ISSN (Online) 2456 -1304

International Journal of Science, Engineering and Management (IJSEM)
Vol 2, Issue 12, December 2017

On-screen Swipe Keyboard Using Gaze-based Method

[1] Morokot Cheat, [2] Manop Wongsaisuwan
[1] Chulalongkorn University, [2] Chulalongkorn University

Abstract: **-- The group of disabled people have limitation in quality of life in everyday activity. Assistive technology using eye gaze is researched to improve their life in learning. It will be beneficial to handicapped people to type words quickly and easily by themselves. Scanning gaze-path will be inaccurate if we scan wrong for the first letter since users will hit other keys unintentionally during saccade for intended letter. In this paper, we introduce on-screen keyboard for text-entry with integration of dwell-free and dwell-time method instead of using fingers. Dwell-time method is applied for the first letter and the last letter to make sure that the users activate the right letter. After users activate the first letter, they can swipe their eye gaze with dwell-free method to make their typing faster. To predict intended words which users would like to type, Revised Levenshtein algorithm is used to calculate distance between scan-path words and words in database. The highest score is considered as intended words that users intend to type. Scoring criteria include distance, word frequency, and stored words that users used to type. In experiment, the performance of less letter is higher than many letters because of missing many gaze-path letters since setting threshold of fixation duration is so high.**

Index Terms— Assistive technology, Dwell-free, Dwell-time, Gaze path, Swipe keyboard.

I. INTRODUCTION

Recently, Human-Computer Interaction (HCI), which is a technique that allows people to have advanced communication with computers, is an interesting topic in research. Since the handicapped people cannot use the computer like usual people, using eye movement for user-to-computer communication becomes an efficient assistive technology for humans. Eye tracking which is the process to measure eye movement or gaze point of eye is used as an input to interact between eye gaze and the computer [1]. It is used for many various areas of research such as neuroscience, psychology, industrial engineering and human factors, marketing and advertising, computer science, user interface for home control, capture technology, and virtual reality research [2].

To help disabled people in typing, researchers investigated dwell-time typing by gaze selection to activation as command without mouse click or mouse cursor [3], [4], [5]. Using dwell time was a good solution in control keyboard because everything would activate all the time everywhere we looked if we did not apply dwell-time method. Majaranta et al. (2009) proposed the system to allow users could adjust duration of dwell time by themselves. Their results showed that reduction of average of dwell-time duration could provide fast gaze typing [6].

In these few years, there have been research on designing on-screen keyboards of the computer using eye-based dwell-free method to swipe through letters for typing text entry [7], [8], [9], [10]. Liu et al. (2016) developed a system to recommend words from swiping the eye gaze with solving three main errors: extra-letter error, neighbor-letter error, and missing-letter error using Longest Common States Mapping algorithm. It predicted typed word from dictionary words stored in database. When users finished their typing, they just needed to look back to candidate lists. Anyway, there was no specific activation for the first and the last letter of words.
Later, Karauchi et al. (2016) improved the interface of on-screen keyboard to enable users to add words that did not have in dictionary into database. To solve the unknown first letter and to get specific detection on the first letter that users intend to type, the researchers used reverse crossing method to select the key by releasing another button as action button when gaze point hit to that key. The users need to move their eye up on action button after they hit their intended letter or word to release that the first letter of words is selected correctly for predictive keyboard. However, the action button is designed with overlapping on other keys surface of keyboard, so the users might still hit the wrong point between action button of key activation and other keys. Nevertheless, those algorithms do not remember words which used to type in the past in criteria of word scoring.

Therefore, our work will be closer to Kurauchi's research in that designing on on-screen keyboard for handicapped people to type text entries using only their eye-gaze path with

ISSN (Online) 2456 -1304

International Journal of Science, Engineering and Management (IJSEM)
Vol 2, Issue 12, December 2017

scoring criteria. We propose combination of dwell-time method and dwell-free method together in swipe gaze-based keyboard including storing of previous typed words to rank intended word. Dwell-time method will be used for activation of the first and the last letter; then system will switch to dwell-free method after activation of staring point. In addition, Levenshtein algorithm will be revised using neighbor weight.

Hopefully, our contribution will enhance advanced communication between humans, especially handicapped people or people with motor impairments, and computers to be more robust and native way in typing.

II. REVISED LEVENSHTEIN ALGORITHM

A. Classical Levenshtein

Levenshtein algorithm is a common method to compare two strings by measurement of the similarity from the source string (s) and the target string (t) [11]. It is used in plagiarism detection, spell checking, speech recognition. The classical Levenshtien algorithm is described as below [12]:

Step 1: Set n to be the length of source string (s). Set m to be the length of target (t).
Step 2: Construct and initialize a matrix containing 0 to m rows and zero to n columns.
Step 3: Examine each character of s (i from one to n).
Step 4: Examine each character of t (j from one to m).
Step 5: If s[i] equals t[j], the cost is zero. Otherwise, the cost is one.
Step 6: d[i,j] = minimum(d[i-1,j]+1, d[i,j-1]+1, d[i-1,j-1] + cost).
Step 7: The distance is found in cell d[n,m].

B. Revised Levenshtein Algorithm

For classical Levenshtein, the cost of same letter between source and target string is zero, otherwise it equals to one. In this paper, Levenshtein with iterative of two matrix rows was used instead of iterative of full matrix. In addition, we revised algorithm of Levenshtein to change the cost with weight for different letter. Weight is considered as w when both letters in comparison are neighbor letter, otherwise the cost is one. Thus, algorithm is revised from simple algorithm on Step 5.

Step 5: If s[i] equals t[j], the cost is 0. If s[i] differs from t[j] but they are neighbor letter, the cost is w. Otherwise, the cost is one.

The benefit of this Revised Levenshtein algorithm is to improve prediction from neighbor-letter error from systematic error of eye tracker which gives inaccurate gaze point due to imperfect calibration.

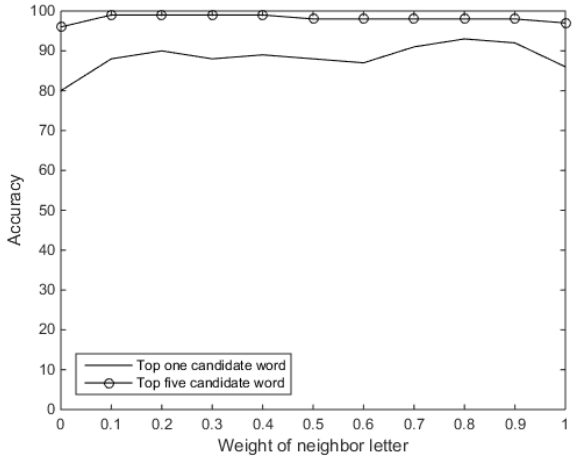


Figure 1. Neighbor weight

Fig. 1 showed the preliminary result on neighbor weight versus performance of accuracy. The data from this graphic were randomly generated from online software tool of typing error generator including missing letter, neighbor letter, inserted letter, and swapped letter error [13]. The top one rate was illustrated in simple line whereas the top five rate was presented in line with dot point. The accuracy of the top five was nearly steady. In contrast, changing weight of neighbor letter could affect recommendation of intended word in the first rank. Hence, neighbor weight equaled to 0.8 was selected in this paper.

III. SYSTEM DEVELOPMENT

A. Character of Eye Movement

Character of human gaze is divided into two main parts such as saccade and fixation. Saccade is a rapid eye movement in scanning process to find the specific point whereas fixation is stationary of gaze point on single interested location that the users need to focus. So, eye fixation is mostly longer than eye saccade. The average duration of a saccade is from 20 ms to 40 ms whereas average duration of a fixation is between 100 ms to 600 ms [14], [15].

B. Concept of System

The swipe keyboard is a kind of keyboard which uses continuous gesture from the first letter to the last letter of word on touchpad screen as predictive text keyboard. The

ISSN (Online) 2456 -1304

International Journal of Science, Engineering and Management (IJSEM)
Vol 2, Issue 12, December 2017

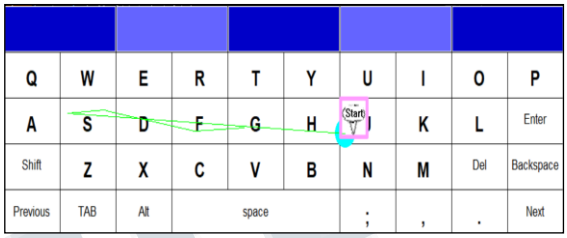
concept of swipe gaze-based keyboard comes up with initial idea of swiping using fingers. Using finger touch as the input, the system can get the actual first letter and the last letter to predict text-entry. The problem of swipe keyboard using gaze-based method is a bit difficult to know which letter users would like to activate as the starting point of words because users need to seek around on-screen keyboard before they can find their preferred letter, but the system will consider all keys to be activated when gaze points hit them [4], [7].

Thus, this system proposes integration of dwell time to activate the first letter of words for more accurate. The users need to fixate on small start box at the left side of the first letter for a limited period. Then, the system will switch to dwell-free method to allow users to swipe on keyboard for their intended words to fasten their typing. In this step, users might hit extra letters that are not intended letters unintentionally. Scanning all gaze path on keyboard will give many letters in analyzing. So, pre-processing is used to deal with this problem. Even dwell-free method does not limit time for users to fixate [16], average duration of a fixation is between 100ms to 600ms. Using threshold value as pre-processing will ignore irrelevant letters which come from eye saccade [17]. If threshold time is small, there are too much input letters, but if threshold is too large, it will lose more intended letters for analysis. Lastly, if users reach the end letter of word, they need to use dwell-time method again to fixate on small left box of intended letter until it is activated.

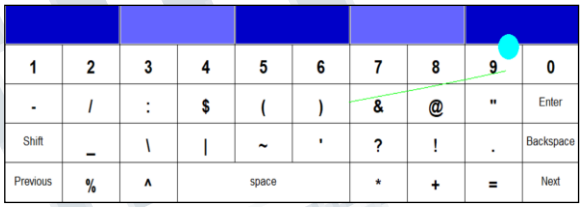
After getting input from scan-path words to the system, the system will calculate the distance between scan-path words and database words from dictionary. The 5000 words with number of frequency in English based on Corpus of Contemporary American English (COCA) are taken from online website: http://www.wordfrequency.info [18]. Those words are stored in database of system with formatting into trie data structure because advantage of trie is to build main roots for existing database words [19]. Trie data structure can reduce computational time in comparison of two strings because it will compare character letter of input words with letters of main roots of dictionary in database and then it will continue to compare adjacent letter. So that, it will not spend time on comparison of the same letters in the dictionary. After that, Revise Levenshtein algorithm is used to compare two strings to find high score by score criteria. The candidate words will be shown in candidate list to give choices for users to select. The first candidate word is recommended due to order of the highest score ranking.

C. Interface Design

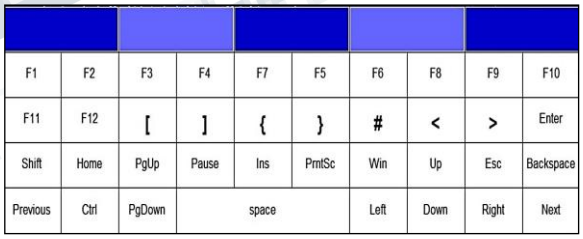
The on-screen keyboard can be categorized into three panels as shown in Fig. 2. The first panel is swipe gazed-based keyboard which shows the English character. Since the second panel and the third panel will display numbers and other symbols, only dwell-time method will be applied to type key-by-key as numerical method normally. To change panel screen, key “Next” or key “Previous” needs to be pressed by gaze point.



(a) The first panel of keyboard



(b) The second panel of keyboard



(c) The third panel of keyboard

Figure 2. Interface of on-screen keyboard

D. The Input and Output of System

The input of the system is gaze point from eye tracker. Tobii EyeX Controller is a low-cost eye tracker which is installed below the computer screen. It will capture the users from camera with near infrared illumination to improve quality to detect eyes movement. When gaze point hit area of on-screen keyboard, the system will store scan-path words. In addition, algorithm of the system will be applied to find better predictive text entry for users after activation of end square box at left of last letter. Thus, candidate words in list is the output to give choice for user in swipe keyboard using gaze-based method. After the users select their intended word from candidate list, process message can be sent out to release that words on location of current mouse cursor in application of Windows such as Microsoft Words, Power

ISSN (Online) 2456 -1304

International Journal of Science, Engineering and Management (IJSEM)
Vol 2, Issue 12, December 2017

Point, or social network.

Scoring Criteria

To be able to recommend candidate words, scoring criteria are important in process of word choosing. There are three main factors to include as linear combination, as in (1). The first one relates to distance between scan-path word and database word. The parameter r_i in (1) is ratio of Levenshtein distance d_i between typed words and i^{th} word in database, $r_i = 1/d_i$ [7]. The word frequency that appear in many usages also affects to score ranking as the second factor. f_i is number of frequency of i^{th} word in database whereas t_i is number of i^{th} word in database that users used to type. The last scoring criterion is from remembrance of typed word in that keyboard.

$$Score_{(word_i)} = w_1\left(\frac{r_i}{\sum_{k=1}^{n} r_k}\right) + w_2\left(\frac{f_i}{\sum_{k=1}^{n} f_k}\right) + w_3\left(\frac{t_i}{\sum_{k=1}^{n} t_k}\right) \quad (1)$$

The constraint of sum of three weight (w_1, w_2, *and* w_3) equals to one where w_1, w_2, *and* w_3 are weight of ratio of Levenshtein distance, word frequency, and stored words with selected value as 0.975, 0.01, and 0.015, respectively.

METHODOLOGY

The methodology of swipe keyboard gaze-based method describes in main steps as shown in Fig. 3:

Step 1: Build words in database into trie data structure.

Step 2: Initialize threshold duration as pre-processing for ignore irrelevant letter from eye saccade.

Step 2: Activate the first letter of word.

Step 3: Store letter sequence which is gazed longer than threshold.

Step 4: Activate the end letter of word.

Step 5: Calculate Revised Levenshtein distance between database and input word.

Step 6: Ranking the candidate words by proposed scoring.

Step 7: Display candidate words.

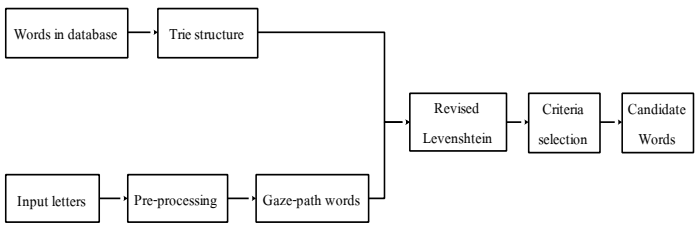


Figure 1. Concept of overall system

EXPERIMENTS

The experiments were conducted in Control Systems Research Laboratory of Department of Electrical and Engineering in Chulalongkorn University. Data were recorded from one male student and one female student who wore glasses. All participants had normal physical condition, and they were familiar in usage of the computer. Handicapped people were encouraged to be volunteer as participants; however, there was no handicapped people in our experiment yet due to time constraint.

Experimental Design

The participants sat in comfortable chair in front of computer which was mounted with eye tracker, Tobii EyeX Controller, below screen of the computer in the middle point and distance between the participants to computer screen was not longer than 50cm. The system was tested on Windows 10 of Laptop of intel® Core i7-7700HQ processor at 2.8 GHz and 8 GB DDR4-RAM with displaying on a large Desktop monitor. Before perform typing task, participants needed to calibrate eye gaze with Tobii Eye Tracking Core software in order to adapt manner of eye movement of each user to refine accuracy and performance on this device.

Participants performed four sessions which each session had 20 random words. Four sessions were provided following vary of word length distribution. For the first session, random words of two-letter words were displayed on screen whereas the second session and the third session were session of typing for random words of three-letter words and four-letter words, respectively. Additionally, the fourth session was mixed session which combined non-uniform word length. The timeline of each session drew as illustration in Fig. 4.

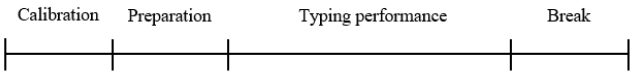


Figure 2. Timeline of experiment

Calibration was the first step to calibrate eye gaze of each participate with real environment individually. Each participant needed to take calibration only once of first session of experiment. They did not need to re-calibrate after first session. However, each participant could request for re-calibrate before starting next sessions if they feel not comfortable with gaze-point detection.

Preparation was a process of helping the participants to be ready for main performance of swipe typing. The main step of experimental timeline was step of typing performance. In this step, each participant would swipe on on-screen keyboard using eye gaze to type words following displayed

ISSN (Online) 2456 -1304

International Journal of Science, Engineering and Management (IJSEM)
Vol 2, Issue 12, December 2017

words on screen. Last of each session, each participant could break for before new session would start.

RESULT AND DISCUSSION

Fig. 5 presented the accuracy following each section. The average accuracy rate of 2-letter words reached 87.5% whereas typing 3-letter words reached only 70%. Section of 4-letters words got only 65%, and the lowest result (52.5%) is in section of mix-letter words which had target words until 11 letters. Accuracy of experiment on section of 2-letter words was higher than other sections because this system for 2-letter words looked like dwell-time typing since users needed to activate for the first and the last letter. Even typing words of many letters was not accuracy like 2-letter words, duration of typing was faster due to integration with dwell-free method.

Results showed that using word generator gave more accurate for candidate words than experiment with actual users. It proceeded from difficulty in control gaze point of eyes. The participants felt unfamiliar in typing the system for the first typing. They reported that it was hard to hit letters on the edge of screen when keyboard was full screen on a large monitor of desktop, so it might be out of bound of tracking.

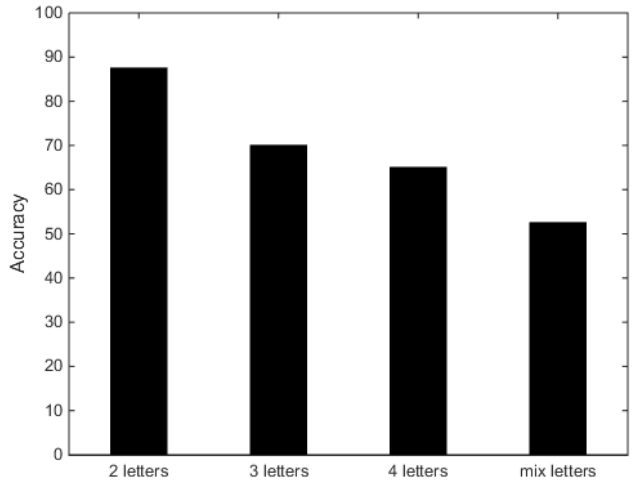


Figure 3: Result of each session

In saved file of gaze-path scanning, it was observed that most words in scan-path words were less than until half of target words that users needed to type. It appeared that the value of threshold of fixation duration might be too high compared to real fixation duration for users who were familiar and fast typing. That was the reason why the performance of words with many letters was somehow low.

CONCLUSION

In conclusion, this system performs well for typing word with less letters. However, there existed some limitations. All weights were given by evaluation from word generator, and threshold was set following theory of eye fixation. To improve the system, number of participants, especially handicapped people, should involve more in experiment. Furthermore, since characteristic of typing using eye gaze of each user is not similar, adaptive weights of neighborhood, weight of scoring, and threshold for pre-processing should be changed to learning of user-to-user in the near future.

REFERENCES

[1] P. Majaranta and A. Bulling, “Advances in Physiological Computing,” London: Springer-Verlag, 2014, pp. 39–65.

[2] A. T. Duchowski, “A breadth-first survey of eye-tracking applications,” Behav. Res. Methods, Instruments, Comput., vol. 34, no. 4, pp. 455–470, 2002.

[3] N. Bee and E. André, “Writing with Your Eye: A Dwell Time Free Writing System Adapted to the Nature of Human Eye Gaze,” in Perception in Multimodal Dialogue Systems, 2008, vol. 5078 LNCS, pp. 111–122.

[4] J. P. Hansen, A. S. Johansen, D. W. Hansen, K. Itoh, and S. Mashino, “Command Without a Click : Dwell Time Typing by Mouse and Gaze Selections,” Proc. Human-Computer Interact. – INTERACT´03, no. c, pp. 121–128, 2003.

[5] M. E. Mott, S. Williams, J. O. Wobbrock, and M. R. Morris, “Improving Dwell-Based Gaze Typing with Dynamic, Cascading Dwell Times,” Proc. 2017 CHI Conf. Hum. Factors Comput. Syst. - CHI ’17, pp. 2558–2570, 2017.

[6] P. Majaranta, U.-K. Ahola, and O. Špakov, “Fast gaze typing with an adjustable dwell time,” Proc. 27th Int. Conf. Hum. factors Comput. Syst. - CHI 09, p. 357, 2009.

[7] A. Kurauchi, “EyeSwipe : Dwell-free Text Entry Using Gaze Paths,” in Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, 2016, pp. 1952–1956.

[8] Y. Liu, B. S. Lee, and M. J. McKeown, “Robust Eye-Based Dwell-Free Typing,” Int. J. Hum. Comput. Interact., vol. 32, no. 9, pp. 682–694, 2016.

[9] Y. Liu, C. Zhang, C. Lee, B. Lee, and A. Q. Chen, "GazeTry : Swipe Text Typing Using Gaze," Proc. Annu. Meet. Aust. Spec. Interes. Gr. Comput. Hum. Interact., pp. 192–196, 2015.

[10] D. Pedrosa, M. D. G. Pimentel, A. Wright, and K. N. Truong, "Filteryedping: Design Challenges and User Performance of Dwell-Free Eye Typing," ACM Trans. Access. Comput., vol. 6, no. 1, pp. 1–37, 2015.

[11] D. Karch, D. Luxen, and P. Sanders, "Improved fast similarity search in dictionaries," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 6393 LNCS, no. Spire, pp. 173–178, 2010.

[12] A. Apostolico and Z. Galil, Pattern Matching Algorithms. Oxford: Oxford University Press, 1997.

[13] "Typo Errors Generator - Typing Mispelling - Online Software Tool," 2017. [Online]. Available: https://www.dcode.fr/typing -error-generator. [Accessed: 02-Oct-2017].

[14] Tobii Technology, "Determining the Tobii I-VT Fixation Filter ' s Default Values," Tobii WhitePaper, 2012.

[15] R. Matos and T. Technology, "Designing eye tracking experiments to measure human behavior," Tobii Technology. pp. 1–42, 2010.

[16] I. S. MacKenzie and X. Zhang, "Eye typing using word and letter prediction and a fixation algorithm," Proc. 2008 Symp. Eye Track. Res. Appl. - ETRA '08, vol. 1, no. 212, p. 55, 2008.

[17] Z. Sharafi, Z. Soh, and Y. G. Guéhéneuc, "A systematic literature review on the usage of eye-tracking in software engineering," Inf. Softw. Technol., vol. 67, pp. 79–107, 2015.

[18] M. Davies, "Word frequency: based on 450 million word COCA corpus," 2016. [Online]. Available: http://www.wordfrequency.info. [Accessed: 15-Aug-2017].

[19] A. Miyachika, R. Yoshinaka, and A. Yamamoto, "Approximate String Matching Based on Extending Levenshtein Automata for Tries," pp. 11–16.