A novel click-free interaction technique for large-screen interfaces

Takaomi Hisamatsu, Buntarou Shizuki, Shin Takahashi, Jiro Tanaka

Department of Computer Science
Graduate School of Systems and Information Engineering
University of Tsukuba
1-1-1 Ten-noudai, Tsukuba, Ibaraki 305-8573, Japan
e-mail {omi,shizuki,shin,jiro}@iplab.cs.tsukuba.ac.jp

Abstract We developed a novel click-free interface for laser-pointer-based systems with wall-sized screens. The technique includes two moving operations: "crossing," which uses four edges as a crossing target to execute a command, and "encircling," which selects objects by drawing a circle around them. This paper describes the proposed interface, its implementation, and an experiment conducted to test the technique.

1 Introduction

A laser pointer is a useful pointing device for presentations on wall-sized screens, such as computer-projected presentations. It allows a presenter to point anywhere on the large screen quickly, even when the screen is located far away. However, when the speaker needs to manipulate the PC, he or she must either physically press some keys, or use a commercial device such as a wireless mouse or remote control to manipulate the PC from a short distance. However, these devices are difficult to use to point to a target (e.g., mouse) or often have several buttons that can easily be confused (e.g., remote). Such complications are compounded when a user needs to use these devices while explaining a slide with the laser pointer.

Several proposed systems [4, 5, 8, 9] have attempted to address this problem using a laser pointer as an input device. However, all of these systems adopted a mouse-compatible interface where the mouse button is pressed while the laser dot is held for a certain amount of time on a specific spot on the screen display. Here, we call this technique the "holding technique." There are several problems with this method: it takes a relatively long time to execute, because the user must wait the specified amount of time to click anything; the click operation requires a user to maintain a fixed posture during the delay period, straining his or her body; and the laser dot vibrates, because of the user's natural hand tremor, making it difficult to interact with small objects on the screen display.

To solve these problems, we have designed and implemented a new technique using moving gestures. The system includes two moving operations, crossing and encircling.

Using the crossing operation, a speaker crosses the edges of the screen using a laser pointer to input and execute predefined commands. The encircling operation is used to select graphical user interface (GUI) objects on the screen. The speaker chooses an object, such as an icon, button, or thumbnail, by circling it. Basically, the system selects the GUI object nearest to the center of the circle drawn by the user.

2 Click-free interaction technique

Myers *et al*. [7] showed that it is difficult for a user to keep a laser pointer fixed on a specific area of a screen for a long time; the beam is unsteady, and users cannot turn the device on or off at will. This is a weakness of using a laser-pointer-based system in combination with the holding technique described above. Therefore, we incorporated the two moving operations, crossing and encircling, into our click-free interaction technique. Note that these techniques do not use holding, but moving gestures for the interaction, because it is a more natural and easy way to control a laser pointer. That is, drawing lines or circles is easier than keeping a laser dot fixed on a specific location for a period of time.

2.1 Crossing operation

The crossing operation is used to execute commands, using goal crossing [1] as the fundamental interaction method. All the commands executed by the user are activated by goal crossings.

The system can detect movements of the laser dot outside the screen, and use these movements for interactions. This is a unique merit of this technique. We use the edge of the projected screen as the main goal of crossing, because the screen edges are long enough for the user to cross easily. There are three basic types of crossing: in-out, out-in, and in-out-in (Fig. 1 left). In-out is performed by moving the laser dot from the inside to the outside of the projected screen; out-in is the opposite movement; and in-out-in consists of two successive crossings (i.e., the user does not turn off the beam between crossings), inside to outside and then back inside the screen. We use combinations of these three basic operations, and the four sides of the display screen to identify different commands.

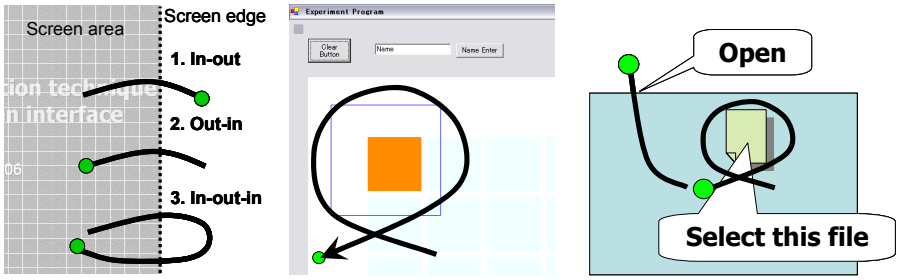

Figure 1. (left) Three types of crossing: (1) in-out, (2) out-in, and (3) in-out-in
(center) Target Encircling: a colored target, laser dot curve, and fine blue-line marker
(right) Encircling and crossing to select and execute

2.2 Encircling operation

The encircling operation is used to select GUI objects. Basically, the system selects the object or objects (e.g., icon, button, thumbnail) nearest to the center of the circle drawn by the user. Using a laser pointer-based system, a user can draw circles easily. Therefore, this method may be easier for selecting GUI objects than the holding technique.

In addition to objects, a user can select an area of the screen, for example, to zoom in on the map viewer application. In this case, the user's circle provides position and zooming ratio information to the system. This is a unique merit of the encircling operation.

Several pen-based systems [3, 6] use encircling selection, and have shown the usefulness of the operation. We incorporated this method into the laser-pointer-based interaction, because the two systems are similar. For example, both a laser pointer and a pen can be used to point to something directly on a screen display. However, a laser pointer cannot be used as precisely as a pen, since a user touches the screen surface with a pen, but operates a laser pointer from a distance. Therefore, we adapted the pen-based method to the laser-pointer-based system.

2.3 Encircling and Crossing

The user can select a GUI object and execute a command by combining the two techniques. Moreover, the system provides combinations of executable commands for various applications. For example, in a file manager, if the top edge is defined as Open File, when the user encircles a file icon and crosses the top edge, successively, the file will be opened (Fig. 1 right). One can select a target and execute a command quickly and easily in one stroke.

3 Prototype system remote pointer

3.1 System overview

The system consists of a PC, a USB 2.0 camera, and a video projector. We use a green laser pointer (wavelength 532 nm, divergence 1.2 m rad, output power about 5 mW). The projector displays the PC's screen, and the camera is fixed on the projected screen display. Figure 2 shows the setting of the prototype system.

Before using the system, one must designate four points for calibration. Usually these points make a four-sided figure, but not a rectangle. The system then converts the polygon (on the coordinate system of the image) to a rectangle (on the coordinate system of the PC display) with the same aspect ratio as the projected display. We applied the projector–camera homography method [9] to our system.

After calibration, the camera produces 640- x 480-pixel monochrome images at 33 frames/sec. The laser dot tracker runs on the PC. The tracker scans to find brighter pixels than a given threshold in each frame. If the tracker finds brighter pixels, the system places a mouse cursor at that position. If the tracker finds no such point, it reports that no laser dot has been found. The mouse cursor moves as the laser dot moves.

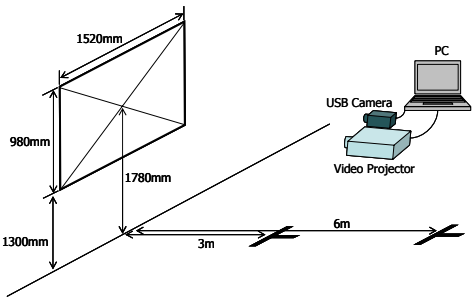

Figure 2. Setting of the prototype system and the experiment

3.2 Detection of Crossing

If the tracker finds a laser dot outside the projected area in a certain frame, and then finds the dot inside the projected area in the next frame, there is an out-in crossing. There is an in-out crossing when the laser dot exists first inside and then outside the projected area. However, surfaces around the actual screen are not usually flat. In contrast, the flatness of the screen is guaranteed. Therefore, in our approach, the tracker locates laser dots only on the screen.

To distinguish between traveling and turning on/off, the tracker performs the inference illustrated in Figure 3. First, it calculates an outgoing vector when it loses the laser dot and an incoming vector when it finds it. The incoming vector is derived

from the first two dots. Then, it makes another vector, beginning at the first dot. Its direction and size are c times the incoming vector, where c is a predefined constant. The tracker infers that the laser beam traveled across the edge if the vector crosses the edge. Otherwise, it infers that the laser beam was turned on. The same inference is performed for the outgoing vector. In our current implementation, we use $c = 4$, which is derived empirically.

The tracker also uses a threshold to distinguish between an in–out–in crossing and an in–out crossing followed by an out–in crossing. If two crossings are detected within the threshold, it is considered an in–out–in. The same value is used to avoid erroneous failure to find the laser dot; that is, the tracker does not consider the beam turned off until it continuously finds no laser dot for the threshold period. The threshold was 500 ms in our implementation.

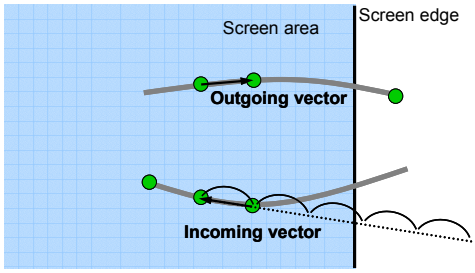

Figure 3. Detection of crossing

3.3 Detection of Encircling

The system detects the encircling gesture as follows. First, it examines whether the laser dot trajectory intersects with itself. If so, then the system calculates the area encircled. If this area is larger than a predetermined value, then the system detects that something has been encircled. The predetermined value is 2000 pixels on the display coordinate system. Next, the system calculates the center of the encircled area, and displays a blue square at that center. This marker has an area proportional to the encircled area. Figure 1 (center) shows a colored object, a laser dot curve, and the blue marker square. When the square includes the center of some GUI object, the object is detected and selected.

4 Usability of the proposed technique

We conducted some informal tests on the crossing operation using two applications. We also conducted an experiment on the encircling operation, to examine the time and proficiency with which a user could successfully encircle a target.

4.1 Informal tests on the crossing operation

We implemented two prototype applications using the crossing operation: a slideshow application for computer presentations, and a map viewer application.

In the slideshow, the most basic operations were next-page and previous-page. We used out–in (2) crossing for these operations: if the laser dot crossed the right edge of the projection, then a next-page command was executed, and if the dot crossed the left edge then a previous-page command was executed. If the dot moved from inside the screen perimeter to outside and then back inside, continuously, the system detected the movements as a sequence of next/prev-page commands. Because such a movement can be replaced with a circular movement at the side of the screen, the user can easily move through pages continuously. Another interesting operation is to switch between presentation and thumbnail modes, performed by executing a circular movement of the laser dot at a corner of the screen, i.e., an in–out–in (3) operation consisting of crossing from the inside to the outside of the right edge of the projection, and then crossing from the outside to the inside across the bottom edge.

In the map viewer application, the following operations are available: go right, left, up, down, zoom in, zoom out, and rotate. These commands are made from three basic movements. Move right and other directional commands are made from the out–in crossing; zoom-in from right-handed in–out–in crossings; zoom-out from left-handed in–out–in crossings; and rotate from out–in crossings, when the laser dot enters with an angle of less than 45 degrees.

We conducted some informal tests, allowing volunteers to use these two applications. Users were able to successfully manipulate these applications after a little practice. Their feedback included that these applications are useful and easy to use; they wanted to use the slideshow application in their presentations; and that it is easy to understand how to use the edges to execute commands. Although the tests were informal, we are certain that the crossing operation is useful.

4.2 Encircling operation experiment

Setting

Figure 2 shows the geometric setting, including the size and position of the screen and the position of the user. Users stood at a fixed distance from the display, which was projected onto the screen by a video projector. The resolution of the displayed screen was 1280 pixels width x 1024 pixels height; the screen scale was 1.5 m width and 1 m height. The experiment was performed with 15 users, none of whom had used a laser pointer device system before. All participants were male, 21–36 years old, and with no disabilities in their hands, fingers, eyes, color vision, or hearing.

Methods

A matrix of squares virtually represented GUI objects typical to any system, such as icons, menus, buttons, and thumbnails. Figure 4 shows the screen of the

experimental program. The matrix consisted of 50 (5 rows x 10 columns) light blue squares. Each square was 90 pixels, and there were 10 pixels between squares.

In the experiment, the target, one of the arranged squares, was turned yellow or orange. Users were asked to encircle the target using a laser pointer. If the user succeeded, the system produced a beep and then showed the next colored target. Each user completed five sets, each comprising 20 possible target squares. The experiments were performed in two standing positions, at distances of 3 and 6 m, called the near and far positions, respectively. Each participant completed five sets from each position.

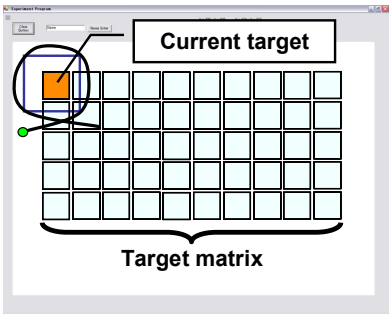

Figure 4. The screen of the experimental program

4.3 Results

Figures 5 and 6 show the average time for users to encircle a target(s), and the error count per set. The average time was 1.4–1.5 s in the fifth set. The most proficient users averaged 0.9–1.0 s. It is interesting to note that the average time was faster in the far position, likely because the hand movements for encircling gestures are smaller from the far position; this makes it more difficult and therefore results in more errors.

Users made, on average, four to five errors in encircling a target, per set. Half of the users made less than three errors. The most proficient users made no errors or only one error in the fifth set. The average success rate was about 60–80% from the far position, and almost 80% from the near position. About half of the users had a 90% success rate.

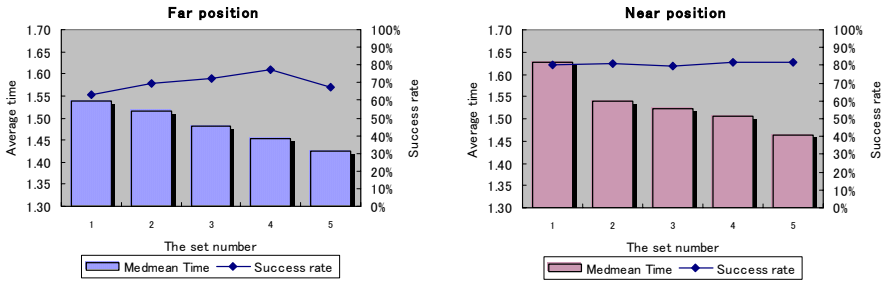

Fig. 5. (Right) Average required time to encircle and success rate at the far position
Fig. 6.(left) Average required time to encircle and success rate at the near position

4.4 Classified errors

We classified errors into three types (see Fig. 7 for examples): user did not select any square (52%); user selected a square other than the target (17%); or user encircled more than one square, including the target (31%). The most common error was failing to select any square.

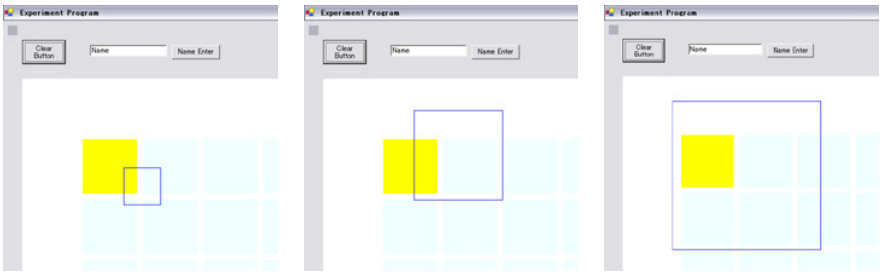

Figure 7. Classified errors: (1) nothing encircled, (2) different target encircled, and (3) multiple targets encircled

4.5 Discussion regarding classified errors

We analyzed the errors classified in section 4.4. In the first case, when a user encircles empty areas, then the system should execute nothing. We considered the second case more important, because this error strongly affects system usability. Therefore, it was necessary to reduce the incidence of this type of error. Based on our observations of users during the experiment, we concluded that it was easier to encircle a target using thin ovals, versus circles. Therefore, square size should be decreased and more blank space should be included on the right and left sides, keeping the top and bottom space. This would make encircling easier, because there would be enough space to encircle a target, and the user could encircle it by drawing a thin oval.

Regarding the third type of error, we believe it would be beneficial for the system to request a user to reselect one object when multiple objects are selected, instead of replying with an error message. We plan to improve the system so that when a user encircles two to four objects, then the system zooms in and arranges them around the oval area, allowing the user to reselect one of them, as it is easy for a laser-pointer user to select from four directions (north, south, east, and west). If there are more than four objects in the encircled area, then the system cancels the operation. The results of our experiments showed that users rarely selected five objects or more.

5 Conclusion

We developed a new interaction technique for directly manipulating applications in laser-pointer-based systems with a large screen. The technique uses two moving operations, crossing and encircling. The crossing operation uses the four edges of the screen as crossing targets, enabling users to give commands to the system. The encircling operation for selecting a target(s) proved to be easier than the holding technique, especially when users stood 3 m away from the screen (versus 6 m). This technique proved effective in terms of speed and success rate. The two proposed operations are effective for manipulating a PC using continuous strokes.

In the future, we plan to perform more experiments to improve the system and further refine it. For example, when the screen edge is divided and commands are defined, how many commands can each edge have? And how accurate is our arrangement of the target matrix and the reselecting method?

References

1. Accot, J. and Zhai, S. 2002. More than dotting the i's --- foundations for crossing-based interfaces. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves (Minneapolis, Minnesota, USA, April 20 - 25, 2002). CHI '02. ACM Press, New York, NY, 73-80.
2. Brown, M.S.; Wong, W.K.H., Laser pointer interaction for camera-registered multiprojector displays, Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on, Volume 1, Issue , 14-17 Sept. 2003 Page(s): I - 913-16 vol.1
3. Hinckley, K., Baudisch, P., Ramos, G., and Guimbretiere, F. 2005. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Portland, Oregon, USA, April 02 - 07, 2005). CHI '05. ACM Press, New York, NY, 451-460.
4. J.-Y. Oh, W. Stuerzlinger, Laser Pointers as Collaborative Pointing Devices, Graphics Interface 2002, AK Peters and CHCCS, pp. 141-149, May 2002.
5. Kirstein, C. (1998). Interaction with a Projection Screen Using a CameraTracked Laser Pointer. Proceedings of the International Conference on Multimedia Modeling (MMM 98), IEEE Cmputer Society Press.
6. Mizobuchi, S. and Yasumura, M. 2004. Tapping vs. circling selections on pen-based devices: evidence for different performance-shaping factors. In Proceedings of the

SIGCHI Conference on Human Factors in Computing Systems (Vienna, Austria, April 24 - 29, 2004). CHI '04. ACM Press, New York, NY, 607-614.

7. Myers, B. A., Bhatnagar, R., Nichols, J., Peck, C. H., Kong, D., Miller, R., and Long, A. C. 2002. Interacting at a distance: measuring the performance of laser pointers and other devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Changing Our World, Changing Ourselves (Minneapolis, Minnesota, USA, April 20 - 25, 2002). CHI '02. ACM Press, New York, NY, 33-40.
8. Olsen, D. R. and Nielsen, T. 2001. Laser pointer interaction. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Seattle, Washington, United States). CHI '01. ACM Press, New York, NY, 17-22.
9. R. Suklhankar, R. Stockton, and M. Mullin. Smarter Presentations: Exploiting Homography in Camera-Projector Systems. In Proceedings of International Conference on Computer Fision, 2001.
10. Shizuki, B., Hisamatsu, T., Takahashi, S., and Tanaka. J., Laser Pointer Interaction Techniques using Peripheral Areas of Screens. Proceedings of the international working conference on Advanced Visual Interfaces (AVI2006), pp. 95-98, May 2006.
11. X. Chen and J. Davis. LumiPoint: Multi-User LaserBased Interaction on Large Tiled Displays. Technical report, Stanford University, 2001.