TBS^3: Two-Bar Single-Switch Scanning for Target Selection

Mathieu Raynal[1(✉)] and I. Scott MacKenzie[2]

[1] ELIPSE Team, IRIT Lab, University of Toulouse, Toulouse, France
mathieu.raynal@irit.fr
[2] Electrical Engineering and Computer Science, York University, Toronto, Canada
mack@yorku.ca

Abstract. We present a two-dimensional (2D) pointing technique for motor-impaired users who generally use single-input switch. The technique, called TBS^3 for "two-bar single-switch scanning", uses two bars and proceeds in two steps: moving a vertical bar then moving a horizontal bar. The user starts and stops each bar by activating the single input switch. Selection is made at the intersection of the two bars when the horizontal bar is stopped by the user. We present two variations of the technique. In the first ("one-way"), the bars move only in one direction, and return to their origin after each selection. In the second ("button"), the user controls the direction by a selecting an on-screen soft button before bar movements starts. The techniques were evaluated in a experiment with twelve participants using the 2D target-selection task described in ISO 9241-411. Due to the two-stage pointing process, the task was inherently slow with a mean movement time of 8.4 s per trial. The mean error rate was just under 10% with the one-way method (8.0%) more accurate than the button method (11.1%). Throughput was 0.31 bps overall with values of 0.26 bps for the one-way method and 0.36 bps for the button method.

Keywords: Motor disability · Pointing technique · Single-switch scanning · Assistive technologies

1 Introduction

Since the 1980s and the emergence of the Apple Macintosh, graphical user interface have become the standard interface on traditional computers. These interfaces use a pointing device to interact in applications, to point and select elements to access different functionalities. The mouse and the trackpad are the main pointing devices for desktop and laptop computers, respectively. However, users with severe motor impairments cannot operate these devices.

In this paper, we propose a pointing technique using single-switch input and inspired by an interaction technique used on virtual keyboards. Indeed, to enter text, motor-impaired users often use a soft keyboard combined with single-switch

K. Miesenberger et al. (Eds.): ICCHP-AAATE 2022, LNCS 13341, pp. 338–346, 2022.
https://doi.org/10.1007/978-3-031-08648-9_39

scanning – a so-called "scanning soft keyboard". The cursor moves automatically from zone and when the cursor is on the right zone, the user validates it through an input action.

We propose a new pointing technique based on the automatic scanning of two bars (one horizontal and one vertical) controlled by a single input switch. We describe the principle of our interaction technique, and the different possibilities of interaction afforded. To validate this technique, we performed an evaluation with pointing tasks following recommendations for the evaluation of pointing devices [3, 9]. We describe the methodology of our evaluation followed by analyses of the results and suggestions for further improvements and research.

2 Related Work

Alternative pointing devices have been proposed to provide access for motor-impaired users. The most common are those using eye-tracking [10] or head-tracking techniques [8]. However, these devices have drawbacks: eye-tracking systems require the user to fixate on the element they want to select. Other techniques based on EEG signals [6] or contraction signals of voluntary muscles using EMG [5] have also been studied to bring WIMP interaction to the most severely disabled people. Alternatives using face tracking [4] or a keypad [1] have also been proposed.

3 TBS³: Two-Bar Single-Switch Scanning

3.1 Principle

Our implementation for single-switch scanning works with a vertical bar and a horizontal bar (see figures below). The bars move automatically over the width of the screen for the vertical bar and the height for the horizontal bar. The pointer is at the intersection of the two bars.

Pointing is done in two main steps: first, moving the vertical bar, then moving the horizontal bar. The user presses a single input switch to start the automatic movement of the vertical bar. Then, the user presses the same input switch again to stop the first bar. Stopping the vertical bar automatically starts the movement of the horizontal bar. Finally, the user stops this bar when desired by pressing the single input switch again. Stopping the second bar then generates a click at the coordinates of the intersection of the two bars. Along with this simple description of two-bar single-switch scanning, there are additional possibilities to enhance interaction.

3.2 "One-Way" Version

We implemented a first version called "one-way" where the vertical bar starts from the left of the screen and moves from left to right. The horizontal bar starts at the top of the screen and moves down.

Two options were considered for the start of a new pointing task. The first is to start the bars from the position of the last pointing, while the second option is to restart the bars from the origin.

When a bar reaches the end of the screen, we offer two modes: "go/return" mode wherein the bar continues in the opposite direction and "circular" mode wherein the bar starts again from the origin of the screen.

3.3 "Button" Version

In the "one-way" version, the user cannot choose the direction of bar movement. If the selected element is close to the bars but in the opposite direction to movement, the user must wait until the bar travels across the entire screen before returning to the element. Similarly, if the element is in the second part of the screen, the bars must travel a long distance before arriving at the element.

To overcome these problems, before the start of each scan, we implemented a "button" version which offers two soft buttons displayed in the center of the bar to allow the user to choose the movement direction (see Fig. 1). Button focus automatically alternates between the two buttons until the user makes a selection to set the direction of movement. This done by pressing the single input switch when the desired direction button is highlighted. The bars start from the center of the screen. Thus, the user can choose the direction of movement of the bars. Although an extra scanning step is required, this makes it possible to divide-by-two the distance required to reach the desired element.

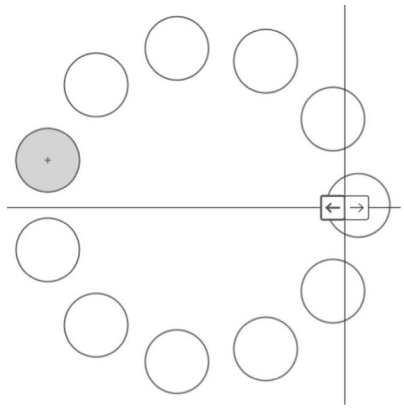

Fig. 1. "Button" version

4 Method

The goal of our user study was to empirically evaluate TBS[3], our two-bar single-switch scanning technique, and compare the two implementation methods described above ("one-way", "button"). The "one-way" method was combined

with "origin" starting mode and the "button" method was combined with the "last pointing" starting mode. A 2D Fitts' law task conforming to ISO 9241-411 was used with three movement amplitudes combined with three target widths. Our hypothesis is that, even if the "button" version requires one more input actions, moving the bars in both directions will yield better results.

4.1 Participants

We recruited 12 participants from the first author's university campus. Nine were male, three were female. Ages ranged from 18 to 43 yrs.

4.2 Apparatus

The experiment was conducted on a Dell Latitude 5490 laptop with a resolution of 1024 × 768 pixels. The experiment tasks were presented using a modified version of GoFitts[1], a Java application implementing the 2D Fitts' law task (see Fig. 2). GoFitts includes additional utilities such as FittsTrace which plots the cursor trace data captured during trials.

The 2D task presented 11 targets per sequence, combining three movement amplitudes (100, 200, 400 pixels) with three target widths (20, 40, 80 pixels). Amplitude and width were included to ensure the conditions covered a range of task difficulties. The result is nine sequences for each test condition with IDs ranging from $\log_2(\frac{100}{80} + 1) = 1.17$ bits to $\log_2(\frac{400}{20} + 1) = 4.39$ bits.

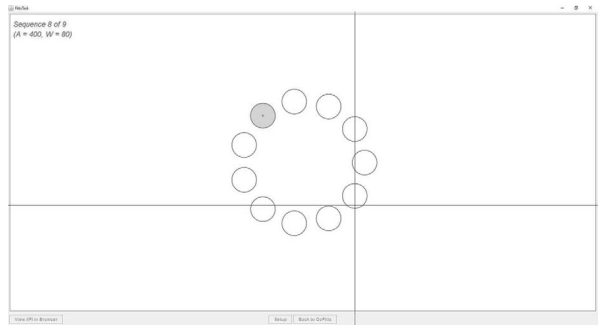

Fig. 2. 2D Fitts' law task in the modified GoFitts application with an amplitude of 400 pixels and a target width of 80 pixels

4.3 Procedure

The software and experiment were explained and demonstrated, following which testing began. Testing took a little under one hour per participant. Before each method, participants could test the method as much as they wanted (they generally performed 2 or 3 target selections).

[1] https://www.yorku.ca/mack/FittsLawSoftware/.

4.4 Design

The experiment was a $2 \times 3 \times 3$ within-subjects design. The independent variables and levels were Pointing Method ("one way", "button"), Amplitude (100, 200, 400 pixels), and Width (20, 40, 80 pixels).

For each sequence, 11 targets appeared. The dependent variables were throughput (bps), movement time (ms), error rate (%). There were two groups for counterbalancing, one starting with the "one way" and the other starting with "button". The total number of trials was 2376 ($= 2 \times 3 \times 3 \times 11 \times 12$).

5 Results and Discussion

5.1 Throughput

The main performance measure in ISO 9241-411 is throughput (TP) [3,9], measured in bits/second (bps). TP is calculated over a sequence of trials as the ID-MT ratio: $TP = ID_e/MT$. The standard specifies calculating throughput using the effective index of difficulty (ID_e). The calculation includes an adjustment for accuracy to reflect the spatial variability in responses: $ID_e = \log_2(A_e/W_e + 1)$ with $W_e = 4.133 \times SD_x$.

Selecting with TBS[3] had a mean throughput of 0.306 bps. By pointing method, the means were 0.256 bps and 0.356 bps for one-way and button, respectively. This represents a 39% performance advantage for button. The effect of selection method on throughput was statistically significant ($F_{1,10} = 19.89, p = .0012$).

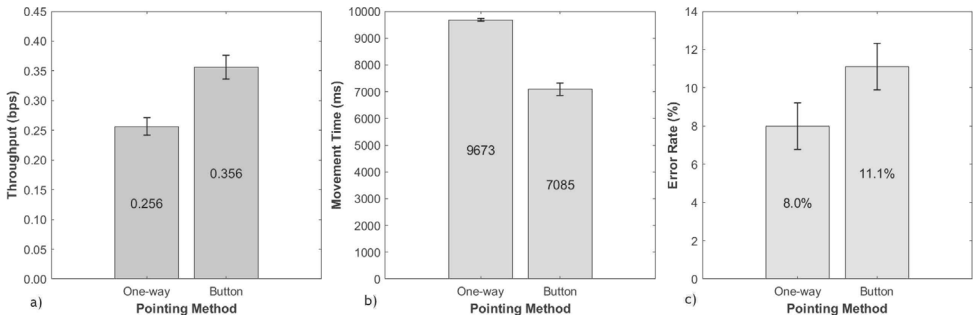

Fig. 3. a) Throughput (bps) by pointing method; b) Movement time by pointing method; c) Error rate by pointing method. Error bars show ± 1 SE

The throughput values are low compared to a mouse [9]. But, we are here in the context of motor impairment, and these values are comparable with those reported in similar work [1,2]. On the other hand, the participants did not control the speed of movement of the two bars. The speed of the bars had been empirically fixed at 3 pixels per 20 ms. This movement speed could be adjusted to each user according to their abilities – a topic for future study.

5.2 Movement Time and Error Rate

The speed and accuracy of the techniques are combined in the throughput calculation, but it is interesting to examine both separately. Indeed, we can see in Fig. 3b that the speed is dependent on the technique in the same way as the throughput: the participants were faster with the button version than with the one way version (7085 ms and 9673 ms, respectively for a reduction of 27%), and this in a significant way ($F_{1,10} = 36.50, p = .0001$).

On the other hand, the error rate shows an opposite trend: Participants were more precise with the one-way method than with button method (8% and 11.1%, respectively), but this difference was not statistically significant. Several errors were observed with the button version when a bar was positioned correctly after the previous click. Indeed, the bars remain in the same position (whereas they are positioned at the origin with one-way) and users are forced to choose a direction of movement. If the bar was already well placed, it was necessary to click twice very quickly (or to let the bar go back and forth). Several participants made mistakes when trying to do two clicks very fast. Possible solutions include adding a third button to allow the user not to move the bar, or using a lower velocity for the initial movement of the bar.

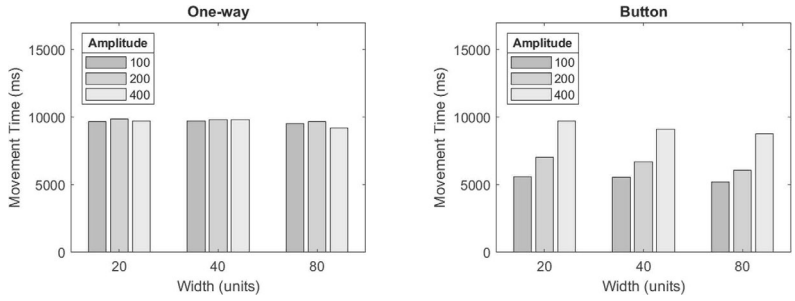

Fig. 4. Movement time by width and amplitude for the one-way (left) and button (right) methods

On the other hand, we can see in Fig. 4 (left) that width and amplitude have no effect on the movement time for the one way method. A Bonferroni multiple-comparisons test confirmed that none of the pairwise differences were statistically significant ($p > .05$). This is because the bars are always positioned at the origin after each click. Therefore, the distance actually traveled is always that between the upper left corner of the screen and the target. This effect is not found with the button version: The bars remain at the position of the last click; so, the distance between the targets influences the movement time to reach the target.

5.3 Cursor Trace Examples

Figure 5 (left) shows an example of pointer traces for the button method. The traces are straight horizontal or vertical lines, which reveal the functioning of the pointing method where the movements of the pointer are guided by successive rectilinear movements of the vertical then horizontal bar.

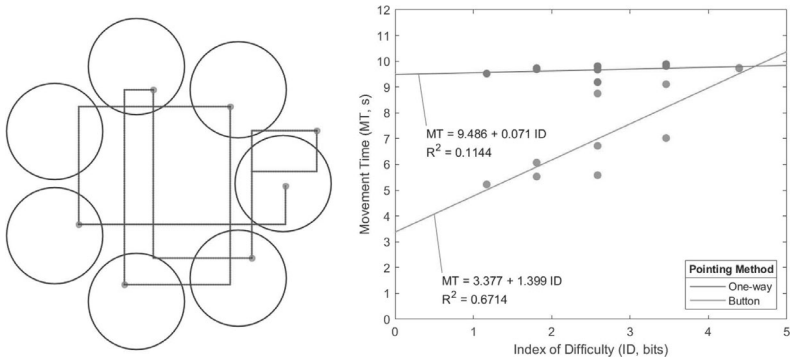

Fig. 5. At left, example pointer trace for button version; at right, Fitts' law models for one-way (blue line) and button (red line) methods (Color figure online)

5.4 Fitts' Law Models

To test for conformance to Fitts' law, we built linear regression models for each pointing method (Fig. 5, right). The one-way model confirms the analysis of movement times carried out in Sect. 5.2. Indeed, we can see that the different points representing the movement time are almost at the same height regardless of the index of difficulty. The slope is almost nil, which shows that the index of difficulty does not influence the movement time.

For the linear regression line of the button method, there are differences for the same index of difficulty. This is because the index of difficulty depends on width and amplitude. Thus the combinations 20/100, 40/200 and 80/400 give the same difficulty index. In the previous section, we saw that the movement time depends on the amplitude but not on the size of the targets. This is why we have here three distinct values for the same *ID*. This difference generates a lower value of the correlation coefficient.

6 Conclusion and Future Work

Single-switch scanning is designed to make pointing easier for users who have limited motor skills and cannot easily use pointing devices that require continuous movement. We propose in this article two methods. The first (one-way) always moves the bars in the same direction. With each click, the bars start again from the origin. This version is slower, but users make fewer errors. The

second (button) allows users to choose the direction of movement. Thus the bars are not repositioned after each click. This allows users to go faster from target to target.

Currently, the user can only perform one type of event when stopping the horizontal bar: a single click. In a future version, we will allow the user to choose the event to perform: "do nothing", "click", "double click", "press", "release", "right click". As for the direction buttons, options will be presented in a pie menu and will be accessible via scanning. Finally, another area of improvement will be the displacement of the pointer. Rather than having two bars, we will present a "software joystick", as described by Raynal et al. [7], where the user selects the direction after an automatic rotation of the "stick" around the axis. This will move the pointer directly in one direction, not necessarily horizontal or vertical.

References

1. Felzer, T., MacKenzie, I.S., Magee, J.: Comparison of two methods to control the mouse using a keypad. In: Miesenberger, K., Bühler, C., Penaz, P. (eds.) ICCHP 2016. LNCS, vol. 9759, pp. 511–518. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41267-2_72
2. Hassan, M., Magee, J., MacKenzie, I.S.: A Fitts' law evaluation of hands-free and hands-on input on a laptop computer. In: Antona, M., Stephanidis, C. (eds.) HCII 2019. LNCS, vol. 11573, pp. 234–249. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-23563-5_20
3. ISO: Evaluation methods for the design of physical input devices - ISO/TC 9241–411: 2012(E). Report Report Number ISO/TS 9241–411:2102(E), International Organisation for Standardisation (2012)
4. Magee, J., Felzer, T., MacKenzie, I.S.: Camera Mouse + ClickerAID: dwell vs. single-muscle click actuation in mouse-replacement interfaces. In: Antona, M., Stephanidis, C. (eds.) UAHCI 2015. LNCS, vol. 9175, pp. 74–84. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20678-3_8
5. Perez-Maldonado, C., Wexler, A.S., Joshi, S.S.: Two-dimensional cursor-to-target control from single muscle site sEMG signals. IEEE Trans. Neural Syst. Rehabil. Eng. **18**(2), 203–209 (2010)
6. Pinheiro, C.G., Naves, E.L., Pino, P., Losson, E., Andrade, A.O., Bourhis, G.: Alternative communication systems for people with severe motor disabilities: a survey. Biomed. Eng. Online **10**(1), 1–28 (2011)
7. Raynal, M., Gauffre, G., Bach, C., Schmitt, B., Dubois, E.: Tactile camera vs. tangible camera: taking advantage of small physical artefacts to navigate into large data collection. In: Proceedings of NordiCHI 2010, pp. 373–382. ACM, New York (2010)
8. Rodrigues, A.S., da Costa, V.K., Cardoso, R.C., Machado, M.B., Machado, M.B., Tavares, T.A.: Evaluation of a head-tracking pointing device for users with motor disabilities. In: Proceedings of PETRA 2017, pp. 156–162. ACM, New York (2017)

9. Soukoreff, R.W., MacKenzie, I.S.: Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. Int. J. Hum. Comput. Stud. **61**(6), 751–789 (2004)
10. Zhang, X., MacKenzie, I.S.: Evaluating eye tracking with ISO 9241 - Part 9. In: Jacko, J.A. (ed.) HCI 2007. LNCS, vol. 4552, pp. 779–788. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73110-8_85