus and the linguistic model used and can be calculated this way:

$$P(w) = \frac{f_w}{\sum_{i=1}^{N} f_i} \tag{2}$$

where f_w is the frequency of word w in the linguistic model, and N is the number of different words in the linguistic model.

In a scanning environment, “keystrokes” are defined as any action taken by the user, such as pressing a switch, with the aim of advancing the cursor to the next item or selecting it. Cursor-movement delays when a single button is used have traditionally been considered keystrokes. From now on, we use *n(w)* instead of *K(w)* when referring to the number of keystrokes in scanning systems. As delays are different for each kind of keystroke, it is better to treat them independently and then obtain a weighted value. Therefore, we consider three types of keystrokes: $n_S(w)$, $n_C(w)$ and $n_L(w)$, where the former refers to the number of scan cycles, the second is the number of switch presses or clicks, and the last, the number of long presses on the switch to enter word *w*. For practical purposes, no distinction is made in the calculations between a click and a release of the switch. A parameter that comprises the different types of keystrokes is the weighted number of keystrokes to enter word *w*:

$$n_{weighted}(w) = w_S \cdot n_S(w) + w_C \cdot n_C(w) + w_L \cdot n_L(w) \tag{3}$$

and the average value is

$$n_{weighted} = \sum_w n_{weighted}(w) \cdot p(w) = w_S \cdot n_S + w_C \cdot n_C + w_L \cdot n_L \tag{4}$$

where w_S, w_C, and w_L are the weights for the scan cycles, the clicks and the long presses on the switch, respectively, and n_S, n_C and n_L are the average values of the respective number of keystrokes.

According to the approximation in Damper's paper [6], who assumed that a switch press required half the scan period, we use the following weights: w_S=1.0 and w_C=0.5. A reasonable duration for the system to detect a keypress as "long" might be half the scan period, so we could assume a duration of one scan period for a long keypress and a weight w_L=1.0. These assumptions will allow us to derive comparative figures for text entry rate.

The main parameter to evaluate the suitability of a text entry device for users is the maximum text entry rate an expert user can achieve when entering text, expressed in wpm. Its value depends on several parameters such as the different weights (w_S, w_C, w_L) and the scan period (*T*).

Rosen & Goodenough-Trepagnier [34] presented an equation in which the average time to enter a word (τ) was expressed as the product of three terms

$$\tau = C \cdot L \cdot T \tag{5}$$

where *C* is the linguistic cost, measured as the average number of characters per word, *L* is the average length of a sequence of input actions necessary to introduce a character, and *T* is the average time to carry out an input action. The product *C·L* is the average length of a sequence of input actions necessary to enter a word. Considering an input action as a keystroke,

$$KSPW = C \cdot L \tag{6}$$

therefore,

$$\tau = T \cdot KSPW = T \cdot n_{weighted} \tag{7}$$

where $n_{weighted}$ is the average value of the weighted number of keystrokes. Therefore, the text input rate, 1/τ in words per minute is shown in (8)

$$\frac{1}{\tau} = \frac{60}{T \cdot n_{weighted}} \quad (wpm) \tag{8}$$

3.4 Procedure

A computer program was written in Microsoft® Excel 2007, more specifically with Visual Basic® for Application (VBA), to evaluate the proposals. This program uses the corpus summary files and using the previous equations calculates the average values of the number of scan cycles, the number of switch presses, the number of long presses and finally the average weighted number of keystrokes. From this data, it is easy to estimate the text entry rate for a particular scan period.

4 Results and Analysis

Using the computer program and the linguistic models explained previously, in word-level disambiguation mode the application predicts the next cell in the first stage and the desired word from the list in the second one. In character-level disambiguation, the application predicts the next cell in the first stage and the desired character in the second one. The number of keystrokes for each mode is shown in Table 1 and Table 2. Although for each disambiguation mode we stated its own linguistic model, it is necessary to use the same set of words to obtain a comparable figure

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762009000100009

of the average number of keystrokes per word. This assures that the mean word length is the same and the results in both modes are comparable. In our tests, we used the reduced dictionary with the 10,000 most common words in English and their frequencies for both disambiguation modes.

Table 1: Detailed number of keystrokes for the three cell keyboard in word-level disambiguation mode

	Phase 1	Phase 2	Total
n_S	3.37	0.95	4.32
n_C	3.50	1	4.5
n_L	1.00	0	1
$n_{weighted}$	6.12	1.45	7.57

Table 2: Detailed number of keystrokes for the three cell keyboard in character-level disambiguation mode

	Phase 1	Phase 2	Total
n_S	3.37	3.26	6.63
n_C	3.50	4.50	8.00
n_L	1.00	0	1.00
$n_{weighted}$	6.12	5.51	11.63

The weights used for the calculation of $n_{weighted}$ were: n_S=1, n_C=0.5 and n_L=1. With these data and a scan period of 0.5 seconds, the entry rates obtained with (8) were 15.86 wpm in word-level disambiguation mode and 10.32 wpm in character-level disambiguation mode.

In order to see the influence of the scan period in the text entry rate, Figure 4 shows the wpm according to *T*, using w_C as a parameter in word-level disambiguation mode. Figure 5 shows the same relation for character-level disambiguation mode. Specifically, in both cases, the scan period varies between 0.2 and 2 seconds and three values of w_C are taken into consideration. The other weights are maintained fixed.

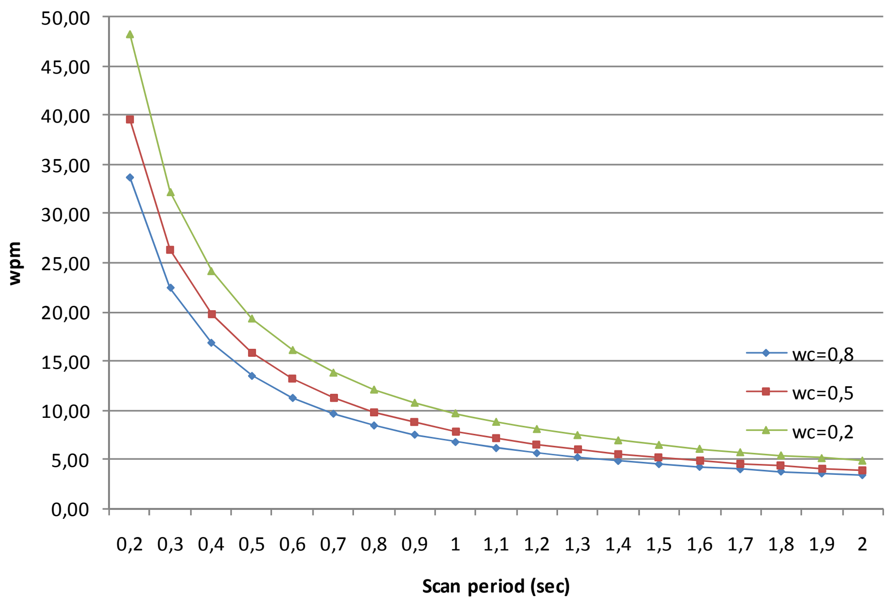

Figure 4: Text entry rate according to scan period with w_C as a parameter in word-level disambiguation mode

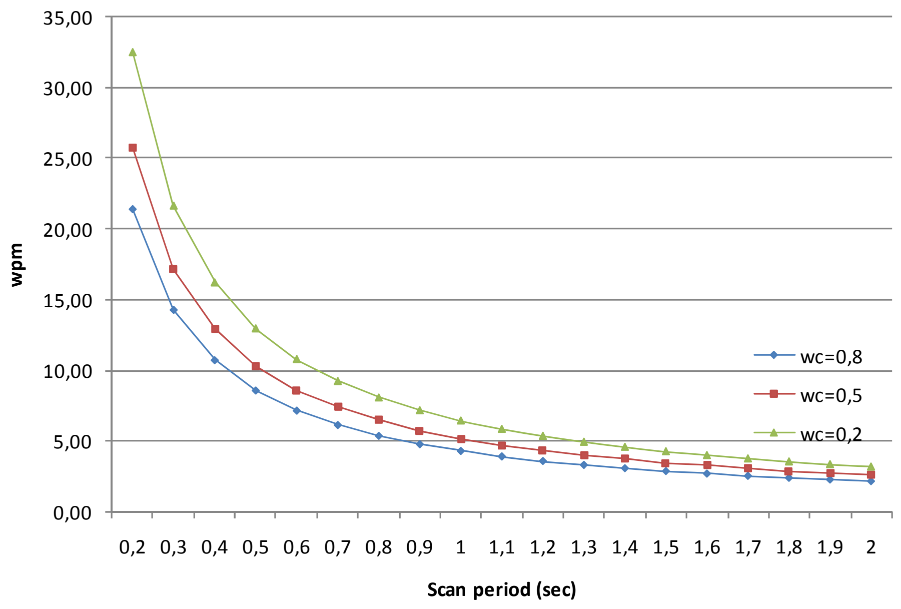

Figure 5: Text entry rate according to scan period with w_C as a parameter in character-level disambiguation mode

The text entry rate in wpm decreases as the scan period increases (Figure 4), (Figure 5), but weights have a large influence especially with low values of the scan period. For example, we obtained 15.86 wpm which matches up with w_C=0.5 and T=0.5 seconds (Table 3). Nevertheless, this value can vary from 13.46 to 19.31 wpm with values of w_C ranging between 0.8 and 0.2 respectively. In character-level disambiguation mode (Table 4), the rate is 10.32 for the same parameters, and ranges from 8.56 to 13.00 wpm.

Table 3: Text entry rate in word-level disambiguation mode

T	w_C=0.8	w_C=0.5	w_C=0.2
0.2	33.65	39.66	48.26
0.3	22.43	26.44	32.18
0.4	16.83	19.83	24.13
0.5	13.46	15.86	19.31
0.6	11.22	13.22	16.09
0.7	9.61	11.33	13.79
0.8	8.41	9.91	12.07
0.9	7.48	8.81	10.73
1	6.73	7.93	9.65
1.1	6.12	7.21	8.78
1.2	5.61	6.61	8.04
1.3	5.18	6.10	7.43
1.4	4.81	5.67	6.89
1.5	4.49	5.29	6.44
1.6	4.21	4.96	6.03
1.7	3.96	4.67	5.68
1.8	3.74	4.41	5.36
1.9	3.54	4.17	5.08
2	3.37	3.97	4.83

Table 4: Text entry rate in character-level disambiguation mode

T	w_C=0.8	w_C=0.5	w_C=0.2
0.2	21.39	25.80	32.51
0.3	14.26	17.20	21.67
0.4	10.69	12.90	16.26
0.5	8.56	10.32	13.00
0.6	7.13	8.60	10.84
0.7	6.11	7.37	9.29
0.8	5.35	6.45	8.13
0.9	4.75	5.73	7.22
1	4.28	5.16	6.50
1.1	3.89	4.69	5.91
1.2	3.56	4.30	5.42
1.3	3.29	3.97	5.00
1.4	3.06	3.69	4.64
1.5	2.85	3.44	4.33
1.6	2.67	3.23	4.06
1.7	2.52	3.04	3.82
1.8	2.38	2.87	3.61
1.9	2.25	2.72	3.42
2	2.14	2.58	3.25

Although in this paper we fix the scan period and weights, this is done merely to obtain an estimation of the performance, and to draw conclusions about the systems, but the parameters have to be measured running an experiment with users, and the scan period adapted specifically for each user [12], [22], [35].

Figure 6 shows the weighted number of keystrokes according to w_C for both disambiguation modes. These data are independent of the scan period. The weighted number of keystrokes for both modes increases with w_C, but the one in character level disambiguation mode in a faster way. Therefore, the text entry rate will increase more quickly when w_C decreases.

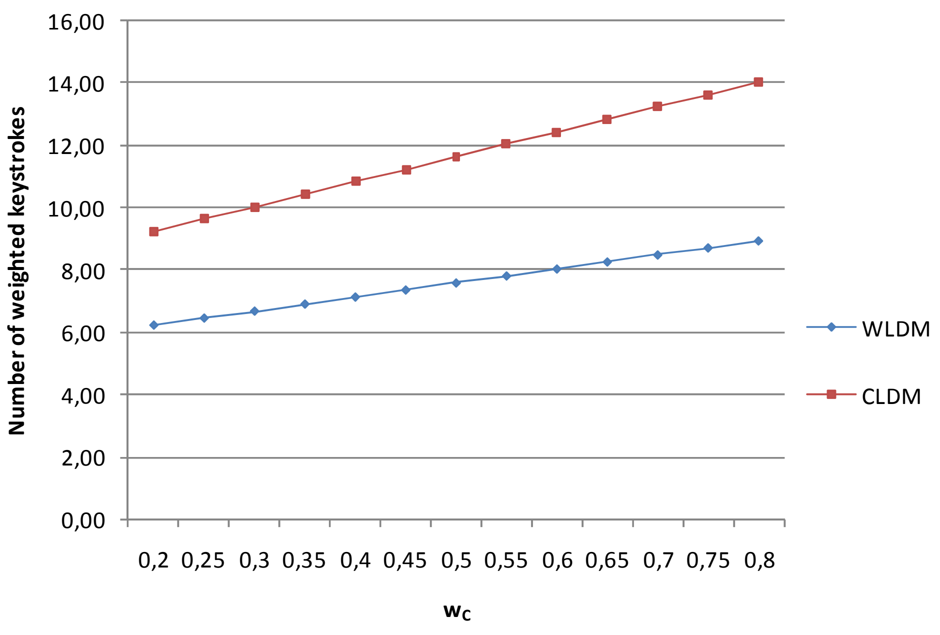

Figure 6: Number of weighted keystrokes according to w_C in word-level and character-level disambiguation modes (WLDM and CLDM respectively)

Although it is not analyzed in this paper, text entry rate speeds up with experience, so it is expected that users reduce scan period and typing errors after several training sessions.

As we said before, both systems predict the next cell in the first stage and the word or characters in the second stage using the linguistic models, but, what is the performance of the predictions like? Clearly, with a perfect prediction, the desired cell, letter or word would always appear in the first position, so there is no need to wait for any scanning delay. In this case, the maximum text entry rate with a perfect prediction is obtained from (4) and (8) with n_S=0. These figures are 37 wpm and 24 wpm for word-level and character level disambiguation modes respectively. Efficiencies of the three cell keyboard for both modes are respectively 42.9% and 43.0%, obtained from (9). These values are not very high, and in order to increase them, several actions could be taken in both stages in each mode. For example, use the n-gram or k-gram model for the prediction of the next cell in stage 1 for both modes. A higher order n-gram model or a k-gram only model could be used in stage 2 of character-level disambiguation mode. In the word-level disambiguation mode, the dictionary could be reduced and dynamically adjusted to the user vocabulary in order to reduce the prediction list of words and reduce the position of the desired word in the list.

$$efficiency(\%) = \frac{wpm_{REAL}}{wpm_{PERFECT\_PREDICTION}} \times 100 \tag{9}$$

5 Conclusions

Current text entry systems based on scanning present some problems such as a low text entry rate, a time consuming task of learning the scan matrix layout, and the need for a large screen surface to show the complete scan matrix. We present a reduced scan matrix of only three cells and two disambiguation modes inspired by the mobile phone text entry systems: word-level and character level disambiguation modes. The text entry rate in wpm depends on individual users, but we estimated 15.9 and 10.3 wpm using a scan period of 0.5 seconds for word-level and character-level disambiguation mode respectively. Both proposals complement each other because word-level disambiguation mode speeds up the typing process while with character-level disambiguation mode the user can type words not included in the dictionary. The main benefits are: a reduced number of clicks, particularly in the word-level disambiguation mode, which is essential for people with motor disabilities; a fixed alphabetical ordering of letters, which requires less mental effort compared with systems where part of the layout is changing dynamically; and an increase in the text entry rate compared with traditional scanning systems using only one switch.

This way, with a well designed e-commerce web site according to WCAG, users can navigate through the site by themselves and with the help of text entry systems based on scanning they will be able to participate actively in all the activities related to the site. Both parties will obtain important benefits: users are integrated in the Information Society and site owners have an opportunity of increasing profits.

The text entry systems presented in this paper may be improved with further research in the following areas: analysis of different layouts of the scan matrix; incorporation or editing keys and other characters such as numbers and punctuation items; exploration of different language models in order to increase efficiency; utilisation of a second contextual dictionary with the specific words needed to make a transaction in a e-commerce page, in order to present these words in the first position of the word list when using word-level disambiguation mode; and finally it is necessary to conduct empirical research with users also.

Websites List

Site 1: BNC database and word frequency lists, by Adam Kilgarriff, University of Brighton.
ftp://ftp.itri.bton.ac.uk/bnc/

References

[1] J. Abascal and A. Civit, Opportunities and risks of the information and communication technologies for users with special needs, in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Yasmine Hammamet, Tunisia, 2002, pp. 264-269.
[2] J. L. Arnott and M. Y. Javed, Probabilistic character disambiguation for reduced keyboards using small text samples, AAC Augmentative and Alternative Communication, vol. 8, pp. 215-223, 1992.
[3] G. Baletsa, R. Foulds, and W. Crouchetiere, Design parameters of an intelligent communication device, in Proceedings of the 29th Annual Conference on Engineering in Medicine and Biology, Chevy Chase, 1976.
[4] W. Chisholm, G. Vanderheiden, and I. Jacobs. (1999, May). Web Content Accessibility Guidelines 1.0, W3C Recommendation. [Online]. Available: http://www.w3.org/TR/WCAG10/.
[5] K. Curran, D. Woods, and B. O. Riordan, Investigating text input methods for mobile phones, Telematics and Informatics, vol. 23, pp. 1-21, 2006.
[6] R. I. Damper, Text composition by the physically disable: A rate prediction model for scanning input, Applied Ergonomics, vol. 15, pp. 289-296, 1984.
[7] J. J. Darragh and I. H. Witten, The Reactive Keyboard. New York, NY, USA: Cambridge University Press, 1992.
[8] P. W. Demasco, Human factors considerations in the design of language interfaces in AAC, Assistive Technology, vol. 6, pp. 10-25, 1994.
[9] N. Garay-Vitoria and J. Abascal, Text prediction systems: a survey, Universal Access in the Information Society, vol. 4, pp. 188-203, 2006.
[10] M. D. Dunlop and A. Crossan, Predictive text entry methods for mobile phones, Personal and Ubiquitous Computing, vol. 4, pp. 134-143, 2000.
[11] P. M. Fitts, The information capacity of the human motor system in controlling the amplitude of movement (Reprinted from Journal of Experimental Psychology, vol. 47, pp 381-391, 1954), Journal of Experimental Psychology: General, vol. 121, pp. 262-269, 1992.
[12] S. Ghedira, P. Pino, and G. Bourhis, Interaction between a disabled person and a scanning communication aid: Towards an automatic adjustment of the scanning rate adapted to the user, in ICCHP 2008, LNCS 5105 (K. Miesenberger *et al*. Eds.). Springer-Verlag Berlin Heidelberg, 2008, pp. 1204-1207.
[13] D. L. Grover, M. T. King, and C. A. Kuschler, Reduced keyboard disambiguating computer, U.S. Patent 5 818 437, October 6, 1998.
[14] K. Harbusch and M. Kühn, An evaluation study of two-button scanning with ambiguous keyboards, in Proceedings of the 7th Conference for the Advancement of Assistive Technology in Europe, Dublin, 2003, pp. 954-958.
[15] K. Harbusch and M. Kühn, Towards an adaptive communication aid with text input from ambiguous keyboards, in Proceedings of 10th Conference on European Chapter of the Association for Computational Linguistics, 2003, pp. 207-210.
[16] C. L. James and K. M. Reischel, Text input for mobile devices: Comparing model prediction to actual performance, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2001, pp. 365-371.
[17] H. H. Koester and S. P. Levine, Learning and performance of able-bodied individuals using scanning systems with and without word prediction, Assistive Technology, vol. 6, pp. 42-53, 1994.
[18] H. H. Koester and S. P. Levine, Modeling the speed of text entry with a word prediction interface, IEEE Transactions on Rehabilitation Engineering, vol. 2, pp. 177-187, 1994.
[19] H. H. Koester and S. P. Levine, Keystroke-level models for user performance with word prediction, Augmentative and Alternative Communication, vol. 13, pp. 239-257, 1997.
[20] P. Kotzé, M. M. Eloff, A. Adesina-Ojo, and J. H. P. Eloff, Accessible computer interaction for people with disabilities: The case of quadriplegics, in Proceedings of the 6th International Conference on Enterprise Information Systems. Porto, Portugal, 2004, pp. 97-106.
[21] M. Kühn and J. Garbe, Predictive and highly ambiguous typing for a severely speech and motion impaired user, Universal Access in Human-Computer Interaction, pp. 933-937, 2001.
[22] G. W. Lesher, D. J. Higginbotham, and B. J. Moulton, Techniques for Automatically Updating Scanning Delays, in Proceedings of the RESNA 2000 Annual Conference, 2000, pp. 85–87.

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762009000100009

[23] G. W. Lesher, B. J. Moulton, and D. J. Higginbotham, Techniques for augmenting scanning communication, Augmentative and Alternative Communication, vol. 14, pp. 81-101, 1998.
[24] G. W. Lesher, B. J. Moulton, and D. J. Higginbotham, Optimal character arrangements for ambiguous keyboards, IEEE Transactions on Rehabilitation Engineering, vol. 6, pp. 415-423, 1998.
[25] I. S. MacKenzie, KSPC (keystrokes per character) as a characteristic of text entry techniques, in Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction, 2002, pp. 195-210.
[26] I. S. MacKenzie, H. Kober, D. Smith, T. Jones, and E. Skepner, LetterWise: Prefix-based disambiguation for mobile text input, in Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology, 2001, pp. 111-120.
[27] I. S. MacKenzie and R. W. Soukoreff, Text entry for mobile computing: Models and methods, theory and practice, Human-Computer Interaction, vol. 17, pp. 147-198, 2002.
[28] I. S. MacKenzie and K. Tanaka-Ishii, Text entry using a small number of buttons, in Text Entry Systems: Mobility, Accessibility, Universality (I. S. MacKenzie and K. Tanaka-Ishii, Eds.). Morgan Kaufmann, 2007, pp. 105-121.
[29] J. Mankoff, A. Dey, U. Batra, and M. Moore, Web accessibility for low bandwidth input, in Proceedings of the 5th International ACM Conference on Assistive Technologies. Edinburgh, Scotland, 2002, pp. 17-24.
[30] R. L. Metts. (2000, February). Disability issues, trends and recommendations for the World Bank. The World Bank Website. [Online]. Available: http://siteresources.worldbank.org/DISABILITY/Resources/280658-1172606907476/DisabilityIssuesMetts.pdf.
[31] J. Miró and P. A. Bernabeu, Text entry system based on a minimal scan matrix for severely physically handicapped people, in ICCHP 2008, LNCS 5105 (K. Miesenberger *et al*. Eds.). Springer-Verlag Berlin Heidelberg, 2008, pp. 1216-1219.
[32] J. Miró-Borrás and P. Bernabeu-Soler, E-everything for all: Text entry for people with severe motor disabilities, presented at the 6th CollECTeR (Collaborative Electronic Communications and eCommerce Technology and Research) Iberoamérica, Madrid, Spain, June 25-27, 2008.
[33] S. E. Palazuelos, J. I. Godino, and S. Aguilera, Comparison between adaptive and non-adaptive word prediction methods in a word processor for motorically handicapped non vocal users, in Proceedings of AAATE, 1997, pp. 158-162.
[34] M. J. Rosen and C. Goodenough-Trepagnier, Factors affecting communication rate in non-vocal communications systems, in Proceedings of the 4th Annual Conference on Rehabilitation Engineering, Washington DC, 1981, pp. 194-196.
[35] R. C. Simpson and H. H. Koester, Adaptive one-switch row-column scanning, IEEE Transactions on Rehabilitation Engineering, vol. 7, pp. 464-473, 1999.
[36] R. C. Simpson, H. H. Koester, and E. LoPresti, Evaluation of an adaptive row/column scanning system, Technology and Disabilities, vol. 18, pp. 127-138, 2006.
[37] C. E. Steriadis and P. Constantinou, Designing human-computer interfaces for quadriplegic people, ACM Transactions on Computer-Human Interaction, vol. 10, pp. 87-118, 2003.
[38] K. Tanaka-Ishii, Word-based predictive text entry using adaptive language models, Natural Language Engineering, vol. 13, pp. 51-74, 2007.
[39] K. Tanaka-Ishii, Y. Inutsuka, and M. Takeichi, Entering text with a four-button device, in Proceedings of the 19th International Conference on Computational Linguistics, 2002, pp. 1-7.
[40] G. C. Vanderheiden, Overview of the basic selection techniques for augmentative communication: Past and future, in The Vocally Impaired: Clinical Practice and Research L. E. Berstein, Ed. Philadelphia: Grune & Stratton, 1988, pp. 5-39.