International Journal of Modern Engineering Research (IJMER)

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

Comparison Among Ambiguous Virtual Keyboards For People With Severe Motor Disabilities

A.J. Molina[1], O. Rivera[1], I. Gómez[1], M. Merino[1], J. Ropero[1]
*(Department of Electronic Technology, University of Seville, Spain)

ABSTRACT

This paper presents an exhaustive study on the different topologies of ambiguous soft keyboards, analyzing the text entry average time per character and the average number of user inputs necessary for its creation. Various topologies and design criteria are investigated. In addition, an analytical model is also proposed. This model allows one to compare among different topologies and estimate the sensitivity that different keyboards offer when compared with dictionary hit rates. It has been found that ambiguous keyboards, with six keys, are better to use.

Keywords **- Descriptive User Models, Human-Computer Interaction, Indirect Text Input, System Model, Virtual Keyboard.**

I. INTRODUCTION

1.1 The Need of Adapted Communication Systems

Communication is a fundamental element to obtain social integration. People with severe motor disabilities often have reduced communication skills. Many of them have speech impairments that make them difficult to be understood, and external aid systems are required to carry out daily communication tasks. These are augmentative and alternative communication systems (AAC).

1.2. Scanning-based Virtual Keyboards

Some of the AAC systems are text entry systems that are equipped with a text-to-speech component, letting an oral communication from an entered text, augmenting communication possibilites. However, conventional text entry devices may not be used by motor disables, especially when they have severe disabilities because of their lack of precision movements. Therefore, devices that are adapted to their skills have to be used.

An alternative that is widely accepted is the use of a virtual keyboard (VK) used as a substitute for a conventional keyboard. A VK is a software application whose graphical user interface represents a keyboard. A VK that has one character in each key is named unambiguous keyboard. On the other hand, an ambiguous VK contains more than one character in some keys. These require disambiguation of the character contained in the key. Ambiguous VK demonstrates some advantages with respect unambiguous ones [1].

1. The efficiency of an ambiguous keyboard is near to one keystroke per letter. 2. Apart from literacy, no memorization of special encodings is required. 3. Attention to the display is required only after the word has been typed. 4. A keyboard with fewer keys can have larger keys for direct selection. 5. The average time to select a key by scanning is reduced considerably. 6. Simple linear scanning can be used efficiently to select a key. 7. Fewer keys may allow direct selection with various input devices.

The keys of a VK may be selected using an interaction method adapted to user's skills, such as a stylus over a touchable screen, an adapted mouse, eye-gaze trackers [2,3], head movement detectors [4,5], etc. However, a simple interaction method is required for severe motor handicapped people. This method is based on detecting a residual voluntary movement or some brain activity, such as the one used in Brain Computer Interaction (BCI) Systems. The most simple device is capable of detecting only one kind of user input, used to select the current item, such as a single switch, blink detectors, wink detectors, saccadic movement detectors, etc. [6-8]. To Use this kind of devices, an indirect text entry method to select the desired option from the currently highlighted options has to be implemented. In this sense, a scanning method is a possible alternative. An option or a group of options is highlighted in each scanning step. If the desired option is in the currently highlighted ones, a selection has to be done.

The scanning method can be automatic or manual. In automatic scanning, an internal timer establishes the dwell time or the time that any key or row is

highlighted. A switch keystroke makes the selection of the current highlighted key or row. In manual scanning, a keystroke makes the current highlighted group to advance to the next. A second switch or the timeout of a timer makes a selection.

On the other hand, the scanning method can be implemented in a linear or row-column mode. In linear scanning, each key is highlighted, one after the other. In the latter, a cluster of keys are highlighted. In matrix VK, a cluster can be made by a row of keys. A selection when a row is highlighted performs a linear scanning of its keys. A new linear scanning on character contained in a key can be performed if a preselected key has more than one character on it. Finally, once a character is chosen, it is convenient to restart scanning from the first row [9].

1.3. The importance of proper configuration

The main drawback of the use of a scanning-based text entry system is its low text entry rate. In [10], it is estimated that the maximum communication rate using this kind of systems is 10 words per minute (WPM). In a normal conversation, able-bodied people may pronounce between 180 and 200 WPM. Some situations in which handicapped people may not participate in a normal conversation could occur because of this threshold, and therefore, this may drive to a social exclusion. Instead of this rate, these systems are the unique alternative in case of severe motor disabilities.

A proper selection and configuration of the system is important to obtain a good communication rate. There are many VKs and scanning methods. Once a given keyboard is chosen, tuning it to the user's skills and preferences may yield significant performance and comfort benefits. A study of optimal configurations of the input devices for people with physical impairments is shown in [11].

VKs are usually set in a rectangular matrix of keys, and each one of them may contain different amounts of characters, although optimal ones can have a button arrangement, different from the rectangular matrix. A study that includes different keyboards is depicted in [12], where a nonmatricial unambiguous keyboard, whose arrangement is determined by character frequency, is compared with other matricial ambiguous ones, such as Huffman, alphabetic, or mobile distribution keyboards. More examples of matricial VK can be found in [13], where a QWERTY-type one is used for brain-injured people, and in [14], where several ambiguous VK with nine keys called Levine, TOC, alphabetic, and frequency are compared.

An important issue that affects the text entry rate is how the characters are distributed on the VK keys. Keyboards, whose character layout is based on their frequency, have better performance than QWERTY, whose layout is a reproduction of the traditional keyboard. This is because the most likelihood characters are placed in or nearby the row or key, where the automatic scan starts, so that the mean character access time is reduced, and therefore, the text entry rate could be increased without any negative effect on the number of user inputs (UI). The fixed character layout can be established based on the character frequency in the language and character access time in VK. On the other hand, a dynamical character layout can also be established, where the characters are automatically rearranged depending on the probability of the previous character sequence. It is obvious that analytically, the second option is the best. However, studies with fluctuating keyboards [15] have shown that there is a toll on time taken using this keyboard owing to the high mental load needed to locate the position of the characters. Thus, the performance of fluctuating keyboard matches or worsens that of fixed ones.

On the other hand, the methods used to improve text entry rates in everyday devices can be applied to improve the communication capabilities for the disables. Thus, for example, the T9[1] method could be implemented in an ambiguous VK to enhance the text entry rate. Character or word prediction can be used to improve performance. Prediction can be accomplished in VK showing the most likely character that follows a preselected character sequence (or prefix) [16], [17] or the most likely word that matches with that prefix [18]. Predictor requires the existence of a dictionary and/or a prefixes table with word/prefixes frequency information included.

1.4. How to Compare

VKs testing can be accomplished by experiments or simulation. The part of the system that is tested has to be made as a prototype or a final product in experiment option, implying a cost in time and budget. In addition, end users have to participate in this option, and a trial programming has to be made

[1] T9 is a registered trademark of Tegic Communication and it means text in 9 keys

carefully to obtain correct measurements. The sample of participants should be representative of the end user group, making it difficult to carry out trials and make prototypes in certain situations. In addition, hardware components are required to test software solutions and the testing results may be influenced by them.

On the other hand, simulation consists of an application software that tries to emulate the system behavior, user behavior and interaction among them. The system behavior is more or less stable and may be translated to a program language. However, other components depend on the user and they may change in each trial or during a trial. Simulators have a group of input parameters that let set the simulation context to obtain correct results. In this way, the participation of users is limited and the need of making a prototype or a final product is removed.

The results obtained by these methodologies are representative of the conditions in which they were elaborated. If we want to know how the keyboard performance is affected by external conditions, the experiment or the simulation should be repeated.

There are many prediction systems, each having different characteristics, making it difficult to compare their performances because of the diversity of heterogenous parameters used to measure them. Some authors, such as Gillette and Hoffman [19] or Heinisch and Hecht [20] have carried out studies on commercial predictive products. A study on non-commercial prediction system is presented in [21].

It is necessary to set some metrics to compare different text entry systems or different configurations of one of these systems. VK for motor disabled people gaims to measure two items: the text entry rate and the number of movements (interactions) that have to be done to enter a text. First, a parameter measured in WPM is often used. On the other side, the most used parameter to measure the second item is keystrokes per character (KSPC). It is also clear that the term number of user inputs per character (UI_c) is more general for that diversity of devices instead of KSPC, and hence, we prefer to use this term, although the meaning is completely equivalent.

Comparison UI_c of different VKs can be carried out by simulation, employing extensive texts from a corpus built using several sources, such as digital journals, magazines, dictionaries, etc. This is due to the UI_c parameter that only depends on the operational mode of a specific KSPC, and it is independent of the user. However, obtaining WPM strongly needs a user's model, and hence, the most frequent method to test a VK is in experiments where the users have to use the VK to type a preselected text fragment with a high degree of correlation with the user's language.

1.5. Goals and document structure

A comparison among different ambiguous VKs is presented in this paper. Different disambiguation modes are considered. A simple user model is used to obtain a proper value of the reaction time, and some system models are presented. The latter are probabilistic models obtained from a dictionary that have been complied. One of these models is a mathematical model associated with various VKs that work in Tn (Text in n-keys[2]) operated by single-switch users. The model estimates both the average time per character ($\overline{t_c}$) and average number of user inputs per character($\overline{UI_c}$). By assuming that the average length of a word, $\bar{l}$, including space character, is 6 for English or 5.5 for Spanish, we have WPM = $\frac{60}{\bar{l} \cdot \overline{t_c}}$. The model lets us to test how a VK layout or a dictionary may influence $\overline{t_c}$ and $\overline{UI_c}$.

In section II, a review of VK software is presented with special emphasis on ambiguous ones. Section III presents a common structure of the proposed models. Section IV describes several topologies and operational modes for ambiguous VKs. Two methods are shown: disambiguation by scanning or word approach. In section IV a simple letter scanning-based model is depicted. In addition, the results are also reported in this section. The Tn mode is presented in section IV, and the required NEXT and SPACE functions are discussed. In addition, a probabilistic model and its validation are shown in this section. Finally, in section V, the model is used to establish a comparison among different ambiguous VKs and to state which VK could be better for an user under different user preferences and external conditions. An appendix, in which VKs considered in the analysis are represented, completes the paper.

[2] A generalization of T9 method

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

II. BACKGROUND

Nowadays, abundant scientific studies about text entry using a numerous varieties of methods and devices, such as mobile phones, PDA, etc., could be found. For instance, a model to predict WPM in mobile based on Fitts' law [22] is depicted in [23]. The study included different methods of text input such as multi-tap, T9, or two-keys. Other studies based on this law are shown in [24], [25], and [26]. These studies try to predict the performance of an expert user. A model for predicting the text entry speed of novice users based on Fitts' law is described in [27]. In [28], a combination of the power law of learning and theorical upper limit predictions is used to describe the development of text entry rates from users first contact to asymptotic expert usage. In addition, interesing studies based on GOMS models, such as those depicted in [29], [30], [31], and [32] have also been carried out. In [33], a study about indirect text entry methods and a model based on the notion of a containment hierarchy are presented. An evaluation of unambiguous VKs with character prediction is depicted in [34]. In this paper, a probabilistic model to predict WPM and $\overline{UI_c}$ using an indirect text entry method is shown.

An experimental study of WPM and KSPC for a mobile using keys disambiguation method based on prefixes (instead of a dictionary, as in T9) is presented in [16]. In that paper, character prediction establishes the likeliest character on the selected key, so that it will be shown in the first place according to the previous prefix. A KSPC next to 1.15 is obtained using this method. Other letter reassignments of a mobile keyboard are shown in [35], where an improved text entry is verified with different users. In other devices, such as PDAs, in which the number of keys are strongly reduced in favor of wider screens, software or VKs are developed for entring text toward the focused application. These VKs are representations of unambiguous keyboards (such as QWERTY) or ambiguous ones (such as mobile keyboards) that are controlled by a stylus. Studies of unambiguous VKs are presented in [24] and [36], where predictive models and user tests are included. In [37], a VK with 4 keys is described and tested using several languages. In [17], a study of the application of character prediction on ambiguous VK is depicted.

II. METHODOLOGY

A comparative study of user perfomance (measured in WPM and $\overline{UI_c}$) using scanning-based ambiguous VKs has been carried out in this paper. VKs with different number of keys using different scanning methods and implementing three disambiguation methods have been studied. Two considerations have been set to obtain the values of performance parameters:

1) A free error context. It is not necessary to implement a method to fix errors.

2) Expert users. Mental times, as search times, that are related to cognitive task, are optimal. The text entry is carried out in the optimal time.

Two methodologies can be applied, as mentioned earlier; one based on simulations and the other based on experiments. The users considered in this study have been suffering from severe motor disability. Thus, the use of a simple input device[3] is required. The chosen device depends on the user's skills. Each device has different characteristics that can influence the selection time, and thus, user performance. This study has tried to compare the VKs independently of the chosen input device. Using a methodology based on experiments, a trial by each configuration of VK is necessary to compare them. Furthermore, some previous sessions have to be carried out to obtain the desired level of experience. Much time and effort may be required by the user. In this study, a methodology based on simulations have been followed because of the difficulty of contact with severe motor handicapped people and aforementioned drawbacks.

To use a methodology based on simulations, some models that lead to predict the value of parameters in each case are required. In this sense, the general structure of these models is shown in Fig. 1.

[3] This is able to detect only one kind of user input

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

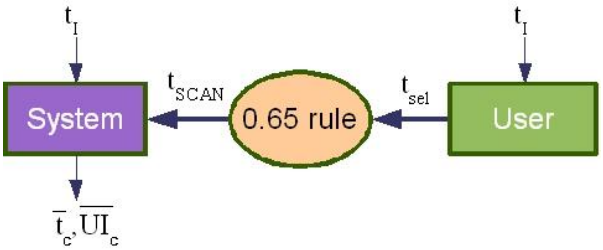

Fig. 1: The structure of models.

First, the reaction time, t_r, is predicted by a model of user behavior. This model is based on Keystroke-Level Model (KLM) [38] and [39]. The t_r is the elapsed time between the presentation of a sensory stimulus and the subsequent behavioral response as a button press. There are three kinds of reaction time experiments [40]: simple, recognition, and choice. In simple reaction time experiments, there is only one stimulus and one response. In recognition reaction time experiments, there are some stimuli that should have a response and others that should get no response. In choice reaction time, the user must give a response that corresponds to the stimulus. The value of t_r depends on the used input device. Therefore, this model has an input parameter that represents the interaction time, t_I[4]. Choice reaction times depend on the number of different stimulus according to Hick-Hymman's law [41]. In our context, with expert users who have a "mental map" of VK layout, the reaction time is reduced to accept or not the highlighted row, key or letter. Thus, the recognition reaction time seems to be more adequate. Henceforth, we will follow KLM notation [42], where the reaction time would include the mental preparation time, M, and the time to make a keystroke, K, or generate a user input. Therefore, the time M can be considered constant for all the processes related to the use of VK.

Once the selection time is predicted, a proper value of dwell time, T_{scan}, has to be calculated. [43], [44], and [45] have shown that the optimal scanning time is related to the reaction time by a constant equal to 0.65. Therefore, the reaction time establishes a lower bound for T_{scan} (Equation 1).

$$\min\{T_{scan}\} = \frac{t_r}{0.65} \qquad (1)$$

Then, the 0.65 rule may be used to obtain proper value of T_{scan}. Subsequently, T_{scan} is used as an input parameter in the model of system behavior. This one is a probabilistic model that emulates the given VK behavior.

III. AMBIGUOUS VKs WITH AUTOMATIC SCANNING

As mentioned earlier, ambiguous VKs require implementation of a disambiguation method. This method allows selection of a character among those on a chosen key. Disambiguation may be carried out in several ways: by using a new scanning of the letters on the chosen key, by character-level prediction, or by word-level approach.

4.1. Disambiguation by scanning: letter scanning

In this case, accessing the characters of a VK is performed by the scanning method. Once a key is selected, the characters on this key are scanned. This scan is usually linear. The value of the dwell time among the characters may be different from that set for scanning a row, column, or key. In this sense, this disambiguation method may be seen as an automatic version of multi-tap that is used on mobiles.

Another alternative is described in [46]. Fig. 2 represents a 12-key VK, where this method has been implemented. Two keys have special codes, called 2nd and 3rd, whose function is to enable next scan through the second or third option of each key, respectively. Instead of other VKs, here, row-column scans only include the first character of each key when the scan starts. To access the second or third character of a key, the user, first of all, has to select the key that contains the code 2nd or 3rd. Subsequently, the scan restarts, including only the characters in the positions indicated by the preselected code. The more likely characters, located in the first position of each key, need only two user inputs, and the rest need four user inputs. It must also be noted that the letter placement in VK with this operation mode differs from the row-column ones.

[4] Time required to interact with the given input device

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

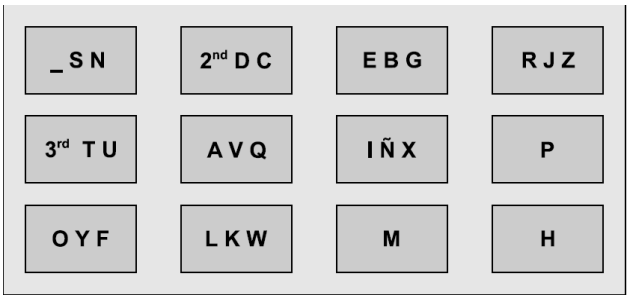

Fig. 2: 12-key VK using scanning depicted in [46]: RED12rcv

4.1.1. System Model

When using this disambiguation method, the value of $\overline{t_c}$ only depends on the times for accessing the characters in the considered context[5]. However, it is necessary to weigh the frequency of usage of each character in the language to obtain the average value of entry time per character. On the other hand, the character access time depends on the position of this character in the layout of VK, the set value of the dwell time, and the interaction time, t_I. The last one is an input parameter of the model, whose value is constant, because it only depends on used input device. As mentioned earlier, the value of the dwell time is obtained by applying the 0.65 rule to the value of the reaction time estimated by the user's model. Therefore, the value of T_{scan} is constant. In this way, once a layout of VK is set, the value of the access time of each character may be considered constant. Hence, the value of $\overline{t_c}$ is given by relationship shown in equation 2, where p represents a matrix of probability of the use of characters in an alphabet and t_{acc} is a matrix containing the access times of characters in a given layout.

$$\overline{t_c} = p^T \cdot t_{acc} \quad (2)$$

Similarly, the value of average UI is obtained in the same way. Let ui_{acc} be the matrix that represents the numbers of necessary user inputs to access a specific character in VK. Then, $\overline{UI_c}$ is given by Equation 3.

$$\overline{UI_c} = p^T \cdot ui_{acc} \quad (3)$$

[5] Free error context and an expert user.

4.1.2. Results

In this paper, ambiguous VKs with 4, 6, 9, 12, and 16 keys have been studied. The chosen layout of VKs is an optimal frequency-letter arrangement of characters in its keys. Two scanning methods have been implemented, a linear and a row-column. The character scanning method is linear. The dwell time is fixed to 1 s, and therefore, the scaling of $\overline{t_c}$ shown in Tables I and II by the appropriated value may give us the results for other dwell times. Obviously, $\overline{UI_c}$ is unaffected by the dwell time.

Although this study is based on ambiguous VKs, the values of $\overline{t_c}$ and $\overline{UI_c}$ using unambiguous VKs may be obtained with the proposed models. In this sense, the values of these parameters have been estimated considering an optimal frequency-letter arrangement of unambiguous VKs with a linear scanning method (CONVl) or a row-column scanning method (CONV).

Table I shows the $\overline{t_c}$ and $\overline{UI_c}$ for the VKs that use letter-scanning. The nomenclature used to identify a VK is RED_NK_KSM, in which NK is the number of VK keys and KSM may be l for linear or rc for row-column.

Table I: Letter-scanning operation mode results for different layouts using row-column or linear key scanning, and linear scanning for characters into a key and T_{scan}=1.}

	$\overline{t_c}$ (s)	$\overline{UI_c}$
RED 4l	5.42	2.00
RED6l	5.35	2.00
RED9l	5.42	2.00
RED12l	5.69	2.00
RED16l	6.65	2.00
CONVl	9.59	1.00
RED4rc	5.94	3.00
RED6rc	5.64	3.00
RED9rc	5.51	3.00

RED12rc	5.51	3.00
RED16rc	5.61	3.00
CONV	5.35	2.00

It can be seen for row-column scanning that $\overline{t_c}$ progressively decreases in relation to the increase in ambiguity, reaching a minimum of 6 keys in VK. Likewise, for linear key scanning, something similar occurs. On comparing the two scanning methods, linear ones are observed to achieve better times than the row-column ones, excluding RED16, and have better number of user inputs. A keyboard with 4, 6, or 9 keys, when a linear scan is used, is able to obtain a $\overline{t_c}$ close to the unambiguous one and with an identical $\overline{UI_c}$.

Table II shows the values that different VKs achieve using the disambiguation method described in [46]. It can be observed that as the ambiguity of the keyboard increases, the benefits worsen. Furthermore, when compared with their counterparts presented in Table I, only keyboards with 12 or 16 keys are found to show improvement. VKs with 9 or less keys show worse results, because they require a larger number of special codes to control the following scan.

Table II: Letter-scanning operation mode results for different layouts using scanning depicted in [46] with T_{scan}=1 s.

	$\overline{t_c}$ (s)	$\overline{UI_c}$
RED4rcv	7.13	4.40
RED6rcv	6.71	3.65
RED9rcv	5.75	2.93
RED12rcv	5.55	2.66
RED16rcv	5.41	2.35

To summarize, better results for ambiguous VKs are obtained by 4-, 6-, 9-, and 12-key VKs with linear scanning of its keys, and RED16rc with the row-column scanning depicted in [46]. Only 4-, 5-, 9-key VKs are found to be very close to unambiguous keyboard.

4.2. Disambiguation by word approach: Tn mode

This disambiguation method is based on T9 method for mobiles. To use this method, two functions are required, NEXT and SPACE. As the text entry is continued, the most likely word associated with the sequence of the selected keys is shown. In most cases (95\% according to [23] for mobile phones), the sought word is predicted. Thus, the SPACE key has to be selected to accept this word and then the text entry is continued by the next word. Only in 5\% of the cases, the suggested word is not accepted. Under these circumstances, the NEXT function has to be selected repeatedly to show other suggestions. In addition, the sought word may not be found in the dictionary and obviously it will not be shown as a suggestion. This is the worst case scenario, and an alternative text entry method has to be used in this case, for example, multi-tap, to type the wished word completely.

In short, the T9 system for mobile phones is very efficient. In [23], it has been shown that WPM is greater using T9 than using multi-tap, and in [16], a comparative table for mobile phones has been presented, where KSPC, using a T9 method, is close to the unit. Nonetheless, in both cases, if the word is not in the dictionary or is not shown in the first position, KSPC will be slightly greater than 1.

4.2.1 NEXT and SPACE functions

To implement this method in an ambiguous VK with n keys (Tn method), a NEXT key is required. Once the NEXT key is selected, the next suggested word is shown. This procedure should be repeated until the sought word is shown or the last suggestion is reached. However, this may produce fatigue to the user and increase the number of user inputs. Hence, it would be preferable to use an automatic variant that can scan the items of the suggestion list in a linear way. This variant requires only an additional user input to choose the word of the list.

On the other hand, the SPACE character has an additional function: to accept the first suggested word. For this reason, the SPACE character may not be integrated with other characters in the same key, because the system would not be able to distinguish if the user is selecting the suggested word or he/she is typing a new character. In this sense, two alternatives of layout may be used to implement the Tn mode: to

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

integrate both the functions, NEXT and SPACE, in the same key, or to use two separate keys.

Some difficulties have been encountered when two new keys are integrated in an ambiguous VK. First, in VKs that are highly ambiguous, for example, from 4 up to 6 keys, WPM may be affected because it would be necessary to carry out a new arrangement of characters in the keys. Second, finding an optimal layout of VK would be a complex task, because the letter arrangement depends on the NEXT key position, and this depends on the used dictionary and, in turn, the letter arrangement in the VK. Third, an optimal VK using Tn mode locates the most likely characters inside the keys that are closest to the scanning start point. In highly ambiguous VKs, this fact could drive to a time penalty, because a sequence of the selected keys is associated with a great number of words. Hence, the probability of the desired word is that the first suggestion is reduced and the use of NEXT key is increased. Such a situation could increase the $\overline{t_c}$ as the user has to search the desired word on the suggestion list. Thus, it is possible that the scrambled arrangement would have a great probability of finding the desired word in the first position, enhancing the text entry rate. Both these situations will be studied in this paper.

The general structure of a VK, where the NEXT and the SPACE functions are separated in two keys, is shown in Fig. 3. The first suggested word is shown on the viewer. If the SPACE key is selected, the suggested word is accepted and the SPACE character between the words is automatically introduced. If the NEXT key is selected, other suggestions are shown in a new window. In this case, a word is highlighted in each scanning step. In such a situation, a user input has to be made to introduce the current highlighted word in the text. Nevertheless, when the end of the list is reached by the scanning method, the alternative text entry method is used to type the desired word completely.

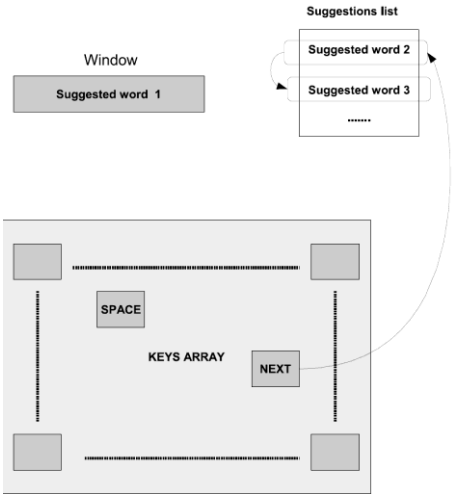

Fig. 3: Ambiguous VK with SPACE and NEXT in independent keys. See text for description.

The general structure of a VK using Tn mode, where the NEXT and the SPACE function are integrated in the same key, is shown in Fig. 4. Once all keys containing all letters of the desired word are selected, the SPACE-NEXT key has to be selected. Subsequently, the list of suggestions appears, and the linear scanning starts. An additional user input is required to select the highlighted word. As before, when the end of the list is reached by the scanning method, the alternative text entry method is used to type the desired word completely. In this solution, an extra user input has to be made if the sought word is shown in the first position. In addition, there is also an increment in the entry time per character, because an additional scanning cycle is always necessary to the focus on the list obtained after the SPACE-NEXT key is selected.

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

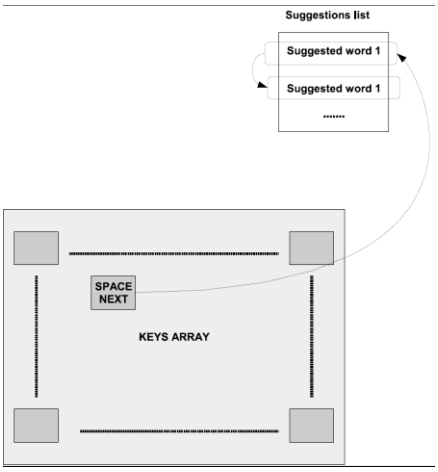

Fig. 4: An ambiguous keyboard with the SPACE-NEXT installed in the same key. See the description of the operation in the text.

4.2.2 System Model

Text entry using Tn mode starts by selecting all keys that contain the characters of the desired word. Accordingly, three cases may be presented:

1) The desired word is the first option in the suggestion list. Therefore, it is shown on the viewer. In this case, the SPACE key has to be selected to enter the word. The average time required to do this operation is represented by $\bar{t}_{first}$. On the other hand, the average number of user inputs required to carry out this operation is represented by $\overline{UI}_{first}$.

2) The desired word is in the suggestion list, but it is not the first option. Hence, the NEXT key has to be selected to scan the next suggestions. Once the word is highlighted, a user input has to be made to select it. $\bar{t}_{nofirst}$ represents the average time to carry out this operation, and $\overline{UI}_{nofirst}$ represents the average number of user input.

3) The desire word is not in the suggestion list. As in the last case, the NEXT key has to be selected to scan the next suggestions. Once the last suggestion is highlighted, the letter-scanning mode is activated. Then, the word has to be entered using this text entry method. The average time required to do this operation is represented by $\bar{t}_{fail}$ and the average number of user input is represented by $\overline{UI}_{fail}$.

The time and the number of user inputs to enter a word depends on the presented case. Therefore, the value of $\bar{t}_c$ and $\overline{UI}_c$ also depends on this. Hence, these cases have to be considered by the system model. A descriptive model of a user interacting with a VK using Tn mode is shown in Fig. 5. Four tasks are represented by a rectangle in this descriptive model: selecting the keys that contain the character of the word (it also includes the selection of NEXT-key or SPACE-key), scanning the suggestion list and typing the word using the letter-scanning mode. In this figure, the three above-mentioned cases are represented. Each case is related to a path that is represented by a dashed line. The first case is related to path 1 (Red), the second one to path 2 (blue) and the third one to path 3 (green).

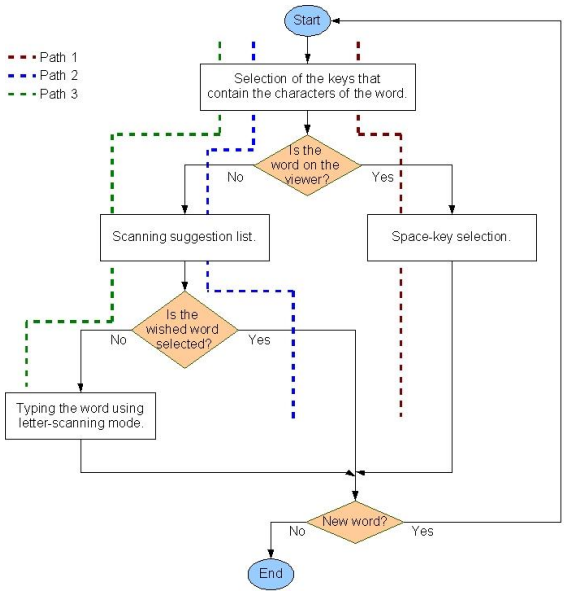

Fig. 5: Model structure.

The average value of time and number of user inputs to enter a word using each path has to be obtained separately. Subsequently, the value of $\bar{t}_c$ and $\overline{UI}_c$ may be estimated by a weighted addition. For that, the average value of time and number of user inputs using each path are calculated by the length of the word and some probabilistic parameters are required. The value of some of these parameters is obtained

using the information contained in a dictionary. Others are input parameters of the model. In this sense, Equation 4 and 5 represent the value of $\bar{t}_c$ and $\overline{UI}_c$ respectively. The probability of entering a word that has n characters (n-word) is represented by $p(w_n)$. The unit in the denominator is due to a space character that follows each word.

$$\bar{t}_c = \frac{\sum_n p(w_n) \cdot T_w(n)}{1 + \sum_n n \cdot p(w_n)} \quad (4)$$

$$\overline{UI}_c = \frac{\sum_n p(w_n) \cdot UI_w(n)}{1 + \sum_n n \cdot p(w_n)} \quad (5)$$

To estimate the value of $T_w(n)$ and $UI_w(n)$, the three aforementioned cases are considered, as it can be checked in Equations 6 and 7. In these, the probability of a n-word in the prediction list is represented by $p(w_n \in L_\Omega)$, and the probability of a n-word in the first position in this list is represented by $p_{in}(w_n \in L_\Omega)$.

$$\begin{aligned} T_w(n) = {} & p(w_n \in L_\Omega) \cdot \left[p_{in}(w_n \in L_\Omega) \cdot \bar{t}_{first}(n) \right. \\ & \left. + (1 - p_{in}(w_n \in L_\Omega)) \cdot \bar{t}_{nofirst}(n) \right] \\ & + (1 - p(w_n \in L_\Omega)) \cdot \bar{t}_{fail}(n) \end{aligned} \quad (6)$$

$$\begin{aligned} UI_w(n) = {} & p(w_n \in L_\Omega) \cdot \left[p_{in}(w_n \in L_\Omega) \cdot \overline{UI}_{first}(n) \right. \\ & \left. + (1 - p_{in}(w_n \in L_\Omega)) \cdot \overline{UI}_{nofirst}(n) \right] \\ & + (1 - p(w_n \in L_\Omega)) \cdot \overline{UI}_{fail}(n) \end{aligned} \quad (7)$$

4.2.3 Validation

To check the model effectiveness, a program has been made in C, which interacts with a database MySQL that houses the dictionary. This dictionary consists of 10,000 words with their absolute and relative frequencies from a Corpus [47]. The words included in the dictionary represent 17,672,326 words from the Corpus (if the Corpus were used as a sample text, it would generate a hit rate in the dictionary of 88%). The program emulates the VK behavior having some input parameters, as mentioned in section III. Furthermore, the context set in that section is considered by this emulation. Besides, this program uses a sample text selected from a collection of some kinds of documents: sports, religion, etc., with a total of 960,180 words and an average of 4.91 characters per word. The sample text is divided into 30 segments of an approximate equal size. For each segment, a set of parameters is gathered: frequency of letters, frequency of n-character words, probabilities vectors described in the above-mentioned sections, etc. Thus, each segment of the sample text represents different simulation conditions. In addition, the value of $\bar{t}_c$ and $\overline{UI}_c$ for each segment is also calculated. The mean hit rate is equal to 85%.

A Matlab program is also made to obtain the model results. This Matlab program reads the dictionary, builds statistical information from it, and calculates the value of $\bar{t}_c$ and $\overline{UI}_c$ according to the model. By using Matlab and C programs, the goodness of the model may be tested.

The simulation (C program) results (blue line) and model results (circles) are shown in Fig. 6. Furthermore, a simplification of the model has been carried out, and its results are shown by a red line with x-marks in this figure. Each segment is represented in x-axis, while the value of $\bar{t}_c$ in seconds is represented in y-axis. As it can be seen, the results from both the models are very close to simulation ones. The maximum relative error for $\bar{t}_c$ is 0.61% using the model or 3.26% using the simplification of the model.

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

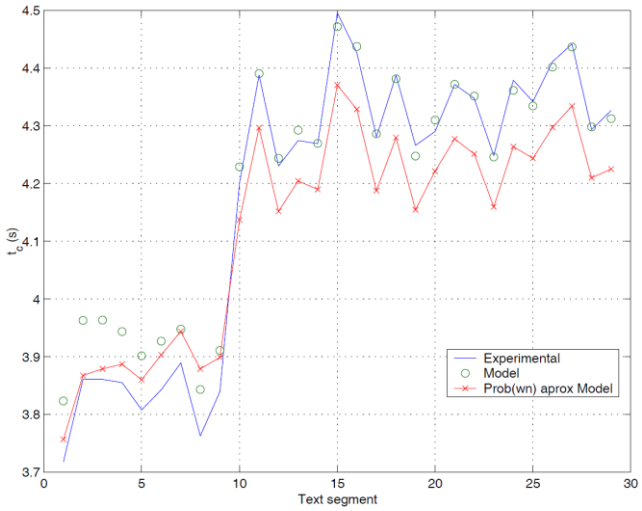

Fig. 6: Average text entry time per character for a RED4 keyboard obtained in each segment: simulation and model results.

The results for $\overline{UI_c}$ are shown in Fig. 7. As in the above-mentioned case, the results of the models are very close to those of the simulation ones: the maximum relative error for $\overline{UI_c}$ is 0.18% using the model or 4.14% using the simplification of the model.

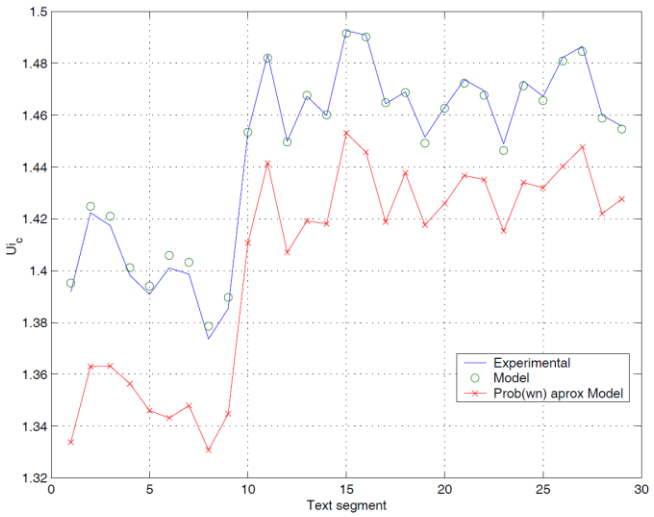

Fig. 7: Average number of user inputs per character for RED4 keyboard obtained in each segment: simulation and model results.

This trial has been repeated using some different VKs obtaining similar results. Thus, it can be concluded that these low relative errors are a proof of the goodness of the model and simplification.

4.2.4 Results

The performance of different ambiguous VKs has been compared in this section. The layouts of these VKs are shown in Appendix A. VKs are sized from 4, 6, 9, and 12 keys, and two different character arrangements are considered: PT and nonPT. In the first case, the time to access a key in Tn mode is used to place the most frequent characters. In the other case, the time to access a position in the letter-scanning method is used to set the character arrangement. In this sense, RED4PT represents an ambiguous VK with 4 keys and PT arrangement, and RED16 is related to a 16-key ambiguous VK with non-PT arrangement. Only a linear-scanning method is considered because of its low value of $\overline{UI_c}$.

The time required to enter a word with n characters (n-word) that are in the dictionary is shown in Fig. 8. Word length is represented in x-axis and the value of this time in seconds is represented in y-axis. In general, as VK ambiguity diminishes, time increases. This is owing to the fact that the scanning method needs longer time to reach different keys. As expected, the PT arrangement results are better in 9-key and 12-key VKs. However, it is completely opposite for 4-key and 6-key VKs. Taking into account the fact that the most frequent words have lengths between 2 and 9, RED4 and RED6 are found to present better results. It must be noted that RED4, RED4PT, and RED6PT show great values of this time, which makes the plot to move away from the constant slope in other cases. For RED4PT VK, this deviation is particularly important. These deviations are due to the fact that as ambiguity is increased, the number of words that are matched with a sequence of selected keys is increased, and hence, the length of the suggestion list is longer, and this involves a greater time to select the desired one. In the case of RED4PT, almost every most-frequent character is placed on the same key, and thus, the number of words that are matched with a sequence of selected keys is huge. Highly ambiguous VKs using a PT arrangement show worse results when compared with the others. Obviously, the probability of a word in the dictionary ($p(w \in \Omega)$) should be close to 1, and hence, the best results of $\overline{t_c}$ may be obtained by RED6, RED4, and RED6PT.

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

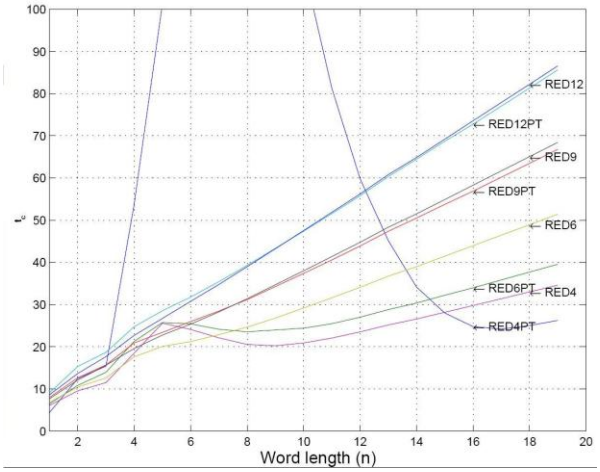

Fig. 8: Time required to enter a word with n characters that is in the dictionary using the studied VKs.

If $p(w \in \Omega)$ is moving away from 1, the letter-scanning mode is used. In such a situation, two cases may occur: 1. the sequence of the selected keys is matched with a word or some words in the dictionary, generating a suggestion list; or 2. this sequence is not matched with any words in the dictionary. Therefore, the time to type a n-word may be broken down into two terms: time to type in the first case, $\bar{t}_{fail\_List}(n)$, and time to type in the second case, $\bar{t}_{fail\_NoList}(n)$. Obviously, $\bar{t}_{fail\_List}(n)$ is greater than $\bar{t}_{fail\_NoList}(n)$. Both the terms are weighted using the probabilistic parameters. Thus, the best scenario is represented by the situation in which if the desired word is not in the dictionary, an empty suggestion list is always predicted. Fig. 9 shows $\bar{t}_{fail\_NoList}(n)$ for the VKs under study. In this situation, the best results are shown by RED6, RED4, and RED6PT.

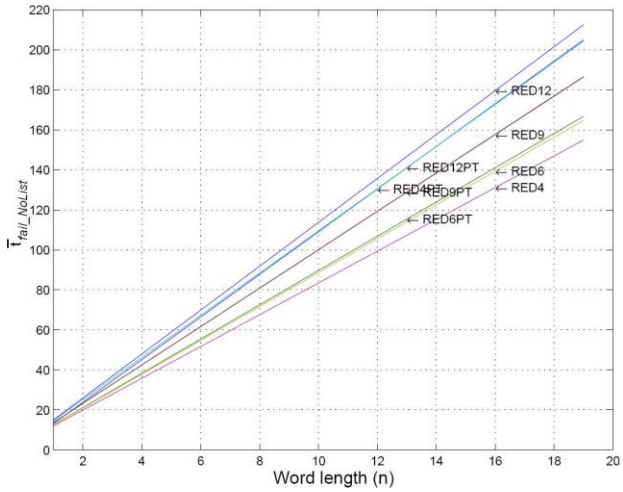

Fig. 9: Time required to enter a word with n characters that is not in the dictionary, using the studied VKs when a suggestion list is never generated.

On the other hand, the worse scenario, where a suggestion list is always predicted when the desired word is not in the dictionary, is shown in Fig. 10. In this case, the user always has to wait for the scanning of the suggestion list before starting to type in letter-scanning mode. The behavior of VKs is similar to that shown in Fig. 8. It must be noted that RED6 continues to be a good choice in the range between 1 and 0 n-words. RED9 and RED9PT also show a good behavior for $\bar{t}_c$ indicator.

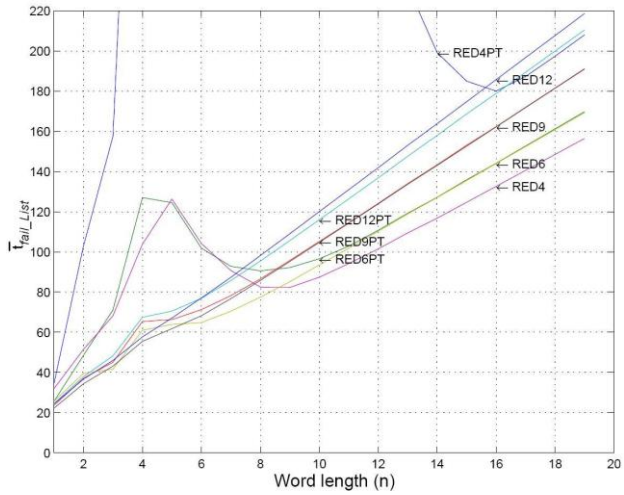

Fig. 10: Time required to enter a word with n characters that is not in the dictionary, using the studied VKs when a suggestion list is always generated.

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

Fig. 11 shows the number of user inputs when the desired word is in the dictionary. Besides, the best VK considering $\overline{UI_c}$ may be set from this figure, which is independent of the value of $p(w \in \Omega)$, because the number of user inputs is the same in any case. The best results are obtained for RED12, RED9, RED6, and RED12PT VKs.

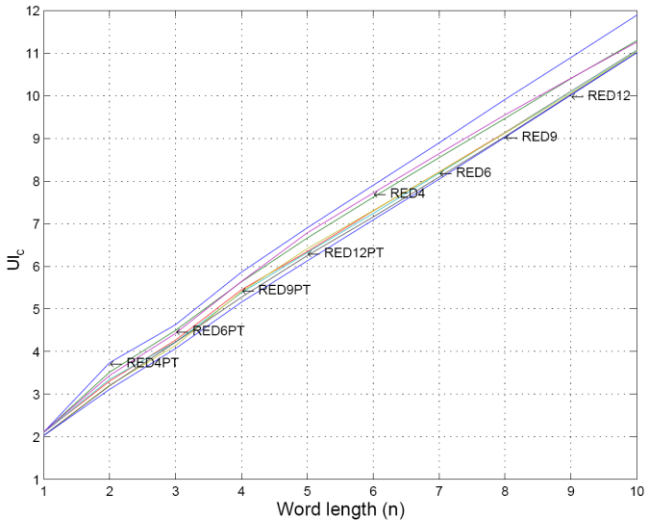

Fig. 11: Number of user inputs when the desired word is in the dictionary.

V. SELECTING A VK

In the previous sections, a study on the behavior of different VKs according to $\overline{UI_c}$ and $\overline{t_c}$ have been presented. The model can be used to estimate $\overline{t_c}$ and $\overline{UI_c}$ for any VK and $p(w \in \Omega)$. A relationship between these parameters has to be established to select the best VK under certain conditions. In this sense, the best option may be defined as the VK that reduces both the parameters. A function F that measures the VK performance may be built. This function must consider user preferences in relation to the text entry rate and the number of user inputs. Accordingly, two variables have been included, a_1 and a_2. Their values could change from 0 up to 1, and a_1 is related to the text entry rate. In this way, a value that is close to 1 shows that the user gives much relevance to the text entry rate. On the other hand, a_2 is related to the number of user inputs. In a similar way, a value that is close to 1 shows that the user gives much relevance to the number of user inputs. These estimations can be employed in Equation 8, so that a numerical value to F_k for the keyboard k can be obtained depending on a_1 and a_2.

$$F_k = a_1 \cdot \frac{\bar{t}_{c_k}}{\max_{\forall j}\left(\bar{t}_{c_j}\right)} + a_2 \cdot \frac{\overline{UI}_{c_k}}{\max_{\forall j}\left(\overline{UI}_{c_j}\right)} \qquad (8)$$

In this equation, the text entry time per character using a given VK, k, is represented by $\bar{t}_{c_k}$. The number of user inputs per character using a given VK is represented by $\overline{UI}_{c_k}$. Both the parameters are scaled according to the maximum values associated with the set of VKs in this study. Subsequently, given a_1 and a_2, the best VK would be the having the lowest F_k.

The function F_k can be made into a bi-dimensional matrix according to values assigned to a_1 and a_2 parameters. For example, if a_1 and a_2 change from 0 up to 1 by 0.1 steps, the matrix will be 11 x 11. The following figures are representations of the comparison results among the F matrices obtained for all the VKs, according to a_1 and a_2 value. The figures show the areas where a VK is the best or have the lowest F_k value. For example, we can see from Fig. 17 that there exists four areas, and therefore, four VKs achieve better performance when $p(w \in \Omega)$ =1. If a user prefers to obtain better type speed (a_1 close to 1) and does not care the number of inputs, an RED4 VK must be chosen. If another user does not care about the type speed and wants to minimize the number of inputs (a_2 close to 1), a CONVL must be chosen. If a third user wants to minimize both the parameters, an RED6 VK must be selected.

As mentioned earlier, in Tn mode, $\overline{t_c}$ and $\overline{UI_c}$ are estimated by the length of the word and some probabilistic parameters are required. Therefore, to predict the value of $\overline{t_c}$ and $\overline{UI_c}$ using a given VK, it is necessary to set the value of these parameters: the probability of typing a n-word, $p(w_n)$, the probability of a sequence of the selected keys generating a suggestion list when the desired word is not in the dictionary, x, and the probability of the desired word is in the dictionary, $p(w \in \Omega)$. First, $p(w_n)$ may be set using the information obtained in the validation. To set an appropriate value of x, it is necessary to take into account the VK layout. For example, in a 4-key VK, the probability of a sequence of the selected keys generating a suggestion list is almost 1, because

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

of the great amount of characters associated with a key. However, for 9-key and 12-key VKs, this probability is close to 0. This is owing to the fact that all words included in the dictionary are matched with only a sequence, and hence, the probability of a sequence that is not matched with a word in the dictionary generates a suggestion list is practically 0. For this reason, the following approximation may be carried out: for 4-key VKs, x = 1, for 9-key and 12-key VKs, x = 0, and for 6-key VKs, x = 0.5. Finally, values from 0 to 1 may be assigned to $p(w \in \Omega)$ to contemplate different conditions.

In this study, the following VKs were included: CONVL, CONV, RED4, RED6, RED6PT, RED9, RED9PT, RED12, and RED12PT, which comprise, all VKs in Appendix A, excluding 4PT. For each VK, the values of F considering different values of a_1 and a_2 have been estimated. In each case, a value of $p(w \in \Omega)$ has been set. In this way, the areas in function of the values of a_1 and a_2 may be defined. Each area represents a VK having the lowest value of F in relation to others. A special area is represented by coordinates a_1, a_2 in [0.9; 1], in which both the parameters, $\overline{t_c}$ and $\overline{UI_c}$, should be minimized. This area is called Maximum Interest Area (MIA).

By assuming a $p(w \in \Omega)$ =0.5 (Fig. 12) only CONVL, CONV, and RED6 VKs may be offered to a user depending on their preferences. By focusing on the piece of area delimited by coordinates a_1, a_2 in [0.9; 1], CONVl can be selected.

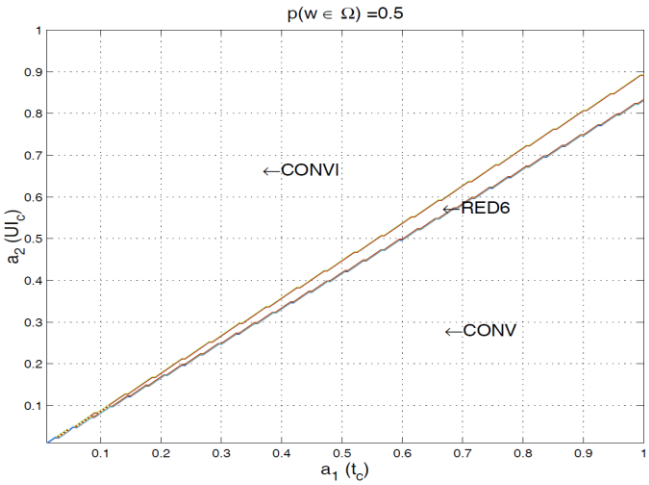

Fig. 12: Areas with $p(w \in \Omega)$ =0.5.

Areas assuming a $p(w \in \Omega)$ =0.6 are shown in Fig. 13. An increase in RED6 area is presented, reducing both CONV and CONVl areas. In this case, using values of a_1, a_2 in [0.8; 1], RED6 could be selected, which includes MIA.

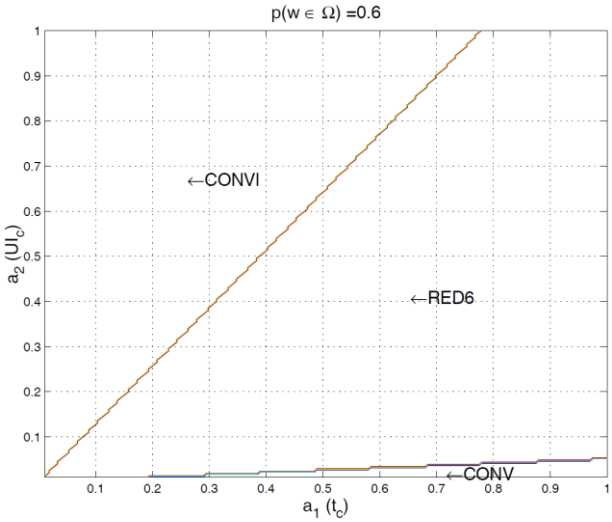

Fig. 13: Areas with $p(w \in \Omega)$ =0.6.

The CONV area disappears when $p(w \in \Omega)$ >0.6 as it may be seen in Fig. 14 - 17. Furthermore, as $p(w \in \Omega)$ increases, the CONVl area decreases. RED6 could be selected for values of a_1 > 0.5 assuming $p(w \in \Omega)$ =0.7, for values of a_1 > 0.3 assuming $p(w \in \Omega)$ =0.8 and for values of a_1 > 0.15 assuming $p(w \in \Omega)$ =0.9.

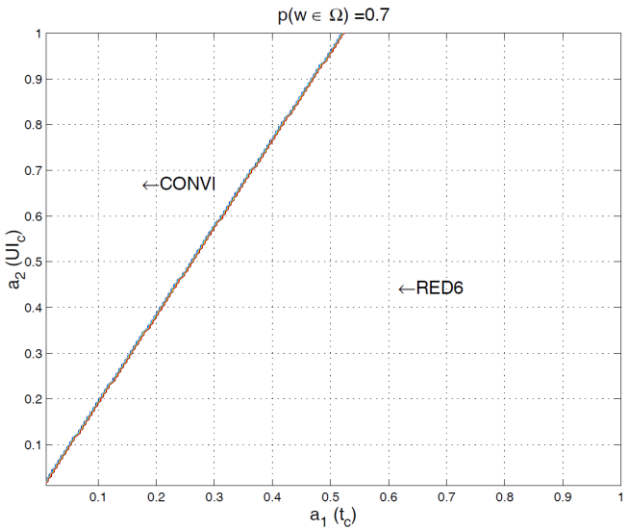

Fig. 14: Areas with $p(w \in \Omega)$ =0.7.

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

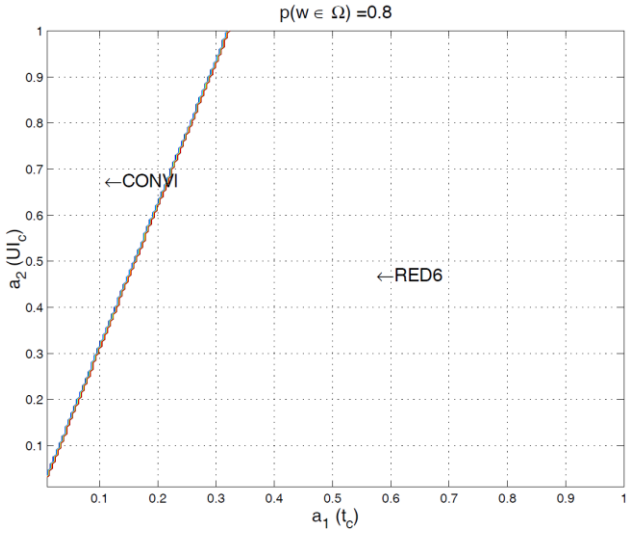

Fig. 15: Areas with $p(w \in \Omega)$ =0.8.

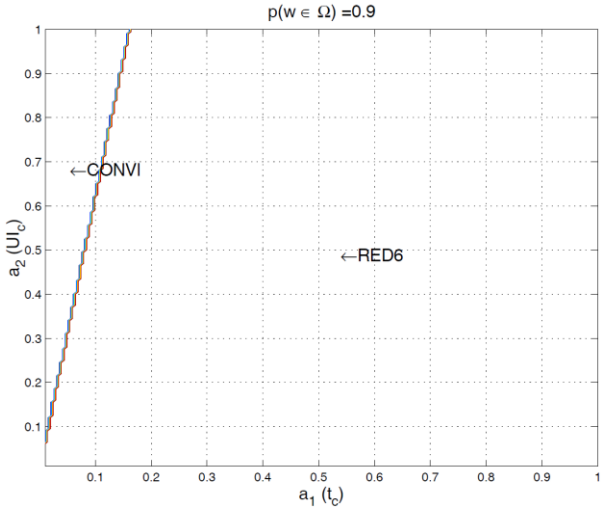

Fig. 16: Areas with $p(w \in \Omega)$ =0.9.

By assuming $p(w \in \Omega)$ =1, where the desired word is always in the dictionary, CONVl, RED12, RED6, and RED4 may be selected. The RED6 area is the greatest one and is placed in the central position. However, for low values of a_2, RED4 is the most likely option.

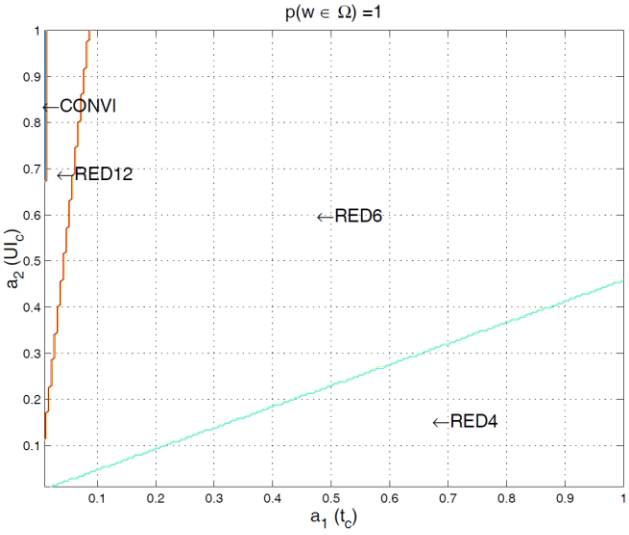

Fig. 17: Areas with $p(w \in \Omega)$ =1.

Furthermore, the effect due to assigning different values of x has also been checked. By assuming that $p(w \in \Omega)$ >=0.8, no changes have been found on the previously obtained values. For lower $p(w \in \Omega)$ <0.8, RED9 or RED12 might replace RED6 on AMI.

VI. CONCLUSIONS

In this paper, some ambiguous VK layouts have been studied. The text entry rate using these VKs in Tn mode is greater than the one using an unambiguous VK. In this mode, a dictionary, a NEXT function, and a SPACE function are required. Furthermore, the convenience of having the SPACE and NEXT key separately on the VK without additional functions or characters in them has also been discussed. Besides, it has been tested that PT arrangement neither improves $\overline{t_c}$ nor $\overline{UI_c}$.

In addition, a model based on the dictionary has been developed. The proposed model allows estimating the values of both the $\overline{t_c}$ and $\overline{UI_c}$. A simple user model based on KLM has been used to estimate the value of t_r. This fact allows making comparison independently of the used input device. The 0.65 rule has been used to estimate a proper value of the dwell time. No simulations are required to compare different VKs. The values of some needed parameters have been set using the information in the dictionary to compare them, and model validity has been demonstrated even in its simplified version. The model allows us to

www.ijmer.com Vol.1, Issue.2, pp-288-305 ISSN: 2249-6645

understand how VKs behave according to the changes in $p(w \in \Omega)$. When using this model, participation of real users in trials is not necessary. Hence, a prototype of the system is not required, and the behavior of any input device can be emulated. A function that assesses the user preferences on $\overline{t_c}$ and $\overline{UI_c}$ indicators and allows comparison of VKs has also been proposed. According to that function, the RED6 VK has been found to show better performances even in a wide range of $p(w \in \Omega)$.

APPENDIX A: CONSIDERED VKs

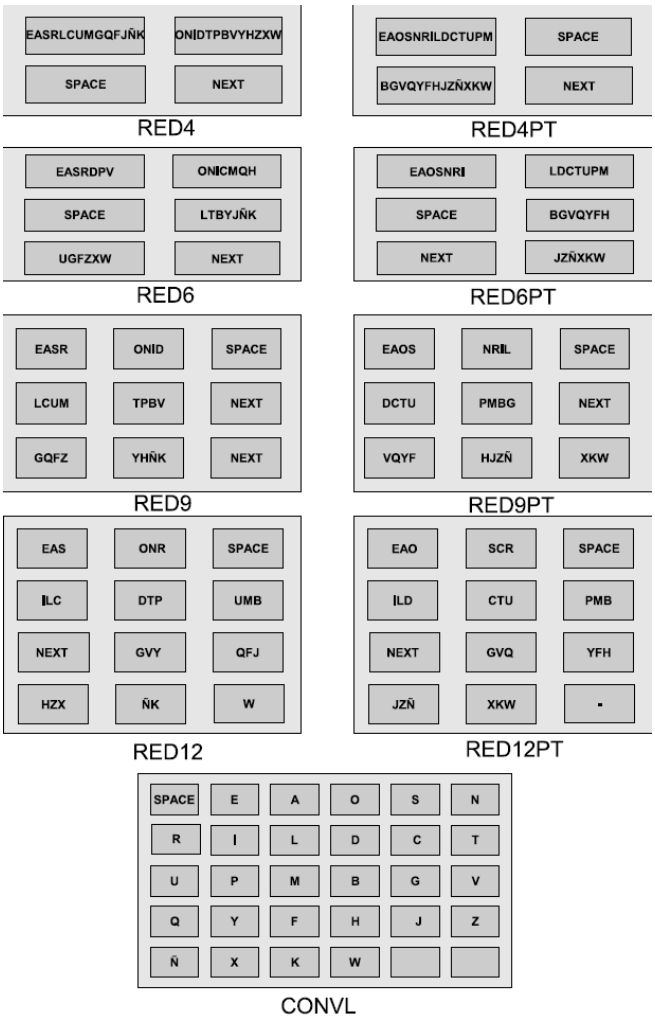

Fig. 18: Considered VKs.

REFERENCES

[1] Kushler, "AAC using a reduced keyboard," *Proceedings of CSUN* 98, 1998.

[2] K.-N. Kim and R. S. Ramakrishna, "Vision-based eye-gaze tracking for human computer interface," vol. 2, 1999, pp. 324–329.

[3] B. Noureddin, P. D. Lawrence, and C. F. Man, "A non-contact device for tracking gaze in a human computer interface," Computer Vision and Image Understanding", vol. 98, n0. 1, pp. 52-82, 2005, special issue on Eye Detection and Tracking. Available: http://www.sciencedirect.com/science/article/B6WCX-4D99BSP-1/2/24b89233ae82e7d1385951212a3151a7

[4] Y.-L. Chen, F.-T. Tang, W. Chang, M.-K. Wong, Y.-Y. Shih, and T.-S. Kuo, "The new design of an infrared-controlled human-computer interface for the disabled," Rehabilitation Engineering, IEEE Transactions on, vol. 7, no. 4, pp. 474–481, Dec 1999.

[5] D.G. Evans, R. Drew, and P. Blenkhorn, "Controlling mouse pointer using an infrared head-operated joystick", IEEE Transactions on Rehabilitation Engineering, vol. 8, no. 1, pp. 107-117, Mar 2000.

[6] O. Rivera, A. J. Molina, and I. M. Gómez, "Versatile system to computer control based on infra-red light," 24th Human Factors and Ergonomics Society Europe Chapter. Num. 24. Shaker Publishing, pp. 4–5, 2007.

[7] I.M. Gómez, A. J. Molina, O. Rivera, "Multipurpose interface for handling general computer applications", AAATE 2007, 9th European Conference for the Advancement of Assistive Technology in Europe, pp. 7-8, 2007.

[8] F. L. Castro, "Class I infrared eye blinking detector", Sensors and Actuators A: Physical, vol. 148, no. 2, pp. 338-394, 2008. Available: http://www.sciencedirect.com/science/article/B6THG-4TDK6SC-4/2/6826bf33afa9efa561f71c97b3992b20

[9] P. O'Neil, C. Roast, and M. Hawley, "Evaluations of scanning user interfaces using real-time-data usage logs", Proceedings of the fourth international ACM conference on Assistive technologies (ACM SIGACCESS Conference on Computers and Accessibility), pp. 137-141, 2000.

[10] N. Alm, J. L. Arnott, and A. Newell, "Prediction and conversational momentum in an

augmentative communication system." Comm ACM, pp. 46–57, 1992.

[11] H. Horstmann, E. Lopresti, and R. C. Simpson, "Toward automatic adjustment of keyboard setting for people with physical impairments." Diability and Rehabilitation: Assistive Technology., pp. 261–274, 2007.

[12]] K. Harsbush and M. Khn, "An evaluation study of two-button scanning with ambiguous keyboards," Proceedings AAATE, pp. 954–958, 2003.

[13] P. Gnanayutham, C. Bloor, and G. Cockton, "Soft keyboard for the disabled," Computers Helping People with Special Needs, Lecture Notes in Computer Science, 2004.

[14] G. W. Lesher, B. J. Moulton, and D. J. Higginbotham, "Techniques for augmenting scanning communication," International Society on Augmentative and Alternative Communication(ISAAC), no. 2, pp. 81-101, 1998.

[15] T. Bellman and I. MacKenzie, "A probabilistic character layout strategy for mobile text entry," In Proc. Graphics Interface, pp. 168–176, 1998. [Online]. Available: citeseer. comp.nus.edu.sg/bellman98probabilistic.html

[16] I. S. MacKenzie, H. Kober, D. Smith, T. Jones, and E. Skepner, "Letterwise: prefix-based disambiguation for mobile text input," in UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology. New York, NY, USA: ACM, 2001, pp. 111– 120.

[17] J. Arnott, Probabilistic character disambiguation for reduced keyboards using small text samples. Taylor & Francis, September 1992, vol. Augmentative and Alternative Communication. Available: http://www.ingentaconnect.com/content/tandf/taac/1992/00000008/00000003/art00004

[18] H. Horstmann and S. Levine, "Modeling the speed of text entry with a word prediction interface," IEEE transactions on rehabilitation engineering, vol. 2, pp. 177–187, 1994.

[19] Y. Gillette and J. L. Hoffman, "Getting to word prediction: developmental literacy and aac." Proceedings of the 10th annual international conference, technology and persons with disabilities, 1995.

[20] B. Heinisch and J. Hecht, "Predictive word processors: a comparison of six programs." Tam News., pp. 4–9, 1993.

[21] N. Garay-Vitoria and J. Abascal, "Text prediction: a survey," Univ Access Inf Soc, pp. 188–203, 2006.

[22] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement." J Exp Psychol, vol. 47, no. 6, pp. 381–391, June 1954. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/13174710

[23] M. Silfverberg, I. S. MacKenzie, and P. Korhonen, "Predicting text entry speed on mobile phones," in CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems. New York, NY, USA: ACM, 2000, pp. 9–16.

[24] I. S. Mackenzie, S. X. Zhang, and R. W. Soukoreff, "Text entry using soft keyboards," Behaviour & Information Technology, vol. 18, pp. 235–244, 1999.

[25] R. W. Soukoreff and I. S. MacKenzie, "Theorical upper and lower bounds on typing speeds using a stylus and soft keyboard." Behavior & Information Technology, pp. 370–379, 1995.

[26] I. S. MacKenzie and R. W. Soukoreff, "A model of two-thumb text entry." Proceedings of Graphics Interface, pp. 117–124, 2002.

[27] A. Pavlovych and W. Stuerzlinger, "Model for non-expert text entry speed on 12-button phone keypads." Proceedings of the SIGCHI Conference of Human Factors in Computing Systems., pp. 351–358, 2004.

[28] P. Isokoski and I. S. MacKenzie, "Combined model for text entry rate development." Proceedings of the ACM CHI 2003 Conference on Human Factors in Computing Systems, pp. 752–753, 2003.

[29] H. Horstmann and S. P. Levine, "Validation of a keystroke-level model for a text-entry system used by people with disabilities." Proceedings of the First ACM Conference on Assistive Technologies., pp. 115–122, 1994.

[30] H. Horstmann and S. P. Levine, "Effect of a word prediction feature on user performance." AAC Augmentative and Alternative Communication, pp. 155–168, 1996.

[31] H. Horstmann, "Keystroke-level models for user performance with word prediction," ISAAC, pp. 239–257, 1997.

[32] H. Horstmann and S. P. Levine, "Model simulations of performance with word prediction." Augmentative Alternative Communication., pp. 25–35, 1998.

[33] M. Baljko and A. Tam, "Indirect text entry using one or two keys." Proceedings of the 8th international ACM SIGACCESS Conference on Computers and Accessibility, pp. 18–25, 2006.

[34] A. J. Molina, O. Rivera, I. M. Gómez, and G. Sánchez, "Evaluation of unambiguous virtual keyboards with character prediction," AAATE 2009. Assistive Technology from Adapted Equipment to Inclusive Environments, pp. 144–149, 2009.

[35] H. Ryu and K. Cruz, "Letterease: Improving text entry on a handheld device via letter reassignment," in OZCHI '05: Proceedings of the 17th Australia conference on Computer-Human Interaction. Narrabundah, Australia, Australia: Computer-Human Interaction Special Interest Group (CHISIG) of Australia, 2005, pp. 1–10.

[36] I. S. MacKenzie and S. X. Zhang, "The design and evaluation of a highperformance soft keyboard," in CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems. New York, NY, USA: ACM, 1999, pp. 25–31.

[37] 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. Morristown, NJ, USA: Association for Computational Linguistics, 2002, pp. 1–7.

[38] S. K. Card, T. P. Moran, and A. Newell, "The keystroke-level model for user performance time with interactive systems." Communications of the ACM, pp. 396–410, 1980.

[39] S. K. Card, T. P. Moran, and A. Newell, The psychology of human-computer interaction., 1983.

[40] A. T. Welford, "Choice reaction times: Basic concepts," Reaction Times. London Academic Press, pp. 73–128, 1980.

[41] W. Hick, "On the rate of gain of information," Journal of Experimental Psychology, vol. 4, pp. 11–36, 1952.

[42] J. Carroll, HCI models theories and frameworks: towards a multidisciplinary science. Kaufmann Publishers, 2003.

[43] R. Simpson and H. Horstmann, "Adaptive one-switch row-column scanning," IEEE Transactions on rehabilitacion engineering vol7. N 4, no. 4, pp. 464–473, 1999.

[44] R. Simpson, H. Horstmann, and E. F. Lopresti, "Selecting an appropriate scan rate: the .65rule," Proceedings of RESNA 2006 Annual Conference, Atlanta, GA. Arlington, VA: RESNA Press, 2006.

[45] G. W. Lesher, D. J. Higginbotham, and B. J. Moulton, "Techniques for automatically updating scanning delays," In Proceedings of the RESNA 2000 Annual Conference, pp. 85–87, 2000.

[46] H. S. Venkatagiri, "Efficient keyboard layouts for sequential access in augmentative and alternative communication," Augmentative and Alternative Communication, 15(2), pp. 126–134, June 1998.

[47] R. Almela, P. Cantos, A. Sánchez, R. Sarmiento, and M. Almela, Diccionario y estudios léxicos y morfológicos. Universitas SA, 2005. [Online]. Available: http://www.um.es/lacell/proyectos/dfe/