CGLXTouch: A multi-user multi-touch approach for ultra-high-resolution collaborative workspaces

Kevin Ponto, Kai Doerr, Tom Wypych, John Kooker, Falko Kuester

Graphics, Visualization and Virtual Reality Lab (GRAVITY)
University of California, San Diego
La Jolla, California, 92093-0436, USA

Abstract

This paper presents an approach for empowering collaborative workspaces through ultra-high-resolution tiled display environments concurrently interfaced by multiple multi-touch devices. Multi-touch table devices are supported along with portable multi-touch tablet and phone devices, which can be added to and removed from the system on the fly. Events from these devices are tagged with a device id and are synchronized with the distributed display environment enabling multi-user support. As portable devices are not equipped to render content directly, a rendered viewpoint is streamed in. The presented approach scales for large numbers of devices, providing a multitude of hands-on techniques for collaborative data analysis.

Keywords: Collaborative Workspaces, Multi-Touch Interaction, High-Resolution Displays, Multi-Device Interaction, Tiled Displays

1. Introduction

Multi-user colocated collaborative research environments present unique opportunities for scientific exploration and analysis. While these collaborative workspaces have been shown to have great utility, developing these

environments effectively is a challenging endeavor [1]. In order to keep users engaged, these working environments must present users with the ability to see and interact with content. As group sizes grow to more than a few users, this requirement presents challenges for standard interface modalities.

Figure 1: Users manipulating data on tiled display system using four different multi-touch devices concurrently.

Newly developed large display environments present the visual capabilities of supporting numerous users simultaneously [2] [3]. Through the ultra-high-resolution nature of these systems, many users can analyze the same data set simultaneously. This kind of physical navigation has been shown to out perform the standard pan and zoom paradigms [4].

Unfortunately, traditional means to interface with these types of systems generally do not work well in multi-user paradigms as interactions are often controlled from standard single user mouse and keyboard interfaces. Additionally, mouse and keyboard interaction paradigms are not readily mobile, limiting the utility of the high resolution display environment. Multi-user interface modalities developed for ultra-high-resolution display spaces have generally provided specialized devices and interaction modalities, requiring users to learn new paradigms before interfacing with the workspace. Furthermore, many of the previously built interaction modalities do not scale for more than a few users.

Recent advancements in consumer electronics have generated an explosion of multi-touch capable devices. Multi-touch devices have been shown to be a powerful interface, harnessing the natural dexterity of the user while pairing established interface modalities when needed. Users carry many of these devices on their person, providing a familiar and readily usable hardware interface suitable for interaction with a collaborative workspace.

In order to build on these recent technological advances, we present an approach enabling multiple users to interface to a digitally enabled workspace, using commodity as well as specialized multi-touch devices (Figure 1). This approach enables users to work collaboratively through ultra-high-resolution distributed display environments serving as a shared canvas. Two different types of multi-touch input devices are utilized consisting of (1) natively rendered multi-touch table devices and (2) mobile, hand-held multi-touch devices that receive streamed visual data. This approach allows complex data sets to be investigated across a broad spectrum of multi-touch capable de-

vices, each of which can be added to and removed from the system on the fly. In this setting, users can be uniquely identified through their devices and as such, this approach lays the foundation for scalable multi-touch, multi-user workspaces / environments.

Below, we first present the related work from fields of collaborative workspaces, interface design and visual analytics. Next, we describe our technical approach, focusing on the software infrastructure. The multi-touch devices are described in the subsequent section, followed by a description of a few initial applications designed for this system. Finally, the system is tested for various capabilities and the results are shown.

2. Previous Work

2.1. Digital Collaborative Workspaces

Digitally enabled collaborative workspaces have shown great potential for a multi-user approach for scientific analysis [5]. Heer and Agrawala outline the strengths and challenges for multi-user environments in their paper "Design Considerations for Collaborative Visual Analytics"[1]. "The office of the future" prototyped a virtual collaborative working environment for the workplace. Churchill et al. examined the history of these types of spaces, the technical issues and challenges these types of systems present, and the types of applications enabled through this technology [5]. Taesombut et al. explored real-time collaborative efforts to analyze earth science data on tiled-dispaly systems [6]. Held and Blochinger studied how to design workflows for collaborative workspaces [7]. Scott et al. examine how to develop colocated collaborative workspaces for table top environments [8]. Shen et

al. continued in the same direction and developed a software architecture for multi-user table interactions [9]. Peltonen et al. studied how multiple users interacted on a a multi-touch enabled display wall [10], analyzing how users negotiated their interaction space and how conflicts between users were resolved. Renambot el al. and Hutanu etl al. explored collaborative environments through high-performance network streaming of visual data [11] [12]. Lloyd et al. explored the challenges of developing collaborative environments for heart and cancer modelling [13].

2.2. Tiled Display Environments

Colocated collaborative workspaces require information to be presented in support of multiple users. Scalable, ultra-high resolution display walls have been shown to address these requirements [2] [3]. In order to facilitate the creation of a unified display canvas, specialized middleware is required to drive these types of environments.

One way of controlling tiled-display walls is to create a virtual unified display, as shown by Chromium [14] and OpenSG [15] . The strength of this approach is that it works with most applications and may not require code to be recompiled. However, this approach generally does not work with textures and shaders, has poor synchronization mechanisms and does not lend itself to ultra-high resolution data rendering.

Another approach is to send pixel content to display nodes as outlined in Scalable Adaptive Graphics Environment (SAGE) [16] [17]. In this approach one system renders content into a buffer that is then mapped to the tiled display environment. This buffer is segmented to match the configuration of the wall and is streamed out via network. The advantage of this approach

is that data needs to be only processed on a single computer, while render nodes only need to display the resulting image, meaning they only need minimal processing power. Also, data driven applications need to exist only on the streaming node to fill the pixel buffer. This technique, however, is not effective for rendering of dynamically changing ultra-high-resolution content, as the network bandwidth requirements are too great.

The distributed application approach as shown in VRJuggler [18], and CGLX [19] allow for a tiled display environment to also act as a distributed computer. In this approach, the same application is started on the tiled display environment and its head node and the viewpoint is changed on each rendering node to match the appropriate subsection of the of the head node allowing the workload of rendering content to distributed over an entire cluster. All user events such as key-presses and mouse movements as well as graphical events such as buffer swaps are synchronized. The advantage of this approach is that it requires very little network bandwidth in order to operate and has the ability to scale well beyond the other approaches mentioned. Furthermore, the approach allows computation to be distributed throughout the display environment and allows graphic card shaders to run natively. The disadvantage of this approach is that code must be designed for the tiled display environment. Additionally, extra control signals must be used to assure every node produces a contiguous display environment.

2.3. Multi-user Interaction Techniques

There have been several approaches taken to enable multi-user interactions on these type of large display environments. "Fluid Interaction with High-resolution Wall-size Displays" by Frangois et al. showed how a pen in-

terface could be used in the context of display walls [20] while also enabling user identification through handwriting analysis. "Lumipoint: multi-user laser-based interaction on large tiled displays" by James Davis and Xing Chen examined techniques for enabling multi-user interaction through laser pointers [21]. Laser pointers position and velocity were tracked through a central server, and multi-user support was enabled by adding laser pointers of differing colors. This approach was found to be scalable by adding additional cameras.

Malik et al. explored the use of touch pads to interact with a display wall in "Interacting with large displays from a distance with vision-tracked multi-finger gestural input" [22]. This system allowed two users to simultaneously use the same working environment. Cameras were used to determine hand placement and allowed hand-specific gestures which were used to overcome the small touchpad size to wall size.

Using a passive wand for interaction with tiled display environments has also been explored by Cao and Balakrishnan [23]. From this system, users could perform gestures to select, move, and rotate. While the system worked well for a single user, it showed performance degradation with additional users.

Ringel et al. explored a multi-user virtual whiteboard in "Barehands: Implement-Free Interaction with a Wall-Mounted Display", using diffuse illumination in order to track upwards of 120 touches simultaneously [24]. Another whiteboard approach presented by Rekimoto proposed a system for allowing many users to interface with a whiteboard through PDAs [25]. Rekimoto's proposed system enable content to be transfered directly from user's

PDA to a virtual whiteboard environment.

3. Technical Approach

The objective of the presented approach is to create an collaborative environment in which multiple users can interface with ultra-high resolution distributed digital workspaces via commodity as well as custom developed multi-touch devices. Standard interface devices limit speed of interaction, creating a bottleneck in data analysis [26]. Multi-touch systems leverage users real world experiences as a computer interface. Huang showed that gestures are physical expressions of mental concepts [27]. Multi-touch interfaces enable simple gestures which can convey complicated interactions [26], many of which have become standardized over different multi-touch platforms. These devices have been shown to successfully interfaced in CAVE type environments [28].

Furthermore, recent advancements in consumer electronics have generated an explosion of multi-touch capable devices. Users carry many of these devices on their person (e.g. cell phones, tablets, etc.), providing a known and readily usable hardware interface into the tiled display system. The familiarity of hardware and interface paradigms provides a major advantage over the approaches shown in Section 2.3 as user's can interface with the system with little guidance.

Additionally, previous approaches which used object tracking were shown not to scale beyond a few users [23] . Other approaches, such as [24], had no ability to differentiate different users, providing limited multi-user support. By building an infrastructure to support multiple multi-touch devices, it is

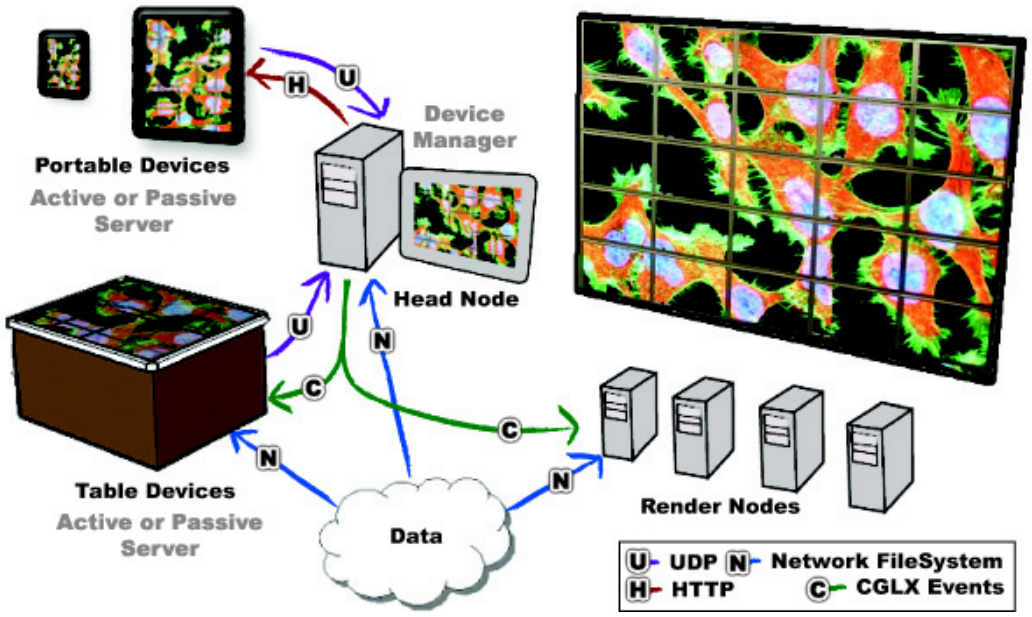


Figure 2: Diagram of the system architecture.

possible to gain the benefits of multi-touch interaction, differentiate users based on device and provide an approach capable of scaling to a plethora of devices.

To achieve this is was critical to

- Develop a framework to manage various multi-touch devices
- Gather input from these devices and forward the touch inputs onto distributed display systems
- Allow these devices to be connected and removed from the system interactively
- Visualize interactions on all devices either natively or through streaming

as shown in Figure 2. In this approach, touch events are from multi-touch devices are first sent to the head node and are then reliably forwarded to tiled display system. As table devices have the capabilities to render content natively, they are also passed these events from the head node and can pull data from remote storage. The portable devices on the other hand do not have the performance characteristics needed, so visual information must be streamed into them on the fly.

3.1. Middleware

With the requirement to render and display data at high-resolution, matching the native capabilities of the displays, a distributed approach was implemented based on the CGLX (Cross-Platform Cluster Graphic Library) middleware [19]. CGLX determines the correct projection and transformation matrices in order to create uniform display environment and provides mechanisms to synchronize user events, such as mouse and keyboard I/O and events are propagated reliably and efficiently throughout the tiled display environment.

As shown in Figure 2, multi-touch devices communicate with the head node directly via UDP, satisfying the needed performance characteristics, at the risk of occasionally unreliable data transport, which was shown to be acceptable in the single device context. From the head node on, data is then distributed reliably via the CGLX event mechanism [19].

3.2. Device Manager

Support for multiple, simultaneously connected devices is provided through the Device Manager. As multiple devices could be connected simultaneously,

a new layer was created to oversee these device servers called the Device Manager. The Device Manager is responsible for establishing connections with new devices either actively or passively and gathering and forwarding events to the system. Active devices are maintained through a server pool data structure.

3.2.1. Passive Servers

Passive Servers are designed for devices which should routinely be connected. When the head node is up and running, it will regularly check to see if these devices are active. In this way, users do not need to explicitly configure connections between the device and the head node. The workflow for the connection process of Passive Servers is shown in Figure 3.

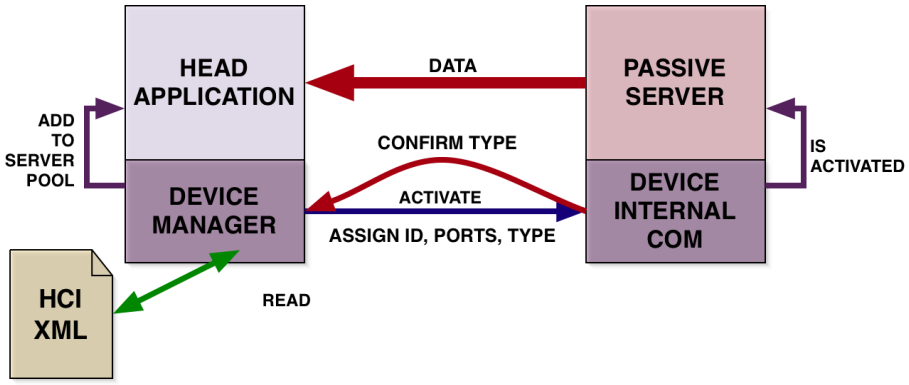


Figure 3: Diagram showing the process of connecting a passive server application.

The desired passive servers are defined via a configuration file in XML format that is parsed during startup. The Device Manager then attempts to activate each of the listed servers. When the Passive Server receives an activation request, it checks if the server type matches the type specified in the XML file. If this server is able to validate this information, it sets its

id and communication ports based on the information sent from the device manager. Additionally, the device manager adds this server to the active server pool on the head node application. The Passive server with the updated information and assigned unique ID is then allowed to send data to the head node.

3.2.2. Active Servers

Active Servers are used for devices which would like to connect to a running CGLX application. This method removes the need for pre-configuration of servers, but requires the device to know the IP address of the head node. Furthermore, the device will need to reconnect each time a new CGLX application is started. The workflow for the connection process of Active Servers is shown in Figure 4.

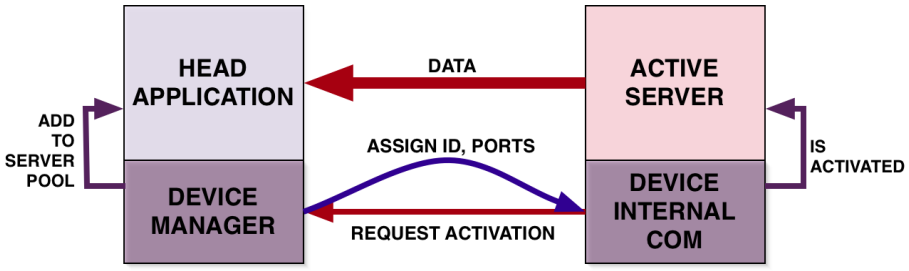


Figure 4: Diagram showing the process of connecting a active server application.

To accomplish this, the Active Server contacts the Device Manager on the Head Node. If the Device Manager accepts the request, it sends the id for the device to use and the port to communicate on. At this point the Device Manager adds this server pool for the head node application, the server is free to send data to the head node.

4. Multi-Touch Devices

In this project two different types of multi-touch devices were utilized; table multi-touch devices and portable multi-touch devices.

4.1. Table Devices

Touch table devices are developed using frustrated total internal reflection (FTIR) technology with a silicone film to enable pressure sensitivity, similar to the approach shown in [29]. Touches are tracked via a camera input, with size and pressure attributes. Using GPU filtering, several hundred touches can be tracked at highly-interactive-rates [29].

As touch table devices do not generally need to be mobile, they can be equipped with powerful computation and network hardware. This localized computational power enables content to be rendered natively. Additionally, the wired network connection allows data to be pulled from network mounted drives. In this way, the machine running the display for the touch table can also be incorporated into the tiled display system either as a head node or as an alternative render node.

4.2. Portable Devices

Portable multi-touch devices allow users to freely move throughout the environment. These devices can be interactively added and removed from the system using the active server approach described above, allowing users to simply walk into the space and begin interacting with the system.

The system is built around multi-touch capable phones and tablet devices. Multi-touch phone devices are generally able to track 5 touches simultaneously, while larger tablets which are able to track 11 touches simultaneously.

These devices are not able to track size and pressure attributes, only centroids for each touch.

4.2.1. Streaming

For many applications, the portable devices do not have the computational power or data access to render content natively. One example comes from localized interrogation application in which requires high-bandwidth network access and GPU based filtering. Furthermore, writing applications designed to work both the tiled-display environment and for portable devices may be extremely challenging and not always possible.

However, giving users visual feedback of their interactions was shown to be extremely important in initial user studies. As the head node has the ability to render preview resolutions of the tiled display system, image data can be streamed back to portable devices. After each frame is rendered on the head node, the contents of front buffer are read from the graphics card, encoded and sent via the network interface.

Initial testing used raw pixel streaming sent via UDP communication. This method was advantageous as data can be rapidly transmitted and multicast communication can stream data to multiple devices via a single stream. Unfortunately, the high network requirements and transport mechanisms caused this approach to be unreliable when transmitted through public networks.

In order to create a reliable transport mechanism, a lightweight http server is created on the head node to which devices could connect. Image data is encoded into a jpeg image to reduce transfer size and is passed to portable device on a fixed interval. The method is effectively able to push

10 fps from the head node to the portable devices.

5. Applications

Developing applications for multi-user environments is a challenging process. As shown in [1], collaborative environments require a substantial amount of thought and effort to be designed effectively. As the purpose of this paper is to build a framework for multi-user support, a few initial test applications are created but the majority of application development is left for future work.

5.1. Whiteboard

One application which works well for this multi-user environment is a collaborative whiteboard as shown in [25] and [24]. To accomplish this we created a framebuffer object on a per monitor basis for each display node, enabling pixel-accurate rendering on the ultra-high-resolution display system. In this application, each user selects a brush and color through the designed interface. For each touch received, a brush texture is rendered into this framebuffer object in a similar approach to [29]. This render-to-texture approach enabled multiple users to fully interact with the system (Figure 5).

This approach was most successful on table devices where users could interface with their fingers, felt tipped markers, or other physical interfaces. As the multi-touch devices had a variety of display and touch resolutions, future work will aim to allow users to select user defined subregion to view and modify. Furthermore, sending markup data will be advantageous, as text information may be easier to create on smaller portable devices.

Figure 5: Users manipulating a virtual whiteboard via touch table device and two separate portable devices. Images shown on the portable devices are streamed from the head node.

5.2. Sorting

A common multi-touch application enables two-dimensional objects to be rotated, translated and scaled using simple gestures. By enabling this on a large-scale display system, multiple users can simultaneously sort through an array of objects as shown in Figure 1. Data was fetched via a network mounted drive for the render node and table devices while screens images were sent to the portable devices.

This application worked well for all devices, but many users did find the small screen of the phone devices to make rotation and scaling gestures somewhat challenging if the images did not fill the screen. The main issue of this application was the contention between users as shown in [10]. These problems can be somewhat mitigated by locking objects to a particular user,

but overlapping viewpoints still pose a visual problem.

5.3. Localized Interrogation

As opposed to locking objects all together, techniques can also be performed on individual sections. Ponto et al. [29] and Bonanni et al. [30] demonstrated how localized interrogation of cultural artifacts could prove effective for the field of cultural heritage. With the addition of the visualization capabilities of the tiled-display system, multiple users with personal devices can readily be supported a shown in Figure 6.

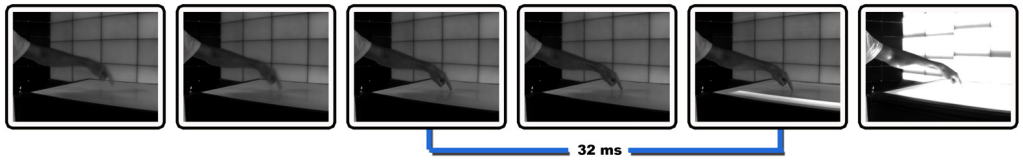
The application developed used a similar approach to that shown in [29]. Large images are converted into a pyramidal tiled TIFF files through Vips [29]. These images are then put on a network mounted drive and data is rendered on a per node basis in a similar fashion to [31]. The framebuffers described in [29] are now created on a per monitor basis allowing for pixel accurate interrogation on the large display environment as described in the section above.

This technique allows users to explore sub-regions of layered data without disturbing the global viewpoint. Unfortunately, translating and scaling content layers causes users workspaces to be disrupted. Additionally, users are not restricted to areas of interaction causing potential conflicts. Future work may enable user selected interaction windows which help mitigate these conflicts and give users a higher precision for their touch interactions.

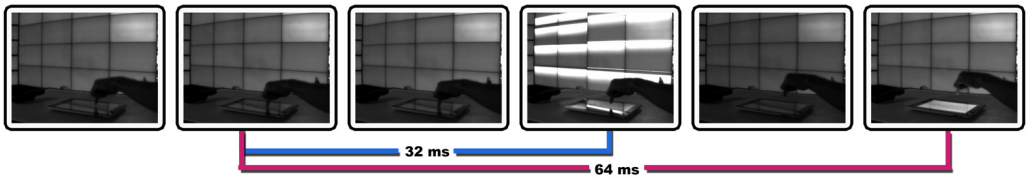
6. Results

To measure the performance of the presented approach, the latency and throughput of the multi-touch interaction is tested. The network bandwidth

Figure 6: User's wiping off individual spectral layers of "The Adoration of the Magi" by Lenardo da Vinci courtesy of Maurizio Seracini and CISA3 at the University of California, San Diego [29].



(a) Table Device



(b) Portable Device

Figure 7: Video frames used to determine latency for table and portable devices. Times are calculated from when the user touches the device to when the system swaps from a black image to a white image with an accuracy of ± 8 ms.

is used to calculate the theoretical maximum number of devices supported via the network constraints.

The table device mirrored the performance characteristics found in [29]. For the tablet device, performance characteristics are described in [32]. The described tests used the OptiPuter network [2]. Each of the tests was performed on a single device.

6.1. Interactivity

As user interaction was of the upmost importance, it was important to measure the time between user actions and visual results. To accomplish this, a simple program is created that displays a black frame which is switched to a white frame when a touch is detected. The entire process is recorded via a 60 Hz grayscale camera giving a measurement of latency ± 8 ms. As

the camera had the same frame rate as the displays, recorded frames may be caught mid update (Figure 7(b)).

As shown in Figure 7(a), it takes two frames from when the table device is touched before the result is shown, resulting in approximately 32 ms of latency. Figure 7(b) shows that the time from touching the portable device until the tiled display system shows the update is also 32 ms. Furthermore, the streamed result takes an additional two frames until the result is shown on the portable device, resulting in a latency of approximately 64 ms. Each of these systems is able to send touch events at 60 Hz even when fully saturated. This means after the initial latency, updates are received on 16 ms intervals.

6.2. Capabilities

As each application may process touch events differently, we instead decided to test our system based on network performance. Assuming that the application could process the events, it is possible to define a theoretical lower bound of the maximum member of devices which could interface with the system given the constraint that all devices are fully saturated. As in practice users tend to vary their interaction, the actual maximum number of devices connected to the system could be much greater.

Figure 8 shows an estimate for the maximum number of devices possible in our system based only on the network bandwidth. This is determined by dividing the maximum network bandwidth for the system by the maximum information the multi-touch device could produce. The assumption is made that each device is fully saturated on tracked touches, with 5 touches for phone devices, 11 for tablets devices and 500 for table devices. Phone and tablet devices are assumed to communicate on a 54Mb wireless connec-

tion and the table devices are assumed to communicate on a 1 Gb wired connection.

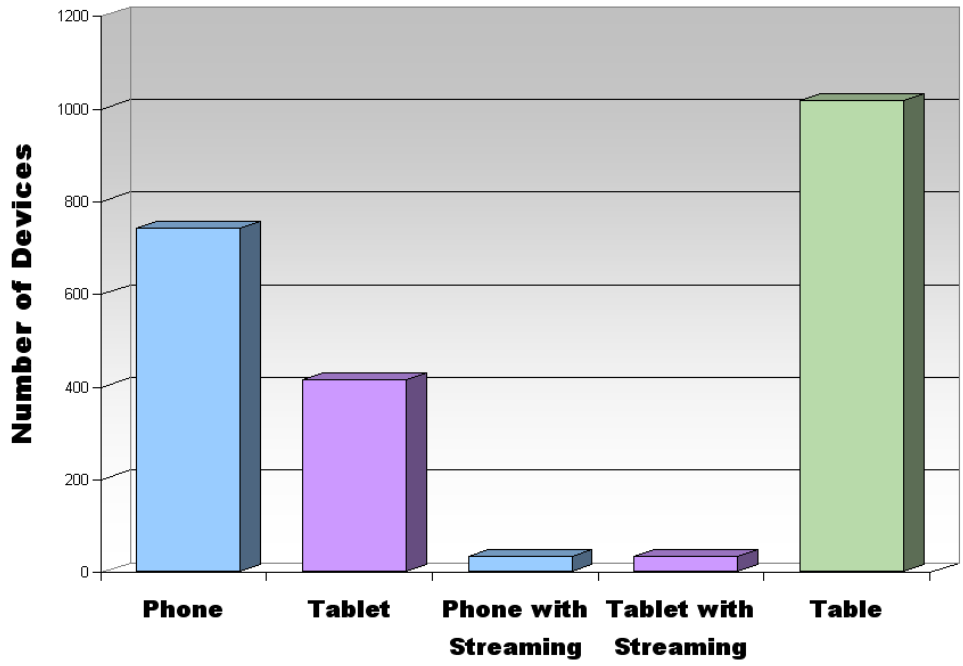


Figure 8: The theoretical maximum number of devices given network constraints

As shown, even when these devices are fully saturated with events, the system can interface with hundreds of phone, tablet and table simultaneously. This number is reduced substantially when the head node is required to stream information back to the devices, falling from several hundred to around 30 devices for both the phone and the tablet. This was due to the substantial increase in network bandwidth required from streaming visual information as opposed to just touch information. This increase could be easily offset by adding extra wireless routers, or adding higher bandwidth network interfaces. Future work for compressing these streams would reduce the net-

work bandwidth requirements, thus increasing the number of streamed to devices.

7. Conclusion

This paper presents a multi-user multi-touch environment for large ultra-high-resolution display systems. Devices can be interactively added and removed from the workspace during runtime and can be uniquely identified, enabling user specific actions. The presented method supports hundreds of multi-touch devices simultaneously, enabling a collocated collaborative workspace.

While this system presents the initial framework for a multi-user, multi-touch system, much work is still left to be done. A software infrastructure to support these kind of multi-user multi-touch scenarios needs to be built from the ground up. Different modalities as to how to deal with conflicting interactions need to be further investigated. Additionally, the proposed framework for adding and removing devices in this type of environment could be extended to other types of devices with additional software engineering.

8. Acknowledgments

The presented research was supported in part by the National Science Foundation (NSF) under award #DGE-0966375, the University of California Chancellor's Interdisciplinary Collaboratories Program, the Jacobs School of Engineering (JSoE), the California Institute for Telecommunications and Information Technology (Calit2), the Friends of the Center of Interdisciplinary Science for Art, Architecture and Archeology (CISA3) and is based in part

on work supported by Award No. US 2008-107, made by the King Abdullah University of Science and Technology (KAUST).

References

[1] J. Heer, M. Agrawala, Design considerations for collaborative visual analytics, Information Visualization 7 (1) (2008) 49–62.

[2] T. A. DeFanti, J. Leigh, L. Renambot, B. Jeong, A. Verlo, L. Long, M. Brown, D. J. Sandin, V. Vishwanath, Q. Liu, M. J. Katz, P. Papadopoulos, J. P. Keefe, G. R. Hidley, G. L. Dawe, I. Kaufman, B. Glogowski, K.-U. Doerr, R. Singh, J. Girado, J. P. Schulze, F. Kuester, L. Smarr, The optiportal, a scalable visualization, storage, and computing interface device for the optiputer, Future Generation Computer Systems 25 (2) (2009) 114–123.

[3] L. Smarr, M. Brown, C. de Laat, Editorial: Special section: Optiplanet - the optiputer global collaboratory, Future Generation Computer Systems 25 (2) (2009) 109–113.

[4] R. Ball, C. North, Effects of tiled high-resolution display on basic visualization and navigation tasks, in: CHI '05 extended abstracts on Human factors in computing systems, ACM, New York, NY, USA, 2005, pp. 1196–1199.

[5] E. Churchill, D. Snowdon, A. Munro, Collaborative virtual environments: digital places and spaces for interaction, Springer-Verlang, 2001.

[6] N. Taesombut, X. R. Wu, A. A. Chien, A. Nayak, B. Smith, D. Kilb, T. Im, D. Samilo, G. Kent, J. Orcutt, Collaborative data visualization for earth sciences with the optiputer, Future Generation Computer Systems 22 (8) (2006) 955 – 963.

[7] M. Held, W. Blochinger, Structured collaborative workflow design, Future Generation Computer Systems 25 (6) (2009) 638 – 653.

[8] S. D. Scott, K. D. Grant, R. L. Mandryk, System guidelines for co-located, collaborative work on a tabletop display, in: ECSCW'03: Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work, Kluwer Academic Publishers, Norwell, MA, USA, 2003, pp. 159–178.

[9] C. Shen, F. D. Vernier, C. Forlines, M. Ringel, Diamondspin: an extensible toolkit for around-the-table interaction, in: CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, New York, NY, USA, 2004, pp. 167–174.

[10] P. Peltonen, E. Kurvinen, A. Salovaara, G. Jacucci, T. Ilmonen, J. Evans, A. Oulasvirta, P. Saarikko, It's mine, don't touch!: interactions at a large multi-touch display in a city centre, in: CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems, ACM, New York, NY, USA, 2008, pp. 1285–1294.

[11] A. Hutanu, G. Allen, S. D. Beck, P. Holub, H. Kaiser, A. Kulshrestha, M. Liska, J. MacLaren, L. Matyska, R. Paruchuri, S. Prohaska, E. Sei-

del, B. Ullmer, S. Venkataraman, Distributed and collaborative visualization of large data sets using high-speed networks, Future Generation Computer Systems 22 (8) (2006) 1004 – 1010.

[12] L. Renambot, B. Jeong, H. Hur, A. Johnson, J. Leigh, Enabling high resolution collaborative visualization in display rich virtual organizations, Future Generation Computer Systems 25 (2) (2009) 161 – 168.

[13] S. Lloyd, D. Gavaghan, A. Simpson, M. Mascord, C. Seneurine, G. Williams, J. Pitt-Francis, D. Boyd, D. M. Randal, L. Sastry, S. Nagella, K. Weeks, R. Fowler, D. Hanlon, J. Handley, G. de Fabritiis, Integrative biology – the challenges of developing a collaborative research environment for heart and cancer modelling, Future Generation Computer Systems 23 (3) (2007) 457 – 465.

[14] G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett, P. Hanrahan, Wiregl: a scalable graphics system for clusters, in: SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, New York, NY, USA, 2001, pp. 129–140.

[15] G. Voß, J. Behr, D. Reiners, M. Roth, A multi-thread safe foundation for scene graphs and its extension to clusters, in: EGPGV '02: Proceedings of the Fourth Eurographics Workshop on Parallel Graphics and Visualization, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2002, pp. 33–37.

[16] L. Renambot, A. Rao, R. Singh, B. Jeong, N. Krishnaprasad, V. Vishwanath, V. Chandrasekhar, N. Schwarz, A. Spale, C. Zhang, et al., Sage:

the scalable adaptive graphics environment, in: Proceedings of WACE, 2004, pp. 2004–09.

[17] B. Jeong, L. Renambot, R. Jagodic, R. Singh, J. Aguilera, A. Johnson, J. Leigh, High-performance dynamic graphics streaming for scalable adaptive graphics environment, in: SC '06: Proceedings of the 2006 ACM/IEEE conference on Supercomputing, ACM, New York, NY, USA, 2006, p. 108.

[18] A. Bierbaum, C. Just, P. Hartling, K. Meinert, A. Baker, C. Cruz-Neira, Vr juggler: a virtual platform for virtual reality application development, in: SIGGRAPH Asia '08: ACM SIGGRAPH ASIA 2008 courses, ACM, New York, NY, USA, 2008, pp. 1–8.

[19] K.-U. Doerr, F. Kuester, CGLX: A Scalable, High-performance Visualization Framework for Networked Display Environments, IEEE Transactions on Visualization and Computer Graphics 99 (PrePrints).

[20] F. Guimbretière, M. Stone, T. Winograd, Fluid interaction with high-resolution wall-size displays, in: UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology, ACM, New York, NY, USA, 2001, pp. 21–30.

[21] J. Davis, X. Chen, Lumipoint: multi-user laser-based interaction on large tiled displays, Displays 23 (5) (2002) 205 – 211.

[22] S. Malik, A. Ranjan, R. Balakrishnan, Interacting with large displays from a distance with vision-tracked multi-finger gestural input, in: UIST

'05: Proceedings of the 18th annual ACM symposium on User interface software and technology, ACM, New York, NY, USA, 2005, pp. 43–52.

[23] X. Cao, R. Balakrishnan, Visionwand: interaction techniques for large displays using a passive wand tracked in 3d, in: SIGGRAPH '04: ACM SIGGRAPH 2004 Papers, ACM, New York, NY, USA, 2004, pp. 729–729.

[24] M. Ringel, H. Berg, Y. Jin, T. Winograd, Barehands: implement-free interaction with a wall-mounted display, in: CHI '01: CHI '01 extended abstracts on Human factors in computing systems, ACM, New York, NY, USA, 2001, pp. 367–368.

[25] J. Rekimoto, A multiple device approach for supporting whiteboard-based interactions, in: CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1998, pp. 344–351.

[26] V. Pavlovic, R. Sharma, T. Huang, Visual interpretation of hand gestures for human-computer interaction: a review, Pattern Analysis and Machine Intelligence, IEEE Transactions on 19 (7) (1997) 677 –695.

[27] R. Sharma, T. S. Huang, V. I. Pavlovi'c, Y. Zhao, Z. Lo, S. Chu, K. Schulten, A. Dalke, J. Phillips, M. Zeller, W. Humphrey, Speech/gesture interface to a visual computing environment for molecular biologists, IEEE Computer Graphics and Applications (1996) 30–35.

[28] Ó. Belmonte, M. Castañeda, D. Fernández, J. Gil, S. Aguado, E. Varella, M. Nuñez, J. Segarra, Federate resource management in a distributed

virtual environment, Future Generation Computer Systems 26 (3) (2010) 308 – 317.

[29] K. Ponto, M. Seracini, F. Kuester, Wipe-Off: An Intuitive Interface for Exploring Ultra-Large Multi-Spectral Data Sets for Cultural Heritage Diagnostics, Computer Graphics Forum 28 (8) (2009) 2291–2301.

[30] L. Bonanni, X. Xiao, M. Hockenberry, P. Subramani, H. Ishii, M. Seracini, J. Schulze, Wetpaint: scraping through multi-layered images, in: CHI '09: Proceedings of the 27th international conference on Human factors in computing systems, ACM, New York, NY, USA, 2009, pp. 571–574.

[31] K. Ponto, K. Doerr, F. Kuester, Giga-stack: A method for visualizing giga-pixel layered imagery on massively tiled displays, Future Generation Computer Systems 26 (5) (2010) 693 – 700.

[32] R. Lewis, iPhone and iPad Apps for Absolute Beginners, Apress, Berkely, CA, USA, 2010.

Kevin Ponto is a Ph.D Candidate at the University of California, San Diego (UCSD) in the Department of Computer Science and Engineering and has background in both the fields of Arts and Engineering. His research is aimed at creating natural and intuitive ways to explore and interrogate large and complex data sets. Furthermore, Kevin's research aims to at creating ultra-high resolution display environments combined with visceral interface technologies to allow for multi-user collaborative

workspaces.

Tom Wypych is a doctoral researcher in the Computer Science and Engineering Department at UCSD. His research encompasses hardware and software design of large-format, high-resolution, high-energy, x-ray computer tomography for non-destructive testing. He is also working on embedded programming on mobile and accelerated processors, implementing real time control systems, dataflow-centric applications, ultra-reliable controls, and media coding applications. Current research is focused on delivering low-latency, high-resolution content over low-bandwidth links to mobile and fixed display terminals.

John Kooker received the B.S. and the M.Eng. degrees from the University of California, San Diego in the Department of Electrical and Computer Engineering, in 2004 and 2010, respectively. He is currently a software engineer with Qualcomm Incorporated, San Diego, CA, and was previously an iPod+iPhone software engineer with Apple Inc., Cupertino, CA.

Kai-Uwe Doerr received his Ph.D. degree from the Darmstadt University of Technology, Germany, in 2004. His Expertise includes virtual cockpit simulation, virtual prototyping, computer vision and 3D database generation. Currently he is a project scientist working at the California Institute for Telecommunications and Information Technology (Calit2) at the University of California, San Diego. His work forcuses on image-based tracking algorithms, cluster-based large scale data visualization and human factors research for interactive 3D visualization technologies.

Falko Kuester is the Calit2 Professor for Visualization and Virtual Reality and an Associate Professor in the Department of Structural Engineering at the Jacobs School of Engineering at the University of California, San Diego. His research is aimed at creating intuitive, collaborative digital workspaces, providing engineers and scientists with a means to intuitively explore and analyze complex, higherdi-

mensional data. In support of this research, he is developing new methods for the acquisition, compression, streaming, synchronization, visualization and hands-on analysis of data, as well as the cyberinfrastructure needed for collaborative data analysis and tele-immersion, including the ultra-high resolution HIPerWall and HIPerSpace visualization environments.