KeySteer : Multidirectional Pointing Input Using A Mechanical Keyboard

Byungjoo Lee
Seoul National University
Daehak-dong Gwanak-gu,
Seoul 151-742, Korea
bjlee@snu.ac.kr

Haesun Park
Seoul National University
Daehak-dong Gwanak-gu,
Seoul 151-742, Korea
gotjs514@snu.ac.kr

Hyunwoo Bang
Seoul National University
Daehak-dong Gwanak-gu,
Seoul 151-742, Korea
savoy@snu.ac.kr

CHI'13, April 27 – May 2, 2013, Paris, France.

Abstract

Here, we proposed a novel method to create multi directional and multi-level pointing input using only a mechanical keyboard. The relative position vectors among pressed keys are calculated into an input vector. The pointing performance of this method was evaluated based on the ISO 9241-9 recommendations. We found that the performance of the method was sufficient to create moderate pointing input as the measurement of the throughput was close to that of a touchpad, which was measured in past studies also using the ISO 9241-9 recommendations. Finally, we implemented two applications to show the other possibilities of the proposed method.

Author Keywords

Pointing input; ISO 9241-9; Fitts' law; Keyboard; Touchpad;

ACM Classification Keywords

H.5.2. User Interfaces: Input devices and strategies (e.g., mouse, touchscreen)

General Terms

Algorithms; Human Factors; Performance; Measurement

Introduction

The keyboard, mouse, trackball, touch screen, touch pad and pointing stick are all typical input devices for personal computers, portable computers and mobile devices. Among these examples, the keyboard, a text-input device, is an irreplaceable device. There are two types of keyboards: the mechanical and the touch-screen types. The mechanical version utilizes arrayed physical switches to type the desired text, while the touch-screen keyboard creates a keyboard image on a screen and allows typing by touching this image with the hand or with some other device. Although a touch-screen keyboard enables the user to perform auxiliary pointing input actions such as selecting icons or moving the cursor; however, this type of keyboard fails to fulfill its intrinsic role as an text-input device because it yields a high level of input error and, even when users become adapted to it, can be operated at a much slower speed than a mechanical keyboard [1]. Thus, touch-screen keyboards are rarely applied to personal and portable computers, except for the devices that are necessarily small, such as handheld mobile devices.

Other types of input devices apart from a keyboard can also facilitate pointing input in circumstances in which the graphical user interface is typically dominant with the operating system. Typical examples are the mouse, trackball, touch screen and pointing stick; these devices detect the user's desired movement and facilitate pointing input. When conducting various tasks on a GUI-based operating system, the user most often uses a mechanical keyboard and a pointing input device such as a mouse alternatively. In such a situation, there is some inconvenience because the user should continually move a hand from keyboard to the pointing input device and back. In a task primarily involving text, although such cases requiring a high level of accuracy are not many, the user should move a hand to the mouse for small and simple movements [2, 3]. This frequently causes fatigue on one's hand or fingers [7].

Related Work

The pointing stick is known to resolve the inconvenience of constant movement by the hands [7]. Nevertheless, although the inconvenience can be mitigated, this device leads to a high level of fatigue in the user's forearm [7], as the user should still apply force with a certain finger constantly. Also, according to one study [4], the current trend is that the usage of the pointing stick has declined because it is impossible to conduct precise movement compared to a track ball or a track pad. Moreover, this device requires zeroing periodically due to the aging of the sensor. Recently, an attempt was made to overcome the inconvenience by means of a vision-based system, with one such system developed that tracked the user's hands on the keyboard as a means of pointing input [4]. However, this also lead to several problems, such as lower DPI due to the limited image resolution, a high cost, and convoluted algorithms for image post-processing, all of which caused a reduction in the processing speed. These products are also sensitive to the surrounding environment [4]. On the other hand, a study changed the normal surface of keyboard into a touch-sensitive display [3]. While they succeeded to show intriguing possibilities of using keyboard for the graphical manipulation, they did not focused on conducting typical pointing task. In addition, their hardware is seemed to be quite complicated and fragile to endure the repetitive force applied during typing task.

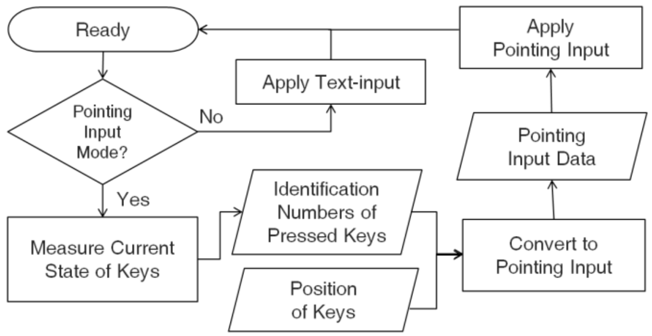

Figure 1. Flow diagram of the system

Additionally, there is a pointing input section on a typical keyboard. It is capable of four directional pointing inputs: up, down, left and right. Nevertheless, the possible directions of input are limited and the gain is not adaptable during a movement. Lastly, the pointing inputs are not conducted in the text input area of the keyboard, which therefore involves the overuse of hand movement, as mentioned above.

This study aims to develop a system that is able to be used as a multidirectional, multi-level pointing input device in the form of a mechanical keyboard, which is indispensable for text input. Consequentially, we expect to minimize the users' hand movements when alternating between text and simple pointing input. Without other devices attached to a computer, we made it possible with this system to use the position vectors of pressed keys. The feasibility and the performance of the system were evaluated by Fitts' law and two more applications that are appropriate for this system are presented.

The Principle of The System Operation

A flow diagram of the system is depicted in figure 1. There are two separated modes in this system and user can switch between those modes using a hotkey or a specially reserved button on the keyboard. If the user selects the mode as the typical text-input mode, the keyboard receives a command and performs as a normal text-input device. Once the pointing mode is selected, however, the system detects which keys are currently pressed by the user and then saves the identification number of the pressed keys. Then the position information of the pressed keys is referred from the previously obtained figure data of the keyboard and is used to yield pointing input with respect to a reference point. The figurative data of keyboard obtained beforehand is comprised of the relative coordinate information of each key; this data for other keyboards with different form factors should be measured and saved in advance. However, typical shape of keyboard is not that various, so once the data is measured, it could be used in other keyboard in most situations. We obtained the relative coordinate data by taking a picture of the keyboard from above and marking the center of each key.

Estimation of Graphic Input

The pointing input is estimated by the vector sum of the pressed keys from a reference point. The reference point is located between the 'U' key and the 'J' key, as depicted in figure 2. The location of the reference point is slightly shifted to the left from the center of the keyboard considering the base location of the user's hand when using the keyboard.

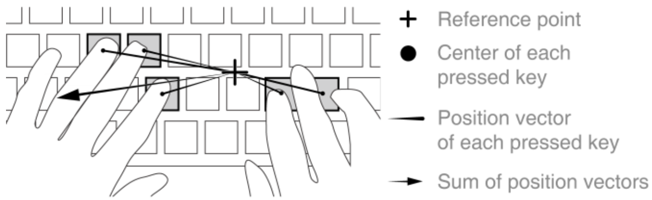

Figure 2. Estimation of user's directional intention

We assume the position vector of the i-th pressed key as $\mathbf{C}_i$ as well as that of the reference point **P**. In addition, n is the number of the pressed keys at the current time and *G* is a constant gain to adjust the sensitivity of the method. In this study, authors examined the appropriate reactivity of the cursor and set G as 20, so the possible slowest speed of cursor becomes 3.35 cm/sec on the display. This value was maintained for all resting experiments. The input vector **v** can be then formulated as follows:

$$\mathbf{v} = \frac{(\mathbf{C}_1 - \mathbf{P}) + (\mathbf{C}_2 - \mathbf{P}) + \cdots + (\mathbf{C_n} - \mathbf{P})}{n} \times \frac{1}{G} = \frac{1}{nG} \sum_{i=1}^{n} (\mathbf{C_i} - \mathbf{P})$$

Then, this input vector is treated as a velocity vector to move a cursor from the current position within a time step.

Examining Performance By Fitts' Law

The pointing task, which is measured and predicted according to Fitts' law, is one of the most fundamental types of graphical input [5]. Fitts' law is a robust model of human psychomotor behavior developed in 1954 [8] for predicting human responses based on rapid targeted movement. It is widely applied to evaluate the performance of pointing devices [11]. There have been several formulations of indexes of difficulty in Fitts' law. We used Shannon's formula, which provides a good fit to the data [11].

$$ID = \log_2\left(\frac{D}{W} + 1\right)$$

$$MT = a + b \times ID$$

MT is the movement completion time. ID is the index of difficulty, whose unit is bits. It is calculated from D, the distance from the target, and W, the width of the target both defined in pixels. Properties 'a' and 'b' are estimated by fitting the formula into the empirical data. The throughput is defined as a reciprocal number of b in units of bit/sec.

Participants

Eight paid participants were recruited from a local university campus. The participants' ages ranged from 20 to 28 years (M = 24 years, SD = 2.62 years). All participants reported using a computer on a daily basis and their usage time per day ranged from 1 to 8 hours (M = 4.33 hours, SD = 1.94 hours). All participants have normal usability of their hands, and none had experienced a system or device similar to that used in the experiment.

Apparatus

A mechanical keyboard, model number RK713A from HP, served as the input device for the task. The task was performed on a 3.30 GHz desktop computer with a 23-inch LED monitor. The resolution of the monitor was 1920 x 1080. The experimental system was built with Java, with the cursor being a standard arrowhead pointing to the upper left. Although we used a wireless keyboard for this experiment, no noticeable lag was found during the control. And generally, mechanical keyboards limit the number of keys that one can simultaneously press. In our case, the limit is ranged from 3 to 5 and while it was found to be sufficient for simple pointing tasks, this could reduce the performance for more complex tasks.

Task

We modified the multidirectional pointing task in the ISO 9241-9 standard [9]. The task was to point between a red and a blue circle that were simultaneously displayed on a black background. The participants were asked to move the cursor inside the red starting circle and to push the spacebar to begin a trial. Then, immediately after the red circle disappeared, the participants were required to move the cursor inside the blue target circle and push the spacebar to end the current trial. The two circles were located symmetrically about the center of the screen. The width of the target, the distance to the target and angle of approach were varied, with 3 (15,25 and 35 pixels), 2 (400 and 800 pixels) and 8 (0, $\pm\pi/4$, $\pm\pi/2$, $\pm3\pi/4$ and π radians) levels, respectively.

Design and Procedure

Participants were seated such that they were aligned with the midline of the computer screen and were free to utilize the entire workspace to accommodate their various control strategies. Before the experiment, they completed a pretest questionnaire about their age, gender and computer usage information. The participants were instructed to point to the blue target after clicking the red target as quickly as possible.

The experiment was a 3×2×8 within-subjects design with a repeated-measures ANOVA. An alpha level of .05 was used for all tests. The experiment consisted of five blocks, resulting in a total of 240 trials for each of the participants. Before starting the trial, one practice block was given. Between two separated blocks, we gave a two-minute break to all participants. For each block, the independent variables were randomly presented to the participants.

Results and Discussion

We considered a learning effect that showed a significant difference in the trial completion time for the block, $F(4, 28) = 6.80$; $p = .001$ (see figure 3(a)). Helmert contrast showed that the effect becomes insignificant ($p = .168$) after the second block. Therefore, we excluded the first block and averaged all subsequent blocks to obtain the throughput of the method. The resulting throughput of the method was 0.95 bit/sec(see figure 3(b)), and the correlation of the fitting was 0.99, the mean of error rate was 6.1% (SD = 1.61 %), which was comparable to the results of past studies [11].

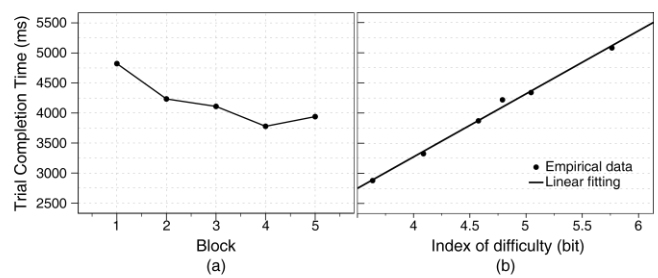

Figure 3. (a) Learning effect and (b) Empirical data fitted to Fitts' law model

Although the performance of the system is not as high as when using a mouse (3~4 bit/sec) [11], the performance of the system was comparable to the throughput of the touchpad as obtained in earlier research (0.99~1.43 bit/sec) [10], indicating that this method is sufficient to realize simple pointing and moving input actions, such as page scrolling, icon selecting and clicking without moving the hand to another device while the user is engaged in a text-dominant task. Another interesting study reported a large number of sub-movements associated with the touchpad [6] resulting from the excessive amount of scrolling action. However, our method does not involve a large amount of hand movement, making it efficient to achieve appropriate throughput.

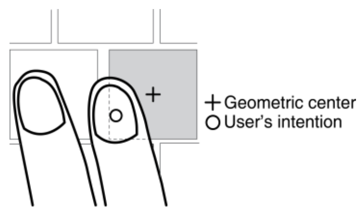

Figure 4. Calibrating user's intention based on human factor

Another advantage of this system compared to a vision or touchscreen system is that it is inexpensive to implement. This may be useful when there is no need for precise control compared to the complexity required to achieve it.

However, this system can still be improved in that it has a high level of error when undertaking minute control during the final stage of a pointing task (the correction phase), as the key arrangement on the keyboard is slightly tilted for typing. Therefore, complete horizontal or vertical movement control is laborious. Moreover, the number of available directions is reduced near the reference point. To solve this problem, tilting the coordinates of keys can be applied, or we can reverse the mapping by assigning smaller input gains for more distant keys so as to make more directions available for users.

Also, to make this system more precise, we can use time information of key-pressing events, such as an approximating the velocity of the hand movement to obtain more precise control of intended directions. In addition, based on the human factor related to the use of a keyboard, we can use different representatives for each key other than merely their center points (see figure 4)

Finally, we suggest two different applications that shows the other possibilities of this method, as discussed below.

Applications

Steering Input for Racing Game

Normally, when people play a computer racing game, they use directional keys on the keyboard in order to provide directional input and to accelerate the virtual car. However, the possible inputs are limited when using directional keys as they provide only simple switching actions. With this system, it is possible to make a racing game more realistic, as there are much more possible steps in each of directions, as in actual driving. One racing game (see figure 5) was implemented to test the feasibility of this application. The game has been developed with Unity3D while a Java host program send input vector into the game object which is, in this case, a car. Left/Right and Front/Rear components of input vector with respect to the reference point are mapped to the actions of steering and acceleration, respectively. After a simple user test, we found that players were more involved in the game when played via our method.

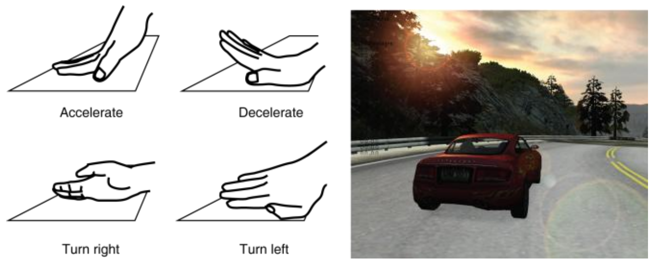

Figure 5. A racing game controlled with our method

Gesture Recognition

Normally, human gestures are measured using a vision-based system. In this research, gestures generated by hand movements are recognized by this input system only when using a mechanical keyboard. When the user sweeps one's hand up, down, left or right on the keyboard, the generated direction is estimated with the coordinate and the time data of the pressed key while sweeping. With a Java application, we conducted a simple experiment to convert a web-page back and forth, showing that typing a new website address can be done by means of a page conversion action easily. A past study also implemented a simple sweeping gesture using a mechanical keyboard [2]. They used a small microphone built in a laptop computer to analyze the sound signal when user lightly sweeping over the keyboard surface. While their method had realized simple gesture capability at a low cost, the robustness of the classifier was not guaranteed and the initial training was needed at the first trial to make it difficult to be applied to the practical applications. Besides, our method does not need more equipments except a mechanical keyboard and it does not need a classifier to be trained, so user can immediately start controlling cursor while still maintaining sufficient robustness.

Conclusion

This research aims to develop a system, with a graphical user interface, that allows a user to create pointing input only with a keyboard. The pointing input is estimated through the vector sum of the pressed keys with respect to the reference point. The performance of the system was evaluated according to Fitts' law and the result showed that the system has throughput comparable to that of a touchpad. Several applications that are able to present the characteristics of the system were also proposed. A racing game, which can have a wide range of directional and acceleration inputs, was played with this system, and a program which shows that the system can be utilized for gesture recognition was realized. This shows that the system has more potential for application than merely as a pointing input device.

Future Work

In this study, we have focused only on the feasibility of the system as a pointing input device. Therefore, future experiment which guarantees that the system mitigate the user's hand movement and this yields high level of comfort compared to conventional keyboard-mouse based input system should be followed.

References

[1] Andrew, S., Doreen, R., Janet, S., Rob, C., and Ben, S. Investigating touchscreen typing: the effect of keyboard size on typing speed. Behaviour & Information Technology 12, 1 (1993), 17-22.

[2] Kato, J., Daisuke, S., and Takeo, I. Surfboard: Keyboard with Microphone as a Low-cost Interactive Surface. In Proc. UIST 2010, ACM Press (2010), 387-388.

[3] Florian, B., Hans, G., and Nicolas, V. Touch-Display Keyboards: Transforming Keyboards into Interactive Surfaces. In Proc. CHI 2010, ACM Press (2010), 1145-1154.

[4] Andrew, W. and Edward, C. FlowMouse: A Computer Vision-Based Pointing and Gesture Input Device. Lecture Notes in Computer Science 3585, 1 (2005), 565-578.

[5] Casiez, G., Vogel, D., Balakrishnan, R., and Cockburn, A. The impact of control-display gain on user performance in pointing tasks. Human-Computer Interaction 23, 3 (2008), 215-250.

[6] Dillen, H., Philips, J. G., and Meehan, J. W. Kinematic analysis of cursor trajectories controlled with a touchpad. International Journal of Human-Computer Interaction 19, 2 (2005), 225-241.

[7] Elisabeth, F. and Mats O.E. Computer mouse or Trackpoint effects on muscular load and operator experience. Applied Ergonomics 28, 5 (1997), 347-354.

[8] Fitts, P. M. The information capacity of the human motor system in controlling amplitude of movement. Journal of Experimental Psychology 47, 6 (1954), 381-391.

[9] ISO/DIS. 9241-9 Ergonomic requirements for office work with visual display terminals (VDTs), Requirements for non-keyboard input devices. International Standard, International Organization for Standardization, (2000).

[10] Mackenzie, I. S. and Aleks O. A. Comparison of three Selection Techniques for Touchpads. In Proc. CHI 1998, ACM Press (1998), 336-343.

[11] Soukoreff, R. W. and MacKenzie, I. S. Towards a standard for pointing device evaluation: Perspectives on 27 years of Fitts' law research in HCI. International Journal of Human-Computer Studies 61, 6 (2004), 751-789.