A Zoomable Virtual Keyboard for Eye Gaze Tracking to Assist People with Motor Disability

Anaelis Sesin[1], Melvin Ayala[1], Malek Adjouadi[1], Mercedes Cabrerizo[1], Armando Barreto[2]

[1]Center for Advanced Technology and Education
[2]Digital Signal Processing Laboratory
Department of Electrical & Computer, Florida International University
10555 W. Flagler Street, Miami FL 33174. USA
anaelis.sesin@fiu.edu, melvin.ayala@fiu.edu, adjouadi@fiu.edu, mcabre001@fiu.edu, barretoa@fiu.edu

Abstract

Onscreen keyboards such as the one provided by Microsoft Windows Operating System are valuable typing aids; however, their use in eyegaze systems are not yet fully exploited. Saccadic eye movement produces sporadic motion of the mouse pointer, which makes it difficult to precisely click on the right keys in a timely manner. The goal of this study was to develop an enhanced virtual keyboard to overcome the inherent effect of jitter by using a trajectory smoothing technique in combination with a zoom-in option of keys nearer the user's eye gaze. The novelty of the proposed onscreen keyboard is that it counteracts the unwanted jitter effect by temporarily magnifying the intended keys for selection. Typing experiments were conducted with seven subjects to assess the suitability of the proposed keyboard with respect to the Microsoft on-screen keyboard in terms of clicking efficiency and speed. The results revealed a significant increment of over 96% in typing speed and 309% in clicking efficiency.

1. Introduction

Computer interface research has known respectable growth in the last decade, and the deployed assistive technology tools have been developed to address the issue of universal accessibility, especially in terms of computer-based applications and information resources [1, 2]. A number of human-computer interfaces (HCI) have relied on eye gaze tracking (EGT) systems as one possibility to establish interaction between user and the computer through eye movement [3, 4].

Unfortunately, the use of EGT systems as the primary mechanism for controlling the mouse pointer has been complicated by inaccuracies arising from saccadic eye movement. Such natural, involuntary movement of the eye results in sporadic, discontinuous motion of the pointer, or "jitter," a term which is used herein to generally refer to any undesired motion of the pointer resulting from a user's attempt to focus on a target, regardless of the specific medical or other reason or source of the involuntary motion. Some attempts at increasing the accuracy of mouse cursor control through eye gazing activity involve the integration of a complementary technology such as Electromyograms (EMG) [5, 6]. However, these approaches result in the user having to wear some device or electrodes which may be perceived as intrusive for users that can overcome the need for EMG mediation.

In earlier publications [7, 8], we have proven that by establishing ANN-based user profiles which were embedded in the HCI interface, the jitter was reduced significantly. However, remnants of this jitter still proved problematic in dealing with small icons or small letters for typing purposes. The objective of this research endeavor is to therefore augment the EGT-based HCI system [9] to accommodate such shortcomings by using the concept of magnification akin to the innovative Zoomable User Interface (ZUI) model. This concept was applied in the development of a Zoomable Virtual Keyboard (ZVK) in order to improve the usability and accessibility to a wide range of applications such as web browsing, email, word processing and editing.

ZVK displays a virtual keyboard on the screen and, when operated in conjunction with the EGT-based HCI, allows users with mobility impairments to type data using only eye movement. Besides providing the basic level of functionality of a keyboard, the ZVK presents two unique characteristics that differentiate it from other virtual keyboards:

1. It displays a list of 1000 most commonly used words from which the user can choose, eliminating the need of typing all the characters in a word, making the typing process faster and less exhausting.
2. It magnifies each key as the mouse is moved over it, eliminating mistyping a letter due to the jitter behavior of the mouse cursor or the unwanted effects of initiating a click with an eye closure.

2. HCI system overview

The EGT-based HCI shown in Figure 1 is based on a remote eye gaze setup, which is less intrusive in contrast to the head mounted version. It consists of a CPU for eye data acquisition, another CPU for user interaction (stimulus computer), an eye monitor, a scene monitor, an eye imaging camera, and an infrared light source. The integrated EGT system in this research is developed around the ISCAN® ETL-500 technology [10]. The corneal and pupil reflection relationship is the foundation for the technique used to determine the movement of the eye [11].

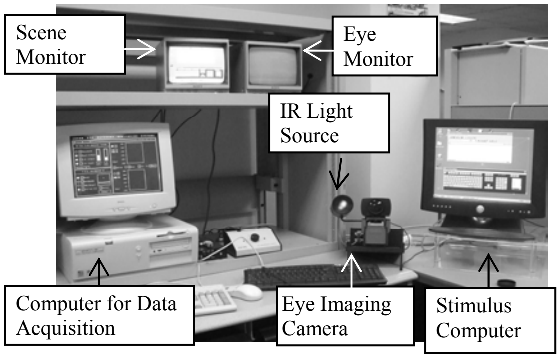


Figure 1. EGT-based HCI components

Figure 2 illustrates the system setup: (a) with a motor able subject using a standard head rest to prevent any abrupt head movement; and (b) with a motor disable individual on a wheelchair.

(a) **(b)**

Figure 2. EGT-based HCI setup

3. Virtual keyboard

The ZVK is composed of 86 buttons arranged in a rectangular fashion based on the QWERTY layout, which is the most commonly used modern-day keyboard layout on English-language computer [12]. It also displays a list of 1000 most commonly used words which can be employed to speed up the typing process. Figure 3 shows a snapshot of the ZVK application. It has the same area as the Microsoft on-screen keyboard.

Figure 3. On-screen keyboard

3.1 ZVK behavior

The basic operations of the ZVK includes: (1) typing a single character or symbol, and (2) pressing and holding several keys simultaneously or in sequence to either produce other type of character, such as uppercase letters, or to generate a computer command, such as opening a menu. For instance, pressing ‘Alt’+’F’ opens the ‘File’ menu of the active application. Notice that in the case of a virtual keyboard the term ‘key press’ actually refers to ‘button click’, thus two or more keys can not be pressed simultaneously in the ZVK. However, with the use of variable flags, the ZVK module stores the information of previously pressed keys allowing the user to replicate the same operations.

For the ZVK application to send characters or commands to an application, it must first capture the program’s window handler. Thus, the user must click in the program in which the characters would be typed.

When a key is clicked, the ZVK application determines if the key pressed is a character, modifier, or lock key. If the stroked key is a modifier or lock key, a flag is set/reset. The value of the flag depends on how many times the same key has been pressed. If the pressed key is a lock key, then the appearance and value of some of the buttons in the keyboard change accordingly. For instance, if the user strikes ‘Caps Lock’ once, the caption of all the alpha character keys (a-z) are changed to upper letter. If it is pressed a second time, the caption and value are restored back to lower case. Figure 4 illustrates the ZVK when the ‘caps lock’ key is ON.

Figure 4. ZVK application with the ‘Caps Lock' ON

If the pressed key is a character key, there are two possible outcomes. If none of the modifier flags is set, the character value of the key is sent to the active window. However, if any of the modifier flags is set, the corresponding command is concatenated to the key’s value before sending it to the active application. For instance, in order to type ‘A’ (capital ‘a’), first the “shift” key is clicked, which sets the shift modifier flag to 1, and then ‘a’ is pressed.

Furthermore, the ZVK displays a list of words that start with the typed character. From this list, the user can select a word to be sent to the active application. Thus, there is no need of typing all the characters in a word, speeding up the typing process.

The flowchart in Figure 5 illustrates the behavior of the ZVK module, when a key is pressed.

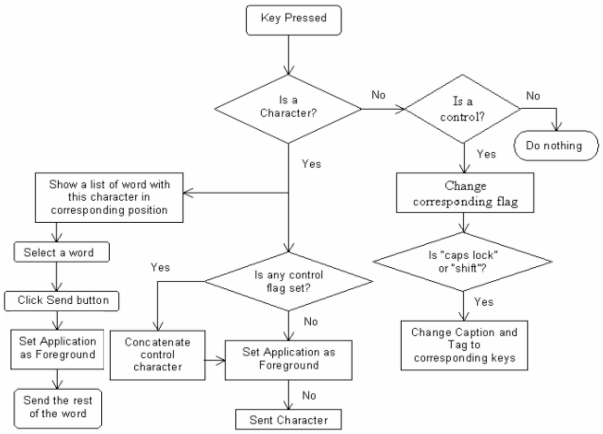


Figure 5. ZVK flowchart when a key is pressed

3.2 Word list manipulation

A list of 1000 most commonly words and their statistics of use was found in the internet (http://esl.about.com/library/vocabulary/bl1000_list1.htm.) and used to create a database. The database contains two fields: 'word', and 'rank', as exemplified in Figure 6. The 'word' field contains the list of words, and the 'rank' field includes the frequency of used for each word, as found on the internet. The ZVK application connects to the database during the STM mode only.

default : Table

word	rank
a	5
able	422
about	58
above	307
act	177
add	168

Figure 6. Distribution of the word list database

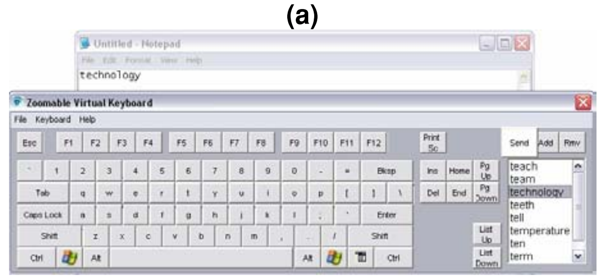
Every time, a character is typed, it is stored in a temporary buffer in the ZVK module and used to execute a query to the database and locate all the words that begin with the typed character. A temporary word list is created using the query results and displayed in the ZVK window as demonstrated in Figure 7. If a new character is typed, it is concatenated at the end of the temporary buffer. Whenever the user types a new character, a new query to the database is performed and a new temporary word list is displayed with the new results. If the user clicks on 'enter' or 'space bar', the temporary buffer is clear, and the default list, containing all the words, is displayed on the window. If the user clicks on 'backspace', the last typed character is removed from the buffer and deleted in the active application. Once again, the list is updated using the new string in the buffer. If the database query does not generate results, the default list is displayed. As mentioned in the previous section, a word can be selected form the list and sent to the active application as shown in Figure 8.

(a)

(b)

Figure 7. ZVK (a) after typing 't', (b) after typing 'te'

(a)



(b)

Figure 8. After typing 'te', the word 'technology' is selected and sent to the Notepad application.

The ZVK application allows the user to customize the list by adding or deleting words from the list. If a word is not in the list, the user can add it by typing the complete word and then pressing the 'add' key. The word is then added to the end of the database, and its 'rank' is set to '1'. Furthermore, the user can also remove a word from the list by selecting the word and pressing the "rmv" button. A query is executed to locate and remove the word in the database.

3.3 Key magnification

Given that ZVK application was developed particularly to be used in the EGT-based HCI, it needed to overcome the remnants of the mouse cursor jitter and the undesired effects of using an eye closure to initiate a mouse click.

When controlling the mouse cursor through eye gazing only, it behaves in a saccade matter, making dealing with small icons or buttons difficult. Furthermore, the proposed EGT-based HCI implements all mouse cursor actions using eye gazing only, thus a mouse click was defined as an eye closure longer than a regular blink avoiding having to integrate any other device to the system. However, as the eyelid closes, the coordinates of the eye drop a bit causing the mouse cursor to deviate from the target where the user wants to click.

The solution to avoid these unwanted effects would be to create larger keys in the ZVK application. However, increasing the size of all the keys was impractical since the ZVK would obstruct the other applications. Thus, a more suitable solution was to magnify each key individually as the mouse cursor moves over it. As a result, the clicking area of each key would be increased without affecting the actual size of the virtual keyboard. Consequently, even though the mouse cursor may not be stabled, the user can still click in any of the keys with greater accuracy. Thus, the key selection process becomes easier and less frustrating. From the user's point of view, it would be like zooming in on each key as the mouse moves over it. Magnification increases the size of each key by 50% to 80% of its original size, depending on the distance of the intended key and the user's eyegaze. Figure 9 illustrates the behavior of the ZVK when the mouse moves over the 'Shift' key.

Figure 9. 'Shift' key is magnified as the mouse cursor moves over it

4. Results

Experiments were conducted in order to asses the suitability for an EGT-based HCI of the ZVK with respect to an existing virtual keyboard application, such as the Microsoft On-Screen Keyboard (OSK) shown in Figure 10. Each application usage was evaluated in terms of clicking efficiency and typing speed. Clicking efficiency refers to the relationship between the total number of clicks (C_T) performed, including clicking in an undesired key, and the number of characters in a sentence which is the ideal number of clicks (C_I) as defined in Equation 1. It is a value between 0 and 1 (for maximum efficiency).

Figure 10. Microsoft on-screen keyboard

$$C_{ef} = \frac{C_I}{C_T} \qquad (1)$$

The typing speed is defined in Equation 2 as the number of characters in a sentence (S_L), divided by the time it took to complete a sentence (t_S)

$$v_S = \frac{S_L}{t_S} \qquad (2)$$

The indicators for performance evaluation were recorded using the 'Keyboard Trial' application which is illustrated in Figure 11.

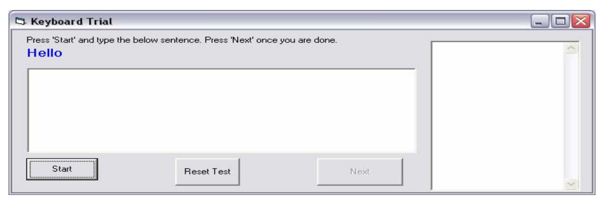


Figure 11. Keyboard trial GUI

Tests were conducted using 7 subjects, each undergoing a total of 3 trials. Each trial consisted on typing 4 sentences using the OSK and the ZVK applications. All sentences differ in character length as shown in Table 1. For simplicity, all characters were typed using lower case only.

Table 1. Experimental sentences

Sentences	S_L
Hello	5
Good Morning	12
I am typing with my eyes	24
I am using a virtual keyboard	29

The results of the typing speed and click efficiency of the 3 trials from each subject were averaged and tabulated in Tables 2 and 3. Furthermore, the results from each subject were averaged and plotted in Figures 12 and 13, which reflect an increase in both typing speed and clicking efficiency.

Table 2. Typing speed results

Subj. #	S_L	OSK t_S (ms)	OSK v_S (char/s)	ZVK t_S (ms)	ZVK v_S (char/s)
1	5	19770	0.25	9328	0.54
	12	40809	0.29	22129	0.54
	24	55371	0.43	51648	0.46
	29	86340	0.34	54980	0.53
2	5	11750	0.43	10068	0.50
	12	54980	0.22	21422	0.56
	24	80961	0.30	24492	0.98
	29	83594	0.35	52340	0.55
3	5	21094	0.24	9668	0.52
	12	70910	0.17	22180	0.54
	24	104461	0.23	42012	0.57
	29	140832	0.21	62062	0.47
4	5	20867	0.24	9812	0.51
	12	37945	0.32	22680	0.53
	24	59320	0.40	40094	0.60
	29	87227	0.33	50469	0.57
5	5	12086	0.41	9719	0.51
	12	27742	0.43	19008	0.63
	24	92984	0.26	41078	0.58
	29	159453	0.18	49367	0.59
6	5	15430	0.32	9000	0.56
	12	34555	0.35	18789	0.64
	24	76297	0.31	42453	0.57
	29	76797	0.38	46078	0.63
7	5	13617	0.37	8398	0.60
	12	61789	0.19	12307	0.98
	24	72719	0.33	36477	0.66
	29	120781	0.24	44867	0.65

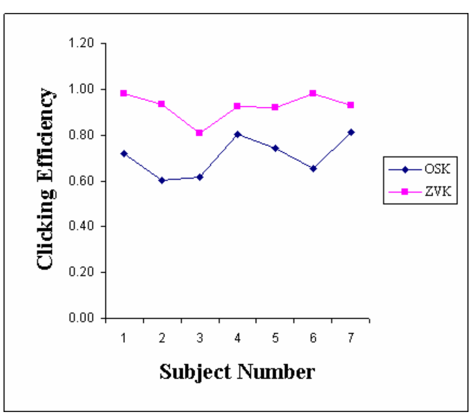


Figure 12. Average typing speed per subject

Table 3. Clicking efficiency results

Subj. #	C_I	OSK C_T	OSK C_{ef}	ZVK C_T	ZVK C_{ef}
1	5	8	0.63	5	1.00
	12	19	0.63	12	1.00
	24	28	0.86	25	0.96
	29	38	0.76	30	0.97
2	5	12	0.42	7	0.71
	12	19	0.63	12	1.00
	24	37	0.65	22	1.09
	29	41	0.71	31	0.94
3	5	6	0.83	6	0.83
	12	26	0.46	13	0.92
	24	44	0.55	42	0.57
	29	47	0.62	32	0.91
4	5	5	1.00	5	1.00
	12	14	0.86	12	1.00
	24	31	0.77	24	1.00
	29	50	0.58	41	0.71
5	5	6	0.83	5	1.00
	12	14	0.86	14	0.86
	24	39	0.62	26	0.92
	29	44	0.66	32	0.91
6	5	9	0.56	5	1.00
	12	19	0.63	12	1.00
	24	34	0.71	26	0.92
	29	40	0.73	29	1.00
7	5	6	0.83	5	1.00
	12	16	0.75	14	0.86
	24	27	0.89	24	1.00
	29	37	0.78	34	0.85

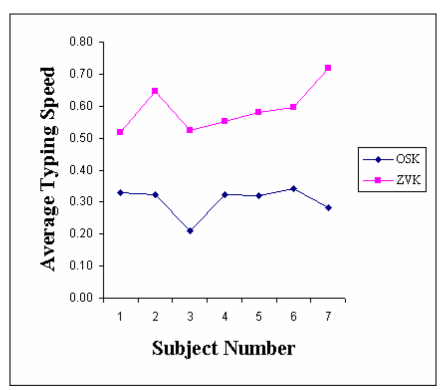


Figure 13. Average clicking efficiency per subject

The overall performance of the OSK vs. ZVK is summarized in Table 4 by taking the average across all subjects and computing the relative increment RI. The results reveal over 96% increase in typing speed and more than 30% increase in click efficiency when using the ZVK application, which demonstrates that the ZVK application is a more appropriate virtual keyboard application when employing an EGT-based HCI.

Table 4. Overall results for OSK vs. ZVK

	OSK (a)	ZVK (b)	$RI = \frac{(b-a)}{a} * 100\ (\%)$
$\bar{v}_S$	0.30	0.59	96.67
$\bar{C}_{ef}$	0.71	0.93	30.99

5. Conclusion

In interacting with computers, eye gaze systems certainly provide invaluable help to people with motor disabilities. However, they can make the interaction with on-screen keyboards a very frustrating experience due to the unavoidable mouse pointer jitter. The zoomable virtual keyboard presented in this study considerably increased typing speed and click efficiency (over 96% and 30% respectively) with respect to the Microsoft on-screen keyboard in experiments involving 7 subjects.

Although having the same area, the proposed keyboard outperformed Microsoft's on-screen keyboard in all experiments regarding typing speed and clicking accuracy. It is assumed that not only the magnification of the keys but also the smoothing algorithm played a decisive role. Magnification of the keys significantly counteracted the mouse jitter since it allowed the keys to be within focus for longer periods of time within the same level of jitter.

The results obtained proved that on-screen keyboard accessibility in remote eye gaze tracking systems can be highly enhanced without affecting the remaining area of the screen.

6. Acknowledgments

The authors are grateful for the support provided by the National Science Foundation under grants CNS-0426125, HRD-0317692, CNS-0540592, and the NSF graduate fellowship for Dr. Anaelis Sesin. Additional infrastructure support from NSF grants CNS-0520811 and IIS-0308155 were very helpful in complementing this study.

7. References

[1] Adjouadi M, Sesin A, Ayala M, Cabrerizo M. Remote eye gaze tracking system as a computer interface for persons with severe motor disability. Computers Helping People with Special Needs. Lecture Notes in Computer Science 3118: 761-769 2004.

[2] Hong P, Huang T. Natural Mouse: A novel human computer interface. IEEE International Conference on Image Processing, Oct. 24-28, 1999 Kobe, Japan.

[3] Hutchinson TE, White Jr. KP, Reichert KC, Frey LA. Human-computer interaction using eye gaze input. IEEE Trans. on Systems, Man, and Cybernetics, 19, Nov/Dec 1989, 1527-1533.

[4] Barreto A, Scargle S, and Adjouadi M. A real-time assistive computer interface for users with motor disabilities. ACM SIGCAPH Computers and the Physically Handicapped, Issue 64, June pp. 6-16, 1999.

[5] Barreto A, Scargle S, and Adjouadi M. A practical EGM-based human-computer interface for users with motor disabilities. Journal of Rehabilitation Research and Development, Vol. 37, No. 1, pp. 53-63, Jan-Feb 2000.

[6] Chin CA, Barreto A, and Alonso M. Electromyogram-based cursor control system for users with motor disabilities. Lecture Notes on Computer Science, K. Miesenberger et al. (Eds.ICCHP-2006, LNCS 4061, pp. 905-912, 2006.

[7] Sesin A., Ayala M., Cabrerizo M., Adjouadi M. "Jitter Reduction in Eye Gaze Tracking System and Conception of a Metric for Performance Evaluation", WSEAS Trans. on Computers, Vol. 3 (5), pp. 1268-1273, Nov. 2004.

[8] Adjouadi M., Sesin A., Ayala M., Cabrerizo M., "Remote Eye Gaze Tracking System as a Computer Interface for Persons with Severe Motor Disability", LNCS 3118, pp. 761-769, K. Miesenberger et al., Eds., Springer-Verlag 2004.

[9] Sesin A., Adjouadi M., Ayala M. and Cabrerizo M., "Configurable, Multimodal Human-Computer Interface System and Method", US Patent Application No. 20070011609.

[10] ISCAN INC., Raw Eye Movement Data Acquisition Software, Instruction Manual.

[11] Guestrin E.D., Eizenman M. General theory of remote gaze estimation using the pupil center and corneal reflections. IEEE Trans. on Biomedical Engineering; 2006. Vol. 53 (6), p. 1124–1133.

[12] Noyes, J. QWERTY-The Immortal Keyboard. Computing & Control Engineering Journal. Vol.9,No.3, 1998, pp.117-122