Interacting At a Distance Using Semantic Snarfing

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

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, which can be important in meetings and when controlling computerized devices. 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. Therefore, we have 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. If the content changes on the PC, the handheld's copy will be updated as well. Interactions can be performed on the handheld using familiar direct manipulation techniques, and then the modified version is sent back to the PC. The content often must be reformatted to fit the properties of the handheld to facilitate natural interaction.

1   Introduction

As ubiquitous computing [18] 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 [2-4, 10, 19], but these interactions are awkward and slow. This is due in part to inherent human limitations. Users do not know exactly where the beam will be when they turn it on, and it takes about one second to move it into position. Users' 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 are carrying computerized devices, including Personal Digital Assistants (PDAs) such as Palm and PocketPC devices, computerized cell-phones (e.g., the Microsoft phones [16]), or even computerized watches [8]. 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 permis-

sion" [11]. 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 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, military command posts, homes, and offices.

2   Related Work

A number of researchers are looking at interacting at a distance using a laser pointer. Eckert and Moore present algorithms and interaction techniques for a system with a camera looking for a laser dot [2]. One technique watches for the laser pointer to be turned 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 [4]. The Stanford iRoom project also is investigating using a laser pointer, and uses special gestures and pie menus for the interaction [19]. 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 [10]. The delay times to detect the laser turning on or off and for dwelling in XWeb are at least one second each. In order to draw Graffiti accurately, the strokes have to be almost full screen, and text is 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" metaphor among different handhelds [12] and between handhelds and a whiteboard [13]. "Hyperdragging" was introduced as a way to move data by dragging it from one device to another [14].

Closely related to the snarfing of pictures is the VNC system [15], 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 Windows CE systems are available.

3   Motivation

In many of today's meeting rooms, there is a computer display projected on the wall, and one person controls the mouse and keyboard connected to it. SmartBoards [17] and other touch sensitive surfaces allow one or two people to interact directly with the large presentation screen, but other meeting participants must resort to manipulation by proxy—trying to describe which objects to manipulate and having the user at the controls do it. In previous work, we investigated having remote cursors controlled by handheld computers [7]. However, the most natural way for people who are distant from the screen to refer to objects is to point at them using a finger, a laser pointer, or simply a gaze. 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 point reliably at anything smaller than about 10 pixels. Furthermore, 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 [2] [10] [19], but they are quite awkward and slow to use. Although future computers will be able to see the laser dot at increased resolution, the amount of wiggle caused by the users' hands shaking will still be a fundamental limitation.

Given these human limitations and the awkwardness of other 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 be snarfed onto the user's handheld mobile device so the detailed work can be performed more quickly and accurately.

Another motivation stems from the prediction that computers in future "smart environments" will be watching and listening for commands, and will be able to respond to voice and gestures. 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., moving a continuous slider to adjust the brightness of the lights, rather than saying "darker, darker, darker, OK" or "lights to 32 percent"). Snarfing the controls onto a handheld device by pointing it at the lights would be very useful.

Other future applications of the semantic snarfing idea might include classrooms, where students might snarf interesting pieces of content from the instructor's presentation; collaborative design sessions, where people might snarf parts of an overall design for detailed analysis and editing; or military command posts, where staff members might be assigned to snarf particular items from an overall map display so they can "drill-down" into the details. This is related to the concept of using the handheld as a "magic lens" [1] for "revisualization" of data displayed on a public display. Individuals could pick which data on the public display was interesting and use their handhelds to view the data in different ways that are more meaningful and personalized.

(a)   (b)   (c)
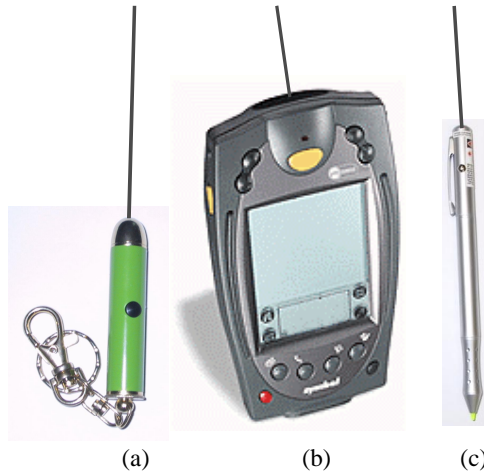
Figure 1. A conventional, inexpensive laser pointer (a), a Symbol SPT 1700 PalmOS device with built-in laser scanner / pointer (b), and a pen with a laser pointer at one end and a stylus or ink pen at the other (c).

4   Semantic Snarfing

Given the delays and inaccuracies inherent in 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 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 more effective. Based on the success of our other Pebbles applications [6], we decided to investigate a multi-machine user interface (MMUI) that shifts part of the interaction to the user's handheld device. We implemented semantic snarfing in our new "Grabber" application that runs both on 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 IEEE 802.11 or other protocols. The Pebbles communication architecture is described elsewhere [5].

In order to make the interaction more natural, we acquired a stylus with an embedded laser pointer (Figure 1-c). 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 (Figure 1-b) 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 [7]. If the laser tracker could follow multiple laser points, then different people could be snarfing at the same time. Currently, they would have to take turns.

Currently, Grabber can snarf three kinds of data: pictures, menus or text. The user chooses which is desired using a menu on the handheld. The area of the PC screen that is snarfed is determined by the initial position of the cursor or laser point when the user hits the "grab," "update," or ⮂ (refresh) button on the handheld. The rest of this section describes these different types of snarfing.
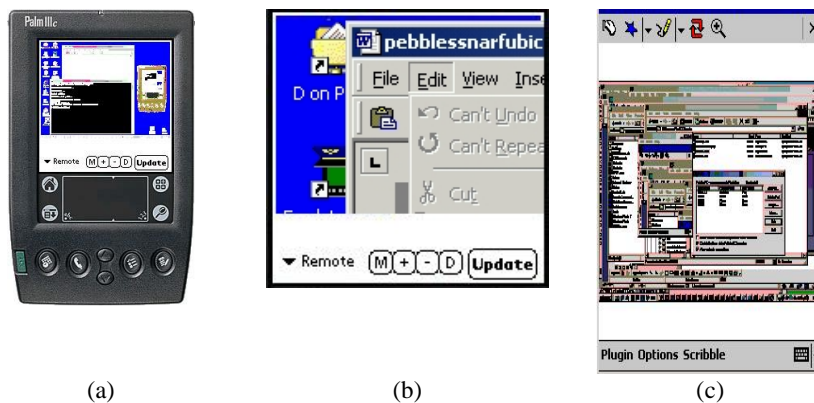

(a) (b) (c)

Figure 2. Snarfing pictures. (a) Full PC screen shown on color Palm IIIc screen. (b) Zoomed in on Palm so pixels are 1-1, but only 3 menu items are visible. (c) PocketPC version.

4.1 Snarfing Pictures

The Grabber program captures a picture of the PC's full screen and shrinks it to fit on the handheld's screen (see Figure 2). This is related to the idea of PalmVNC [15], but we supply more features to make it easier to interact on the handheld. In Grabber, the user can control the level of zooming. When the full PC's screen in visible (Figure 2-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 2-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. The picture of the requested size is generated on the PC.

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 second mode, called "scribble," the user's strokes 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.

4.2   Snarfing Menus

Figure 2 demonstrates a problem with grabbing only the picture of the PC's screen, and why we needed to implement *semantic* snarfing. In Figure 2-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 3 shows the top-level menu and second level menus reformatted as Palm clickable lists.

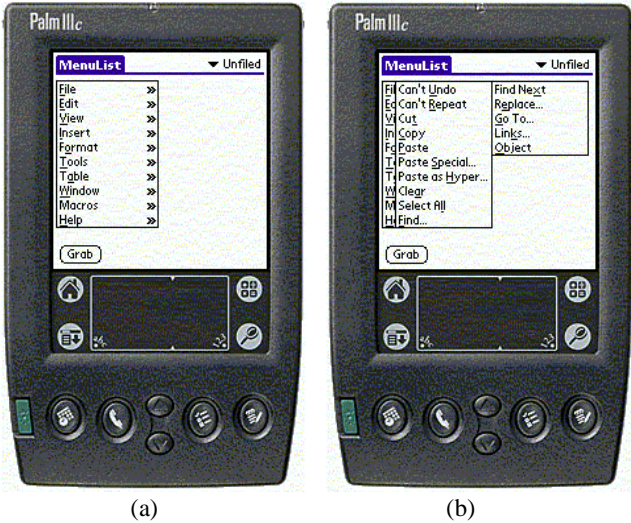(a)   (b)

Figure 3. Snarfing the menus onto the Palm reformats as multicolumn clickable lists. When the user taps on "Edit" in (a), all the items in the Edit submenu are displayed in two columns, as shown in (b).

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 the COM facilities. 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 an item without a submenu, 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 [6], and integrating this with the laser pointer focus mechanism might also be helpful for interacting with web pages at a distance.

4.3   Snarfing Text

Figure 2 shows that it is impossible to read or edit 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 4). 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.
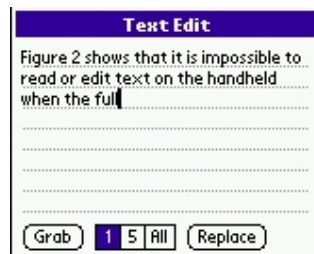

Figure 4. Snarfing text to the Palm.

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 COM.

After the user is finished editing, a button 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 text formatting information because it is much more difficult to render on the Palm, which ships with only 3 fonts.

5   Future Work

In addition to developing the above ideas further and performing user tests of semantic snarfing, we will be investigating how 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 use a larger font for the text. A text-to-speech engine could read the text that is snarfed to the handheld. 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://babelfish.altavista.com/.

As a new part of the Pebbles project, we are working on how to automatically create control panels for appliances on the handheld [9]. 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 [9]. 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.

6   Conclusions

"Semantic snarfing" is a new interaction style, where a laser pointer, finger or eye gaze indicates the region of interest, and then the item at that point is copied ("snarfed") to the user's handheld device, where further interaction takes place. 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 IEEE 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.

7   Acknowledgements

8   References

1. Bier, E.A., *et al.* "Toolglass and Magic Lenses: The See-Through Interface," in *Proceedings SIGGRAPH'93: Computer Graphics.* 1993. **25**. pp. 73-80.

2. 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.

3. Horn, G.A.V., "Proxima's new Ovation+ projection panels do up multimedia." *Byte (online)*, 1995. http://www.byte.com/art/9501/sec12/art9.htm.

4. 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.

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. 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/.

12. 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.

13. 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.

14. 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.

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

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

17. SMART Technologies, "SMART Board 580," 2001. http://www.smarttech.com/.

18. Weiser, M., "Some Computer Science Issues in Ubiquitous Computing." *CACM*, 1993. **36**(7): pp. 74-83. July.

19. 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/.