Easy Swipe Keyboard: A Touch based on Screen Text Entry Method for Smartphone

Md. Ashaduzzaman[1], Mehrab Zaman Chowdhury[2], Mehedi Hassan[3], Ahmed Iqbal Pritom[4]

[1,2,3,4]*Dept. of Computer Science & Engineering, Green University of Bangladesh, Dhaka, Bangladesh*

---***---

Abstract - *An on-screen keyboard or virtual keyboard or soft keyboard layout and a text entry system for smartphones. The main challenge in designing an on-screen keyboard is to fit a huge number of buttons in the small touch screen area of the mobile phones. User dissatisfaction and also typing mistakes often result from using such type of soft keyboards. Our proposed methodology for the keyboard layout, Easy Swipe Keyboard, presents full-text entry facilities with a minimal amount of finger movement. Our aim is to achieve faster typing speed with less amount of typing mistakes or error occurrence by using rectangular buttons. Our proposed layout containing only 8 functional buttons significantly improved the layout design compared with the TYPEHEX keyboard layout which contains 10 functional buttons. The main 3 buttons are given rectangular shape allowing us to place 9 characters in a single button (1 in the center and other 6 are placed in six remaining sides of the rectangular). Letter-frequency data, bigrams and trigrams are considered while placing each letter to ensure that mostly used characters, character pairs and character trios are placed close to each other. The evaluation of our design is done by combining an application of Fitts' law named movement time model and a model named linguistic model. We found that our design approach gives improved wpm values by calculating word per minute (wpm) values for both index finger and thumb compared with soft QWERTY and T9 input method.*

***Key Words*: Interface development, on screen layout, character frequency, Fitts' law, bigram analysis, trigram analysis, gross wpm, net wpm.**

1. INTRODUCTION

As popularity of smart phones is growing rapidly, people are getting more habituated with touch enabled on screen soft or virtual keyboards. To deal with the issues like better typing speed and less typing error, several virtual keyboard models has been proposed. But, even the most popular models like soft QWERTY [1], KALQ keyboard [7], MessagEase [3] and TYPEHEX Keyboard [2] have individual lacking. TYPEHEX Keyboard is latest. It has 10 buttons. So, decreasing the number of buttons is an significant factor in designing on screen virtual keyboards. Some important issues like character per minute (CPM), word per minute (WPM) and Key Stroke per Character (KSPC) depends heavily on design technique. To ensure minimum movement time to reach from one character to another, some additional techniques like letter- frequency analysis and integrating bigram and trigram issues can be very helpful [4]. Unfortunately, none of today's existing on screen keyboard is properly dealing all these issues.

Combining all these issues discussed above, we present a unique text entry system for smart phones. Our main objective is to achieve maximum typing speed with minimum error occurrence. We dealt with all the unsolved issues of previous designs. Ensuring a simple user interface and making people habituated with the design within shortest possible time are some of the major challenges that we faced while designing.

Our proposed design aims at maintaining a good typing speed for both novice and expert users. We combined some well-developed and well experimented methodologies to fit all the characters in the layout. The layout is not only easy to handle, but also requires minimum number of buttons to generate all possible features. Button sizes are increased while comparing with other popular on screen keyboard buttons. Minimum finger movement on touch screen is ensured using Fitts' law and movement model of MacKenzie [5]. For the first time, letter-frequency data and punctuation character frequency data are combined with bigram and trigram analysis. Bigram and trigram are two special cases of n-gram, where values of n are 2 and 3 respectively. Every series of two or three adjacent components in a string of tokens which are typically letters, syllables, or words is called bigram or trigram. Applying these methods, comparing with TYPEHEX Keyboard layout, we found 7.75% improved word per minute (WPM) value using index finger. While comparing with T9 input method, our proposed layout results in 7.88% better WPM using thumb.

2. Related works

There are many keyboard layouts for smart phone. Before starting our work, we learnt about some layouts. We also learnt about some important laws, letter frequency chart and N-Letter Sequences (N-grams). Some popular layouts are Soft QWERTY, MessagEase, KALQ keyboard and TYPEHEX Keyboard. An important law to be discussed here is Fitts's law.

Soft QWERTY [1] is the most popular smart phone keyboard layout with all buttons and functionalities similar to the Desktop QWERTY keyboard. But this layout is hardly feasible because of space limitations on Smartphone screen. Typing mistake is a common problem in this layout while users apply their thumbs to press those tiny buttons.

With the implementation of English letter frequency model MessagEase[3] proposed a novel on screen keyboard layout with minimized movement time. It also reduced the total number of buttons on screen. Reducing some of the major drawbacks, MessagEase drew public attention. MessagEase proposed 9 large keys with an ingenious letter assignment which design led to maximize the speed and ease the writing. It may take a user a few minutes to learn this new way of writing, but it'll save hours by texting and writing faster.

KALQ is optimized for rapid two thumb typing on touch screen devices. KALQ keyboard [7] layout splits the whole keyboard area into two easily reachable groups. It works on tablets as well as smart phones with a large display. The design is based on work by researchers from Max Planck Institute of Informatics, Montana Tech and University of St.Andrews. The design considers multiple human factors affecting the movement of thumbs. This paper also shows a user study where it is shown users could speed up typing by more than 30% over their regular speed than with a default QWERTY layout.

TYPEHEX keyboard [2], presents full text entry facilities with minimum required finger movement. Their proposed keyboard has only 10 functional buttons. The main 5 buttons are given hexagonal shape. It allows us to place 7 characters in a single button (1 in the center and other 6 are placed in six remaining corners of the hexagon). During placing each letter (a-z), they considered letter-frequency data, bigrams and trigrams in order to ensure that frequently used characters, character pairs and character trios are placed as close as possible.

The predictive model of human movement - Fitts's law [5] is extensively used in many cases of Human-Computer Interaction (HCI) and ergonomics. A prediction given by the scientists is that the time needed to expeditiously move to a target area is a function of the ratio between the distance to the target and the width of the target. Fitts's law is employed to model the act of pointing whether physically touching an object with a finger, or using a pointing device by aiming to an entity on a computer monitor.

Fitts's Law has been applied to many scenarios and under a variety of conditions, such as: with many different limbs (hands, feet, eye gaze, head-mounted sights, and the lower lip), physical environments (including underwater), manipulanda (input devices), and user populations (young, old, special educational needs, and drugged participants).

$$MT = a + b\, log2(AW + 1)$$

Here, A is the length of a movement and W is the size of the object, a and b are empirically determined values. Inter button distance, A was measured manually. W is the width of the button.

For cryptanalysis, and frequency analysis in particular, the frequency of letters [4] in text has been studied for a long time, dating back to the Iraqi mathematician Al-Kindi (c. 801–873 CE). Al-Kindi was the one who formally developed the method (Julias Ceaser invented Caesar cipher which can be breakable by this technique even though this method could have been explored in classical times.)

With the development of movable type in 1450 CE, Letter frequency analysis gained additional importance in Europe, where as evidenced by the variations in letter compartment size in typographer's type cases, one must estimate the amount of type required for each letterform.

Few rudimentary techniques for language identification are used by Linguists. One of the widely used techniques is letter frequency analysis. To determine whether an unknown writing system is alphabetic, syllabic or ideographic, letter frequency analysis can be particularly effective. For instance, Hawaiian alphabet has a mere 13 letters, or English which has 26. On the other hand, the Japanese Hiragana syllabary has more than most phonetic alphabets, which contains 46 distinct characters.

An N-gram [4] is a contiguous sequence of n items from a given sequence of text or speech in the fields of computational linguistics and probability. According to application, the items can be words, letters, phonemes, syllables, or base pairs. The n-grams typically are collected from a text or speech corpus. N-grams may also be called shingles when the items are words.

N-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "digram"); size 3 is a "trigram". Greater sizes are sometimes mentioned to by the value of n, e.g., "four-gram", "five-gram", and so on.

It can be said that an n-gram model is a type of probabilistic language model for forecasting the next item in such a series in the form of a (n − 1)–order Markov model. N-gram models are now widely used in probability, communication theory, computational linguistics (for instance, statistical natural language processing), computational biology (for instance, biological sequence analysis), and data compression. Simplicity and scalability are the two advantages of n-gram models (and algorithms that use them) – with larger n, a model can store more context with a well-understood space–time tradeoff, allowing small experiments to scale up efficiently.

3. Proposed Architecture

3.1 Keyboard Design

We are going to present an efficient on screen keyboard layout called Easy Swipe Keyboard. Our design objective is to increase typing speed and reduce typing errors as much as possible. To develop an efficient layout, we integrated some additional designing factors like bigram, trigram and 4-gram [4] along with English letter frequency model.

Our proposed keyboard layout contains only 8 buttons. Distribution of button is given in table-1.

Table-1: Distribution of buttons of Easy Swipe Keyboard

3 buttons	Alphabets, full stop and comma
1 button	Upper/Lower case selection
1 button	Enabling numeric keypad
1 button	Space
1 button	Backspace
1 button	Enter

Our main concern was to place the 26 English letters (a-z / A-Z) and mostly used punctuation characters in a systematic way. Among the 8 buttons, 3 are rectangular shaped. In the center of those rectangular buttons, we placed‘t’, ‘i’, and ‘p’. While placing them we considered top 50 bigrams (frequently used character pairs) in English Alphabet [4]. Single click on these three buttons will make the corresponding alphabets visible on the screen. Up to this, all measurements are done based on letter frequency and top 50 bigrams in English Language.

Fig-3.1: Distribution of Letters, Full stop and Comma

The other 23 letters are placed using the eight sides of each rectangular button. Here also we used top 50 Bigrams of English Language. For example, we found ‘th’ is the mostly used bigram in English. As ‘t’ was previously placed in the center of one rectangular, our primary decision was to place ‘h’ in the closed rectangular where ‘t’ is placed. But among the 8 sides of that rectangular, a randomly chosen side has greater chance of violating Fitts’ law [8]. So while deciding where ‘h’ should be placed, we considered the next few bigrams of ‘h’.

From our bigram list we found that next few bigrams containing ‘h’ are ‘he’ and ‘ha’. So, we narrowed down our decision. ‘h’ must be closed where contains ‘t’ and in that specific side from where ‘e’ and ‘a’ can be most easily accessed. All button’s width, height and inter button distance were previously calculated [10]. So, it was not a very difficult task to determine the side of that rectangular where exactly ‘h’ should be placed. In order to display the characters which are placed in different sides of the button, we need to slide from the center of the button towards the specific side of the rectangular.

Another example will make our designing idea clear. While placing ‘e’ we found that ‘the’ is on top of our 3gram list. While deciding the side, we considered next few bigrams containing ‘n’ which are ‘ion’. Other example will imclude, ‘ur’, ‘ly’, ‘ha’, ‘ion’ etc.

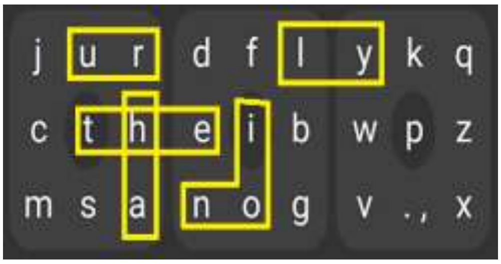

Fig-3.2: Some Example of n-grams

We noticed that some letters are absent in our top 50 bigrams list. It’s an indication that they are not very frequently used. ‘w’, ‘v’ and ‘p’ are some of those letters [4]. But again, randomly placing them is a bad idea. So, we decided to introduce the idea of trigram (frequently used 3 letters). Again we considered top 50 trigrams [4] in our designing. Mostly used trigrams containing ‘w’ are ‘wit’ and ‘was’. Using previous techniques, we placed ‘w’ in a side where optimal distances for typing words containing ‘w’ are minimized.

The design leaves 8 empty places in the major 3 buttons. We noticed that some punctuation letters are very frequently used. So, next we placed 2 most frequently used punctuation letters in those nine positions. Minimum movement time was maintained carefully. For example, we placed the mostly used two punctuation characters ‘.’ and ‘,’ in central button for the ease of the user. Our finally implemented layout is shown in Fig-3.3.

Fig-3.3: Final Layout of Easy Swipe Keyboard

Fig-3.4 shows the design of MessagEase onscreen keyboard. In this layout, there are 14 buttons in total. In Easy Swipe Keyboard, we placed only 8 functional buttons using same screen space. Easy Swipe Keyboard has a better key/button width (10mm) than MessagEase (6mm) [3]. While placing the letters in the button, only letter frequency data is used. They divided the 26 English letters in 3 groups. Most frequently used 9 letters were first placed in 9 central positions. Next 8 were placed on the side of one of the peripheral keys closer to the central key. Next 8 letters are assigned to the peripheral positions of the center key. At last, 'z' is placed beside 'e'.

Fig-3.4: MessagEase's Letter Assignment [12]

Here, no bigram or trigram issues are analyzed. This layout is not a best example of following Fitts' law. So, movement time for key pressing is relatively high. On the other hand, combining Fitts' law with bigram and trigram analysis issues, Easy Swipe Keyboard presents more complete solution towards achieving better typing speed with minimum error occurrence.

4. Performance Evaluation

To measure the efficiency and performance of Easy swipe keyboard's design, we used the methodology presented in Silverberg [5]. Using Fitts' law, we measured the movement time which is the time to reach from one character to another. Fitts' law is expressed as:

$$MT=a+b\ log2(AW+1) \qquad (1)$$

Here, A is the length of a movement and W is the size of the object, a and b are empirically determined values. Inter button distance, A was measured manually. Fig-4.1 shows our calculation for determining inter button distance, A.

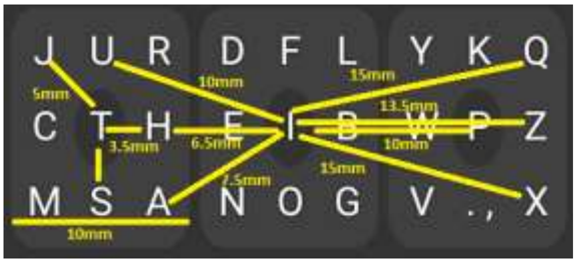

Fig-4.1: Inter Button Distance (A) Measurement

In our layout, W is the width of the button. Easy swipe keyboard's rectangular buttons are exactly 1 cm in width. This decision was carefully chosen because a single finger of average human hand has the width of 1 cm. So,

$$W=10\ mm$$

To evaluate our system, we combined the Movement Model and Linguistic Model (Digraph Probabilities) described by MacKenzie in [5].

4.1. Combining Movement Model and Linguistic Model

The movement model is a one dimensional model based on Fitts's law. The target of this model is to achieve speed tradeoffs in rapid movement. To determine average movement time for different on screen devices, movement model is an essential factor. Considering all 27 major characters of English language (a-z and space) and forming a 27*27 bigram data, we used the linguistic model in our evaluation. This model gives a clear idea about all bigram frequencies of English language. Combining the Linguistic and the Movement model we determined all character pairs movement contributions to the on screen keyboard.

Easy swipe keyboard's input giving pattern is very much similar to the T9 Input Method [5], where single pressing is necessary for displaying any character. Therefore, we simplified our movement model by using the following equation to determine Character Entry time:

$$CT=MT_0 \qquad (2)$$

Where, MT_0 is the initial movement and represents the time to move the finger over the desired key and pressing the key.Average character entry time (CTav) can be expressed as a weighted average of character entry times for all digraphs:

$$CTav=\Sigma\Sigma\ (Pij\times CTij) \qquad (3)$$

For Pij, we used the same 27*27 bigram published in [8].To determine the frequency probability of each pair, we divided total count of that pair by 107199, the size of the character set explained in [8]. Frequency probability of each pair were then multiplied by the previously calculated movement time of that character pair on touch screen. It represents the character entry time of that particular pair. Summing up all 27*27 character entry time, we determined average character entry time, $CTav$. Finally, assuming 5 characters per word, the theoretical upper bound of text entry speed is calculated using same equation used in [5]:

$$\boldsymbol{WPM} = (\mathbf{1}/\boldsymbol{CTav}) \times (\mathbf{60}/\mathbf{5}) \quad (4)$$

4.2. Result and Experiment

To determine the coefficients a and b in the Fitts's law equations, (MT) for pressing any character pair is calculated using the following equation:

$$\boldsymbol{MT} = \mathbf{10000}/(\boldsymbol{N} - \mathbf{1}) \quad (5)$$

MT is calculated for both index finger and thumb [5]. 10 undergrad students took part in the experiment. Among them, five were male and five were female. Their ages were between twenty to twenty four. Six students were right handed where rest of the four were left handed. While calculating the average movement time of character pair, we took the average number of character (N) pressed by them in 10 seconds.

The data sheet based on which values of a and b were calculated is shown in Table 2.

Table-2: Linear Regression Calculation of a and b

Bigram	n	x	y	xy	x^2
th	51	0.678	238.10	161.43	0.46
he	42	1.632	263.16	429.48	2.66
in	46	0.678	243.90	165.36	0.46
er	49	0.678	238.10	161.43	0.46
an	40	2.036	285.71	581.71	4.15
re	42	1.137	270.27	307.30	1.29
on	41	2.137	294.12	628.53	4.57
at	43	1.926	263.16	506.85	3.71
en	44	1.807	263.16	475.53	3.27
nd	43	1.887	270.27	510.0	3.56
ti	42	2.263	303.03	685.75	5.12
es	40	0.848	285.71	242.28	0.72
or	39	1.807	303.03	547.58	3.27
ti	39	1.536	303.03	465.45	2.36
of	41	0.678	277.78	188.33	0.46

Although we worked based on top 50 bigrams, this chart contains result of top 15 bigrams. Analyzing the result sheet, we calculated values of both a and b for index finger and thumb. Table 3 contains our experimental result.

Table-3: Calculating a and b of Fitts's Law

Fingers	Intercept, a (ms)	Slope, b (ms/bit)
Index finger	154	46
Thumb	169	53

Applying this result, initially we found impressive WPM result for both index finger and thumb. Index finger eventually resulted in 47.10 WPM while using thumb it resulted in 44.25 WPM. Some errors occurred during typing and weren't corrected instantly. So, this result actually gives us the theoretical upper bound of Gross WPM [13]. Net WPM will be slightly smaller than our measured data. The equation for Net WPM used in [13] is:

$$\boldsymbol{Net\ WPM} = \boldsymbol{Gross\ WPM} - (\boldsymbol{Uncorrected\ Errors}/\boldsymbol{Time}\ (\boldsymbol{min})) \quad (6)$$

Using this formula, final result of Gross WPM and Net WPM using both index finger and thumb are presented in Table 4.

Table-4: Calculation of Gross and Net wpm

Fingers	Gross WPM	Net WPM
Index finger	48.15	47.65
Thumb	45.20	44.85

4.3. Comparison with other existing layouts

According to MacKenzie and Zhang [9] the theoretical upper bound entry rate of soft QWERTY keyboard is 43.2 WPM. While comparing with soft QWERTY keyboard, Easy swipe Keyboard shows an 8.25% improved WPM value for index finger. While using thumb, this improvement is measured 1.85%. While comparing with T9 input method [5], our result shows 2.06% and 8.15% better WPM for index finger and thumb respectively. The improvement comparison is given in Table 5.

Table-5: Performance Evaluation of Easy swipe keyboard

Fingers	Percentage of improvements	
	QWERTY	T9 Input
Index	8.25%	2.06%
Thumb	1.85%	8.15%

5. Conclusion and future works

In this paper we tried to introduce a smartphone keyboard layout. We employed Fitts's law, letter frequency, bigram, trigram letter distributions as the parameters of designing the keyboard interface and also evaluated our results using the combination of movement and linguistic models. We got improvements over traditional QWERTY based keyboard in respect to WPM and number of errors. All of our experiments were done in 'Samsung Galaxy Core Prime' Smartphone. Easy swipe keyboard showed small but very necessary improvement in increasing typing speed and in reducing typing error. This newer looking interface may need some time to get public attention, but we believe, it will take minimum training time for unskilled users and it's simple and user friendly interface will be highly appreciated by all. Automatic spell correction system and automatic suggestion inclusion is out next target. We strongly believe that adding these features will significantly improve the typing speed in our proposed system.

REFERENCES

[1] Luis A. Leiva, Alireza Sahami, Alejandro Catala, Niels Henze, and Albrecht Schmidt. 2015. Text Entry on Tiny QWERTY Soft Keyboards. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). ACM, New York, NY, USA, 669-678. DOI: https://doi.org/10.1145/2702123.2702388

[2] A. I. Pritom, H. Mahmud, S. Ahmed, M. K. Hasan and M. M. Khan, "TYPEHEX keyboard: A virtual keyboard for faster typing in Smartphone," 2015 18th International Conference on Computer and Information Technology (ICCIT), Dhaka, 2015, pp. 522-526. doi: 10.1109/ICCITechn.2015.7488126

[3] Saied B. Nesbat. 2003. A system for fast, full-text entry for small electronic devices. In Proceedings of the 5th international conference on Multimodal interfaces (ICMI '03). ACM, New York, NY, USA, 4-11. DOI=http://dx.doi.org/10.1145/958432.958437

[4] English Letter Frequency Counts: Mayzner Revisited, http://norvig.com/mayzner.html, Feb. 2018

[5] Miika Silfverberg, Scott MacKenzie and Panu Korhonen, "Predicting text entry speed on mobile phone" in CHI '00 Proceedings of the SIGCHI conference on Human Factors in Computing Systems, Pages 9-16, 2000.

[6] Yanshan Wang, "A novel soft keyboard for touchscreen phones: QWERT" in International Journal of Human Factors and Ergonomics, 2013 Vol.2, No.4, pp.246 – 261.

[7] Antti Oulasvirta, Anna Reichel, Wenbin Li,Yan Zhang, Myroslav Bachynskyi, Keith Vertanen and Per Ola Kristensson, "Improving Two-Thumb Text Entry on Touchscreen Devices" in CHI '13 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Pages 2765-2774, 2013.

[8] R. William Soukoreff and I. Scott MacKenzie, "Theoretical upper and lower bounds on typing speeds using a stylus and keyboard" in Behaviour & Information Technology, 1995, 14, 370-379.

[9] I. Scott MacKenzie and Shawn X. Zhang, "The Design and Evaluation of a High-Performance Soft Keyboard" in Proceedings of the ACM Conference on Human Factors in Computing Systems - CHI '99, pp.25-31.

[10] Shumin Zhai, Michael Hunter and Barton A. Smith, "The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design" in UIST '00 Proceedings of the 13th annual ACM symposium on User interface software and technology, Pages 119-128.