Virtual Keyboard Design: State of the Arts and Research Issues

Sayan Sarcar[1], Soumalya Ghosh[2], Pradipta Kumar Saha[3] Debasis Samanta[4]
IIT Kharagpur, Kharagpur - 721302, West Bengal
Email: [1]sayans@sit.iitkgp.ernet.in, [2]sghosh@sit.iitkgp.ernet.in, [3]psaha@sit.iitkgp.ernet.in, [4]dsamanta@sit.iitkgp.ernet.in

***Abstract*—Recent advancement in Information and Communication Technologies (ICT) open up a scope for computing in Indian languages. Off late, computing devices are evolved in different shapes like cell phone, PDA, iPod etc. In these devices, it is not possible to afford hardware keyboard because of the limitations in size and weight of devices. To alleviate this problem, application developers propose virtual keyboard. The virtual is an on screen graphics keyboard and is flexible to compare with its hardware keyboard counterpart. There are many reported strategies towards the development of virtual keyboards in English language, but those well known strategies are not properly applicable for virtual keyboard design in Indian languages. In this paper, we survey the existing strategies and critically examine their applicability to design virtual keyboard in Indian languages. We then propose an approach to design any Indian language compatible virtual keyboard.**

***Index Terms*—Virtual keyboard, text entry interface, keyboard layout optimization, automatic virtual keyboard design, design space exploration**

I. Introduction

Enormous advancement in communication and information technology instantiates the flourishment of the mobile and handheld devices in urban and rural areas in India. These devices become popular to the common people in India. In these devices, text entry task is not possible through conventional hardware keyboard due to mobility restriction, size factor etc. This motivates researchers to propose a virtual keyboard model which imitates the hardware keyboard ambiance for an effective text entry system. Further, the utilization of virtual keyboard appears in space saving situations and also can be extended into requirement in virtual programmability of keys or systems avoiding mechanical failure or in movement situations where usability of standard keyboard is limited. In these days, virtual keyboards find their position in transport environments e.g. rail, plane or automotive. Virtual keyboards are also designed for public kiosks. This reveals the fact that virtual keyboard has become a very essential needs in every area like performing text entry, simulating hardware keyboard, and adequately in accessing Internet to perform different kind of tasks like E-mailing, chatting, blogging etc. So far the design of virtual keyboard is concerned, the English language virtual keyboards are based on some common design strategies like key arrangement based on character frequency, maintaining distance between character pair etc. One of the basic key arrangement is in alphabetic fashion. This arrangement can be maintained in horizontally or vertically or by some other order. The method behind designing another keyboard layout called Fitaly [1] uses the character frequency of English alphabet as their factor in deciding the position of different character keys in the virtual keyboard. In Fitaly, the frequent keys are placed in center. Space character which is most frequent has two buttons and can be selected by tapping in any of the button. MacKenzie and Zhang [2] uses to produce an optimized keyboard arrangement named Opti [2]. This layout requires four keys for space and places the other keys based on other procedure. Researches also use physical laws [3] to develop the virtual keyboards. One of the approaches used mechanical simulation of a mess of springs of Hooke's law [4] and tries to minimize the distance between character pairs. The method uses digraph probabilities [5] of English alphabets. They called this keyboard as Hooke's keyboard [6]. In another approach[7], developers try to minimize the distance between strongly associated pairs and call the keyboard arrangement as Lewis keyboard.

The virtual keyboard design approaches in English language are broadly confined within basic methodologies of language processing tasks like character frequency calculation (unigram and bigram) [8] from a corpus and also defining an association rule of characters in forming meaningful words in the English language. On the other hand, the structure of Indian languages are completely different from English having indiscriminate usage of inflexed characters (Matra) [9], it is also complex to compose texts. Also, the increased number of characters than English causes the formation of large number of keyboards consisting different combination of characters in the design space. The major difference between these two different groups of language which should be critically analyzed in designing virtual keyboard in Indian language and summarize as follows.

- *Design space exploration*: In almost all Indian languages, the number of characters are much greater than English. Hence, the number of possible layouts for a keyboard in design exploration space are much higher than that of in English language.
- *Number of character set*: There are twenty six characters in English language consisting vowels and consonants. But, in case of Indian language, the number of characters are much higher (like for Hindi and Bengali language, the characters are around 50).

 978-1-4244-5974-2/10/$26.00 © 2010 IEEE

- *Presence of Matra*: Unlike English language, many of the Indian languages have inflexed forms of characters which are commonly used in the corresponding languages.
- *Existence of complex characters*: Many Indian language users are keen about using complex characters which are unlikely in English.

As number of vowels and consonants in Indian languages are more than English, the key arrangement and key placing subtasks become a difficult task. There is no such standardized rule present in the current context for placing next character. The optimal arrangement, which is a challenging issue, would be decided by analyzing the digraph probability of vowel and consonant characters and different principles defined in popular keyboards in English. The evaluation metrics of virtual keyboard usually depend on the interface layout. There can be K! possible layouts for a keyboard with K keys. If only vowels and consonants are taken for Indian languages, there are 50! (approx) ways of designing an Indian language virtual keyboard. Therefore, after analyzing the basic characteristics of designing popular English virtual keyboards (like Qwerty [10], Opti, Fitaly, Dvorak [11] etc.), it has been observed that method for calculating digraph probabilities of every pair of characters or word from standard corpus in English are widely accepted in forming basic design of major popular virtual keyboards. However, in Indian languages, there lies a problem in above mentioned methods for increasing number of characters and inflexions (like matras) in these languages. To solve the problem, designers have tried to reuse the language resources like language corpus and digraph probability chart towards defining a method which helps the users by partially exempting them from a very hectic task, tapping the keys and also not committing much errors.

In addition to this, another hindrance is the absence of standardization of Indian language compatible keyboard layout. A variety of keyboard layouts are being used in Indian languages. In fact, present keyboard layouts are designed on ad-hoc basis. Words in most of the Indian languages are full of inflexions (Matra) and complex characters. A standardized algorithm needs to be defined to help the user in typing those critical characters, layout generation and key management issues. The layouts are irregular in terms of statistical distribution of the keys. As a result, the keyboard layout puts excessive and disproportionate stress on the fingers of user which on long term can cause adverse effects. Though widely used forms of keyboard layout exist, detailed analysis of optimality of those layouts has not been done. The drawbacks of existing Indian language based keyboard layouts lead to the requirement of the detailed scientific and statistical study (n-gram language modeling citengram) of the language corpus towards designing the new standard and widely acceptable layout which is simple in nature (with respect to text entering through keyboard task). Furthermore, the design should be consistent with the capability of human fingers and having fluency with the Indian language structure. In this paper, we propose a generic design strategy applicable for any Indian language based virtual keyboard which will also address majority of the linguistic and layout exploration problem in a easy and convincing way.

The rest of the paper is organized as follows. In Section 2, we discuss the existing keyboard design approaches in English. The comparison of keyboard performance is presented in Section 3 and proposed solution addressing the issues of virtual keyboard with existing approaches is discussed in Section 4. Finally, Section 5 concludes the paper.

II. Existing keyboard design approaches

A number of Virtual keyboards have been proposed in English language. In this section, the design principles of these keyboards are discussed followed by their performance evaluations.

A. Dvorak layout

Dvorak (1943) [11] propose an alternative arrangement for the keys (Figure 1(a)) which rejects the design of the Qwerty layout through an analysis of the relative frequency of used letters. The Dvorak layout is designed for the following tasks:

- To allocate more work to the right hand than the left.
- To distribute letters based upon the strength of each finger.
- To place the most frequently used letters on the home row.
- To place vowels and frequently used consonants so that they can be typed with alternate hands.

Since the Dvorak layout is well known as a Qwerty alternative, it is a logical starting point. In its physical form, the Dvorak keyboard is similar to a Qwerty keyboard. A Qwerty layout can be transformed into a Dvorak layout by renaming the keys. The Dvorak keyboard is designed to optimize two-handed touch typing. The idea is that higher entry rates can be obtained if common digraphs are entered by fingers on opposing hands instead of on the same hand As well, the most common letters (e.g., E, T, A, H) are positioned along the home (viz. middle) row.

In general, the novice prediction is dominated by the visual scan time [12], [13], so any layout permutation that minimizes movement has only a minor impact on the novice entry rates. Novice predictions will always be lower than 12.6 wpm (words per minute) by an amount determined by the movement component of the task.

B. Fitaly layout

The Fitaly *One-Finger* keyboard [1] is designed to minimize hand movement during text entry with one finger, a stylus or a pen (Fig. 1(b)). This keyboard is a commercial product designed to optimize text entry with a stylus. The most important feature of the layout is the presence of two space bars. The proximity of the most common letters in English (e.g., E, T, A, H) to the space bars is also immediately apparent. The keyboard's name is taken from the letter sequence along the second row of keys.

The Fitaly keyboard is designed to minimize the travel

from one letter to the next. According to the developer, the average travel is 1.8 keys compared to 3.2 keys for a Qwerty [10] layout. These figures are obtained using a corpus of digraph probabilities similar to that described by Soukoreff and MacKenzie (1995)[14].

The layout of Fitaly keyboard has been modified to single handed (Fig. 1(c)) (Right handed, Left handed or Mouse driven) and as well as double handed (in contrast with Qwerty layout) to increase the typing speed of using the keyboard. The following rules are being introduced.

1) Measuring frequency of letter-to-letter transitions:: We have measured the frequency of letter-to-letter transitions for a representative corpus of the English language with several millions of character . For example, this produces the number of times the letter o is followed by the letter a.

2) Rearranging the keyboard layout:: They have rearranged the keyboard layout to obtain minimum average traveling time. The effect of this optimization is that the most frequent key transitions are between adjacent keys.

C. Opti layout

It is one of the optimized virtual keyboard layouts for the English language. Figure 1(d) shows the Opti layout as described by MacKenzie and Zhang mackenzie [2]. The keyboard layout is optimized to increase the typing speed using trial and error method, Fitts' law [15], and character and digram frequencies in English. Fitts' law [15] gives a function for computing the key tapping time given the length of the movement and width of the target are needed. These enable a researcher to compute a prediction for the upper bound of user performance given the keyboard layout. Trial and error is needed to generate the keyboard layouts.

According to the calculations of MacKenzie and Zhang is one finger keyboard layout. The Opti layout [2] is theoretically 35% faster than Qwerty and 5% faster than Fitaly layout. In a longitudinal study described by MacKenzie and Zhang the speed difference between Opti and Qwerty seems to exist in the real world too. Their aim is to achieve both speed and accuracy. Emphasis on speed may have contributed to the error rate which is over four percent for both keyboard layouts. The error rate with Opti is consistently slightly lower than the Qwerty [10].

D. Cirrin layout

Figure 1(e) shows the input area of the Cirrin [16] text input method with a stylus. One puts the stylus down inside the ring and then moves it over the areas labeled with the characters. Input is generated from the coordinates of the points where the pen crosses the circumference of the inner circle. Only lowercase alphabets are shown along the circumference in figure 1(e). Putting all characters of the ASCII character set or the thousands of Unicode [17] is clearly not an option. The slices would become too narrow to hit with the pen. The authors suggest to use the non-dominant hand to input the characters which are not found in the Cirrin input area. The device is operated with the non-dominant hand may be chosen freely. Mankoff and Abowd [16] have used a regular Qwerty-keyboard. They also discuss using a regular handwriting recognizer for the task. In a mobile setting a Qwerty-keyboard is not an option. A handwriting recognizer might be workable, but that raises the question of whether Cirrin offers enough benefits to be worth the screen real estate if we will have a handwriting recognizer anyway.

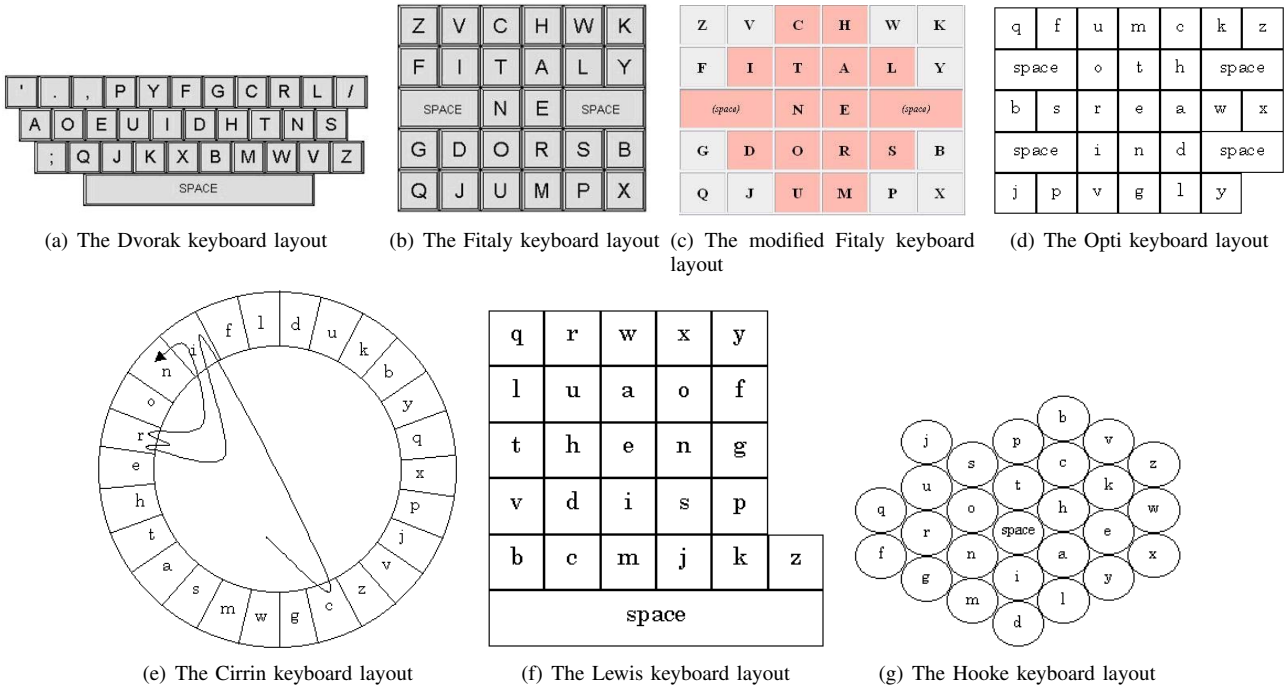
According to test reports on Cirrin, it seems that the accuracy could be good enough for text input. The speed may prove to be a bottleneck since visual feedback is needed in order to hit the relatively small areas with the pen. Cirrin is very dependent on the pen-interface and offers no eyes-free operation.

E. Lewis Keyboard

MacKenzie, Zhang, and Soukoreff [18] designed a virtual keyboard where the letters are laid alphabetically in two columns, which did not show a performance advantage, probably due its elongated shape. Lewis et al [7] proposed a 5 by 6 virtual keyboard (Fig. 1(f))with a strictly alphabetical sequence, which is suffered from the same problem as discovered by Norman and Fisher[19] - the alphabetical discontinuity caused by row breaks.

F. Hooke's layout

As discussed earlier, the goal of a good keyboard design is to minimize the statistical travel distance between characters. The more frequent digraphs should be closer together than less frequent digraphs. In order to achieve this goal, a dynamic system technique has come up. As example, a spring connecting every pair of the 27 keys whose initial positions were randomly placed with spaces between the keys. The elasticity of the springs, when turned on, was proportional to the transitional probability between the two keys so that keys with higher transitional probability would be pulled together with greater force. In addition, there is viscous friction between the circle shaped keys and between the key surface and the table surfaces. The steady state when all keys are pulled together forms a candidate virtual keyboard design. In the simulation, the springs are "virtual". They did not stop other objects passing through them, hence preventing the springs from being tangled. The final positions of the keys might still not be at the minimum tension state, because some keys could block others from entering a lower energy state. Two methods were used to reduce the deadlock or local minimum states. The length of this segment was manually adjusted in the dynamic simulation process. At the end of each simulation cycle, the length of the adjustable segments have been reduced to zero so all the keys were pulled against each other, forming a layout of a virtual keyboard. The performance of the design is then calculated and compared with known results. When not satisfactory, the layout could be *"stretched"* out to serve as another initial state for the next iteration of the same process. The iteration is repeated until a satisfactory layout is formed. Figure 1(g) shows the best layout achieved with this approach. To capture the gist of the spring simulation



(a) The Dvorak keyboard layout (b) The Fitaly keyboard layout (c) The modified Fitaly keyboard layout (d) The Opti keyboard layout

(e) The Cirrin keyboard layout (f) The Lewis keyboard layout (g) The Hooke keyboard layout

Fig. 1. Different popular virtual keyboard layouts in English

technique, it is called as best design achieved through this method Hooke's keyboard (after Hooke's Law)[6].

III. Performance of the existing keyboard

A. Evaluation Technique

1) ***Experimental procedure****:* Performance models are essential to evaluate designs. These models are computational models of users, which computes user performance for a given design. User performance is commonly measured in terms of text entry rate for text entry systems. The design approach works as follows. Performance models compute text entry rate for each design in the given a set of design alternatives. The designs are then compared with the computed text entry rates, in order to determine the best among the set.

Soukoreff and MacKenzie [14], [20] propose a model, often called the FD model to evaluate virtual keyboards. This model uses on small sized mobile devices like PDAs. The model does not consider any physical disabilities on the part of a user and it is assumed that the user selects keys from the interface with finger (touch) or stylus movement. Following components are considered to develop the model.

a) Visual search time: The user first needs to locate the corresponding key on the interface to select a character from a keyboard interface. The time to locate a key is called the visual search time and is denoted by RT. In the FD model, RT is calculated using the Hick-Hyman law [12], [13] depicted in Eqn. (1).

$$RT = A^{'} + B^{'} \log_2 N \tag{1}$$

where $A^{'}$ and $B^{'}$ are constants and N is the total number of keys present on the interface.

b) Movement time: The user needs to move mouse from the current location to select the key to locating the character. The time to make a manual movement from one key to another (movement time) is calculated using the Fitts's law [15] The formula is stated in Eqn. (2).

$$MT = a + b \log_2 (D_{ij}/W_j + 1) \tag{2}$$

where MT is the movement time from the source to target key, a and b are constants, D_{ij} is the Cartesian distance between the two keys and W_j is the width of the target key.

c) Digraph probability: In the FD model [14], [2], the probability of occurrence of character pairs or digraphs is considered. The digraph probability [5] is calculated from a text corpus by Eqn. (3)

$$P_{ij} = f_{ij} / \sum_{i=1}^{N} \sum_{j=1}^{N} f_{ij} \tag{3}$$

where f_{ij} is the digraph frequency and P_{ij} is the digraph probability of characters k_i and k_j . N is the total number of individual characters present on the interface.

d) Words per Minute (WPM): Words per minute (WPM) is the most widely reported empirical measure of text entry performance. Since 1905, it has been common practice to consider a "word" as 5 characters, including spaces, in English and 5.1(approx) in Bengali [21]. Importantly, the WPM measure does not consider the number of keystrokes or gestures made during entry, but only the length of the resulting transcribed string and how long it takes to produce it. Thus, the formula for computing WPM is illustrated in Eqn. (4). Equation (6) illustrates the formula for calculating average performance.

$$Averagemovementtime(MT_{Mean}) = f_{ij} / \sum_{i=1}^{N}\sum_{j=1}^{N} MT_{ij}P_{ij} \quad (4)$$

$$\begin{aligned} Performance of Novice(CPS) &= 1/(RT + MT_{Mean}) \\ Performance of Expert(CPS) &= 1/MT_{Mean} \end{aligned} \quad (5)$$

$$AverageTextEntryPerformance(WPM) = CPS * (60/W_{Avg}) \quad (6)$$

*2) **Text selection**:* For conducting experiments in any language by using user evaluation or simulation technique, one must select some text which can be given to the computing system. We select the English and Bengali text for evaluation. According to census report, Bengali is spoken by more than 210 million people as a first or second language. Among them 100 million Bengali speakers are in Bangladesh and about 85 million are in India, primarily in the states of West Bengal, Assam, and Tripura. The language is also being spoken among immigrant communities in the United Kingdom, the United States, and the Middle East. It is the national language of Bangladesh and one of the languages officially recognized in the constitution of India [22].

Selecting a text for evaluating user performance of text entry as well as performing automatic simulation of computing devices are also a very tough assignments in Bengali language which is enriched with thesaurus of huge word variation of inflexed and complex characters majorly. There are many reading text available in Bengali which can be either restrained with large number of words having complex and inflexed characters vastly (like novels like *"KAPALKUNDALA"* written by *Bankim Chandra Chattopadhyay* etc.) or inscribed with very simplest form of words (consists of simple characters with less number of inflexions) like short stories of Satyajit Ray. Apart from the other criteria, we also take care the occurrence of almost all characters in corresponding language text. To maintain the proper balance between the criteria mentioned above in finding average text entry time for the user, we have chosen portion of the text from two above mentioned titles. Then we assemble these texts of three such paragraphs as set TB_1, TB_2 and TB_3 into a file which is identified as a text corpus for typing.

Similarly, the English text has been selected from popular conversation or dialog, narrative stories etc. and also distributed into three subsets same as Bengali. In the text selection phase, we have tried to choose the text consisting of different words which cover many of the possible meaningful occurrences of almost all characters in corresponding language. The English typing text also consists of three paragraphs namely TE_1, TE_2 and TE_3 which are among the above mentioned English text categories. The major classifications of selecting texts for user evaluation in both English and Indian language are clearly demonstrated in Table I.

We provide a detailed statistics related to user evaluating corpus in Table III (character frequency) and Table II (Text length, number of complex characters, number of inflexions) in favor of argument for selecting such kind of texts in Bengali because we know that variation can affect the performance of users, particularly novice one.

TABLE I
TEXT SELECTION

Text under Test	Number of Characters	Reference Text
TE_1	940	"The Benaras" by Aldous Huxley
TE_2	1034	"My Week with Gandhi" by Louis Fischer
TE_3	1069	"The Gift of the Magi" by O.Henry
TB_1	761	"Kapalkundala" by Bankim Chandra
TB_2	791	"Ramayana" in Bengali
TB_3	801	"Ashamanjababur Kukur" by Satyajit Ray

TABLE II
CORPUS STATISTICS I

Text under Test	Text Length	Number of Complex Characters	Number of Inflexions
TB_1	761	23	304
TB_2	791	53	379
TB_3	801	25	327

TABLE III
CORPUS STATISTICS II

character	frequency	character	frequency	Character	Frequency
অ	23	ণ	7	য়	25
আ	29	ত	50	ৎ	1
ই	33	থ	14	ং	4
উ	8	দ	44	ঁ	13
এ	20	ধ	21	া	119
ও	9	ন	64	ি	81
ক	63	প	39	ী	27
খ	16	ফ	4	ু	38
গ	31	ব	65	ূ	4
ঘ	9	ভ	14	ৃ	2
ঙ	8	ম	43	ে	73
চ	26	য	15	ৈ	2
ছ	23	র	67	ো	27
ঝ	34	ল	40	ৌ	1
ঝ	8	হ	32	্	92
ঞ	2	শ	15		
ট	31	ষ	60		
ঠ	5	স	41		
ড	5	ড়	16		

*3) **Selection of Users**:* In every user-centric design, the users play a significant role in evaluating the product which follows the design. Here our target users are mostly English illiterate people who had already crossed the basic barrier of literacy but can not properly read and write in English

language. For user selection task, we have visited nearby places like local markets, schools, domestic help centers etc. Experiment has been done with three types of users (according to level of English literacy) of different occupation and 22 people in average belonging to each user type. The interface has been tested by educated persons like office staff, college students, business person in both urban and rural areas. User category U_1 consists of people who are well familiar with English language such as Office executives and Businessmen of Non-Bengali community. People belong to U_2 user category have knowledge in both the language, English and Bengali. College Teacher and Office staff whose mother language is Bengali are the member of this group. U_3 user class includes people who have meager knowledge in Bengali language in its writing form. The shopkeeper, domestic help person, rickshaw pullers are members of this user categories. The user category and their basic language skills are described in Table IV.

TABLE IV
USER SELECTION

User Category	Language Efficiency	Number of Subjects
U_1	English	13
U_2	English + Bengali	28
U_3	Bengali	25

B. Experimental Design

1) ***Text entry rate calculation****:* The experiments have been carried out through simulation and user evaluation techniques. Initially the keyboards following the design principles of popular English keyboards like Dvorak, Fitaly, Opti, Lewis, Hook's and Cirrin (Google's Bengali On-Screen keyboard developed in same logic) are being designed. In the process of designing the keyboards, some rules have been set which are given below.

a) Grouping rule: Grouping concept has been introduced by making vowels and consonants of Bengali in one group and inflexions in separate group.

b) Bengali word frequency rule: Usually major keyboard layouts in English use the digraph probability [5] calculated from English corpus. Similarly, the frequency of occurrence of one Indian language character (Here Bengali character including space) after another has been found out from Bengali corpus. As Greedy method is followed in placing the characters in the final layout, some stringent rules are defined.

I. We sort the digraph probability values in descending order for every possible two character combination. Then find out the probability values of each pair and also their counterpart (i.e. in Bengali, ক after র and also র after ক). If the difference is moderate (frequency difference greater than 100), then that combination need not to be reflected in the keyboard layout, other combination with the individual keys would be considered.

II. The highest valued combination keys are placed in contiguous location within Virtual keyboard. Middle row of a keyboard mostly consists of most occurring characters in language (Unigram frequency of individual characters has been calculated and based on that, decision is taken). The upper row consists of characters having less unigram frequency than middle row characters and the lower block is the next block to be filled. Using the steps, situation has occurred where all the keys have been placed and finally, iteration is stopped.

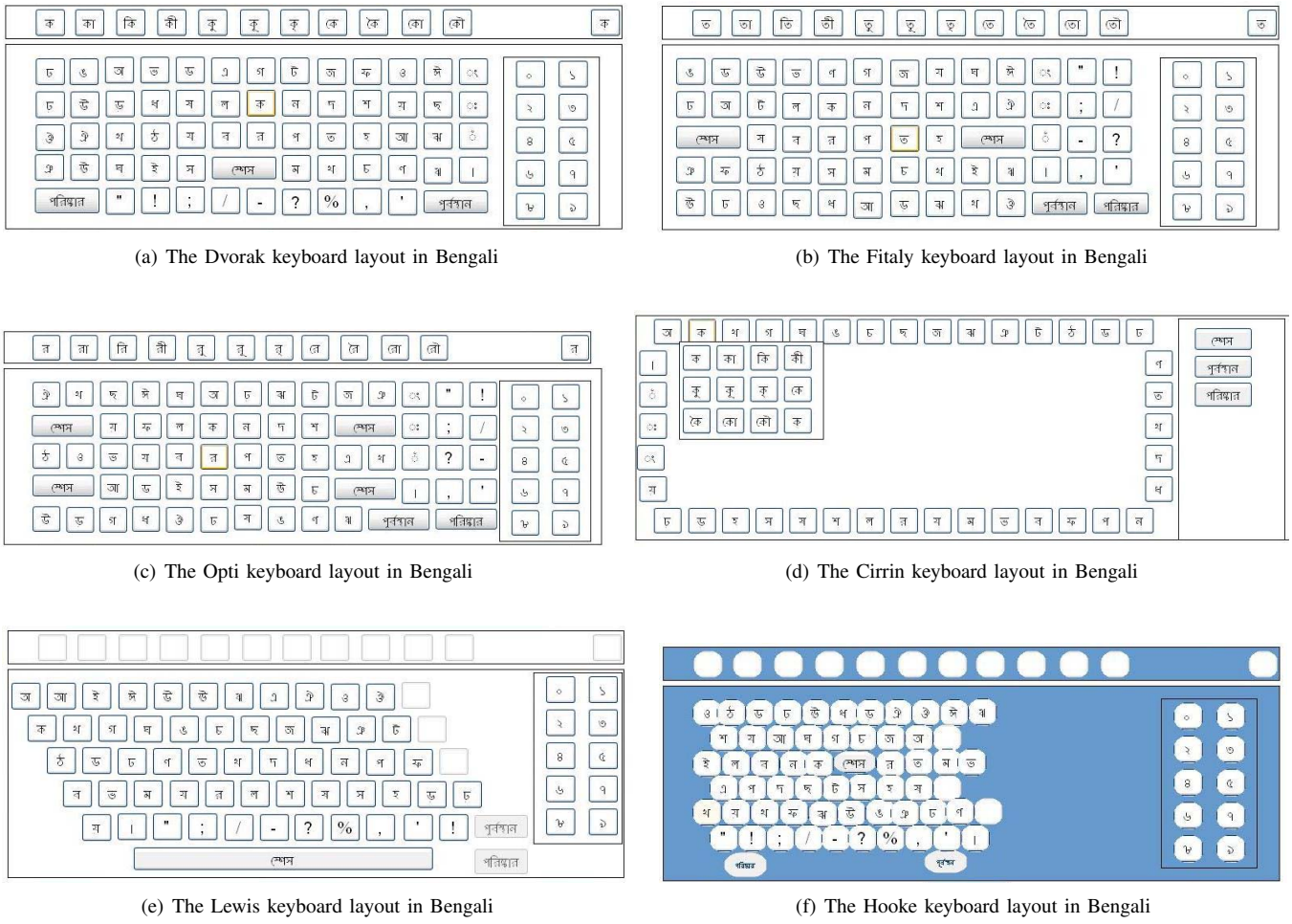
c) Management of Complex character: Most of the Indian languages are full of complex characters [23]. So one need to take number of characters entered in forming complex characters as one at the time of measuring text entry rate. In case of user evaluation, the number of errors in typing measure have been increased by one when button like backspace or clear (for Bengali, পুর্বস্থান or পরিষ্কার) is pressed.
A one and half page corpus (both in English and Bengali language), consisting of three sets, has been prepared and total number of characters (including space character) in each set has also been calculated (assuming complex form of a character as a single one).

In the user evaluation based approach, users of three above mentioned categories are requested to type the English or Bengali in virtual keyboard of corresponding languages. The average text entry rate (Eqn. 6) has been empirically calculated from the total time of typing the whole text by the program in background and the results are also being documented in a log file.

On the other hand, programs are being executed to simulate the motor process through typing behavior which consists of performing notion actions in the corresponding languages (English and Bengali) virtual keyboard. The experiment has been performed for every keyboard layout and results are stored into appropriate file.

Some keyboard layouts are implemented based on methodologies of English popular layouts are shown in Fig. 2.

Observation has been drawn from conducted experiments of two categories namely user evaluation and simulation. Interestingly, the simulation based experiment produces improved text entry rate (Words per minute parameter) over other. The reason behind the happening is exclusion of visual search time parameter in case of simulation based computation. It has been concluded that visual search time depends on human cognition and perception at the time of user testing,. In case of novice user, the average visual search time of finding a character on the keyboard from any other location on it fluctuates between 1.5 and 5.0 seconds (It happens at the starting of a session where user completely new to the keyboard pattern and unaware about the mechanism of forming word. Also during non-motivated phase, user can not find exact normal or inflexed character for sufficiently large time which has to be typed next). Whereas for the same situation, the visual search time of expert user also variates from 1.0 to 2.0 seconds. In English language, the Hick-Hyman Law [12][13] states that the search time would become zero for expert user. It is not appropriate in case of Indian language based experts as experimental results prove that fact. So, the average search time between two keypress becomes 3.0 seconds for all the users (as we have not differentiated the two groups of users,



(a) The Dvorak keyboard layout in Bengali

(b) The Fitaly keyboard layout in Bengali

(c) The Opti keyboard layout in Bengali

(d) The Cirrin keyboard layout in Bengali

(e) The Lewis keyboard layout in Bengali

(f) The Hooke keyboard layout in Bengali

Fig. 2. Different virtual keyboard layouts in Bengali

TABLE V
TEXT ENTRY PERFORMANCE WITH USER EVALUATION

Text Type	Text Length	Text Entry Rate (Word per Minute)						WPM (AVG)
		DVORAK	FITALY	OPTI	LEWIS	HOOKE	CIRRIN	
English	TE_1(940)	7.653	7.532	7.624	7.241	7.414	7.524	7.975
	TE_2 (1034)	8.257	8.008	8.091	7.954	8.013	8.042	
	TE_3(1069)	8.479	8.268	8.552	8.138	8.332	8.443	
Bengali	TB_1(761)	3.472	3.316	3.594	4.142	3.194	3.332	3.914
	TB_2 (791)	3.705	3.682	3.841	3.877	3.483	3.585	
	TB_3 (801)	4.199	4.505	5.292	4.651	4.216	4.392	

TABLE VI
TEXT ENTRY PERFORMANCE WITH SIMULATION

Text Type	Text Length	Text Entry Rate (Word per Minute)						WPM (AVG)
		DVORAK	FITALY	OPTI	LEWIS	HOOKE	CIRRIN	
English	TE_1(940)	11.6549	11.4178	12.016	9.2175	10.3938	11.2547	11.109217
	TE_2 (1034)	11.7813	11.5927	12.163	9.3309	10.5146	11.2938	
	TE_3(1069)	11.8951	11.6573	12.252	9.4152	10.6046	11.5109	
Bengali	TB_1(761)	6.7396	6.5179	7.2809	5.9128	6.2144	6.7528	6.7527556
	TB_2 (791)	6.7958	6.5729	7.3498	6.2236	6.2569	6.8937	
	TB_3 (801)	6.8328	6.8901	7.8573	6.2754	6.8687	7.3142	

namely novice and expert). The user evaluation and simulation results are shown in Table V and Table VI, respectively.

C. Lessons Learned

According to the above discussion, the fact is clearly revealed that the popular text entry mechanisms in English

language perform poor in case of Indian languages where complex and inflexed characters (Matra) are present. The issues are discussed in following.

- Text entry performance
- Spacing requirement
- Error proneness
- Positioning of inflexed window

1) Text entry performance: Though the average text entry performance for the English keyboard is lying within satisfiable range, the scenario drastically deteriorates in case of Indian languages. The text entry mechanism produces poor results for Hindi, Bengali, Oriya etc. and other languages.

2) Space requirement: In any Indian language, the number of character set is much larger than English. the major implementation area of virtual keyboards are different handheld devices like mobile phones, PDA, iPod etc. where limited space is available for placing virtual keyboards. The space optimization is therefore a challenging matter for placing unambiguous virtual keyboards. Also, one has to look after the issues like the positioning of inflexion window, grouping of character related with same context etc. from space requirement point of view.

3) Error proneness: Users frequently have committed errors during typing with the developed keyboards. Even, the error rate has not been decreased significantly after several sessions by expert users. So, the probability of error proneness becomes relatively higher for the keyboards.

4) Inflexion window: The English language based virtual keyboards consist of different combination and placement of 26 letters (5 vowels and 21 consonants) within whole layout. English language is also a very simple and easy word constructed language having no complex symbols and signs. The context completely differs in different Indian languages like Hindi, Bengali etc. These language are full of complex characters made by combination of single letters and also inflexions (as কা, কি, কু in Bengali language). The number of unique characters (including Matras) is different in several languages exist in Indian context (In Bengali, it is more or less 61).

The common structure for English language is not directly applicable toward constructing virtual keyboard in most of the popular Indian regional languages. After rigorous user study of several single pointer based virtual keyboards, language modeling found to be effective for placement of characters one after another to form optimal keyboard layout. Majorly the bigram model maintained in many effective virtual keyboards in English language would help the users to enter English text in nominal time (more WPM) with comparatively less visual search time (Eqn. 1). In India, the occurrence of inflexions is very frequent in language corpus. Many common words in different domains are mostly full with inflexed and complex characters. In the context, unigram and bigram modeling alone are unable to produce good results. Apart from language related issues, the other concerns are to minimize visual search time (Eqn. 1) and mouse pointer movement time (Eqn. 2) at the time of forming texts. In language related contexts, a few methods for constructing virtual keyboards had been already proposed. But our approach also addresses the problem of minimizing other human psychology related issues like visual search time (Eqn. 1), mouse movement time (Eqn. 2) etc. which are proved to be important stakeholders in virtual keyboard design research. At the time of implementation, one has to take care the following issues.

a) Clustering of different character type: After analyzing the Bengali language corpus, the conclusion has been drawn that the vowel characters (অ , আ , ই ,এ) do not have inflexed forms and also some characters (ঈ , ঊ , ঐ , ঔ) have poor result in frequency of occurrence. If the clustering has been done with respect to character type (vowel and consonants), then some vowel characters which have good frequency results would loose significance and placed in a cluster which has much greater movement time (Eqn. 2). If the clustering would have been done maintaining the frequent and infrequent character list, some vowel characters which are as much frequent as consonant characters may void of inflexed window because vowel characters do not have inflexions. So, formation of two distinct character clusters by providing rules for maintaining inflexed window etc. is a very challenging issue towards offering better user-centered computing environment.

b) Positioning of Normal characters: The characters may be placed in alphabetical or user-centered order through n-gram language modeling (upto bigram) which can help the user by consuming less time in constructing a word. The developer must look after both on the word constructing time of user and optimal visual search time (Eqn. 1).

c) Positioning of Inflexed characters: Many of the Indian popular regional languages are biased with inflexions as their language corpuses contain majorly inflexed and complex characters. So, to construct keywords, user has to type several inflexions and as a result, the procedure takes more time. The visual search time (Eqn. 1) also has to be added if the individual inflexion exists separately in the layout. So, the developers who have the plan to develop efficient virtual keyboard, must find out a time effective solution regarding the proper placement of the keys and also inflexed character set.

With proper addressing of the above mentioned issues, a different robust keyboard layout has been proposed for Bengali language which may applicable for any Indian languages which are prone to complex or inflexed characters(Matra) or other different symbols. The user testing results also proved more user satisfaction than the previously mentioned keyboards.

IV. Proposed solution addressing the issues

The above mentioned issues have been addressed by proposed generic menu-augmented layout suitable for Indian languages enriched with inflexion and complex characters. As a case study, we take Wikipedia [24] bengali corpus and analyze both unigram and bigram frequencies of single non-inflexed characters. It has been observed previously that all designed virtual keyboards may contain a separate panel of

inflexed characters which is apart from the character panel. Hence, a user has to move to and fro from one panel to other to construct most of the words in any Indian languages. As a result, the movement time (Eqn. 2) got increased in all the cases. Also, user has to search each character of a word and construct the inflected form of that specified character (if any). This phenomenon is bound to increase the visual search time (Eqn. 1) of some users.

The layout is being divided into two major zones to support the normal human cognition of common people for concentrating towards accessing virtual keyboard. This major zones are with most frequent characters and less frequent characters. The first zone resides at the important place in terms of eye contact of most of the human being, the center of the keyboard and the other zone surrounds the previous one. The first zone consists of frequent characters (for example, in Bengali language, space bar, র, ন, ত, ক, ল etc). The surrounding area of zone one contains the next most probable characters (after both unigram and bigram analysis assuming previous character occurred). The less probable characters are then placed at the outer side of the keyboard layout. The distance between the characters and their inflexed forms has been reduced as the inflexion set of a particular character dynamically generated on mouse hovering event and displayed on the root character.

Further, modification of the shape of inflexion window is proposed in this layout to minimize the key distance further. The keyboard design also introduces background color concept on inflexion window to help the user by clearly distinguishing the root characters with inflexed ones. The window size has modified into circular by keeping in mind the fact that the distance of each key from the center in the inflexed window is same. The layout produces astonished results which may lead to form of standardized layout (for Bengali language).

The empirically calculated results after testing in Bengali

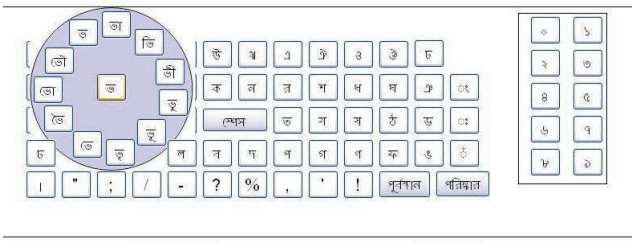


Fig. 3. The proposed layout in Bengali language

language with two types of users (Novice and Expert) are given in Table VII. The simulation provides improvised simulation results which can be shown in Table VIII.

A number of issues can further be addressed for proposed layout in connection with minimizing the error rate like minimizing Levenshtein distance [25] (Minimum String Distance) in word level evaluation and "Key Stroke Per Character" (KSPC) [26] in character level evaluation. Prediction list can also be provided for preventing the user from not committing more user error and decreasing number

TABLE VII
EMPIRICAL RESULTS

Text under test	Number of characters	Novice User			Expert User		
		Time(s)	Average WPM	Number of Errors	Time(s)	Average WPM	Number of Errors
TB_1	761	4471	2.03981	8	2301	3.97818	4
TB_2	791	4764	1.98992	13	2378	3.98654	7
TB_3	801	4848	1.98019	17	2402	3.99667	7

TABLE VIII
RESULTS OF SIMULATION

Text under test	Number of Characters	Average WPM
TB_1	761	7.5514
TB_2	791	7.7921
TB_3	801	7.8723

of key tapping at the time of word formation.

Rigorous testing has been carried out with users mainly who are able to cope with English and Bengali both languages in terms of speaking and writing and also have strong knowledge of accessing computing devices like mobile, PDA, computers etc. Many different texts in Bengali language have been given to them at three stages of time. They have typed each of them and results have been accumulated into corresponding log files. Average value of evaluation metric(Eqn. 6) has been taken for each of the text.

We have conducted experiments with users with different cognition level like with more or less visual search time (Eqn. 1) for finding the keys, having problem of writing correct spelling of words, committing wrong typing of characters due to lack of concentration at the time of performing typing tasks etc. We collect results for previously described keyboards and also proposed keyboard. We have scrutinized the results and documented the resolution evolving from them. We have plotted a part of the results of the average text entry performance with the English like keyboards as well as with proposed keyboard into a graph (Fig. 4) and observed its nature carefully. The curves reflect more stable and better result for our proposed keyboard after lot of testing sessions. The result signifies that the users feel more comfortable with the keyboard and in consequence, the curve would make very small diversion after large number of sessions. Though initially the average result performs better, but after 30 sessions the proposed keyboard would produce better results and also become much more adjusted with the users' psychology. Beside improved performance, this keyboard layout would also become more acceptable among our targeted users for its quick adaptiveness with their cognition level. Also, several genetic algorithm based approaches [27] can be implemented to find out optimized keyboard layout.

V. CONCLUSION AND FUTURE WORK

We have reviewed a number of mechanisms on design and evaluation of virtual keyboards in English language. We found that the mechanisms in their present forms are not applicable

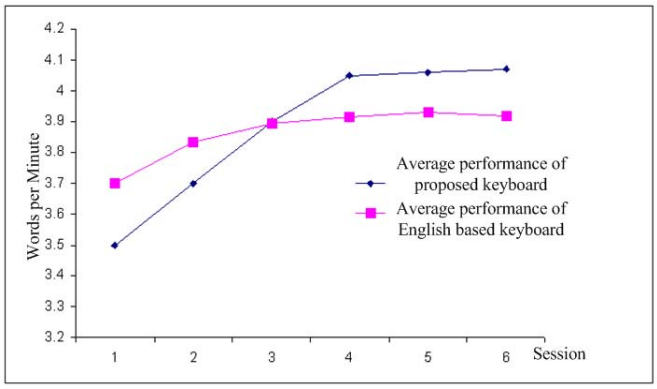


Fig. 4. User evaluation in modified layout

for designing effective virtual keyboard for Indian languages. This statement signifies the necessity for further modification in the methods. We have stated several issues that should be considered for modifications. A comparison among the most relevant approaches is presented in table V and VI. In addition, we proposed a new mechanism in the virtual keyboard design. Our proposed research constitutes text entry rate based evaluation. We reviewed several promising areas of a good keyboard design to achieve more text entry rate. Based on our review, we proposed a improved mechanism for accomplishing moderate text entry rate at the time of *"virtual keyboarding"*. Moreover, we have proposed better keyboard layout for Indian languages which can help other researchers associated with same field.

The research of finding efficient virtual keyboard design can be driven further through *"Keyboard Personalization"* concept. The method can be formulated keeping in mind the issues of keyboard design. In the working phase, users can choose one interface among many which is more suitable to their thought. It is not mandatory that every user should select one or two keyboard as best among all. The best result would come when the distribution of user's choice is spread over the result set evenly. A naive approach is to count the vote for each keyboard layout and find out maximum whereas another approach is to select the proper keyboard which is selected by individual user.

Another direction of research associated with virtual keyboard design can be carried out by finding optimized design in context of Indian language. The approach is to identify the best design in a given set consisting of design alternatives, assuming that the alternatives are limited in number. In other words, the approach is suitable for a small subset. It is, however, also possible to find out a good design by exploring the entire space of alternative designs. A number of design space exploration algorithms report in the literature to achieve this objective. It is found that among the many exploration algorithm like *"Genetic Algorithm"* and *"Simulated Annealing"* based approaches are producing good results. To design efficient function which can provide good result in virtual keyboard design area through selection from exploration space by *"Genetic algorithm"* or *"Simulated Annealing"* based approaches is also evolved to be a good research issue.

REFERENCES

[1] "The fitaly one-finger keyboard," http://www.fitaly.com/fitaly/fitaly.htm (Accessed on January 2010).
[2] I. S. MacKenzie and S. X. Zhang, "The design and evaluation of a high-performance soft keyboard." in *Proceedings of the CHI 99 Conference on Human Factors in Computing Systems.*, 1999.
[3] S. Zhai, M. Hunter, and B. A. Smith, "The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design." ACM, 2000, pp. 119–128.
[4] "Hooke's law - wikipedia," http://en.wikipedia.org/wiki/Hooke's law (Accessed on January 2010).
[5] "Bigram - wikipedia," http://en.wikipedia.org/wiki/Bigram (Accessed on January, 2010).
[6] M. Zhai, S. Hunter and B. A. Smith, "The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design." in *Proceedings of the ACM Symposium on User Interface Software and Technology - UIST 2000, 119-128.* New York: ACM., 2000.
[7] P. Lewis, J.R Kennedy and M. LaLomia, "Development of a digram-based typing key layout for single-finger/stylus input." in *Proceedings of the Human Factors and Ergonomics Society 43rd Annual Meeting.*, 1999.
[8] C. D. Manning and H. Schutze, "Foundations of statistical natural language processing. mit press, 2000. isbn 0-262-13360-1. 620 pp." *Nat. Lang. Eng.*, vol. 8, no. 1, pp. 91–92, 2002.
[9] "Languages of india - wikipedia," http://en.wikipedia.org/wiki/Languages_of_India (Accessed on January 2010).
[10] "Qwerty - wikipedia," http://en.wikipedia.org/wiki/QWERTY (Accessed on January 2010).
[11] W. L. Dvorak A. Merrick N. L. Dealey and G. C. Ford, "Typewriting behavior." 1936.
[12] W. E. Hick, "On the rate of gain of information," *Quarterly Journal of Experimental Psychology*, vol. 4, pp. 11–26, 1952.
[13] R. Hyman, "Stimulus information as a determinant of reaction time," *Journal of Experimental Psychology*, vol. 45, pp. 188–196, 1953.
[14] R. W. Soukoreff and I. S. MacKenzie, "Theoretical upper and lower bounds on typing speed using a stylus and soft keyboard." *Behaviour and Information Technology*, vol. 14, pp. 370–379., 1995.
[15] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement." *Journal of Experimental Psychology*, vol. 47, p. 381391., 1954.
[16] J. Mankoff and G. D. Abowd, "Cirrin: a word-level unistroke keyboard for pen input," in *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology*. New York, NY, USA: ACM, 1998, pp. 213–214.
[17] "Unicode consortium," http://www.unicode.org (Accessed on January 2010).
[18] S. X. MacKenzie, I. S. Zhang and R. W. Soukoreff, "Text entry using soft keyboards." *Behaviour and Information Technology*, vol. 18, pp. 235–244., 1999.
[19] D. A. Norman and D. Fisher, "Why alphabetic keyboards are not easy to use: Keyboard layout doesn't much matter," *The Journal of the Human Factors and Ergonomics Society*, vol. 24, no. 11, pp. 509–519, October 1982.
[20] S. X. MacKenzie, I. S. Zhang, "Text entry using soft keyboards." *Behaviour and Information Technology*, vol. 18, pp. 235–244., 1999.
[21] R. S. Akshar Bharati, Prakash Rao K and S.M.Bendre, "http://www.iiit.net/techreports/2002_4.pdf," IIIT Hyderabad, Tech. Rep., 2002.
[22] "Bengali language - britanica," http://www.britannica.com/EBchecked/topic/60785/Bengali-language (Accessed on January, 2010).
[23] "Multilingual support-wikipedia," http://en.wikipedia.org/wiki/Help:Multilingual_support_(Indic) (Accessed on January, 2010).
[24] "Wikipedia," http://en.wikipedia.org (Accessed on January, 2010).
[25] R. A. Wagner and M. J. Fisher., "The string-to-string correction problem." *Journal of the Association for Computing Machinery*, vol. 21, no. 1, pp. 168–173, January 1974.

[26] I. S. MacKenzie, "Kspc (keystrokes per character) as a characteristic of text entry techniques." in *Proceedings of the Fourth International Symposium on Human-Computer Interaction with Mobile Devices, pp. 195-210. Heidelberg, Germany: Springer-Verlag.*

[27] F. Hoffmann, "Soft computing techniques for the design of mobile robot behaviors," *Information Sciences*, vol. 122, no. 2-4, pp. 241–258, 2000. [Online]. Available: citeseer.ist.psu.edu/hoffmann94soft.html