

Laser Pointer Interaction

Dan R. Olsen Jr. and Travis Nielsen

Computer Science Department, Brigham Young University, Provo, UT

{olsen, nielsent}@cs.byu.edu

ABSTRACT

Group meetings and other non-desk situations require that people be able to interact at a distance from a display surface. This paper describes a technique using a laser pointer and a camera to accomplish just such interactions. Calibration techniques are given to synchronize the display and camera coordinates. A series of interactive techniques are described for navigation and entry of numbers, times, dates, text, enumerations and lists of items. The issues of hand jitter, detection error, slow sampling and latency are discussed in each of the interactive techniques.

Keywords

Laser pointer interaction, group interaction, camera-based interaction.

INTRODUCTION

A very interesting setting for interactive computing is in a meeting where the display is projected on the wall. Projection of the large image allows all participants sitting in their chairs to see the information under discussion. This provides a shared environment that can ground the discussion and provides an equal discussion point for everyone. However, if the information is interactive, only one of the participants has control of the changes. Interaction may occur through a computer in front of one of the participants whose screen image is being projected or it may occur through some on-the-board interaction device such as a Mimio pen [1]. In such scenarios only one person is in control. It is possible for multiple people to interact at the board using pen-based tools, but it is generally not feasible for more than two unless a very large screen is used. There is also the problem that when people are at the board, the rest of the participants have a hard time seeing what they are doing. What is needed is an inexpensive mechanism for people to interact at a distance from a display surface.

This paper describes an inexpensive technique whereby every person in the room using a \$15 laser pointer can interact with the information on a large projected display. Interaction is performed by using the laser to point at displayed widgets to manipulate their functions. The

equipment consists of a computer attached to a projector and a camera to detect the laser pointer position. We used a standard 1024 x 768 projector connected to a laptop PC. For the camera we used a \$500 WebCam that can deliver up to 7 frames per second over TCP/IP. This camera connection is very slow, but adequate for our initial tests.

In addition to meeting situations, this technique is useful wherever the user is in a situation for which a large projected display is possible, but a local personal display would be awkward. Examples include a repair shop with service information displayed on the wall, a laboratory where instrument controls are displayed on the wall, or as an alternative to the traditional television IR remote. In situations where the hands are occupied, the laser could be mounted on the back of a half-finger glove with the actuator switch on the side of the glove. This would require use of the hand to point, but would eliminate searching for and grabbing the pointer.

This work is distinct from other camera-based interaction techniques in that it bypasses the image processing problems of tracking fingers [2], head regions [3], or face features [4]. The focus of the camera is on the work surface rather than on the user. We are also concerned not with demonstrating or measuring a technology but developing a full suite of interactive techniques that can work as practical information manipulation tools. Kirstein and Muller [8] have reported a similar approach to interactive input. Their approach was to map the laser appearance, movement and disappearance to mouse down, move, and up events in X-Windows. As we will show in this paper, such a simple mapping is not sufficient for general information manipulation.

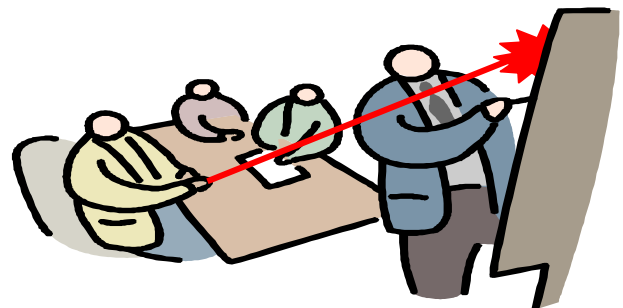


Figure 1 - Laser Pointer Interaction

System architecture

Our laser pointer system is implemented as an interactive client for the XWeb system[5]. XWeb is a client/server architecture for interactive manipulation of information. It

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGCHI'01, March 31-April 4, 2001, Seattle, WA, USA.

Copyright 2001 ACM 1-58113-327-8/01/0003...\$5.00

is designed to allow network information services to support interactive access from a wide variety of interactive platforms. The laser pointer system is one of a number of interactive clients that we have implemented for XWeb.

XWeb achieves its independence from particular interactive platforms by defining a set of general interactors that are based on the information to be manipulated rather than the interactive techniques that might be used. The interface specification is concerned with the range of possible information modifications and on the way information is organized rather than the handling of input events or actual presentations. For example XWeb defines an interactor for a finite set of choices. A given client may implement this as a menu, radio buttons, combo box, marking menu or any other technique. The information result would be the same. This independence of interactive technique has allowed us to create services that can be accessed via speech, button gloves, pen systems, and traditional desktops using the same interface specification. To fit with this architecture, the laser pointer system provides interactive techniques for each of the interactors in XWeb. By integrating with XWeb the laser pointer system can be used with any XWeb service and can also collaborate with any of the other interactive clients.

The laser pointer client is divided into three layers as shown in figure 2. The laser recognition layer handles the laser spot recognition and coordinate mapping, and is also responsible for cursor feedback on the state of the recognition. The recognizer layer communicates with the interaction layer using a set of specialized events. The interaction layer is responsible for the specific interactive techniques required for each type of interactor. The information-editing layer is uniform across all clients. It handles the interactor descriptors, propagation of data changes to network services, and management of collaborative sessions. In this paper, the areas discussed are the recognition layer and the interactive techniques found in the interaction layer.

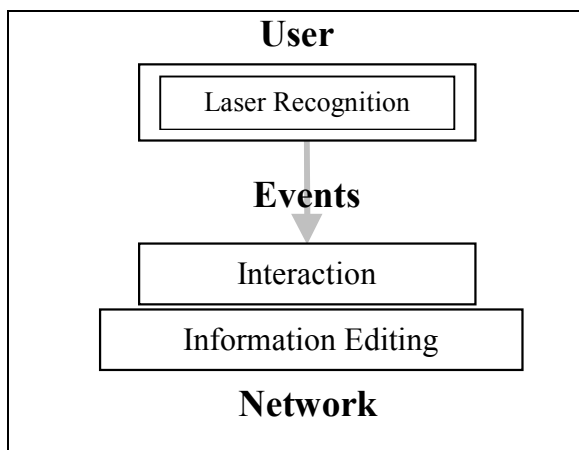


Figure 2 - System architecture

There are three fundamental problems to making this system work.

- Detecting the laser spot
- Calibrating the camera to the projector
- Developing appropriate interactive techniques

DETECTING THE LASER SPOT

Fundamental to the interactive techniques is the ability to locate the laser spot as shown in Figure 3. We must not only reliably determine the spot position but also reliably detect whether or not it is present. The spot recognition software can sometimes lead to delays of greater than 200 milliseconds. Much slower sampling rates make the movement of the cursor appear jerky.

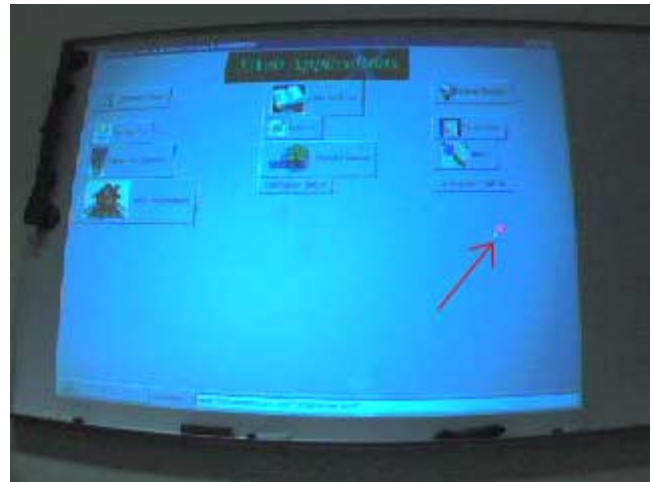


Figure 3 - Camera's View

The recognizer communicates with the rest of the interactive system in terms of five events.

- LaserOn (X,Y)
- LaserOff(X,Y)
- LaserExtendedOff(X,Y)
- LaserMove(X,Y)
- LaserDwell(X,Y)

Detecting laser on and laser off is somewhat problematic when using cheap cameras with automatic brightness control and low resolution. The automatic brightness controls continually shift the brightness levels as various room lighting and interactive displays change. This causes the detection algorithm to occasionally deliver a false off. The low resolution of the camera will occasionally cause the small laser spot to fall between sampled pixels also causing false off. The false off problem is partially handled by voting for on or off over the last 5 frames. False LaserOff events are also mitigated by careful interactive technique design. Similarly we detect dwell (holding the laser pointer in one spot) when the spot position over the last 5 frames lies within a small neighborhood. This essentially means that the LaserOn, LaserOff and LaserDwell events cannot be detected at faster than one per

second. This has not been a problem for our users. In addition we have introduced the LaserExtendedOff event, which is generated when no laser spot is detected for more than 2 seconds.

To detect the spot we use a two level technique. We first search for the brightest red spot. If we find an acceptable one, we return it as the laser position. Otherwise, we search using a convolution filter that is somewhat slower. To speed up the process and to reduce false cursor jumps, we first look in a small window around the last spot detected. If we do not find an acceptable spot in that window, we then search the whole screen in the next frame. Our most difficult problem is that the brightness of the laser spot tends to saturate the CCDs in the camera to produce a white spot rather than a red one. This is a problem when working over white areas of the display. We resolve this by turning down the brightness adjustment of the camera. Although we have achieved recognition rates that are substantially better than the 50% reported in [8] they are still enough of a problem that interactive techniques must be specially designed to mitigate the recognition problems.

CALIBRATING THE CAMERA

It is intended that this system be portable and usable in a variety of situations and with a variety of projectors and cameras. The fact that cameras, projectors and rooms are all different in their optics and their positioning poses a problem. What is needed is a function that will map a detected laser spot (X,Y) position in the camera image to the corresponding position in the coordinates of interactive display. As can be seen in figure 3 there are keystoneing and non-linear pincushioning effects from both the projector and the camera. These change from situation to situation.





We resolve this using a calibration step when our interactive client initializes. We project a series of 25 points on the screen whose interactive coordinates are known. Each of these points is then located in the camera image. With this set of 25 point pairs, we use least squares approximation to calculate the coefficients for two polynomials of degree -1 through 3 in X and Y. These learned polynomials are then used by the point detection software to map detected points into interactive screen coordinates.

INTERACTIVE TECHNIQUES

The challenge in designing interactive techniques for the laser is mitigating the effects of latency and errors in the laser tracker. Simple mapping of MouseUp/Down to LaserOn/Off does not work.

Handling the interactive techniques happens in three parts: recognition feedback, navigation and editing. Recognition feedback allows the user to adapt to the noise, error and misrecognition found in all recognizer-base interactions. The feedback is through an echoing cursor and through the

selection mechanism for the widgets. There are four cursor modes:

Tracking		Dwell detected	
Scrolling		Graffiti	

All of the cursors are positioned where the laser spot is detected. When no spot is detected, no cursor is shown, indicating to the user when there are recognition errors. If the LaserDwell event is detected over an interactor that responds to LaserDwell then the circle is added. This normally indicates that an interactive dialog fragment is beginning. The scrolling cursors appear when an interactor enables scrolling and LaserMove or LaserDwell events are reported. The Graffiti cursor appears when LaserMove or LaserDwell events are being interpreted as Graffiti strokes.

Navigation

In addition to the cursor echo there is also selection echo. The selected widget is surrounded by a red rectangle. Selection of widgets is the primary navigation mechanism for working through a full-sized interface. Information in XWeb is organized into hierarchic groups, lists and hyperlinks. All of these navigation tasks are handled by the widget selection. A widget is selected whenever LaserDwell is detected over that widget.

We introduced the LaserDwell event, where the laser pauses in a given region, because of the relative uselessness of LaserOn. One would naturally equate LaserOn with MouseDown in more traditional interfaces. However, there is no echoing cursor when the laser is off, as there is in the traditional “mouse up” condition. When the user first turns on the laser they have little confidence in where that spot will appear on the screen. The natural technique is to turn it on and the visually bring the laser to the desired location. This means that the initial position of LaserOn as well as the position of subsequent LaserMove events are not interactively useful because they do not convey any user intent, only the settling on the desired location. LaserDwell allows us to detect the desired location and forms our primary selection mechanism. This issue was also addressed by Kirstein and Muller[8] by mapping laser dwelling to MouseDown. This is similar to the interactive problems found in eye tracking [6, 7].

As mentioned earlier, XWeb supports collaboration among multiple interactive clients. An effective use of the laser pointer client is to slave it to the XWeb speech client. Speech-only interactions are relatively weak in navigation among complex structures of widgets. However, once a widget is selected, simply saying the desired new value is very effective. By using the laser pointer as a navigation device and the speech client to change values, an effective combination is achieved. The connection between the laser

pointer client and the speech client is entirely handled by the XWeb infrastructure.

Interactors

The XWeb interface specification is structured around interactors, which each embody a particular set of editing and navigation semantics. A particular XWeb client will create widgets for each of these interactors to provide interactive techniques that are appropriate to that interactor's semantics and the interactive devices available to the client.



Figure 4 - Selecting buttons

For purposes of this discussion, the laser pointer widgets can be divided into button, enumeration, scrollable, text and list categories. Buttons are currently only used for hyperlinks as in figure 4 and for global navigation tasks such as going back or forward along the hyperlink history. "Pressing" a button is done by selecting it using LaserDwell (hold the laser over the button until the laser dwell cursor appears) and LaserOff (releasing the laser pointer button). If the user moves outside of the button for a sustained period (approximately one second) before LaserOff then the button is not activated. This is similar to mouse-based interfaces where selecting the wrong button can be remedied by moving out of that button before releasing the mouse. Requiring movement outside for a sustained period rather than any movement outside the button is necessary because the natural hand jitter frequently causes inadvertent movements outside of the target widget.

Enumeration

The Enum allows the user to select from among a statically defined set of choices. LaserDwell over an Enum will cause the set of choices to pop up and then the user can navigate through the list by moving the laser over them as shown in figure 5. Any laser detection over any of the options will select that option. As shown, when the list of choices is too large, scroll arrow buttons are provided. Scrolling behavior is discussed in the section on scrollable widgets. LaserOff is useless in this interaction because of the frequency of false LaserOff events. We handle this in two ways. Once a selection is made, the user can begin working elsewhere. Any sustained (1/2 second) detection of the laser spot elsewhere on the screen will confirm the selection and close the popup. In addition, a LaserExtendedOff event, where the laser spot is not detected for 1.5-2 seconds, confirms the end of the selection dialog. Such a delay is normally way too long for interactivity. However, it works well in this situation because the user is not continuously interacting at this

point, but changing context. Any attempt to begin work somewhere else will confirm the change immediately, as will any pause in the work. This technique fits with the natural rhythm of the interaction.

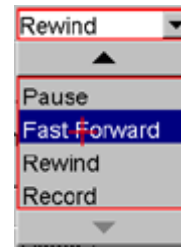


Figure 5 - Enumeration Interactor

Scrollable

The Number, Date and Time interactors are similar in that they each interact in a continuous range of possible values. They are also similar in that they are composed of parts that can be scrolled independently. Each of these interactors is shown in figure 6 in their inactive state. When each is selected they enter an active state, which pops up addition displays for the actual editing of the value.

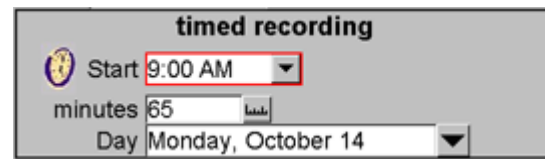


Figure 6 - Inactive Scrollable Interactors

Numbers

Our Number interactor can represent many numerical values such as minutes in figure 6 or multi-level units as in figure 7. Take for example, a length that can be expressed in feet and inches as well as in meters. The Number interactor allows interface designers to define the relationship among feet, inches and meters so that the user can interact in any of them. Similarly Fahrenheit and Celsius conversions are possible as well as any other linear combination of multilevel units.

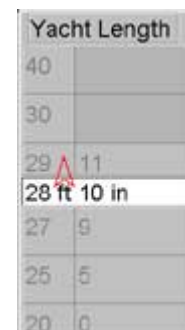


Figure 7 - Number Interactor

In designing interactive techniques for the laser, the primary constraint is that position change is easy for the

user and on/off/dwell is much more sluggish. Therefore, where possible, values are changed using spatial selection and scrolling rather than multiple events such as mouse clicks. Hand jitter also precludes the traditional scroll bar selection of values because the user cannot accurately hold a given scroll position.

Our basic interaction technique is to use the laser to express increment and decrement operations. When the interactor is selected using LaserDwell a panel of new values pops up, both above and below the current number value. Each of the parts of the number (in our example, feet and inches) can be incremented or decremented independently by holding the laser over one of the possible new values. As can be seen in Figure 7, the possible values are larger or smaller depending on how far they are from the current value.

Using LaserDwell or LaserOff to select increment and decrement values makes for a very sluggish interface. Therefore placing the laser over an increment value will select it without waiting for a dwell. The inherent variability and latency in the recognizer, however, make this an erratic scrolling technique that is hard to control. This is damped out by imposing a limit of 800 milliseconds between increment/decrement selections. Confirmation of the change is handled using activity outside the interactor or LaserExtendedOff.

Dates and Times

Figure 8 shows the Date and Time interactors. Both date and time displays can consist of several parts, any or all of which might appear in a particular presentation. The possible parts of a date are: century, last two year digits, full month name, abbreviated month name, month number, day number, full day of the week and abbreviated day of the week. Each of the parts can be scrolled independently using the same event dialog used for numbers.



Figure 8 - Date and Time Interactors

Text

The Text interactor is a standard type-in box for entering characters. There are two interactive problems, 1) selecting the appropriate insertion point and 2) entering the characters. A text box is first selected using LaserDwell followed by LaserOff. Once selected it presents the display in Figure 9 and sets the text insertion point at the location provided with LaserDwell. Because of hand jitter this

insertion point is rarely accurate. The set of arrow icons can be used to scroll the insertion point by character, word boundary or beginning/end of line. The interactive behavior of these scrolling arrows is similar to the scrolling used in number, times and dates.

When the text interactor is selected it captures the entire window to use as input space for entering text using Graffiti-like character strokes. We had to retrain our Graffiti recognizer to handle the relatively sparse point sets from the spot tracker. Most users find this form of text input rather cumbersome. The problem seems to lie in the latency and slowness of the spot recognition. Users seem to accurately generate the character strokes at about one third of the speed of writing on a Palm Pilot. However, the spot recognizer latency forces a slower stroke speed. Better cameras and faster camera connections should resolve this.



Figure 9 - Text Entry

List

The List interactor is our basic mechanism for structuring arbitrary amounts of data. A list is an ordered collection of rows containing other interactors. Rows can be selected and then modified using cut, copy and paste, as shown in Figure 10. Opening and closing the list as well as the other operations are all handled with buttons to the left of the list using the standard button dialog. The elements of the rows can themselves be selected directly.



Figure 11 - List interaction

USER STUDIES

We performed a quick test of the usability of the laser pointer interactions using eight subjects. The task was to input some settings for an automated lawn sprinkler timer. This task required entry of a start time, seven on/off

selections for which days of the week sprinklers were to run, and six times in hours and minutes for how long each zone was to be watered. Each user was given the same task on each of three different user interfaces. The interfaces were 1) the laser pointer widgets, 2) mouse driven Java/Swing widgets that look like the laser pointer widgets, and 3) a physical sprinkler timer with identical controls that we bought at a lawn and garden store. We shuffled the order of the interfaces for each user to mitigate learning effects between interfaces.

Before the laser pointer test each user was given 6 minutes to view a short video demonstrating the use of the widgets and practice with the laser pointer on a sample interface. All of the users were familiar with mouse-based interfaces and were not given training time for that interface. On the manual sprinkler timer each user was given 6 minutes to read the instruction card that came with the timer and to work with the timer before being given the task.

The average times to complete the task were as follows.

Average Task Times in Seconds

Mouse	Timer	Laser
90	206	215

Both the physical timer and the laser pointer are more than twice as slow as the mouse-based interface. On the physical timer the display and the buttons are highly multiplexed, requiring the user to learn a number of special modes to control the settings. On the laser pointer the latency in the recognizer and their unfamiliarity with that style of interface seemed to be the major problems. In this data there are also clear ordering effects among the tests. Among samples where the laser was used after the timer, the laser took less average time. When the laser was used before the timer, the timer performed better.

The only conclusions that we can draw from these tests are 1) that mouse-based interactions are clearly faster than either the laser pointer or the highly multiplexed buttons and display on the physical timer and 2) that the laser pointer display performs about the same as the physical timer interface. However, considering the significant noise and latency in the recognition, along with the unfamiliarity of the technique we are highly pleased with the performance of the laser pointer relative to other interactive techniques.

REFERENCES

- [1] <http://www.mimio.com/>
- [2] Crowley, J. L., Coutaz, J., and Berard, F. "Things that See" Communications of the ACM 43, 3 (March 2000), pp 54-64.
- [3] Berard, F. "The Perceptual Window: Head Motion as a New Input Stream" Proceedings of the 7th IFIP conference on Human-Computer Interaction (INTERACT), (Sept 1999).
- [4] Reilly, R. B., "Applications of Face and Gesture Recognition for Human-Computer Interaction," Proceedings of the 6th ACM International Multimedia Conference on Face/Gesture Recognition and their Applications, (1998), pp 20-27.
- [5] Olsen, D. R., Jefferies, S., Nielsen, T., Moyes, W., Fredrickson, P., "Cross-modal Interaction Using XWeb," Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST '00), (Nov 2000).
- [6] Zhai, S., Morimoto, C., and Ihde, S., "Manual and Gaze Input Cascaded (MAGIC) Pointing," Proceedings of Human Factors in Computing Systems (CHI '99), (May 1999), 246-253.
- [7] Jacob, R. J.K., "The Use of Eye Movements in Human-Computer Interaction Techniques: What You Look at is What You Get," ACM Transactions on Information Systems, 9, 3 (April 1991), pp 152-169.
- [8] Kirstein, C. and Müller, H. "Interaction with a Projection Screen Using a Camera-Tracked Laser Pointer." Proceedings of The International Conference on Multimedia Modeling. IEEE Computer Society Press, (1998)