AN ADAPTABLE SCAN-BASED TEXT ENTRY FOR MOBILE DEVICES: DESIGN, PREDICTIVE MODELING, AND EMPIRICAL EVALUATION

Sandi Ljubić[*] – Damir Arbula – Krunoslav Smrekar

Department of Computer Engineering, Faculty of Engineering, University of Rijeka, Vukovarska 58, 51000 Rijeka, Croatia

ARTICLE INFO

***Article history*:**
Received: 08.11.2016.
Received in revised form: 09.12.2016.
Accepted: 09.12.2016.

Keywords:
Human-Computer Interaction
Text entry
Assistive keyboard
Mobile devices
Predictive modeling
Empirical evaluation

Abstract:

This paper presents a highly customizable assistive on-screen keyboard for mobile devices, which supports several text entry methods based on row-column and bisection scanning techniques. Text entry can be accomplished using a zone based touch screen interface and/or via hardware keypads, involving configurable input control which can range from single-switch solution up to 5-key design. Apart from the presentation of a novel user interface, the paper contributions are as follows: development of movement models for all scan-based methods involved in text entry solution, computation of related upper-bound text entry speed predictions, and empirical investigation of their validity. In order to assess model predictions, a specific instance of row-column scanning technique was juxtaposed to bisection scanning principle in a user study involving 16 participants. Methods are evaluated against text entry performance, required workload, and general usability attributes. Although theoretical models predicted higher entry speed for bisection scanning, the results obtained from experiment demonstrated the row-column technique as significantly more efficient. This outcome discrepancy is specifically discussed by putting emphasis on factors that affect identified relation.

1 Introduction

In the area of Human-Computer Interaction (HCI), scanning technique represents a simple interaction pattern which abandons direct manipulation in a graphical user interface and relies on selections being made using switch. Generally, a sequential highlighting through a set of options is used for recommending an action (e.g. the selection of a particular item), with a scan delay typically ranging from 0,5 s to 2,0 s [1]. The user can select the highlighted item by issuing a selection command which can be activated in a number of ways, depending on type of a provided interface. For example, confirmation can be utilized by pressing a single key, toggling a physical switch, tapping on a

[*] Corresponding author. Tel.: + 385 – (0)51 – 651436; fax: + 385 – (0)51 – 651416
E-mail address: sandi.ljubic@riteh.hr.

touchscreen area, head movement, eye blinking, etc. As such, scanning has become an important part of assistive technology, and a valuable asset in Ambient Assisted Living (AAL) environments [2], providing interaction support for elderly users and people with motor impairments who have trouble using conventional input devices.

Text entry is a use case of a particular concern in HCI. A vast number of text entry methods and input modalities has been developed and analyzed, targeting different platforms and technologies. Scan-based techniques are frequently used in assistive text entry, and can be found within contemporary on-screen keyboards. Apart from accessible computing, texting on small or reduced form-factor devices gains interest in other domains, namely in gaming, ubiquitous computing, and wearable computing. These settings and related devices (e.g. smartwatches) often require a physically constrained input mechanism comprising only a few keys or a small set of input primitives [3]. In such setups scan-based text entry methods can also become very handy.

In this paper, an implementation of on-screen keyboard for Android mobile devices which supports different scanning schemes is presented. Several customization options are additionally provided so as to enable users to adapt input control according to their individual preferences. Both the keyboard functionalities and underlying design decisions are explained in detail. Apart from the keyboard design presentation, there were two main objectives of this work: (i) to comparatively assess text entry efficiency of two main scan-based strategies involved in the proposed solution (row-column vs. bisection), and (ii) to utilize both prediction models and controlled user study for making this assessment. The main contributions of the paper are summarized in the following:

- Different scan-based strategies are implemented and provided in a single text entry solution for mobiles, involving a novel design of input control and a number of adaptability options.
- Movement models for presented scan-based text entry methods are developed, thus formalizing the effect of different scanning schemes on the time required for character selection.
- Upper-bound text entry speed predictions for scan-based methods in question are provided, by combining previously developed movement models with language models (English and Croatian) built from available text corpora.
- Validity of the obtained theoretical predictions is investigated a posteriori, by conducting empirical study involving sixteen able-bodied participants. A discrepancy between theoretical and achieved entry speeds is found.

Finally, an argumentation is offered that can put a new light on the respective relation between theoretical predictions and real text entry speeds achieved in laboratory setting. In provided discussion, workload aspects and usability attributes are highlighted as factors that clearly affect this relation.

2 Related work

2.1 On scan-based text entry

A nice and constructive insight into the scan-based interaction techniques, and their specific impact on supporting motor impaired users, can be found in [4]. Important aspects of scanning systems specifically applied in text entry solutions are described in the following.

The most used variant of scan-based techniques is the row-column scanning [5], in which the highlighting usually starts with row-by-row iteration. When the user makes a selection, items within the selected row become subjects of highlighting pattern. The following selection results in inputting a corresponding symbol to the currently active text stream. Naturally, such text entry method is considerably slow, especially when compared with traditional keyboard-based input. However, numerous complementary techniques augmenting the row-column scanning principle have been developed. They can be grouped into four main categories:

- Scan delay manipulation;
- Word completion;
- Scanning pattern manipulation;
- Character layout optimization.

Scan delay, i.e. the dwell time between highlighting two neighboring elements directly corresponds to text entry speed: the shorter the delay is, the longer the input rate. But one must be aware of speed-accuracy tradeoff, because very short scan delays often imply wrong selections. Hence, one approach to enhance row-column scanning is to manipulate scan delay accordingly. Simpson and Koester [6] dealt with speed-accuracy tradeoff using dynamical run-time adaptation of the scan delay, showing that such adaptation could provide text entry enhancement without increasing task complexity.

Word completion is another method for enhancing text entry performance by making use of natural language properties. Based on the users' input history and the already known text sources (e.g. corpora and dictionaries), an entry system can help by predicting what the user might want to enter. A word candidate list is usually presented in an additional row, accessible via currently active interaction modality. Word completion typically causes an enhancement of on-screen text entry speed [7], proving that inherently slower scan-based methods can profit from text prediction, regardless of increased cognitive costs.

The scanning pattern defines the order in which items are highlighted within a scanning cycle [8]. Linear scanning pattern represents the most straightforward principle wherein all individual items are highlighted one-by-one in a recurrent cycle. This is the pattern that results in the slowest text entry rates. The row-column scanning pattern is, on the other hand, the most common implementation of grouped scanning – a scheme where progressively smaller groups of items are highlighted until a final selection is made. Grouped scanning is often generalized by a three-level scan, also known as the block–row–item scan [9]. In a block–row–item scan, 2D character matrix is divided into blocks, comprising logically grouped items (e.g. uppercase letters, lowercase letters, numbers, symbols, candidate words, etc.).

Character layout optimization is a technique of assigning letters to keys, and/or rearranging the overall keyboard layout by making use of the statistical properties of a given language. For example, within finger-based touchscreen interaction, the goal of layout optimization would be to minimize finger movements when tapping on a virtual keyboard. Optimization process usually relies on heuristic search, taking into account both language digraph frequencies and relative distances between keys. Zhai et al. [10] presented two quantitative approaches for determining optimized virtual keyboard layouts based on the related performance considerations.

2.2 On text entry predictive modeling

Soukoreff and MacKenzie [11] introduced an exemplary quantitative prediction technique which is based on two components: a movement model, and a language model. The movement model aims to predict the total time CT_{ij} required to enter character j preceded by previously entered character i. The language model uses digraph frequencies in a given language, and determines probabilities of occurrence P_{ij} for each digraph. These two models have to be combined in order to predict an average character entry time CT_L:

$$CT_L = \sum_{i \in C} \sum_{j \in C} (CT_{ij} \cdot P_{ij}) . \tag{1}$$

In the expression above, L denotes a given language, and C represents the existing character set used in the respective text entry method. The reciprocal of CT_L yields the average number of characters per second, which is converted into the standard text entry metric WPM (words per minute):

$$\mathrm{WPM}_{\max} = \frac{1}{CT_L} \cdot \frac{60}{5} . \tag{2}$$

The conversion presented in Equation (2) is grounded on the customary definition from the text entry domain, which assumes a word to be five characters long. Obtained $\mathrm{WPM}_{\max}$ represents the theoretical upper-bound text entry rate, because the utilized predictive model entirely relies on the time to input characters, and ignores additional time cost for mental activities such as thinking or visual searching.

When it comes to movement model, Fitts' law is typically used in order to predict the movement time between keys on a soft keyboard. In general, the use of Fitts' law is justified in scenarios where typing procedure can be decomposed into trivial pointing tasks, i.e., in cases where input is performed via pointing device (human finger included).

Since scan-based interaction modality is significantly different from the point-and-select paradigm, Fitts' law cannot be applied to the scanning movement model. Instead, the total time required to enter a character has to be modeled using different parameters.

3 Design of an adaptable scan-based text entry for mobile devices

A simple QWERTY-like character layout is chosen for the main visual interface of the proposed keyboard. It resembles the look of the standard keyboards used in touchscreen mobiles, with main difference being the rigid 10×4 design with equally dimensioned keys. Along with 26-letter English alphabet, Croatian letters and some punctuations symbols are additionally included. Text entry can be achieved using standard touch-typing, however, scanning techniques are enabled as well. The keyboard itself serves as a “touch command panel”, offering five distinct areas for manipulation when scan-based input is activated. The basic layout of the keyboard along with the anatomy of said touch command panel is presented in Fig. 1.

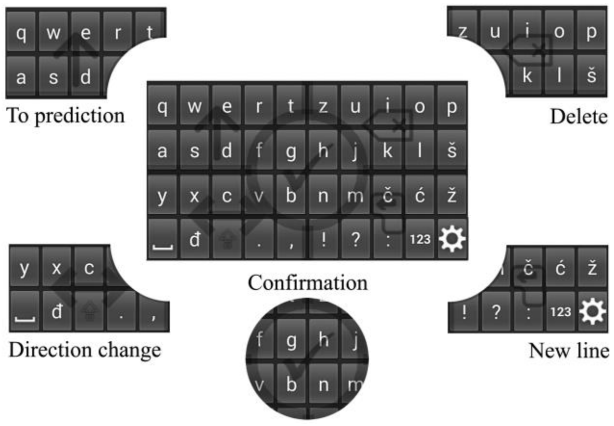

Figure 1. The main interface of the proposed text entry solution.

There are maximum five commands that can be activated when using scanning modalities:

- *Confirmation* – the essential switch for selecting the desired character/symbol, located in the very center of the touch panel;
- *Delete* – the touch area used for error correction;
- *New line* – used for entry confirmation in single-line input fields as well as for new line insertion in multi-line text areas;
- *To prediction* – switching the “scan cursor” from regular keyboard to the word candidate list (and vice-versa);
- *Direction change* – toggling the current direction of scanning.

3.1 Row-column and column-row designs

Fig. 2 presents the basic use case of text entry with row-column scanning modality. Exactly two confirmations are expected for a single character selection when using this technique. In case when input stream contains an error, delete command can be used for correction. This implies that minimally two commands have to be enabled for text entry with a delete option (2-key design). Alternatively, the delete command can be placed inside the character layout if single-switch text entry is preferred or strictly required. Consequently, the keyboard with the presented touch panel allows different control schemes, ranging from single-switch solution up to 5-key design.
The usage of word completion and scan direction change is a question of the users’ individual needs. If change direction command is enabled, the user can benefit from minimizing the scanning path between two subsequent confirmations. Word completion will be used if prediction option is previously enabled in the keyboard settings. In such a case, a candidate list is presented in the additional row above the keyboard layout, containing the most probable words with currently entered prefix. A custom dictionary can be easily imported into the presented text entry solution.
In order to augment overall user experience, sound and haptic feedback are also provided for scan-based text entry. The user can customize feedback options according to her/his own preference, as follows:

- *Scan tick sound* – if enabled, every single scan will be accompanied by appropriate sound;
- *Confirmation sound* – if enabled, the confirmation of the letter/symbol will be followed by a distinct sound signal;
- *Touch vibration* – if enabled, activating the specific touch area (i.e. issuing a command) will be supplemented by a short vibration of the mobile device.

When using row-column scanning, the user can accidently make an error by selecting the wrong row or the wrong column. While the wrong column directly implies entering the wrong character and the need for error correction, making the unwanted row-selection can be recovered without altering the input stream.

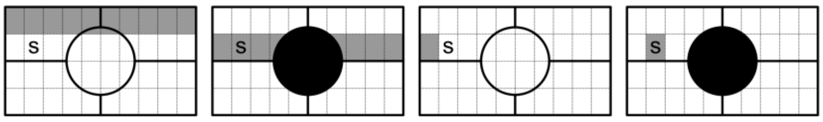

Figure 2. The basic principle of text entry using the provided row-column scanning modality.

In the provided text entry solution, the confirmation touch area is used for that purpose, altogether with the long tap gesture. Therefore, the role of confirmation touch area is twofold: if standard tap is used, a selection will be made; if long tap gesture is used instead, then the scanning cycle will be forced to its beginning.

Within row-column technique, the scanning cycle usually starts with the first row. After the selection of the particular character has been made, the cycle is restored to a starting position. Nevertheless, a different instance of the row-column scanning is additionally implemented, in which the scanning cycle always continues from the last selected row. The respective technique is entitled "row-column with no reset".

Apart from the scanning cycle starting point, row-column technique is furthermore customized so as to provide the possibility to start scanning from either rows or columns. Hence, altogether four versions of group scanning are implemented in the provided text entry solution.

3.2 Bisection scanning design

Along with the row-column principle, the proposed text entry method implements bisection scanning as well. Fig. 3 presents the basic use case of text entry when bisection scanning modality is utilized.

The main idea of bisection scan is rather straightforward: a confirmation command is used for gradually dividing the initial character layout, up to the smallest subset consisting of only few characters. The final selection in the smallest subset is then quickly achieved by choosing between these few available items. Seeing that the provided keyboard design involves 10 columns, "total bisection" is not possible, and the smallest subset will contain either 2 or 3 characters, according to its position in the character layout.

As opposed to row-column principle, bisection scanning is supposed to be more burdensome since both physical and mental activities at a higher level are expected. Exactly 5 confirmations are required to enter a single character, while at the same time the scanning subset constantly changes its position and size, thus imposing a certain cognitive load.

All options provided for the row-column scanning can be used for bisection scanning as well. Long tap on the confirmation area will force the scanning cycle to its starting point which can be, according to the user's preference, either the first row-based block or the first column-based block. Indeed, two similar versions of bisection scanning are implemented and included in the text entry solution.

3.3 Adaptability

The provided text entry solution supports no less than 6 scan-based input modalities, which can all be complemented with other available options, according to the users' preference towards scan direction control, scan delay duration, sound/haptic feedback, and word completion utility.

Finally, customization set is encompassed by the possibility to control any scan-based method using external hardware keypad. This option is feasible only for mobile devices supporting USB OTG (*On-The-Go*), i.e. for those devices able to act as a USB host.

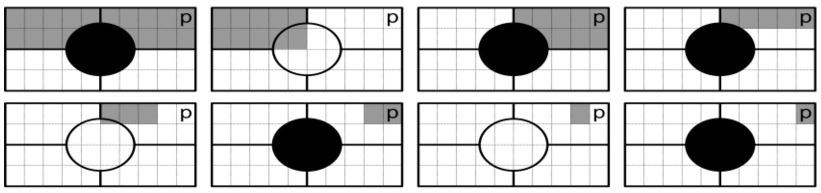

Figure 3. The basic principle of text entry using the bisection scanning modality.

Scan-based text entry via external keyboard can be of particular concern in assistive technology domain since various keypads can be designed and constructed in order to help people with different physical impairments.
High level of adaptability is achieved by providing a number of adjustable settings that can reflect users' individual preferences. Customization possibilities provided in the keyboard settings are summarized in Table 1.

Table 1. Customization of the provided text entry solution (defaults are highlighted)

Settings	Provided options)
Input modality	Normal (touch typing)
	Row-column: **RC**
	Row-column (no reset): **RC_{NR}**
	Column-row: **CR**
	Column-row (no reset): **CR_{NR}**
	Bisection, rows first: **B_{RF}**
	Bisection, columns first: **B_{CF}**
Feedback	Scan tick (sound) [on / off]
	Character entry (sound) [on / off]
	Command (vibration) [on / off]
Direction change	Enable direction change [on / off]
Scan delay	Configurable [seconds]; 0.75 s default
Word completion	Use word completion [on / off]
	Reset frequencies in dictionary
	Import new dictionary
Input interface	Touch panel with 5 commands
	Keyboard via OTG: 5 active keys

4 Predictive modeling of text entry speeds

In order to predict theoretical upper-bound text entry speeds for scan-based methods in question, both the movement models and language models have to be developed.

4.1 Movement models for scan-based methods

For a regular row-column scanning technique (RC), the time required to input a particular character *j* does not depend on previously entered character *i* since the scanning cycle always resets to the starting position. So, the corresponding movement model can be formalized as follows:

$$CT_{ij}(\mathrm{RC}) = D_j \cdot T_{sd} + 2T_c\,. \qquad (3)$$

In the expression above, D_j stands for the minimal distance from the starting position to the position of the character *j*. Each scan in this path consumes time, namely the duration of a scan delay (T_{sd}). In addition, exactly 2 confirmations are required for selection of the target row and column, thus the respective parameter T_c has to be counted twice.
In order to identify the minimal number of scans in the scan-based shortest path, a keyboard coordinate system is defined, in which the character 'q' has the origin position (0, 0), while the settings symbol stands at the right-bottom position (9, 3). Generally, if character *j* at position (x_j, y_j) needs to be entered, then the scan-based shortest path consists of exactly (x_j + y_j) scans. The model from Equation (3) can thus be adjusted accordingly:

$$CT_{ij}(\mathrm{RC}) = (x_j + y_j)T_{sd} + 2T_c\,. \qquad (4)$$

Since the shortest scan-based paths do not differ within the column-row (CR) design, the corresponding movement model is identical:

$$CT_{ij}(\mathrm{CR}) = CT_{ij}(\mathrm{RC})\,. \qquad (5)$$

As opposed to the basic row-column and column-row principles, in their "no reset" versions, after the letter confirmation, the scanning cycle continues from the last selected row. Hence, the shortest path between two consecutive characters *i* and *j* in these cases depends on both positons: (x_i, y_i) and (x_j, y_j). For row-column with no reset (RC_{NR}), exactly x_j scan delays are required to reach character *j* once the target row has been selected. If target row y_j lays beneath the previously selected row y_i, or these rows are in fact the same, then (y_j – y_i) scan delays have to be consumed before the actual row selection. Conversely, if target row is positioned above the previously selected row, the scanning cycle will involve the switch between the last and the first row in the default top-down direction. The movement model for RC_{NR} modality can therefore be defined as follows:

$$CT_{ij}(\mathrm{RC_{NR}}) = \\ = \begin{cases} (x_j + y_j - y_i)T_{sd} + 2T_c, & y_j \geq y_i \\ (x_j + 4 - (y_i - y_j))T_{sd} + 2T_c, & y_j < y_i \end{cases}. \quad (6)$$

Movement model for column-row with no reset ($\mathrm{CR_{NR}}$) is analogous to $\mathrm{RC_{NR}}$ model although they are not identical. The main difference is that cyclical switching will occur in horizontal scanning, between the last and the first column of the keyboard layout. The following equation represents the $\mathrm{CR_{NR}}$ movement model:

$$CT_{ij}(\mathrm{CR_{NR}}) = \\ = \begin{cases} (y_j + x_j - x_i)T_{sd} + 2T_c, & x_j \geq x_i \\ (y_j + 10 - (x_i - x_j))T_{sd} + 2T_c, & x_j < x_i \end{cases}. \quad (7)$$

The effect of scan direction change can be predicted as well, because it is possible to model the optimal usage of the corresponding option. The actual benefit can be demonstrated by a simple comparison between the regular row-column scanning technique (RC) and its version which enables the direction change ($\mathrm{RC_{dc}}$). For example, if selection of the NUM key is considered, reaching the related position (8, 3) involves 8 scan delays less if direction change is properly triggered.
Given that issuing the direction change command consumes time T_{dc}, then $2T_{dc}$ have to be included in all cases which involve switching both the vertical and horizontal direction. While changing the horizontal scan direction is cost-effective for all characters located in columns 6 and above, toggling the default top-down route is effective if (and only if) target character is located in the bottom row. The movement model for $\mathrm{RC_{dc}}$ scanning is defined accordingly:

$$CT_{ij}(\mathrm{RC_{dc}}) = \\ = \begin{cases} (x_j + y_j)T_{sd} + 2T_c, & y_j \neq 3; x_j \leq 5 \\ T_{dc} + (1 + x_j)T_{sd} + 2T_c, & y_j = 3; x_j \leq 5 \\ T_{dc} + (y_j + 10 - x_j)T_{sd} + 2T_c, & y_j \neq 3; x_j > 5 \\ 2T_{dc} + (1 + 10 - x_j)T_{sd} + 2T_c, & y_j = 3; x_j > 5 \end{cases}. \quad (8)$$

Finally, movement models for bisection scanning have to be determined. As explained before, bisection strategy always involves exactly 5 confirmation commands for selecting a particular character. Apart from $5T_c$, additional time is spent on scan delays, i.e. for shifting through the character subsets that gradually decrease in size. The number of required scan delays depends on the position of the target character and can range from 0 to 5. As opposed to the previous cases, writing the formula for the bisection movement model is not convenient if variables x_j and y_j are used. Since optimal path to every character j can be determined in advance, the following form can be used:

$$CT_{ij}(\mathrm{B_{RF}}) = \\ = \begin{cases} 5T_c, & j \in \{q\} \\ T_{sd} + 5T_c, & j \in \{w, r, a, z, y\} \\ 2T_{sd} + 5T_c, & j \in \{e, t, s, f, u, o, h, x, v, n, SP\} \\ 3T_{sd} + 5T_c, & j \in \{d, g, i, p, j, l, c, b, m, đ, ć, ., !\} \\ 4T_{sd} + 5T_c, & j \in \{k, š, č, ž, ,, ?, CAPS, NUM\} \\ 5T_{sd} + 5T_c, & j \in \{:, SET\} \end{cases}. \quad (9)$$

The movement model thus defined holds for both bisection modalities presented in this paper. Hence, the following can be stated:

$$CT_{ij}(\mathrm{B_{RF}}) = CT_{ij}(\mathrm{B_{CF}}). \quad (10)$$

All presented movement models apply to characters displayed on the initial layout. If uppercase letters are also needed within the text entry models, corresponding calculations have to be extended. Word completion features are not involved in modeling because this research focuses solely on the interaction aspects of scan-based methods.

4.2 Language models

Two language models have been constructed. The first one applies to the English language with a character set consisting of 27 elements, including the lowercase letters a-z, and the space character. The second model refers to the Croatian language, more specifically to character set comprising altogether 32 lowercase items: 27 from the said English charset, and 5 more specific letters: 'č', 'ć', 'đ', 'š', and 'ž'. Therefore, the models provide a 27×27 matrix of digraph probabilities P_{ij}(EN) for English, and a corresponding 32×32 P_{ij}(HR) matrix for Croatian.
For the computation of the P_{ij} values, both the English and the Croatian text corpora were obtained from the freely available Open Subtitles repository

(version 2013) [12]. All text within the available corpora was transposed to lowercase, and only predefined characters were considered valid for the computation of the language statistics.

4.3 Scan-based text entry speed predictions

In order to obtain upper-bound text entry speed predictions, the movement models defined in equations (4-10) have to be combined with the language models P_{ij}(EN) and P_{ij}(HR). Furthermore, the values for T_{sd}, T_c, and T_{dc} have to be set.
Since all of available commands for scan-based text entry are jointly located on the relatively small touch panel, it is reasonable to assume they have equal execution time. For both the confirmation and the direction change command, a duration of 0.24 s is used in subsequent model calculations.
The 0,24 s value is based on the model human processor (MHP) [13]. MHP predicts simple reaction time by combining perceptual, cognitive, and motor aspects of interaction. It is therefore assumed that issuing a particular command (confirmation or direction change) represents the task which can be executed within the simple reaction time. This task completely corresponds to MHP, as command execution inherently includes: perception of the scan-cursor location, decision making (e.g. confirmation should take place only for the proper location), and actual physical movement (tapping the touch panel).
Scan delay duration remains the only variable in the developed models. It is used for obtaining text entry rate predictions for different T_{sd} values without the need of formally testing scan-based methods. The effect of scan delay on predicted input speed is analyzed using three different values of T_{sd}: 0.5 s, 0.75 s, and 1 s. Text entry speed predictions were calculated using originally developed application. Results are summarized in Table 2.

Table 2. *Text entry speed predictions for proposed scan-based methods (settings chosen for empirical evaluation are highlighted)*

Method	WPM_{max}					
	EN; T_{sd} [s]			HR; T_{sd} [s]		
	0,50	0,75	1,00	0,50	0,75	1,00
RC	4,50	3,19	2,47	4,14	2,92	2,26
RC_{dc}	4,99	3,84	3,12	5,00	3,85	3,13
CR	4,50	3,19	2,47	4,14	2,92	2,26
RC_{NR}	4,08	2,88	2,22	3,74	2,62	2,02
CR_{NR}	3,49	2,44	1,87	3,57	2,50	1,92
B_{RF}	5,36	4,35	3,66	5,21	4,20	3,52
B_{CF}	5,36	4,35	3,66	5,21	4,20	3,52

Inspection of theoretical predictions revealed some interesting relations:

- As expected, the shorter scan delay should yield higher entry rates for all scan-based methods;
- No-reset versions of CR/RC methods are expected to be less efficient than their default types, meaning that scanning cycle which returns to its initial position is the better option;
- If no-reset versions are actually considered, then RC_{NR} promises better results than CR_{NR}, irrespective of the used language;
- In general, scanning within the QWERTY-like character layout seems to be a better fit for the English language, seeing that predicted speeds for Croatian are sufficiently higher only for CR_{NR} method;
- Proper use of direction change truly augments the input efficiency of row-column scanning, with $RC \rightarrow RC_{dc}$ enhancement being more prominent for the Croatian language;
- Bisection-based methods are supposed to be the most efficient, promising higher text entry speeds than any version of the RC/CR modality.

The relation last mentioned, i.e. row-column vs. bisection, is of particular concern in this paper. From the interaction standpoint they represent significantly different strategies: while row-column assumes more time spent on numerous scan delays, bisection relies on more demanding command rate. Since empirical evaluation that tests all considered settings (method × scan delay × language) would be both time-consuming and unpractical, it was decided to comparatively assess two main scanning principles. Hence, RC_{dc} and B_{RF} methods were specifically selected for further inspection in text entry experiment with real users. In order to support assessment reproducibility within a wider research community, English was chosen as the target language for input tasks. The most common value for scan delay was selected for the same reason.

5 Empirical evaluation

Contrary to predictive modeling approach, a user study can reveal text entry performance in realworld scenarios. The main goal was to carry out an experiment in which scan-based methods could be comparatively assessed against input efficiency, required workload, and usability issues in general.

5.1 Participants, apparatus, and procedure

Sixteen users were involved in empirical research (15 males, 1 female), their age ranging from 21 to 25 with an average of 22.81 years (SD = 1.10).

Scan-based text entry methods were tested on Samsung Galaxy S5 smartphones (5.1" display size) running Android Lollipop OS. Previously developed Android application was used for logging relevant text entry events and the corresponding timing data. This application presents transcription-based text entry tasks, requiring the user to rewrite a displayed text phrase. A single task is considered done when a presented phrase is fully and correctly transcribed using the provided input method. Short and easily memorable text phrases (in English) were used for testing, specifically those provided in [14]. All phrases in question consist exclusively of lowercase letters and space character, without any punctuation symbols. Parameters for scan-based text entry experiment was set as shown in Table 1: no word completion, touchscreen interface only, scan direction change enabled, sound feedback for scan shifts, and T_{sd} = 0.75 s.

In order to familiarize with scan-based methods and testing application features, a detailed demonstration of text entry was given using both RC_{dc} and B_{RF} method. There were no training sessions involved whatsoever. In the actual experiment, participants were instructed to enter 10 different text phrases using each scan-based method. To get around the possible learning effects in the experiment, the sequence of experimental conditions were properly counterbalanced. According to the best practice in text entry empirical research, users were additionally instructed to input text "as quickly as possible, as accurate as possible". For every completed text entry task, corresponding log record was generated in the testing application, providing WPM and TER (total error rate) metrics.

After testing each scan-based method, users were asked to estimate perceived workload by completing a survey based on the rating part of the NASA-TLX [15]. The concluding assessment was carried out using a short post-study questionnaire, asking participants to compare two methods from the usability standpoint.

5.2 Results and discussion

The sequential entry of 10 different phrases was analyzed from the descriptive statistics point of view. The effect of inherently involved practice can be seen in Fig. 4, showing text entry performances averaged across ten consecutive trials.

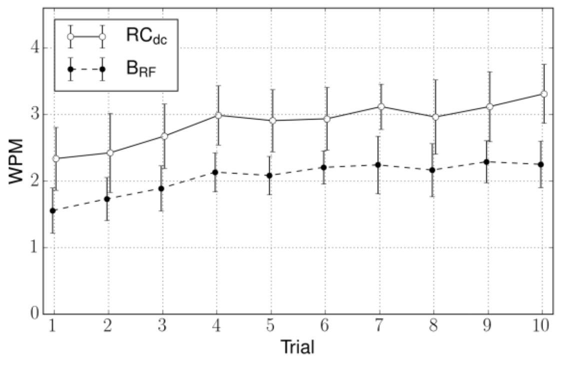

Figure 4. Text entry performances averaged across ten repetitive trials.

Input performance generally improves with repetition, irrespective of the used scan-based method. Seeing that users had no training sessions prior to actual testing, it is expected and understandable outcome. Task replication itself allowed for learning and enhancing interaction skills. It can be seen that learning curve is somewhat steeper for RC_{dc} method, indicating row-column scanning technique to be easier to learn.

Negative correlations between text entry speed and total error rate metrics are revealed for both scan-based methods. The observed speed-accuracy relations represent a direct consequence of implemented tasks that required fully correct transcription. On average, users tend to make more errors when using bisection modality (7.20 % for B_{RF} vs. 6.69 % for RC_{dc}).

Since predictive modeling results refer to expert-level text entry, it was decided to further investigate empirical results which correspond to users' best individual performances. For each participant, a task with the highest obtained input speed is selected (regardless of trial in which such result occurred) and henceforth used as the steady-state performance level of the respective user. The best performances among the scan-based methods are then compared mutually as well as with theoretical predictions of upper-bound text entry speed. The respective relations are shown in Fig. 5.

When performing the related *t*-test on best-of datasets obtained from experimental setting, a significant difference was found in text entry performance between the two scan-based methods: $t(15) = 12.116$; $p<0.001$. Indeed, users were significantly faster using row-column technique (3.66 ± 0.37 WPM) than using the bisection-based method (2.55 ± 0.26 WPM).

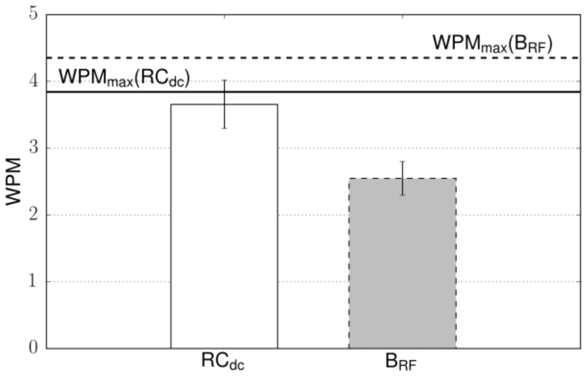

Figure 5. Predicted and achieved text entry speeds.

Text entry speeds achieved within user testing are ordered the opposite from what theoretical predictions suggest. While predictive models assume that bisection principle maintains higher input rate potential, in conducted experiment the RC$_{dc}$ showed to be significantly faster than the B$_{RF}$ method. This discrepancy between theoretical predictions and empirical outcomes can be attributed to the several factors.

At first, there is a difference in handling wrong row selections in RC$_{dc}$ and incorrect selections of character subset in B$_{RF}$. While utilizing B$_{RF}$ method, the user can make few correct bisection commands prior to unintentional selection of the wrong character subset. In such scenario, the user can reset the scanning cycle and perform all five required bisections once again, thus eliminating errors in the input stream. On the other hand, RC$_{dc}$ involves only two levels for single character selection, hence the effects of resolving wrongly selected row are generally less time consuming.

Furthermore, while mental activities required for scan-based methods are ignored in predictive modeling, they are nevertheless inherently involved in empirical setting. Bisection scanning seems to be more demanding in that respect because highlighted elements change in both position and size. Reaching the text entry expert level with RC$_{dc}$ requires skillful usage of direction change command. Conversely, expertness in B$_{RF}$ assumes mastering the bisection principle, i.e. learning and remembering patterns made up of confirmation commands and scan delays for every character. The latter certainly imposes much more interaction burden, especially if required physical activities are also considered. Simplicity makes the row-column scanning more natural to use, so users tend to acquire control skills much faster than with the bisection scanning.

Aforementioned arguments are corroborated by the results of qualitative evaluation. Questionnaire based on Raw-TLX format was used so as to obtain comparative ratings of perceive workload on a 20-point Likert scale. The respective outcomes are shown in Fig. 6 (top).

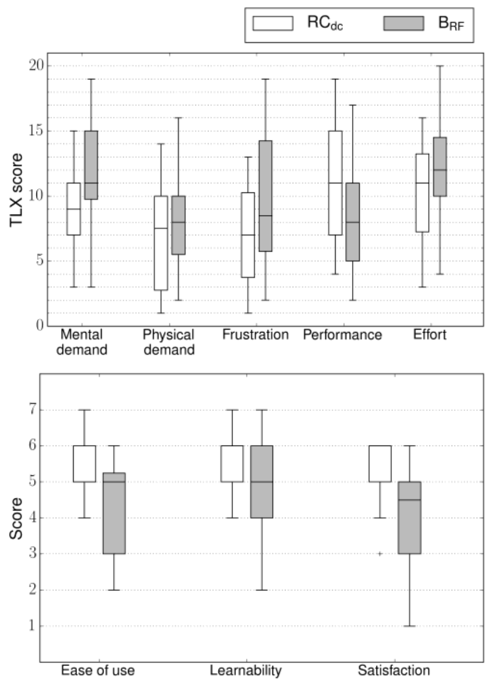

Figure 6. Users' opinions on perceived workload (top), and usability ratings (bottom).

The Wilcoxon signed-rank tests were used to assess obtained TLX-based scores. Significant differences were found for each considered factor, all of them clearly favoring the row-column technique and thus confirming the issues previously discussed.

In the concluding survey, participants used 7-point Likert scales for rating two scan-based text entry methods against the ease of use, perceived learnability, and overall satisfaction. The obtained results are presented in Fig. 6 (bottom). Wilcoxon signed-rank tests statistically confirmed that the

RC_{dc} method was indeed significantly easier to learn ($Z = -2.069$, $p = 0.039$), as well as significantly easier to use ($Z = -1.974$, $p = 0.048$). When it comes to overall impression, users were much more satisfied with row-column scanning than with bisection scanning.

6 Conclusion

An adaptable on-screen keyboard for mobile devices, which supports several variations of row-column and bisection scanning, is developed and presented in this work. Its design and functionalities are described in detail, with special emphasis on provided customization options that support tailoring the text entry process according to the user's individual preferences.

Predictive modeling approach was used in order to obtain theoretical upper-bound input speeds for all scan-based methods encompassed within the text entry solution. It is demonstrated how different scanning strategies affect the time required for character selection, thus inferring different WPM values. The effect of target language on entry rate predictions is also confirmed. Predictive models are shown to be particularly useful for evaluating and comparing various interaction designs without real users. Among provided methods, bisection scanning was predicted to be the fastest one.

Empirical evaluation was conducted in order to validate previously obtained predictions. Text entry experiment, involving sixteen users and targeting two different versions of scanning, was carried out. Obtained entry rates were generally low, but in line with existing solutions implementing similar interaction techniques. Contrary to model predictions, RC_{dc} method showed to be significantly faster than B_{RF}.

Differences between theoretical predictions and empirical outcomes should not raise the questions about validity of the modeling procedure. The results of the qualitative evaluation revealed issues associated with the related discrepancy. Bisection concept was reported to be more demanding (both physically and mentally), more frustrating, and much more difficult to learn. Therefore, expert-level text entry efficiency seems to be much harder to achieve when bisection scanning is utilized. The comparison of various trends in learning methods corroborates this consideration.

It would be very interesting to see which level of text entry efficiency more trained users could achieve. To put things into perspective, it should be noted that real text entry speeds were obtained from the experiment wherein users spent no more than one hour per method, including breaks between text entry tasks. It is therefore reasonable to expect higher levels of text entry expertise in the long run. This especially applies to bisection scanning, characterized by a combination of moderate learning curve and higher entry potential. According to the model predictions, a longer learning cycle could indeed provide a valuable payoff for the target users.

References

[1] Felzer, T., MacKenzie, I.S., et al.: *Qanti: A software tool for quick ambiguous non-standard text input*, Proc. Int'l Conf. ICCHP, Vienna, Austria, 2010, 128-135.

[2] Brestovac, G., Marin, D., et al.: *Ambient orchestration in assisted environment,* Engineering Review, 34 (2014), 2, 119-129.

[3] MacKenzie, I.S., Soukoreff, R.W., Helga, J.: *1 thumb, 4 buttons, 20 words per minute: design and evaluation of H4-writer*, Proc. Symp. User Interface Software and Technology (UIST), Santa Barbara, USA, 2011, 471-480.

[4] Ntoa, S., Margetis, G., et al.: *Scanning-based interaction techniques for motor impaired users*, In: Kouroupetroglou, G.: Assistive Technologies and Computer Access for Motor Disabilities, IGI Global, Hershey, 2014.

[5] Baljko, M., Tam, A.: *Indirect text entry using one or two keys*, Proc. Int'l Conf. Computers & Accessibility, Portland, USA, 2006, 18-25.

[6] Simpson, R.C, Koester, H.H.: *Adaptive one-switch row-column scanning*, IEEE Trans. Rehabil. Eng., 7 (1999), 4, 464-473.

[7] Felzer, T., Nordmann, R.: *Alternative text entry using different input methods*, Proc. Int'l Conf. Computers & Accessibility, Portland, USA, 2006, 10-17.

[8] Simpson, R.C., Mankowski, R., Koester, H.H.: *Modeling one-switch row-column scanning with errors and error correction methods*, The Open Rehab. Journal, 4 (2011), 1-12.

[9] Lafi, S.M., Hock, S.K.B.: *An adaptive text entry method based on single-key minimal scan matrix for people with severe motor*

disabilities, International Journal of Scientific & Engineering Research, 3 (2012), 8, 1-5.

[10] Zhai, S., Hunter, M., Smith, B.A.: *Performance optimization of virtual keyboards*, Human-Computer Interaction, 17 (2002), 2-3, 89-129.

[11] Soukoreff, R.W., MacKenzie, I.S.: *Theoretical upper and lower bounds on typing speed using a stylus and soft keyboard*, Behaviour & Information Technology, 14 (1995), 6, 370-379.

[12] *http://opus.lingfil.uu.se/OpenSubtitles2013.php*

[13] Card, S.K., Newell, A., Moran, T.P.: *The psychology of human-computer interaction*, Lawrence Erlbaum Assoc., New York, 1983.

[14] MacKenzie, I.S., Soukoreff, R.W.: *Phrase sets for evaluating text entry techniques*, Proc. Human Factors in Computing Systems (CHI EA), Ft. Lauderdale, USA, 2003, 754-755.

[15] *https://humansystems.arc.nasa.gov/groups/tlx*