System Design for the WeSpace: Linking Personal Devices to a Table-Centered Multi-User, Multi-Surface Environment

Hao Jiang[1,2] Daniel Wigdor[3] Clifton Forlines[4] Chia Shen[1]
[1]Initiative in Innovative Computing at Harvard University [2]Tsinghua University
[3]Microsoft Surface [4]Mitsubishi Electric Research Labs
h-jiang@mails.thu.edu.cn dwigdor@microsoft.com forlines@merl.com chia_shen@harvard.edu

Abstract

The WeSpace is a long-term project dedicated to the creation of environments supporting walk-up and share collaboration among small groups. The focus of our system design has been to provide 1) groups with mechanisms to easily share their own data and 2) necessary native visual applications suitable on large display environments. Our current prototype system includes both a large high-resolution data wall and an interactive table. These are utilized to provide a focal point for collaborative interaction with data and applications.

In this paper, we describe in detail the designs behind the current prototype system. In particular, we present 1) the infrastructure which allows users to connect and visually share their laptop content on-the-fly, and supports the extension of native visualization applications, and 2) the table-centric design employed in customized WeSpace applications to support cross-surface interactions. We will also describe elements of our user-centered iterative design process, in particular the results from a late-stages session which saw our astrophysicist participants successfully use the WeSpace to collaborate on their own real research problems.

1. Introduction

The amount of data that collaborators are bringing to group meetings to share, explore, and make sense of is growing exponentially. With this growth, challenges emerge not only in visually presenting large quantities of data, but also in enabling collaborative interaction with this data in a natural and efficient way.

As a potential remedy to these challenges, shared display surfaces in various form factors are becoming commercially available. These include multi-megapixel data walls that offer a large physical display area without compromising dots-per-inch (DPI), as well as multi-user, multi-touch tabletops that enable direct touch on the display surface and foster collaboration through a face-to-face setting. The availability of these products enables the construction of collaborative visual exploration spaces. In such a space, information, from any source or user, can be visually presented with high fidelity, and simultaneously and collaboratively explored by multiple users.

The construction of such a space requires not only the careful design of the user interface, but also the engineering of the system architecture. The designer must consider the needs of multiple simultaneous users and their needs in terms of data transfer, visualization, and communication. The engineer must provide careful structural architecture to accommodate massive data transmission, information visualization, and multiple views rendering in real time.

The WeSpace is our research into a multi-surface collaboration space. Among similar systems, the WeSpace particularly aims at building a general-use tool for workplaces in which visual data from different sources are rendered simultaneously for exploration. The WeSpace works in a walk-up and share manner: group members simply connect their laptops and start pouring the visualization of their data to the shared surfaces. Our current prototype has been successfully used by a group of astrophysicists in their collaborative

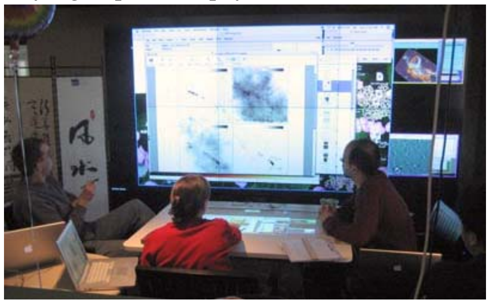
Figure 1. Astrophysicists meeting in the WeSpace.

research meetings.

The development of the WeSpace has undergone several iterative cycles as revisions of the infrastructure design and feedback from user interfaces came in. The challenges we faced in those iterations mainly focused on 1) the system structural design to accommodate visual feeds from multiple laptops and have them updated and rendered on multiple surfaces at real time, and 2) the interaction design for a multi-user application crossing multiple shared surfaces. We regard the above two issues key to delivering a multi-user system of real use, and believe them common to all multi-user, multi-surface system designers.

The goal of this paper is to describe our experience building WeSpace and detail the designs we arrived at that address the challenges of multi-user, multi-surface collaborative spaces. It is our hope that our approach will inform future work on building such systems. We start with an overview of the WeSpace project, narrating its basic requirements and design iterations. We then describe the infrastructure of the system, followed by the table-centric cross-surface interaction techniques employed in two native WeSpace applications: Layout Manager and LivOlay. We also report results from actual users using the WeSpace on real research problems.

2. Related Work

Numerous research projects have explored creating digital meeting rooms that include shared interactive surfaces. The Colab [25] system allows teams to work together or remotely on multiple desktops and a digitized whiteboard. Dynamo [12] allows users' media to be transferred to a server and presented on the server's shared display. Streitz et al [20][26] embedded computers into meeting room furniture, such as whiteboards (Dyna Wall), tables (InteracTable) and chairs (CommChair). Their work also described interaction techniques to support spontaneous collaboration. Rekimoto and Saitoh's Augmented Surfaces [22] interconnects digital devices and physical objects in the workspace, allowing media data to be drag-and-dropped across surfaces or carried with physical objects. Shen et al's UbiTable [23] provided a mechanism for the spontaneous, walk-up-and-use sharing of data, such as photos and notes. The iRoom [15] project aimed to investigate and build seamless interactive spaces, in which group activities benefit from coordinated views on multiple large displays [8].

Redirecting locally running applications to remote displays has been shown to keep group members aware of the ongoing activities of their collaborators. ARIS [2] and SEAPort [3] provide textual and iconic interfaces respectively to transplant applications to other co-located devices. A selected application will maintain its current context and continue running on a different machine. Multibrowsing [17] allows web pages to be displayed and viewed on multiple displays in a collaborative way. In another approach, screen sharing techniques such as the VNC protocol [21] allow visualizations of native applications to be shared and even interacted with remotely. Tan et al [28] and Wallace et al [30] respectively described their work on showing multiple remote application images on a shared display. Furthermore, comDesk [18], Mighty Mouse [5], and IMPROMPTU [4] enable a user to re-direct their shared application windows to other collaboration members' personal displays to be viewed and/or interacted with. IMPROMPTU [4] also supports shared displays in the collaborative environment. The key new features that set the WeSpace apart from the above systems are the provision of 1) native applications for domain specific group usage inside the WeSpace large display environment, and 2) a shared multi-user multi-touch table as a central and visible input space for shared group input.

Efficient graphics rendering is also a major concern when building user interfaces for shared surfaces. DiamondSpin [24] provides a SDK for building collaborative tabletop applications. DiamondSpin application windows have flexible orientations to be viewed from different sides of the table. Isenberg et al [11] presented a buffer framework for rendering multiple digital artifacts on the screen with high performance. Tuddenham et al's T3 framework [29] simplifies rendering on high-resolution tiled displays.

To support interaction with an application that resides on multiple displays, researchers have investigated mechanisms for moving a pointer around the space. For example, in the iRoom [15] project, Johanson et al used the PointRight [16] technique, allowing a single mouse to navigate across all the display surfaces in the space. User input from a keyboard is then directed to the mouse's focus. Later work based on cross-surface movement of a pointer made improvements in maintaining visual continuity when a mouse travels across the seam between displays. Nacenta et al's Perspective Cursor maps displays' 3D positions in the space to the user's 2D visual plane [19]. Baudisch et al's Mouse Ether [1], which leverages the discrepancy between display DPIs, works similarly.

An interactive tabletop, once introduced into the space, can be used as a centric control to other surfaces. Forlines et al [9][10] use direct touch on a horizontal surface to coordinate views on vertical displays. The MultiSpace [7] system creates portals on the table for other devices, so that transferring data to a device is done by drag-and-drop to the target portal. Wigdor et al [31] added a world-in-miniature representation to each portal on the table that enabled direct control to layout of peripheral surfaces.

3. The WeSpace: Overview

The WeSpace project originated from our desire to develop a general tool to support scientists conducting collaborative research across multiple disciplines. We began by seeking out a research group to serve as partners in our user-centered, participatory design process. The target group we chose was the COordinated Molecular Probe Line Extinction Thermal Emission Survey of Star Forming Regions group (http://www.cfa.harvard.edu/COMPLETE) at Harvard Smithsonian Center for Astrophysics, known as the COMPLETE group. The WeSpace is the outcome of our year-long close collaboration with COMPLETE group members: from intensive interviews, to attending and observing their research meetings, to identifying system requirements, to iterative development cycles, to the delivery of the WeSpace system.

Initial interviews with COMPLETE members revealed a challenging need for a better collaboration tool to support their work. The highly variable individual practices of the groups' members make software standardization impossible. In particular, data sources and types examined by the COMPLETE group vary widely within a project. As a result, the software tools employed by group members varies widely, with some being custom-built, one-of-a-kind solutions pieced together by the team members themselves or other astrophysics practitioners. Collaboration is further complicated by the fact that these tools do not conform to any standard form of output.

After the investigation, we sat down with the COMPLETE members and derived the following system requirements for a collaborative tool that would address these challenges.

Provide a sharable display: the environment should include displays that would sufficiently allow the researchers to present their work to the group. The displays should ideally function at a high resolution to ensure the effect of visualization applications running on them.

Allow the use of their laptops: because of the necessities of using different data types and custom software and configurations, the collaboration tool must allow users to run applications from their own laptops, while visualizing and analyzing the output rendering on a shared display.

Maintain interactivity of existing applications: the ideal tool would allow data being shown on the large display interactively, within the application generating its view. This allows for a faster iterative process, while maintaining the fidelity of data.

Retain user control over their own data: on certain occasions such as meeting with people outside the research group, the collaborators needs to maintain control over their own data, ensuring that only the renderings they choose are shown, and that proprietary underlying data is not shared. Ignoring this requirement might limit how well the system would generalize to other uses.

Support egalitarian and visible input: because each member brings a different expertise to the group, the system should provide group members with equal opportunity of controlling the meetings.

Our first version of the WeSpace was built in fulfillment of the above requirements. The system was deployed on a single data wall. Users' laptop screen images, once connected, were shared on the wall. Each user had his/her own pointer (distinguished by color) roaming on the wall display, controlled by his/her laptop mouse. A user's pointer could move into other users' desktop windows on the wall, thus gain control over other laptops. Feedback from COMPLETE group members showed that they had confusion over keeping track of each user's pointer and the connection between which mouse was associated with which pointer. In addition, after working on the existing system they brought up new functional requirements to facilitate visual analysis, such as overlaying different application images. These comments resulted in two extra system requirements, listed below.

Provide an interactive table: a multi-touch tabletop adds possibilities of simultaneous direct-touch input from multiple users, and promotes more egalitarian input to the system. Besides, a user's input with their hand on the table is more apparent to other members than input with a mouse. This promotes task awareness among multiple users and reduces confusion.

Support native applications on the system: different from users' applications on their laptops, native system applications reside and execute on the system server. A native application receives input from all connected laptops, applies visualization functions such as overlaying multiple sources, and renders output on the shared surfaces. Supporting such native applications in the system would allow users share their data at the same time, and make sense of them by applying visualization functions.

The current WeSpace system includes a 10ft by 5ft Mitsubishi MegaView data wall with the resolution 3072x1536, and a 42-inch DiamondTouch [6] multi-user interactive table with a projected resolution 1280x1024, both driven by a 3.2 GHz Windows PC. With a light-weight client installed on the laptop, users simply walk up to the space, connect to the server, and start sharing their ongoing applications on the large displays. Two WeSpace native applications are currently built and installed in the system: Layout Manager and LivOlay. The former provides automatic and customized layout arrangement of multiple laptop images on the shared surface, and the latter allows live applications to be visually overlaid.

The WeSpace system has been used several times by the COMPLETE group to conduct collaborative meetings on real research topics. Figure 1 shows a picture taken from one of these sessions. We received positive feedback from all group members, and witnessed tangible outcomes (research proposals) coming from those collaborative sessions, one of which is describe in detail in a later section.

4. The WeSpace: Infrastructure Design

As our goal was to create a collaborative space for users to walk-up and share their visual data with minimum interruption to their day-to-day scientific practices, we based the WeSpace on screen sharing techniques that allow real-time application images to be sent to a remote computer. Using screen sharing techniques, any existing application can be shared on the large display without relinquishing the underlying data. In contrast, other sharing models fall short in fulfilling some of our system requirements. One of those models, for example, relocates an application from a user's laptop to public surfaces [2][3]. Despite losing control over proprietary data, relocation every single application requires ad hoc installation and configuration of that application on the server, making it impossible to generalize the system to a truly walk-up-and-use tool.

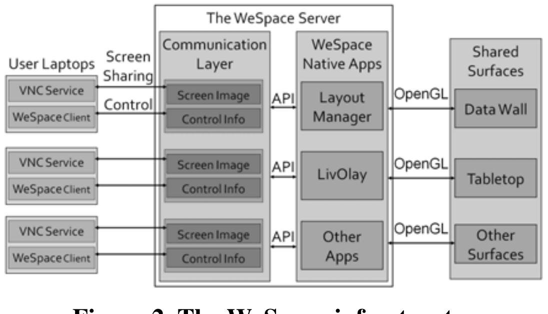

Figure 2. The WeSpace infrastructure.

We leverage the standard VNC protocol [21] for screen sharing, which is widely used and well implemented on all major operating systems. For example, Mac OS has built-in VNC Sharing Service, and Windows users can also enable the VNC service by installing a freeware called RealVNC [21].

Figure 2 shows the infrastructure of the WeSpace. All shared surfaces are driven by a single server. Shared data and control information are transferred between the server and laptops via a wireless network. The WeSpace is implemented using Java.

A Light-weight WeSpace Client is installed on each user's laptop. The WeSpace Client exchanges control information with the server and provides a switch to turn on/off screen sharing service for privacy protection. By intercepting low-level laptop mouse and keyboard events, the WeSpace Client is also able to direct user input to the server to control native WeSpace Applications on the shared displays. To connect to the WeSpace, a user simply launches the client, types in the server IP address and hits the "Connect" button. A TCP channel for exchanging control information is established between the client and the server, followed by a VNC connection between the WeSpace server and the VNC Service provider in the user's laptop.

The Communication Layer in the server manages network connections to users' laptops. It creates an instance for each connected laptop. Each instance in the communication layer maintains an image buffer for the client's live screen data, as well as the control information for that client.

WeSpace APIs (Application Programming Interfaces) are exposed at the communication layer. Client screen images and other control information are accessible through those APIs for developing WeSpace native applications. To make it a general tool, we built the WeSpace with this open structure to support collaboration in different visual exploration domains. Using exposed APIs, WeSpace programmers can build customized visual applications that fit a particular domain purpose, and run these applications in the space.

WeSpace Native Applications running on the server collect shared data in image format from all connected users, apply customized visualization techniques, and render the output on the shared surfaces in the WeSpace. With the interactivity of the multi-touch tabletop and input from client laptops, WeSpace native applications enable users to explore and make sense of the collective visual data from different sources. As of now, two applications have been built and installed in the WeSpace: Layout

Manager and LivOlay. Their designs are described in the next section.

Shared Surfaces in the current WeSpace prototype include a large data wall and a multi-touch tabletop. However, the WeSpace infrastructure allows the number and arrangement of display surfaces to be configurable to WeSpace application developers. Shared surfaces are rendered by native WeSpace application using JOGL [14], a Java wrapper for OpenGL. With four client laptops connected, our prototype refreshes at an average rate of 15 frames per second on both the 4.5-million-pixel data wall and the tabletop.

Although screen-sharing based collaborative spaces exist in other research projects [4][18][30], the WeSpace mainly differs from them in two facets. First, while other systems use mouse as the only input mechanism to the system, a multi-touch tabletop is introduced into the WeSpace, which helps to maintain the focus on the visual data, and better support awareness and egalitarian input. Second, the WeSpace supports native visualization applications running on the shared surfaces, which are extensible to fit specific domain purposes. Instead of displaying screen images without modification, in WeSpace users' live data goes through image processing, and is optimally presented for visual exploration and collaboration.

5. The WeSpace: Native Applications

While visually intensive tasks greatly benefit from physically large, pixel-rich displays [27], the absence of effective interaction techniques on the shared displays may hinder users from "feeling control over the data" during the exploration. By having a multi-touch tabletop in the WeSpace, we leverage its support of under-the-finger direct-touch interaction to minimize the diversion of user attention incurred from giving input to the system. In addition, an interactive tabletop creates a face-to-face seating arrangement among users and has better support to egalitarian input, both fostering collaboration productivity.

A table-centric approach is employed in the WeSpace to interact with multi-surface applications: the tabletop serves as the major channel to provide input to the system, as well as the viewport control to other vertical displays in the space. Specifically, a WeSpace application works in **separate views** when users want to give input on the table while watching the visual outcome on the high-resolution data wall. User interface widgets are rendered on the tabletop but not the wall. **Synchronized views** between the table and the data wall are formed when the viewport of the wall needs to be adjusted. The table now has a mirrored view of the visualization on the wall with decreased resolution. Multi-touch gestural input on the table, such as zooming and panning, alters the viewport on both surfaces.

These concepts embodies in our design of the two WeSpace native applications: Layout Manager and LivOlay.

5.1. Layout Manager

As many pieces of shared data from all users are brought up to the collaborative space, only one or a few of them need attention at any moment. Layout Manager allows users to arrange the layout of connected laptop screen images on the shared surfaces, both manually and automatically, so that the shared data that is the current focus of discussion occupies the central position and more display area on the data wall.

Each client laptop connected to the space is assigned a display status: important, public, or private. Important and public laptops are both displayed on the shared surfaces, whereas the former are enlarged to highlight their importance. Private screens indicate their owner's desire for privacy, thus will not be displayed on the shared surfaces. The default status for a connected laptop is public.

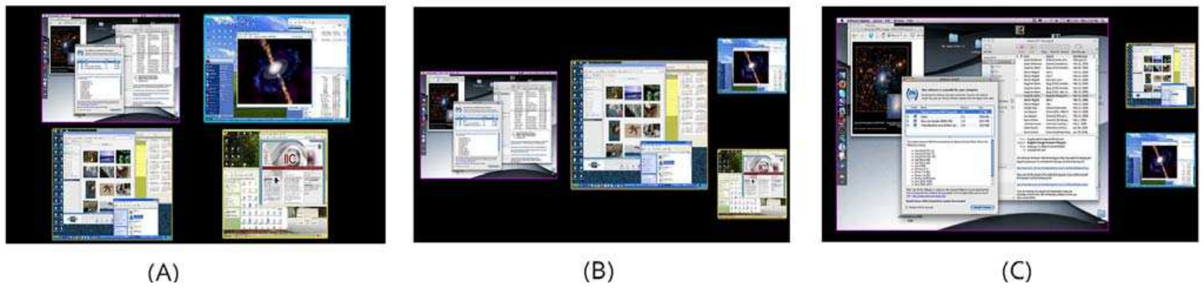

Figure 3. Automatic layout arrangement in Layout Manager, with four laptops connected. (A) All laptops are in Public status. (B) Two Important laptops and two Public laptops. (C) One Important laptop, two Public laptops and one Private laptop (hidden).

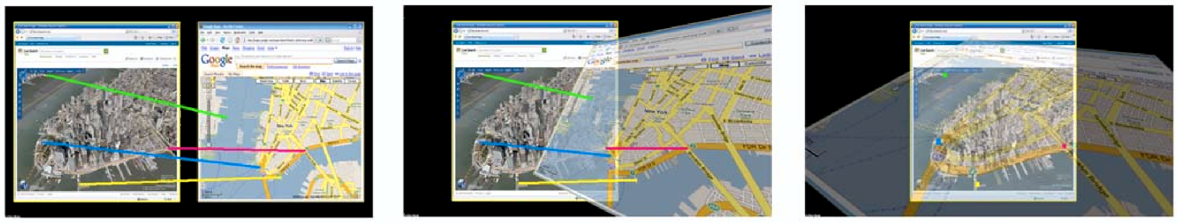

Figure 4. The transition of views on the data wall in LivOlay, with three applications to be overlapped. (left) Linked view. (center) Transition. (right) Overlapped view.

Layout Manager maintains synchronized views between the wall and the tabletop: important and public screens are displayed with identical layout on both surfaces. To switch a laptop screen to another status, any user may tap on the control buttons rendered next to that screen on the tabletop. Status controls are also provided on each laptop's native WeSpace client interface, and can be altered by its owner.

Layout Manager applies an automatic arrangement when a display's status changes: important screens will fill the center space of the shared surfaces, while public ones are set aside (Figure 3). Transitions are animated to ensure visual fluidity. Users can also manually control the size and position of the laptop images with gestural input performed on the tabletop, similar to [24]. This change in layout is reflected on shared surfaces.

The multi-touch tabletop can also redirect touch input to individual connected laptops in Layout Manager. Double-tapping a laptop's image on the table severs the synchronized view between surfaces: the wall keeps the layout display of multiple screen images, while the table zooms in to a full-screen display of the selected laptop. User actions on the tabletop are interpreted as mouse input and sent to the client laptop. Exiting the full-screen mode restores the synchronized view.

5.2. LivOlay

As a complement and a necessary addition to the side-by-side comparison function provided in Layout Manager, we incorporated LivOlay in to the WeSpace to address the need in visual exploration tasks to overlay imagery data from different sources. LivOlay works by users selecting corresponding landmark points in visualizations to be overlaid. With these common points, a transformation is first calculated then applied to screen images. The result is that screen images are distorted, registed, and overlaid. Visual data to be overlaid in LivOlay are live application screens from connected laptops, which can reflect users' real-time input. An early version of LivOlay without multi-touch table support is presented in [13].

LivOlay in the WeSpace supports multiple applications from all connected users to be overlaid at the same time. When LivOlay is launched, all users' desktop images are displayed on the table. Users tap on applications to select them, selected applications have their boundaries acquired and visually highlighted using the WeSpace API.

LivOlay leverages the pixel-rich data wall as the main surface for displaying the resulting visualizations. To help users analyze applications images both separately and in an overlapped stack, LivOlay supports two display styles on the wall: **linked view** and **overlapped view**.

In the linked view, applications are displayed separately side-by-side, each application showing registered points as well as links to corresponding points on other application images (Figure 4, *left*); in the overlapped view, live application renderings are overlapped according to the transformation calculated using their registration points [13] (Figure 4, *right*). Toggling between views is achieved by tapping on the button rendered on the tabletop, and transitions are

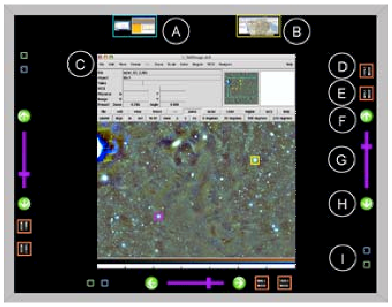

Figure 5. Control Panel in LivOlay. (A) Portal icon to Layout Manager. (B) Portal icon to LivOlay. (C) An application to be overlaid, with registered points displayed as pins. (D) Wall content switch between linked view and overlapped view. (E) Table content switch between control panel and overlapped view. (F,H) Display next/previous application. (G) Slider for transparency control. (I) Unused pins.

animated (Figure 4, *left-to-right*).

A **control panel** interface on the interactive tabletop provides users with control over LivOlay (Figure 5). The application image to be analyzed and overlapped is displayed in the center area of the control panel. Toolbars are duplicated and positioned along each edge of the tabletop to support egalitarian input. Registering and modifying a feature point in one application is achieved by dragging a pin from the toolbar and dropping it to the target position. A slider is provided to adjust the transparency of that application in the overlapped visualization. An overlapped view is also provided on the table, which is synchronized with the overlapped view on the wall. Switching between the control panel and the overlapped view on the table, as well as between the linked view and overlapped view on the wall, are triggered by touching corresponding buttons rendered on the table.

LivOlay also enables users to annotate on the table using a stylus. Annotations appear both on the table and on the wall, and are correctly transformed to register with each displayed application.

6. Feedback

After the development of the WeSpace, we made it available to the COMPLETE group. Three researchers conducted collaborative research sessions in the WeSpace. Two sessions were held, approximately 4 hours each, during which they brought in the charts and data that they were currently working on. We took notes of interesting events, video-taped the sessions, and logged input and system events. We also asked members of the group to describe their experiences with the system.

The positive results of these sessions demonstrated the WeSpace as a useful platform for visual exploration tasks. Conducting these collaboration sessions was extremely beneficial to the users, and to their delight (and ours) resulted in a new finding that led to a major research proposal.

Walk-Up and Share & Native Applications: Although two native applications were present in the system we tested (Layout Manager and LivOlay), it was LivOlay which received the most clear and positive responses from our users: the usefulness of being able to view and overlay data from different sources and from each user's laptop was clear, as they spent most of the session working with overlaid data.

Egalitarian Input: Our system recorded the number and type of input from each of the three scientists in the collaborative session. We analyzed these logs to address questions concerning the relative contribution from each member in terms of controlling the system, and directing the conversation.

Overall, the relative contribution from the three group members (22%, 33%, and 45%) in terms of number of input actions was fairly equal, which stands in contrast to a group's use of a single-user system: a situation in which the group member controlling the mouse and keyboard greatly influences the direction of conversation.

While the overall distribution of input was relatively even, the logs also showed that the input to the system in each short five or ten minute period was majorly given by one user. This suggested that the control of the system passed from user to user at different times during the meeting as the three participants took turns directing the conversation, which matches our observations of the meeting that the scientists took turns introducing new data to the conversation with their colleagues reacting to these additions.

Tangible Outcomes: The clearest evidence of the success of the WeSpace is the multiple, tangible scientific outcomes produced during the sessions. In all, the users reported that the work done during the sessions will enable them to submit a new observing proposal, as well as provide significant content to three scientific ongoing journal papers which otherwise would not have been made.

7. Conclusion and Future Work

WeSpace provides seamless integration of personal devices to a table-centered, multi-user, multi-surface environment, as well as customized visualization facilities for visual exploration. In this paper, we present the system design of the WeSpace, our contributions to the multi-surface collaborative system community are 1) the system infrastructure which allows users to connect and visually share their laptop content on-the-fly, and supports the extension of native visualization applications, and 2) the table-centric design employed in customized WeSpace applications to support cross-surface interactions.

Future work of this ongoing project includes: a) the development of WeSpace native applications to fit the particular requirements of other scientific domains, b) the incorporation of other cross-surface interaction hardware or techniques to the system, and c) a session recording facility, which saves ongoing screen images and control information through the session and allows interactive retrieval to the session afterwards.

8. Acknowledgements

We deeply thank the COMPLETE group, without whom this research could not have been performed. This research was conducted at Mitsubishi Electric Research Labs. Hao Jiang is partially supported by the 863 Program, grant number 2006AA01Z131 and 2008AA01Z132.

9. References

[1] Baudisch, P., Cutrell, E., Hinckley, K., Gruen, R. (2004). Mouse ether: accelerating the acquisition of targets across multi-monitor displays. In *Proc. CHI 04*, 1379-1382.
[2] Biehl, J. T., Bailey, B. P. (2004). ARIS: an interface for application relocation in an interactive space. In *Proc. of Graphics interface 04*, 107-116.
[3] Biehl, J. T., Bailey, B. P. (2006). Improving scalability and awareness in iconic interfaces for multiple-device environments. In *Proc. AVI 06*, 91-94.
[4] Biehl, J.T., Baker, W.T., Bailey, B.P., Tan, D.S., Inkpen, K.M., Czerwinski, M. (2008). IMPROMPTU: A New Interaction Framework for Supporting Collaboration in Multiple Display Environments and Its Field Evaluation for Co-located Software Development. In *Proc. CHI 08*, 939-948.
[5] Booth, K.S., Fisher, B.D., Lin, C.J., Argue, R. (2002). The "mighty mouse" multi-screen collaboration tool. In *Proc. UIST 02*, 209-212.
[6] Dietz, P., Leigh, D. (2001). DiamondTouch: a multi-user touch technology. In *Proc. UIST 01*, 219-226.
[7] Everitt, K., Shen, C., Ryall, K., Forlines, C. (2006). MultiSpace: Enabling Electronic Document Micro-mobility in Table-Centric, Multi-Device Environments. In *Proc. Tabletop 06*, 27-34.
[8] Fischer, M., Stone, M., Liston, K., Kunz, J., Singhal, V. (2002). Multi-stakeholder collaboration: The CIFE iRoom. In *Proc. CIB W78 Conference 2002: Distributing Knowledge in Building*, 6-13.
[9] Forlines, C., Esenther, A., Shen, C., Wigdor, D., Ryall, K. (2006). Multi-user, multi-display interaction with a singleuser, single-display geospatial application. In *Proc. UIST 06*, 273-276.
[10] Forlines, C., Lilien, R. (2008) Adapting a Single-user, Single-display Molecular Visualization Application for Use in a Multi-user, Multi-Display Environment. In *Proc. AVI 08*.
[11] Isenberg, T., Miede, A., Carpendale, S. A buffer framework for supporting responsive interaction in information visualization interfaces. (2006). In *Proc. Int. Conf. Creating, Connecting and Collaborating through Computing (C5'06), IEEE Computer Society*, 262–269.
[12] Izadi, S., Brignull, H., Rodden, T., Rogers, Y., Underwood, M. (2003). Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media. In *Proc. of UIST 03*, 159-168.
[13] Jiang, H., Wigdor, D., Forlines, C., Borkin, M., Kauffmann, J., Shen, C. (2008). LivOlay: Interactive Ad-Hoc Registration and Overlapping of Applications for Collaborative Visual Exploration. In *Proc. CHI 08*, 1357-1360.
[14] JOGL API. https://jogl.dev.java.net/
[15] Johanson, B., Fox. A., Winograd, T. (2002). The Interactive Workspaces Project: Experiences with Ubiquitous Computing Rooms. *IEEE Pervasive Computing*, v.1, n.2 (April 2002), 67-74.
[16] Johanson, B., Hutchins, G., Winograd, T., Stone, M. (2002). PointRight: experience with flexible input redirection in interactive workspaces. In *Proc. UIST 02*, 227-234.
[17] Johanson, B., Ponnekanti, S., Sengupta, C., Fox, A. (2001). Multibrowsing: Moving Web Content across Multiple Displays. In *Proc. UbiComp 01*, 346-353.
[18] Miura, M., Shizuki, B., Tanaka, J. (2004). comDesk: A Cooperative Assistance Tool Based on P2P Techniques. In *Proc. KES 04, LNAI3214*, 883-890,
[19] Nacenta, M.A., Sallam, S., Champoux, B., Subramanian, S., Gutwin, C. (2006). Perspective cursor: perspective-based interaction for multi-display environments. In *Proc. CHI 06*, 289-298.
[20] Prante, T., Streitz, N., Tandler, P. (2004). Roomware: Computers Disappear and Interaction Evolves. *Computer 37, 12 (Dec. 2004)*, 47-54.
[21] RealVNC. http://www.realvnc.com/
[22] Rekimoto, J., Saitoh, M. (1999). Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *Proc. CHI 99*, 378-385.
[23] Shen, C., Everitt, K.M., Ryall, K. UbiTable: Impromptu Faceto-Face Collaboration on Horizontal Interactive Surfaces. In *Proc. UbiComp 03, LNCS 2864*, 281-288.
[24] Shen, C., Vernier, F. D., Forlines, C., and Ringel, M. (2004). DiamondSpin: an extensible toolkit for around-the-table interaction. In *Proc. CHI 04*, 167-174.
[25] Stefik, M., Foster, G., Bobrow, D.G., Kahn, K., Lanning, S., Suchman, L. Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Commun. ACM 30, 1 (Jan. 1987)*, 32-47.
[26] Streitz, N.A., Geißler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., Steinmetz, R. (1999). i-LAND: an interactive landscape for creativity and innovation. In *Proc. CHI 99*, 120-127.
[27] Tan, D.S., Gergle, D., Scupelli, P., Pausch, R. (2006). Physically large displays improve performance on spatial tasks. *ACM Trans. Computer-Human Interaction 13, 1 (Mar. 2006)*, 71-99.
[28] Tan, D.S., Meyers, B., Czerwinski, M. (2004). WinCuts: manipulating arbitrary window regions for more effective use of screen space. In *CHI ' 04 Extended Abstracts*, 1525-1528.
[29] Tuddenham, P., Robinson, P. (2007). T3: Rapid Prototyping of High-Resolution and Mixed-Presence Tabletop Applications. In *Proc. Tabletop 2007*, 11-18.
[30] Wallace, G., Li, K. (2007). Virtually Shared Displays and Input Devices. In *Proc. of the 2007 USENIX Annual Technical Conference*, 375-379.
[31] Wigdor, D., Shen, C., Forlines, C., Balakrishnan, R. (2006). Table-centric interactive spaces for real-time collaboration. In *Proc. AVI 06*, 103-107.