PointRight: A System for Pointer/Keyboard Redirection Among Multiple Displays and Machines

Brad Johanson, Greg Hutchins, Terry Winograd
Stanford University
Gates 3B-376
Serra Mall
Stanford, CA 94305-9035
{bjohanso,gmh,winograd}@graphics.stanford.edu

ABSTRACT

As hardware becomes cheaper, rooms with many large displays become practical. The machines driving the displays need to run standard operating systems in order for legacy applications to be used. Getting keyboard and mouse input to all the screens in the room creates complexity and potential confusion—although it is possible to have multiple wired or wireless keyboards and mice, it becomes confusing to know which one to use. PointRight is a software system that allows pointer and mouse control to be smoothly switched among displays. A single free-floating mouse and keyboard can be used to control multiple screens. When the cursor reaches the edge of a screen it seamlessly moves to the adjacent screen and keyboard control is simultaneously redirected to that machine. Laptops may also redirect their keyboard and pointing device, and multiple pointers are supported simultaneously. The system automatically reconfigures itself as displays go on or off or change the machine they are currently displaying.

KEYWORDS: Mouse control, Pointer redirection, Large displays, shared display groupware, interactive workspaces.

INTRODUCTION

The Interactive Workspaces Project at Stanford [2] is experimenting with an environment containing multiple large simultaneously visible displays (Figure 1). Each display is driven by a projector that has several possible machines as sources. A drop cable allows display to any of the screens from a laptop. The machines all run standard Windows operating systems, to allow the use of legacy applications. The three side screens are touch sensitive.

Our goal is to allow a user to control all of the machines that display to the screens in the room in an intuitive fashion. For example, if something is on the touch screen the user should be able to interact with it by touch, independently of the machine being displayed.

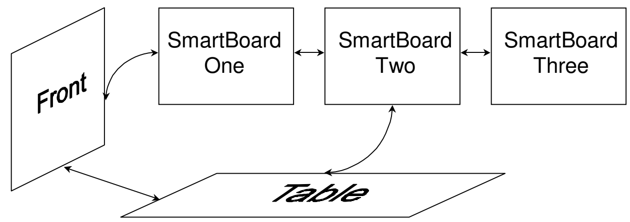

Figure 1 - Pointer Transition Paths Between Screens

The PointRight system allows a free-floating wireless mouse and keyboard to control all visible screens. The wireless mouse controls a cursor on one screen at a time. Going off the edge of the screen takes it to the next adjacent screen, if there is one. This "pointer space" of adjacent screens functions similarly to a large virtual desktop except that each display is a different machine. Touch screens automatically re-route themselves to the displaying computer, and laptop users can redirect their keyboard and pointer into the pointer space. The functionality is accomplished by installing a client on all machines that can receive pointer and keyboard events, and a server on those that can generate them.

RELATED WORK

In addition to the obvious solution of a separate input device for each screen, there are several other solutions to the problem of controlling multiple machines Keyboard-Video-Mouse switches allow several machines to be controlled with one keyboard and mouse in a modal fashion (the user must perform explicit switching actions and remember the current mapping).

Apple has supported multiple display desktops from one machine for many years, and Microsoft Windows now supports this as well. VNC [6] and similar commercial products allow keyboard and mouse re-routing from one machine to another along with mirroring the remote display. The Pebbles system [4] and other CSCW systems allow multiple-mice controlled by PDAs (in the case of Pebbles) simultaneously on a single machine, but don't support transferring control between many adjacent screens. Rekimoto in [5] has shown an augmented desktop where a laptop mouse can drag items on and off a top projected table display.

POINTRIGHT FUNCTION

With PointRight, user-controlled pointers move seamlessly across screen boundaries in a pointer space according to an adjacency map, as illustrated in Figure 1. Pointer and keyboard events go to the machine that is currently the source of that display. If a display is off or not running the PointRight software, the pointer skips over that display. In order to avoid losing track of the cursor, a special larger moving cursor is displayed when the mouse is wiggled.

Touch screens are also automatically re-routed, sending input to whatever machine is displaying, rather than the one to which the touch screen hardware is connected. Client-side mapping compensates for screen-resolution differences among machines. Any laptop running the PointRight Software can also become the controller for any machine, in the same mode as the shared wireless mouse.

The system supports an arbitrary number of pointers at once, including those redirected from laptops. It currently does not attempt to resolve actions by multiple pointers on one screen (as addressed in [4]), allowing the operating system to intermix mouse events as they arrive. Since we set cursor position as absolutes, when multiple people move control pointers on one screen at the same time, they see ghosts of each cursor as the cursors are drawn in a round robin fashion.

POINTRIGHT MECHANICS

There is a client for each machine that receives pointer and mouse input, and a server for each machine that can act as a source. Clients listen for absolute coordinate pointer events that they convert to local screen coordinates and then submit to the local event queue. Each server maintains a database of the display topology and a current position of their pointer. The database contains information about the screens and machines in the space and their current connections and is used by servers to determine which machine should receive the events. Since the database is per server, each pointer can potentially apply to different sets of screens and different interconnection topologies.

Each server sends events to the current target screen over a direct connection formed the first time the server has an event for that source. The local databases track state by receiving configuration update events through the Event Heap [2]. These events come from PointRight clients saying their machine is on or off, from projector controllers, and from laptops indicating that they are available to be connected to displays in the room.

The system currently works with Windows 9x/NT/2000 in both client and server mode, and in Linux in server mode

DISCUSSION

PointRight provides flexible layout. Screens need not be tiled on a plane—note that in Figure 1 the left side of the table display connects to the bottom of the front display. The layout can be changed dynamically through the use of database update events.

The system is in active use for meetings in our Interactive Room. Its simplicity seems quite intuitive—on several occasions we have come across groups using the PointRight mouse and keyboard without any training. Future plans include control of mouse acceleration across the different servers and machines in a uniform manner and convenient specification of configurations.

ACKNOWLEDGEMENTS

The Interactive Workspaces project is the result of efforts by many students; see [2] for an exhaustive list and more complete project information. Thanks to Bryn Forbes, and Rito Trevino in particular, who helped figure out how to tap into the Windows and Linux mouse and keyboard event systems. Thanks to Susan Shepard for her help making the video and keeping the iRoom stable enough to develop in. The work described here is supported by DoE grant B504665, and by donations of equipment and software from Intel Corp., InFocus, IBM Corp. and Microsoft Corp.

REFERENCES

[1] Fox, A., Johanson, B., Hanrahan, P., Winograd, T., “PDAs in Interactive Workspaces,” Computer Graphics and Animation, May, 2000.

[2] Interactive Workspaces Project at Stanford University, http://graphics.stanford.edu/projects/iwork. See also http://graphics.stanford.edu/projects/iwork/pointright.

[3] Johanson, B., Fox, A., Hanrahan, P., Winograd, T., “The Event Heap: An Enabling Infrastructure for Interactive Workspaces”, submitted to UIST’2000, San Diego, CA, USA, 2000.

[4] Brad A. Myers, Herb Stiel, and Robert Gargiulo. "Collaboration Using Multiple PDAs Connected to a PC." Proceedings CSCW'98: ACM Conference on Computer-Supported Cooperative Work, November 14-18, 1998, Seattle, WA. pp. 285-294.

[5] Rekimoto, J. "Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments." CHI'99, pp. 378-385.

[6] Richardson, Tristan, Quentin Stafford-Fraser, Kenneth R. Wood & Andy Hopper, "Virtual Network Computing", *IEEE Internet Computing*, Vol.2 No.1, Jan/Feb 1998 pp33-38.