# Interacting At a Distance Using Semantic Snarfing, Laser Pointers and Other Devices

Brad A. Myers, Choon Hong Peck, Dave Kong, Robert Miller, and Jeff Nichols

Human Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213
`bam@andrew.cmu.edu`
`http://www.cs.cmu.edu/~pebbles`

**Abstract.** It is difficult to interact with computer displays that are across the room. A popular approach is to use laser pointers tracked by a camera, but interaction techniques using laser pointers tend to be imprecise, error-prone, and slow. Our user study measured best-case performance numbers for laser pointers, and found that it takes people about one second each to move the beam to a target and to turn off the beam. Due to the unsteadiness of people's hands, the beam jitters about 10 pixels, but this can be filtered to a 2-pixel accuracy using a window of about 1 second. This adds to a three-second delay for any laser pointer interaction technique. Therefore, we developed a new interaction style, where the laser pointer (or other pointing technique such pointing with a finger or even eye tracking) indicates the region of interest, and then the item there is copied ("snarfed") to the user's handheld device, such as a Palm or PocketPC handheld. Interactions are then performed on the handheld using familiar direct manipulation techniques. The content often must be reformatted to fit the properties of the handheld to facilitate natural interaction.

## 1 Introduction

As ubiquitous computing [22] becomes more common, rooms will contain devices, appliances and displays that are computer-controlled. Many research and commercial systems have investigated using laser pointers to interact with screens that are across the room [3-5, 12, 23], but these interactions have seemed awkward and slow. Our measurements, reported below, show that this is due to inherent human limitations. People do not know exactly where the beam will be when they turn it on, and it takes about 1 second to move it into position. People's hands are unsteady, so the beam wiggles. And when the button on the laser pointer is released, the beam usually flies away from the target before the beam goes off.

Increasingly, people will be carrying computerized devices, including Personal Digital Assistants (PDAs) such as Palm and PocketPC devices, computerized cell-phones (e.g., the Microsoft [18] or Symbian [20] phones), or even computerized watches [10]. As part of the Pebbles project, we are researching how these kinds of mobile devices will interact with other devices such as those in such a "smart room." We describe here our concept of "semantic snarfing" as a model for how mobile devices might be used to control large displays at a distance.

"Snarf" is defined by *The New Hackers Dictionary* as "To grab, especially to grab a large document or file for the purpose of using it with or without the author's permission" [13]. We are using it here to refer to grabbing the contents of the PC screen onto a handheld device, usually without the knowledge of the application running on the PC.

We use the term *semantic* snarfing to emphasize that it is not always acceptable to just grab a picture or an exact copy of the interface. Instead, the meaning, or *semantics,* is often required. For example, if you want to edit some text displayed on a screen across the room, it would not be useful to copy a *picture* of the text to your mobile device. Instead, the text string itself must be snarfed, so it can be edited. Typically, the interface displayed on the mobile device must be different than on the large display, due to differences in screen size, screen properties (e.g., color or not) and the available interaction techniques (e.g., stylus on the handheld vs. 2-button mouse on the PC). When editing is finished, the result must typically be transformed semantically before it is transferred back to the PC.

This research is being performed as part of the Pebbles project [7] (http://www.pebbles.hcii.cmu.edu), which is investigating the many ways that handheld and mobile devices can be used *at the same time* as other computerized devices for meetings, classrooms, homes, offices and military command posts.

## 2 Related Work

A number of researchers have looked at interacting at a distance using a laser pointer, but none has reported performance numbers or user studies as described in this paper. Proxima has commercial data projectors that incorporate a camera for tracking the laser dot [4] and thereby controlling the mouse cursor. Eckert and Moore present algorithms and interaction techniques for a system with a camera looking for a laser dot [3]. Interaction techniques are described, including looking for the laser pointer to be off for at least one second to signify a mouse action. The specific actions are chosen using a global mode switch displayed at the bottom left corner of the screen. Kirstein and Muller describe a simple system with minimal interaction techniques for laser point tracking [5]. The Stanford iRoom project also investigated using a laser pointer, and uses special gestures and pie menus for the interaction [23]. Recently, the XWeb system was extended with a variety of laser pointer interaction techniques, including ways to use a laser pointer for menu selection, scrolling and Graffiti-based text entry [12]. The times to detect laser on, off and dwell in XWeb were at least one second each. Graffiti strokes needed to be full screen to work, and text was entered at about 1/3 the speed for text entry on the Palm.

The snarfing concept is related to work on multi-machine user interfaces (MMUIs), where the handheld is used at the same time as a PC. Rekimoto has studied how to move information fluidly using a “pick-and-drop” among different handhelds [14] and between handhelds and a whiteboard [15]. “Hyperdragging” was introduced as a way to move data by dragging it from one device to another [16].

Closely related to the snarfing of pictures is the VNC system [17], which displays the screen from one computer on another. The user can operate the remote screen by clicking on the local picture. Implementations for handhelds running the Palm and WindowsCE systems are available.

## 3 Motivation

In many of today’s meeting rooms, there is a computer display projected on the wall, and one person will control the mouse and keyboard connected to it. Smart-Boards [19] or other touch sensitive surfaces can allow one or two people to interact directly with the large presentation screen, but the rest of the people are left to try to describe objects and to manipulate them by proxy by having others do it. In previous work, we investigated having remote cursors controlled by handheld computers [9]. However, the most natural way for people distant from the screen to refer to objects is to point to them using a finger or laser pointer, or simply to look at the objects. Therefore, it seems desirable to have a camera track a laser pointer dot or follow where a finger is pointing in the air, or even to use eye tracking at a distance. Cameras tracking hand or eye movements will not be able to get more than a very crude estimate of where the user is pointing. Even with a laser pointer, the shaking of the user’s hand and the resolution of today’s cameras results in an inability to reliably point at anything smaller than about 10 pixels, and there is no “mouse button” on a laser pointer. Previous studies of laser pointer or gestural interaction at-a-distance have proposed a number of interaction techniques that try to overcome these problems [3] [12] [23], but they seem to be quite awkward and slow to use.

Surprisingly, we were unable to find any measurements in the literature of laser pointer performance, so we designed two studies to determine the inherent limits of the laser pointing technique. These are described in sections 4 and 6. For example, although future computers will be able see the laser dot at increased resolution, the amount of wiggle caused by the users’ hands shaking will still be a fundamental limitation (and will likely increase as we all get older).

Given these fundamental limitations and the awkwardness of other’s attempts at retrofitting today’s interaction models for use by laser pointing, we decided to try a different strategy. Rather than trying to interact at a distance, we use pointing for what it is good at: referencing a broad area of interest. The objects of interest can then “snarfed” onto the user’s handheld mobile device so the detailed work can be performed at a more effective speed and accuracy. This concept is described in section 7.

Another motivation stems from the prediction that in future “smart environments,” computers will be watching and listening for people’s commands, and will be able to respond to voice and gestures. If people want to manipulate a control that is across the room, sometimes it may be appropriate to speak a command (e.g., saying “lights on”), but other times, it may make more sense to use a control panel on a handheld mobile device (e.g., to move a continuous slider to adjust the brightness of the lights, rather than saying “darker, darker, darker, OK” or “lights to 32 percent”). Therefore, “snarfing” the controls onto the user’s device from wherever the user is pointing would be useful (see section 7).

## 4 Studying Laser Pointer Performance

We were frustrated that prior papers about laser pointer interaction techniques did not differentiate between problems resulting in limitations of today’s technology (camera resolution, processor speed, etc.) and problems inherent in the human use of laser pointers. We felt studying these issues would improve the design of future interaction techniques. Therefore, we performed two studies to discover some of the more fundamental parameters of laser pointing and, as a result, aid in the design of more usable laser interaction techniques. Specifically, we hoped to discover answers to some of problems related to selecting, clicking and dragging objects on a projection screen using a laser pointer.

We also wondered to what extent the numbers would depend on the particular design of the laser device. We compared a conventional inexpensive laser pointer (Figure 1-a) with a Symbol SPT 1700 handheld device [21], which is like a Palm Pilot with a built in laser and radio transmitter (see Figure 1-b). The laser in the Symbol is designed to perform laser scanning of bar codes, but on the SPT 1700, the rotating mirror can be turned off so the laser will be stationary and can serve as a laser pointer. We were particularly interested in the SPT since it provided a single platform that could serve as a pointing and snarfing device.

### 4.1 Setup for the First Study

A camera was placed 3 inches behind a piece of tracing paper, facing a target, which was a small dot with crosshairs drawn on the tracing paper (see Figure 2). A computer program then grabs frames from the camera with 320x240 resolution at a rate of 18-20 fps and tracks the location of the laser. This was the maximum rate and resolution we could achieve with our computer. The camera was put this close to the target so we could get the maximum possible resolution and investigate fundamental human limitations instead of camera and processor limitations. The camera’s view was about 5 square inches. To get this resolution on a typical projected screen at about 5 feet would require the camera to have a resolution of about 7680x5760, which may be possible in the future.

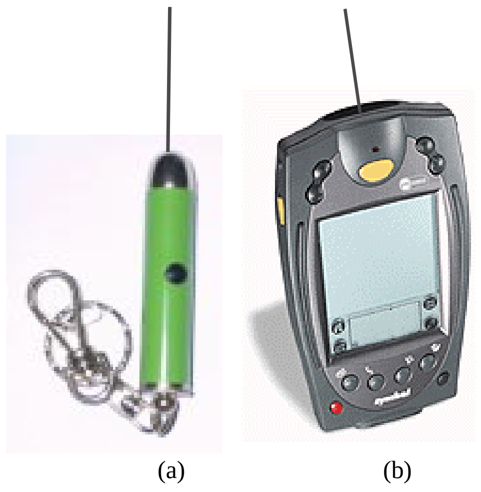


**Figure 1. Conventional, inexpensive laser pointer (a) and Symbol SPT 1700 Palm with built-in laser scanner / pointer (b).**
*(simulated laser beams)*

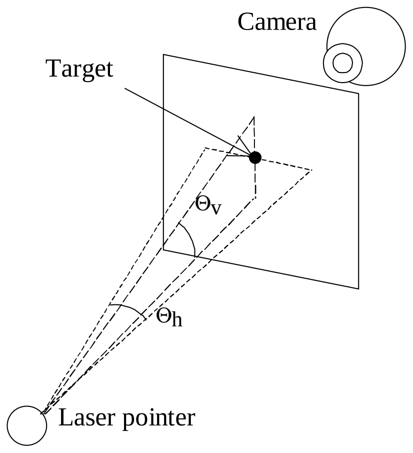
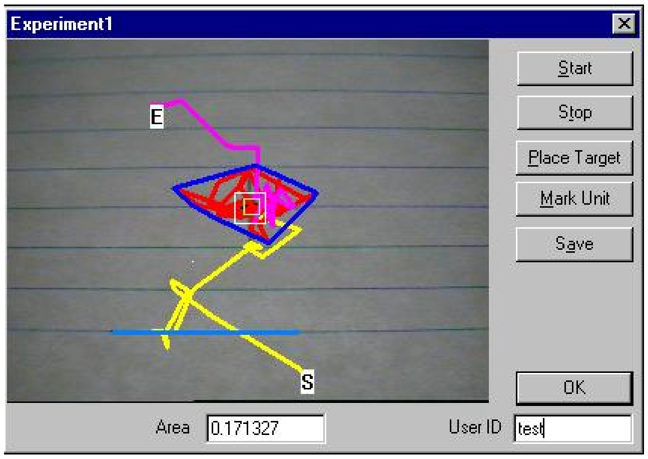


**Figure 2. Study setup.**



Area 0.171327 User ID test

**Figure 3. Sample plot as captured by the camera.**

For each trial, the user was asked to aim the laser beam at the target indicated on the tracing paper using the laser pointer. Figure 3 shows a sample plot that was captured from our program for one trial. It is superimposed over the actual view captured by the camera. The horizontal lines were on the original tracing paper, and are ¼ inch apart. A one-inch line was drawn on the paper, and the "Mark Unit" button allows the experimenter to draw on the computer a line of the same size (shown here as the darker blue horizontal line near the bottom), which is used to calibrate the camera's view. The light green inner square in the center is the target that the user was aiming at. The computer program records each trial in three phases. The acquisition phase records the entry trail of the laser as the user moves towards the target. In the trace, the user's starting position is marked with an "S", and the part of the trace before the target is acquired is in yellow (from the "S" to the center). The second phase records the dwell of the laser point on the target. When the target is acquired, the computer beeps and the user tries to hold the pen as steadily as possible on the target for 3 seconds. The red part of the trace near the center in Figure 3 is this phase. A second beep signals the user to turn off the laser. The third phase records the laser's exit trail. This is shown in purple in Figure 3, and goes from the center to the "E" which is where the beam disappeared. The dark blue polygon shows the convex hull of the points during the dwell phase, and the text display at the bottom shows that the area of this polygon is about 0.17 inches. The larger white square in the middle of Figure 3 is centered on the average of all the points during the second phase.

Each user performed 8 trials (a within-subjects design). Four of the trials are conducted with a conventional small laser pointer and the remaining trials were conducted using the Symbol SPT 1700 handheld device, which is like a Palm with a built in laser (see Figure 1). Out of the 4 trials using the conventional laser pointer, in 2 of the trials the user was 5 feet from the target and in the other two trials, the user was 10 feet away from the target. All trials using the conventional laser pointer were conducted with the user's dominant hand. With the SPT 1700, the user did one trial at 5 feet and one at 10 feet with each hand. Studying the non-dominant hand will provide parameters for designing interaction techniques that require the user to hold the handheld and operate the laser with the non-dominant hand while leaving the dominant hand free to write on the handheld device. The order of the trials was randomized to control for ordering effects.

Ten participants between the ages of 20 to 23 took part in this study, 4 women and 6 men. Prior to the study, the users filled out a questionnaire on their experience level and frequency of using a laser pointer. The mean score of the users were, on a scale of 1 to 5, 2.6 for experience and 1.8 for frequency of usage (where 1 = none and 5 = a lot), so they had used laser pointers some, but not very much.

## 4.2 Results of the First Study

Table 1 shows the amount of time that it took the subjects to acquire the target in phase one of the study. The average time ranged from 0.5 to 1.4 seconds, and the maximum time took up to 3.5 seconds to acquire the target. Comparing the time required to acquire the target for the various conditions, we find that the time at 10 feet is significantly ($p < 0.01$) greater than the time at 5 feet for both devices using the dominant hand. We also find a statistical significant difference for the dominant hand being faster than the non-dominant hand for the SPT at 5 feet ($p < 0.08$). However, there are no statistical significant differences between the dominant hand and the non-dominant hand for the SPT at 10 feet, or between the times for the laser and the SPT either at 5 feet or 10 feet.

We couldn't measure the maximum distance that the beam started away from the target, because it often started out of the camera's view. We observed that it was not unusual for the beam to start several inches away from the target.

**Table 1: Time to acquire target.**

| **(seconds)** | **0Laser Pointer in Dominant Hand** | | **SPT in Dominant Hand** | | **SPT in Non-Dominant Hand.** | |
|---|---|---|---|---|---|---|
| | **5 ft** | **10 ft** | **5 ft** | **10 ft** | **5 ft** | **10 ft** |
| Mean | 0.686 | 1.121 | 0.529 | 1.435 | 0.907 | 1.161 |
| Standard Dev. | 0.406 | 0.843 | 0.345 | 0.842 | 0.489 | 1.424 |
| Max | 1.700 | 3.545 | 1.102 | 2.674 | 2.000 | 5.017 |

Table 2 shows the results for the subjects' steadiness with the regular laser pointer. The first two rows show the deviation of the laser beam in inches. The first row is the horizontal deviation ($d_h$), and the second is the vertical deviation ($d_v$). These measures are the diameter of the convex hull. We wondered whether people's jitter tended to be equally distributed or more oriented more along one direction. The second two rows show the same measurements, but converted to angles. We conjectured that the angle of jitter would be the same independent of the distance, but the data contradicted this prediction. We found that $\theta_h$ at 10 ft is significantly ($p < 0.005$) less than $\theta_h$ at 5 ft whereas there is no significant difference in $\theta_v$. We speculate that this reduction in $\theta_h$ may be because the absolute distance of the deviation is larger and users can correct the position more easily when they can see the deviation more clearly. When we compared the $\theta_h$ at 5 ft against $\theta_v$ at 5 ft, we find that $\theta_h$ is significantly greater ($p < 0.001$) than $\theta_v$. This result suggests that the shape of the laser dwell is more like a horizontal ellipse at 5 ft and more circular at 10 ft. We have no theory for why this would happen.

**Table 2: Angle and diameter deviations for the Laser Pointer.**

| | **5 ft** | **10 ft** |
|---|---|---|
| Mean $d_h$ (inches) | 0.586 | 0.916 |
| Standard Deviation | 0.177 | 0.294 |
| Mean $d_v$ (inches) | 0.439 | 0.843 |
| Standard Deviation | 0.124 | 0.243 |
| Mean $\theta_h$ (degrees) | 0.570 | 0.437 |
| Standard Deviation | 0.167 | 0.140 |
| Mean $\theta_v$ (degrees) | 0.419 | 0.402 |
| Standard Deviation | 0.119 | 0.116 |

For the Symbol SPT 1700 handheld device (Table 3), we find that $\theta_h$ and $\theta_v$ at 10 ft are significantly ($p < 0.05$) less than $\theta_h$ and $\theta_v$ at 5 ft for the dominant hand. However, we did not find any statistical difference ($p \approx 0.05$) in both $\theta_h$ and $\theta_v$ between the distances for the non-dominant hand. This result implies that the user is able to correct the position more effectively at the further distance with his/her dominant hand but not so with his/her non-dominant hand. Also, the result suggests that users will generally produce a more circular drift using the SPT 1700.

Comparing the laser pointer with the SPT, we find no statistical significant differences between the amounts of wiggle, except that the $\theta_v$ and $d_v$ for the SPT are slightly smaller than the laser pointer at 10 feet ($p < 0.01$).

**Table 3: Angle and diameter deviations for the Symbol SPT 1700.**

| | SPT Dominant | | SPT Non- Dominant | |
|---|---|---|---|---|
| | **5 ft** | **10 ft** | **5 ft** | **10 ft** |
| Mean $\theta_h$ (degrees) | 0.557 | 0.455 | 0.592 | 0.482 |
| Standard Deviation | 0.153 | 0.153 | 0.162 | 0.143 |
| Mean $\theta_v$(degrees) | 0.397 | 0.298 | 0.535 | 0.385 |
| Standard Deviation | 0.088 | 0.073 | 0.277 | 0.151 |
| Mean $d_h$(inches) | 0.583 | 0.951 | 0.620 | 1.010 |
| Standard Deviation | 0.160 | 0.320 | 0.170 | 0.299 |
| Mean $d_v$(inches) | 0.415 | 0.625 | 0.561 | 0.807 |
| Standard Deviation | 0.093 | 0.152 | 0.290 | 0.316 |

We were also interested in how accurately a filtered average location could be. Figure 4 shows a graph of the data for one trial, with the dashed line at the bottom representing the moving average of all previous points. We found that this moving average would get within 0.1 inch of the target (which corresponds to about 2 pixels on a normal screen) after about 1 second. The data also shows that the beam can stay on for 0.5 to 1.0 second after the beep, and the deviation can be larger than the view area of the camera.

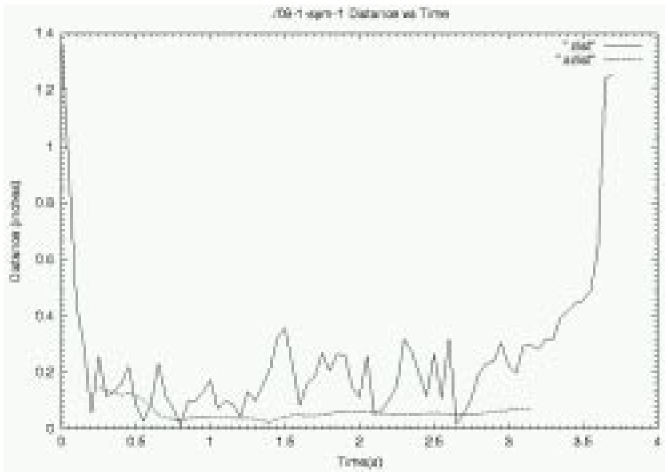


**Figure 4. Example trace of data recorded for one trial**

### 4.3 Discussion of the First Study

The brief summary of all the data from the first study is that people cannot turn on the beam where they want, they cannot hold it still where they want, and they cannot turn it off where they want. Before the laser point is where the user intends, we have to track the laser point for at least 0.9 seconds at 5 feet and 1.4 seconds at 10 feet, taking the maximum mean times. For a typical projection screen, in a small room where the user is 5 feet from a 6 feet wide screen, the size of the wiggle in the x and y axes will be about 10 and 7 pixels respectively. Standing 10 feet from a large 8 feet wide screen, the wiggle will be about 12 pixels in both axes. This implies that widgets designed for laser interaction must be fairly big. However, the moving average of the laser points can be a good indication of the user's intended location. The moving average of the dwell points over 1 second at 18 fps is usually within 0.2 inches (or about 2 pixels on a projection screen) from the target. These measurements provide a basis for the one-second delays built into many of the laser tracker systems reported in the related work section.

The implications of these numbers on the design of laser pointer interaction techniques are that to correctly track a user's dwell on a location to within 2 pixel accuracy, we have to track the laser for at least 0.9-1.4 seconds to wait for the user to get to where they want the beam to be, and then wait about another one second to get an accurate moving average, which will be within 2 pixels of the target point. This is a total of about 2.5 seconds to execute a selection. If start and end-point pair is desired (such as for a drag of an object to a new location), the location where the beam disappears cannot be used, so the system must look for the first and last dwell-points, which adds further delays. People are able to do almost as well with their non-dominant hand, which leads us to believe that two-handed interaction techniques using the SPT's laser may be effective.

## 5 Laser Tracking for Interaction

In order to experiment with interacting at a distance and semantic snarfing, we needed a laser tracking system that would work across the area of an entire projected display, rather than the tiny area used in our first study. Therefore, we tried pointing our inexpensive camera at a front-projected SmartBoard, without much success. The problem was that the automatic-gain-control on cheap cameras makes the bright laser dot saturate to white and be indistinguishable from other white areas of the screen. Therefore, we moved to a more expensive camera that has a manual brightness control. Setting the brightness almost all the way off causes the camera to still see the laser dot, but most of the projected image is no longer visible. This makes the tracking quite easy and accurate. Also, the more expensive camera does not have the curved-image problems that were reported by Olsen [12].

We use a Winnov image capture board in our PC, into which we plugged the S-Video output from the camera. The Pebbles laser tracker software is written in MFC Visual C++ and uses the Video For Windows API for grabbing frames from the device. This software layer abstracts the underlying hardware from laser-tracking software, so our software should work on any frame grabbing hardware. We grab each frame and subtract it from the previous frame to get a difference frame. In this difference frame, we look for the laser dot by looking for the pixel with the highest red value that is above the noise level. On a 450mhz PC, we can process images with a 320x240 resolution in RGB format at a rate of 18-20 frames per second.

Configuring the laser tracker is a two-step process. First, the user pushes the “Calibrate” button, which causes the software to calculate the normal noise level of the hardware device by looking for the maximum difference in the value of corresponding pixels in successive frames without any laser point or other changes in the picture Next, the user indicates four points on the screen (compare this to XWeb which requires 25 points [12]). The four points enable the software to perform a linear mapping of the camera’s view to the screen coordinates, since often the image seen by the camera is trapezoidal.

To control the PC’s cursor, we connect the laser tracker’s results to our RemoteCommander software that already maps operations on the handhelds’ screens to the PC’s cursor and keyboard actions. We added a special PC-side version of the RemoteCommander client that can accept input from the laser tracker program as if it had come from a handheld. Since we are exploring multi-machine user interfaces, we assume that the button presses for the mouse buttons will be performed using another device, such as a Palm or PocketPC held in the user’s non-dominant hand (see section 7). In the future, we might add the ability for a dwell of the laser dot to signify a button press, as in other laser tracking systems.

Currently, our laser tracker software is only set up to track a single laser dot, although techniques described by others (e.g., [23]) could be used to add tracking of multiple dots.

## 6 Study Two: Speed of Interaction

In this study, we wanted to use our new laser tracking setup described above to see how the speed and accuracy of interaction using the laser pointer compared to conventional devices such as a mouse. Therefore, we repeated the classic pointing experiment from Card, Moran and Newell (Experiment 7B, pp. 250) [2]. A picture of the testing application is shown in Figure 5. The test consists of clicking between a central stationary target and two outlying targets. The current target to click in is shown in red, while the other targets are shown in black. The current target also changes in a particular pattern, alternating from side to center to other side and back again. Each time the subject clicks in the center target, the new current target will also change in width and distance from the central target in a random pattern. The outer targets could have one of three widths (16, 64, or 128 pixels) and be two distances from the central target (36 or 144 pixels) for 2 x 3 = 6 possible conditions. The sequence of target conditions was randomly generated beforehand and used for every trial.

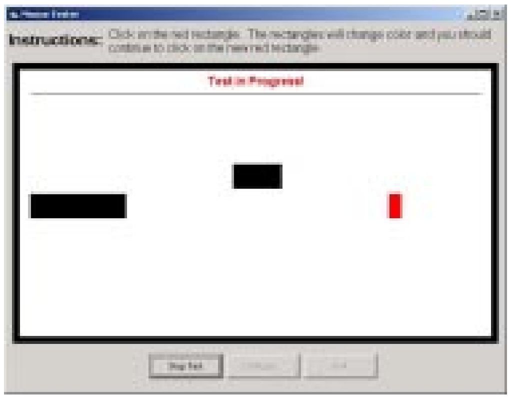

Instructions: Click on the red rectangle. The rectangles will change color and you should continue to click on the new red rectangle.

**Figure 5. Screen for study 2.**

We tested five pointing devices in this study:

1. A conventional mouse.
2. A SmartBoard, where the user could tap with a finger on the large board.
3. A Symbol device (Figure 1-b) where the laser is attached to the handheld. Any of the physical buttons on the Symbol could be pressed to signify the mouse button. In this condition, the laser beam is always on, so the acquiring and leaving times measured in study 1 are not relevant.
4. A laser pointer (Figure 1-a) in one hand and a handheld in the other hand that served as mouse button (we used the Symbol device in this condition, but only for its buttons). All subjects left the beam on for the entire trial.
5. A joystick built into the wireless remote control that came with our data projector (an Epson Powerlite 710C).

In the mouse condition, the screen was a conventional monitor, but in all other conditions, the screen was projected onto the SmartBoard. Each subject performed three thirty-second trials with each pointing device. The order of the devices was randomized to control for ordering effects. Ten participants between the ages of 21 to 30 took part in this study, 1 woman and 9 men. All participants were students in the School of Computer Science at Carnegie Mellon University.

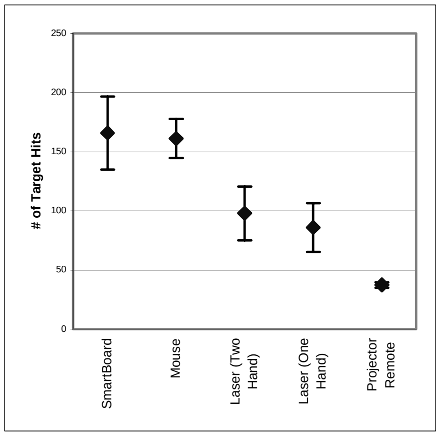
## 6.1 Results of the Second Study

Table 4 and Figure 6 shows the average performance for each pointing device in the second study in order of decreasing performance. A repeated-measures ANOVA shows the differences are reliable (F(5,50)-147.0, $p < .0001$). The devices can be categorized into three performance groups, most clearly shown in the number of targets hit metric. The SmartBoard and mouse are significantly better than the other four devices in number of targets hit. The projector remote and our picture snarfing technique are significantly worse. There is also a statistically significant different between the two pointer conditions.

**Table 4: Average User Performance Using All Six Pointing Devices.**

| | # of Targets Hit | | Miss % | | Time per Hit (sec) | |
|---|---|---|---|---|---|---|
| **Device** | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** |
| SmartBoard | 165.9 | 30.9 | 3.1 | 3.1 | 0.51 | 0.20 |
| Mouse | 161.3 | 16.5 | 3.5 | 3.8 | 0.52 | 0.16 |
| Pointer (Two Hands) | 97.9 | 22.8 | 10.6 | 7.8 | 0.85 | 0.41 |
| Pointer (One Hands) | 86.0 | 20.7 | 12.9 | 12.3 | 0.96 | 0.72 |
| Projector Remote | 37.5 | 2.3 | 20.1 | 10.8 | 2.09 | 0.70 |

(a)



(b)

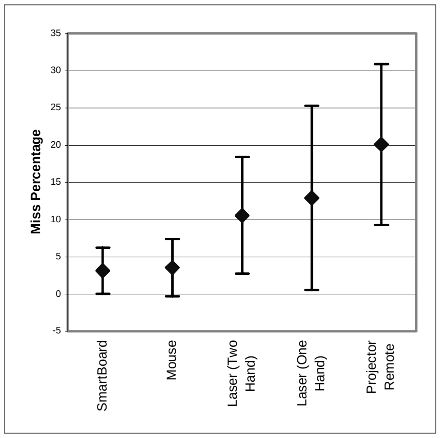


**Figure 6. Graphs of results of second study.**
**In (a), larger numbers are better, and in (b) smaller numbers are better**

### 6.2 Discussion of the Second Study

This study suggests that laser pointers are more difficult to use than close-range pointing devices such as the mouse and SmartBoard. Of the laser pointers, people performed better and preferred using two hands, which confirms other studies that showed good two-handed performance [1]. The error rates using the laser pointers were high, supporting the results from study 1. The joystick on the remote control was very hard to use.

## 7 Semantic Snarfing

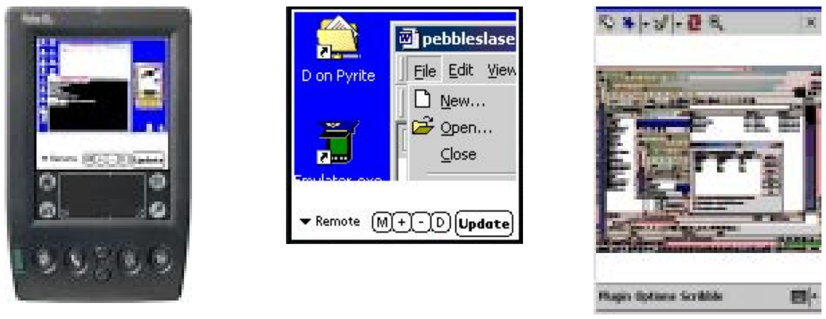
Given the inherent delays and inaccuracies we measured for using a laser pointer, it is not surprising that laser pointer interaction techniques are quite awkward and slow. If a camera were trying to track where the user's finger was pointing or even where a user's eyes were looking, the accuracies would be even worse. Therefore, we decided to investigate ways to make interacting at a distance be more effective. Based on the success of our other Pebbles applications, we decided to investigate a multi-machine user interface, by moving part of the interaction to the user's handheld device. This is embodied in our new "Grabber" application for Palm and PocketPC handhelds. Like all Pebbles applications, Grabber communicates to the PC using various methods. It can communicate using a serial cable or wirelessly using 802.11 or other protocols. The Pebbles communication architecture is described elsewhere [6].

In order to make the interaction more natural, we acquired a stylus with an embedded laser pointer. Thus, the user can hold the stylus/laser pointer with their dominant hand and use it both to point to the big screen across the room, and to tap on a Palm or PocketPC device held in the non-dominant hand. Alternatively, an integrated device like the Symbol SPT can be used. In either case, the laser pointer can be used to indicate the approximate area of interest on the main screen, and then the handheld's screen can be used for the detailed operations.

This mode of operation also helps support multiple users collaborating on a shared display. In real meetings, multiple people rarely seem to need to interact at exactly the same time, but turn taking is rapid and fluid. The snarfing style of interaction lets each person quickly grab what they want to work on to their private handheld, perform the necessary edits there, and then put the changes back. The Pebbles architecture already supports multiple people using their handhelds to operate on a single display at the same time [9]. If the laser tracker could follow multiple laser points, then Pebbles already will allow user to have a separate cursor in custom multi-cursor applications.

### 7.1 Snarfing Pictures

The Grabber program will capture a picture of the PC's full screen and shrink it to fit on the handheld's screen (see Figure 7). This is related to the idea of PalmVNC [17], but we supply more features to make it easier to interact on the handheld. The user can control the level of zooming. When the full PC's screen in visible (Figure 7-a and -c), it is very difficult to see any details, but the general screen layout is visible. When zoomed in all the way (Figure 7-b), the user can easily read the text, but very little of the PC's screen can be viewed at a time (a Palm screen with 160 pixels across can only show 1/40th of a 1024x768 screen at a one-to-one pixel ratio). Intermediate levels of zooming are also available.



(a) (b) (c)

**Figure 7. Snarfing pictures. (a) Full PC screen shown on color Palm IIIc screen. (b) Zoomed in so pixels are 1-1, but only 3 menu items are visible. (c) PocketPC version.**

If the view is zoomed in, the user can pan using various buttons on the handheld. Alternatively, the Grabber can be set to automatically pan to wherever the PC's cursor is, or wherever the user's focus with the laser pointer or other coarse-grain pointing device is. This makes it seem like the picture is being snarfed back along the laser beam.

Drawing on the handheld's screen can perform various operations on the PC. In one mode, pen operations on the handheld are sent through to the PC as if they were normal mouse operations with the appropriate coordinate mappings. This makes it easy, for example, to tap on buttons, or even to draw if a drawing program is running on the PC. In the "scribble" mode, the user's drawings on the handheld are drawn on the PC on a transparent layer in front of what is there, so the user can draw arbitrary annotations on the screen. These scribbles can be easily erased or saved for later.

## 7.2 Snarfing Menus

Figure 7 demonstrates a problem with grabbing only the picture of the PC's screen, and why we needed to implement *semantic* snarfing. In Figure 7-b, the File menu has been popped up, but only 3 out of the 23 items in the menu fit onto the Palm screen. Scrolling down to get to the "Exit" option at the bottom, for example, would be tedious and slow.

Therefore, Grabber can instead snarf the *contents* of the menu at the focus of interest, and redisplay the menu on the handheld as a regular menu. For example, Figure 8 shows the top-level menu and second level menus reformatted as Palm clickable lists.

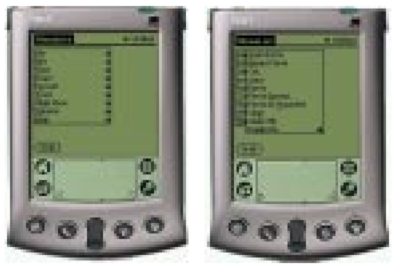


**Figure 8. Snarfing Menus onto the Palm reformats it to be Palm menus.**

Grabber snarfs the menus and toolbars out of unmodified, conventional PC applications using various heuristics. We can get standard windows menubars and toolbars using various Windows calls, and menus and toolbars from Microsoft Office applications using OLE Automation. The menu or toolbar items are then sent to the handheld, where they are displayed using an appropriate set of widgets for the handheld. When the user clicks on an item with a submenu, the submenu is displayed on the handheld without involving the PC. When the user clicks on a leaf item, an appropriate message is sent to the PC to cause the selected operation to be performed.

In the future, we would like to add support for menus in other kinds of applications, in particular for menus implemented in Java Swing. We already have the capability to snarf the links out of a web page onto the handheld [8], and integrating this with the laser pointer focus mechanism might also be helpful for interacting with web pages at a distance.

### 7.3 Snarfing Text

Figure 7 shows that it is impossible to read or edit the text on the handheld when the full screen is showing, but when zoomed-in, you cannot see the whole line. Therefore, to enable text editing, Grabber can also snarf the text at the focus of interest. The user can choose whether to grab one line, 5 lines, or the whole text string (see Figure 9). The text is then reformatted into a text string in the format of the handheld, and the user can edit it. After editing, the user can have the string put back onto the PC, to replace the old string that was there.

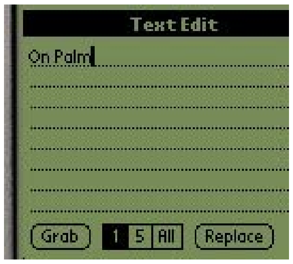


**Figure 9. Snarfing text to the Palm.**

In order to capture the text, various heuristics are used. We can get the text out of a standard edit control (which is used by dialog boxes, text fields, and the Notepad window), and from a Microsoft Office application using standard procedure calls.

After the user is finished editing, a command on the handheld will replace the original string on the PC with the edited string. This may not work if some other user on a handheld or at the keyboard has edited the same string. We currently use very simple heuristics to check whether it is OK to put the string back, and simply check whether the text that was originally grabbed is still at the same location. In the future, we could use more sophisticated matching techniques such as those in others' multi-user text editors. We also hope to add support for snarfing the text from many other kinds of applications, including Microsoft Word and Java text widgets. We currently do not capture any formatting of the text, and this would also be good to add, but much more difficult to render on the Palm, which typically only has 3 fonts.

### 7.4 Snarfing of Control Panels

As a new part of the Pebbles project, we are working on how to automatically create control panels for appliances on the handheld [11]. We call this creating a "Personal Universal Controller" (PUC) since it is customized to the one user, and the handheld will be able to control any appliance. Our preliminary studies suggest that interfaces on a handheld may be operated in 1/2 the time with 1/5 the errors as the manufacturer's interface [11]. Using the handheld in this way can be considered snarfing the user interface off of the appliance. We are using *semantic* snarfing, since a fundamental goal of the research is to automatically reformat the controls to be appropriate to the properties of the handheld, and the preferences and experience of the user. We hope to report more about this research in the future.

## 8 Future Work and Conclusions

In addition to developing the above ideas further and performing user tests of the Grabber application, we believe the concept of Semantic Snarfing can be applied to many other areas, and can enhance other kinds of interactions. For example, for handicapped people, the handheld might reformat the PC's screen to be much larger or the text to use a larger font. The text that is snarfed to the handheld could be read by a text-to-speech engine. The text (or the labels for menu items) might even be translated into a different natural language by using a web service such as http://www.dictionary.com/translate/.

It is predicted that more and more of people's everyday devices such as cell phones and watches will become wirelessly networked to computers and appliances through technologies such as 802.11 and BlueTooth. Billboards, stores, and other information services may be able to deliver their content in different formats, so the mobile devices will then be able to snarf information from many kinds of information displays, and format it in an appropriate way for the mobile device's screen.

In this ubiquitous computing world, people will want to use whatever devices they have at hand to operate or investigate screens or appliances at a distance. It will therefore be increasingly important that the information and controls be able to be semantically snarfed to any kind of mobile device.

## Acknowledgements

For help with this paper, we would like to thank Al Corbett, and all the users of the Pebbles applications who have provided useful feedback.

This research is supported by grants from DARPA, Microsoft and the Pittsburgh Digital Greenhouse, and by equipment grants from Symbol Technologies, Palm, Hewlett-Packard, Lucent, IBM and SMART Technologies, Inc. This research was performed in part in connection with contract number DAAD17-99-C-0061 with the U.S. Army Research Laboratory. The views and conclusions contained in this document are those of the authors and should not be interpreted as presenting the official policies or position, either expressed or implied, of the U.S. Army Research Laboratory or the U.S. Government unless so designated by other authorized documents. Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

## References

1. Buxton, W. and Myers, B. "A Study in Two-Handed Input," in Proceedings SIGCHI'86: Human Factors in Computing Systems. 1986. Boston, MA: pp. 321-326.
2. Card, S.K., Moran, T.P., and Newell, A., The Psychology of Human-Computer Interaction. 1983, Hillsdale, NJ: Lawrence Erlbaum Associates.
3. Eckert, R.R. and Moore, J.A., "The Classroom of the 21st Century: The Interactive Learning Wall." SIGCHI Bulletin, 2000. 23(2): pp. 33-40.
4. Horn, G.A.V., "Proxima's new Ovation+ projection panels do up multimedia." Byte (on-line), 1995. http://www.byte.com/art/9501/sec12/art9.htm.
5. Kirstein, C. and Muller, H. "Interaction with a projection screen using a camera-tracked laser pointer," in Multimedia Modeling; MMM '98 Proceedings. 1998. pp. 191 -192.

6. Myers, B.A., An Implementation Architecture to Support Single-Display Groupware. Carnegie Mellon University School of Computer Science Technical Report, CMU-CS-99-139 and Human Computer Interaction Institute Technical Report CMU-HCII-99-101, May, 1999. http://www.cs.cmu.edu/~pebbles/papers/pebblesarchtr.pdf.

7. Myers, B.A., et al., “Using Hand-Held Devices and PCs Together.” ACM Communications of the ACM, 2001. pp. To appear.

8. Myers, B.A., et al. “Extending the Windows Desktop Interface With Connected Handheld Computers,” in 4th USENIX Windows Systems Symposium. 2000. Seattle, WA: pp. 79-88.

9. Myers, B.A., Stiel, H., and Gargiulo, R. “Collaboration Using Multiple PDAs Connected to a PC,” in Proceedings CSCW'98: ACM Conference on Computer-Supported Cooperative Work. 1998. Seattle, WA: pp. 285-294. http://www.cs.cmu.edu/~pebbles.

10. Narayanaswami, C. and Raghunath, M.T. “Application Design for a Smart Watch with a High Resolution Display,” in Proceedings of the Fourth International Symposium on Wearable Computers (ISWC'00). 2000. Atlanta, Georgia: pp. 7-14. http://www.research.ibm.com/WearableComputing/factsheet.html.

11. Nichols, J.W. “Using Handhelds as Controls for Everyday Appliances: A Paper Prototype Study,” in ACM CHI'2001 Student Posters. 2001. Seattle, WA: pp. 443-444. http://www.cs.cmu.edu/~pebbles/papers/NicholsRemCtrlShortPaper.pdf.

12. Olsen Jr, D.R. and Nielsen, T. “Laser Pointer Interaction,” in ACM CHI'2001 Conference Proceedings: Human Factors in Computing Systems. 2001. Seattle, WA: pp. 17-22.

13. Raymond, E.S., The New Hacker's Dictionary. Second Edition ed. 1994, Cambridge, MA: The MIT Press. See also: http://www.fwi.uva.nl/~mes/jargon/.

14. Rekimoto, J. “Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments,” in Proceedings UIST'97: ACM SIGGRAPH Symposium on User Interface Software and Technology. 1997. Banff, Alberta, Canada: pp. 31-39.

15. Rekimoto, J. “A Multiple Device Approach for Supporting Whiteboard-based Interactions,” in Proceedings SIGCHI'98: Human Factors in Computing Systems. 1998. Los Angeles, CA: pp. 344-351.

16. Rekimoto, J. and Saitoh, M. “Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments,” in Proceedings SIGCHI'99: Human Factors in Computing Systems. 1999. Pittsburgh, PA: pp. 378-385.

17. Richardson, T., et al., “Virtual Network Computing.” IEEE Internet Computing, 1998. 2(1): pp. 33-38. http://www.uk.research.att.com/vnc/.

18. Shim, R., “First look at MS 'Stinger'-based phone.” ZDNet UK Online, 2000. http://www.zdnet.co.uk/news/2000/31/ns-17218.html.

19. SMART Technologies, “SMART Board 580,” 2001. http://www.smarttech.com/.

20. Symbian, “Simply a Better Phone: Symbian's Smartphone reference design,” 2001. http://www.symbian.com/.

21. Symbol Technologies, “SPT 1700 Pocketable Computers,” 2001. http://www.symbol.com/products/mobile_computers/mobile_palm_pi_hdwr_spt1700.html.

22. Weiser, M., “Some Computer Science Issues in Ubiquitous Computing.” CACM, 1993. 36(7): pp. 74-83. July.

23. Winograd, T. and Guimbretiere, F. “Visual Instruments for an Interactive Mural,” in ACM SIGCHI CHI99 Extended Abstracts. 1999. Pittsburgh, PA: pp. 234-235. http://graphics.Stanford.EDU/projects/iwork/papers/chi99/.