The ASPECTA Toolkit: Affordable Full Coverage Displays

Julian Petford
University of St Andrews
St Andrews, UK
jp438@st-andrews.ac.uk

Miguel A. Nacenta
University of St Andrews
St Andrews, UK
mans@st-andrews.ac.uk

Carl Gutwin
University of Saskatchewan
Saskatchewan, Canada
gutwin@cs.usask.ca

Joseph Eremondi
University of Utrecht
Utrecht, Netherlands
joey@eremondi.com

Cody Ede
University of Saskatchewan
Saskatchewan, Canada
cody.ede@usask.ca

ABSTRACT

Full Coverage Displays (FCDs) cover the interior surface of an entire room with pixels. FCDs make possible many new kinds of immersive display experiences - but current technology for building FCDs is expensive and complex, and software support for developing full-coverage applications is limited. To address these problems, we introduce ASPECTA, a hardware configuration and software toolkit that provide a low-cost and easy-to-use solution for creating full coverage systems. We outline ASPECTA's (minimal) hardware requirements and describe the toolkit's architecture, development API, server implementation, and configuration tool; we also provide a full example of how the toolkit can be used. We performed two evaluations of the toolkit: a case study of a research system built with ASPECTA, and a laboratory study that tested the effectiveness of the API. Our evaluations, as well as multiple examples of ASPECTA in use, show how ASPECTA can simplify configuration and development while still dramatically reducing the cost for creators of applications that take advantage of full-coverage displays.

Author Keywords

Full Coverage Displays; Hemispherical Projection; Toolkits.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI)

INTRODUCTION

Full Coverage Displays (FCDs), which cover the interior surfaces of an entire room with display pixels, have been discussed for many years in HCI [12, 41]. Predictions have stated that displays will become a commodity purchased by area, and so inexpensive that we will be able to use them like wallpaper [45]. FCDs are very different from current technologies - current large displays typically cover only a small area of the environment and so FCDs can provide benefits and opportunities that are unrealized with current setups.

Figure 1. A mockup, built using ASPECTA, of potential ASPECTA applications

For example, FCDs can help realize Augmented Reality and Ubiquitous Computing visions of reactive environments in which digital information is made available where it is needed [27, 1]. Previous work in psychology also suggests that the space around users is fundamental for cognition [28] and memory [9]. FCDs constitute an obvious step forward towards leveraging natural human skills for improved interaction with technology.

Despite their desirability, there are significant hurdles to the adoption of FCDs in real-world settings. Although multiple technologies have been available to implement fairly sophisticated FCDs since the 1980's (e.g., [12]), FCD installations are usually ad-hoc, expensive, difficult to program and deploy, and require significant expertise.

Existing FCD systems (e.g., [34, 19]) are primarily research prototypes requiring specialized equipment such as servo-motorized devices, multiple displays and projectors, a wide variety of sensors, and sophisticated calibration procedures. The considerable difficulties involved in building these types of environments and their cost likely prevents all but small number of researchers from using FCDs. In turn, the result-

PerDis '16, June 20 - 22, 2016, Oulu, Finland

ACM 978-1-4503-4366-4/16/06...$15.00
DOI: http://dx.doi.org/10.1145/2914920.2915006

ing lack of example applications and empirical evaluations may have delayed our ability to develop better FCD systems and turn them into commercial products.

In this paper, we present ASPECTA, a toolkit designed to facilitate the implementation and deployment of FCDs, as can be seen in Figure 1, without expensive equipment. Although there are still limitations (e.g., projector brightness and resolution: a problem which is reducing over time as high resolution, high luminosity projectors become more affordable), we simplify the technical issues of creating full or partial coverage spaces and lower the bar for developing full coverage applications. ASPECTA is an open-source toolkit that includes an API to project content on multiple (or all) surfaces of a room. Anyone with a commodity projector and a cheap hemispherical mirror [53] can turn a room into a full-coverage interactive space. Distortion and input are dealt with through a simple, graphical configuration step that does not require any tracking or sensors beyond a regular mouse.

The main contributions of this paper are:

- The design and implementation of the ASPECTA API for full coverage displays.
- The design and implementation of a sensor-less configuration method and tool for FCDs.
- A separate evaluation of each of the toolkit's APIs and the configuration tool.
- A design space of existing and future applications enabled by the availability of inexpensive FCDs.

RELATED WORK

We discuss existing related work in two groups, projection-based environments and multi-display environments. The "Comparison with Other Projection Systems" subsection at the end of the paper provides a qualitative comparison of our system with the most relevant related work.

Projection-Based Environments

The CAVE system [12], inspired by Sutherland's earlier "ultimate display" [50] is often cited as one of the earliest immersive interactive displays. It was designed to enable virtual reality applications that surround the user and provide a visually-uninterrupted simulated environment through rear projection on the walls of a purpose-built small room. Full and partial-coverage CAVE environments are still popular, and are used in areas such as industrial design and military simulation. A modern implementation of such a system is Benko and Wilson's dome display [4], which uses a spherical projection device. CAVE-like environments typically require expensive installations and are designed as single-purpose spaces to display virtual reality. The work we present here is different in motivation since we are interested in full-coverage displays that coexist in home and office environments and are thus not limited to specialist installations for a specific purpose. Also, unlike CAVE systems, we are proposing the use of a single projector and front projection to simplify the development of these immersive environments.

A closer source of inspiration for our work is Raskar et al.'s "Office of the Future" [41], in which they propose (but only partially implement) multiple integrated projection screens and sensing mechanisms to create an environment that combines a regular office with virtual objects and distributed locations. Several other projector-based systems have been implemented for office and work environments, with different degrees of sophistication and different goals, such as the OMNI display [6] (with the goal of improving group awareness) and Kimura [23] (with the goal of supporting office work and office activity). These systems often use multiple projectors concurrently, and use blending techniques to deal with overlaps, or with environments where projection areas are not flat (e.g., [2, 16, 21, 40, 39]).

Closest to our work are Microsoft's Illumiroom [20] and Roomalive [19] systems, Ubi Displays [14, 15] and the prototype developed for the Ubiquitous Cursor project [53]. Microsoft's IllumiRoom uses a projector to extend the field of view in video games, and makes objects in the room around the display change their appearance in response to events in the digital environment. Microsoft's RoomAlive uses multiple synchronized projectors, each of which is paired with a Kinect 3D sensor to transform rooms into full coverage gaming environments. The software for this project is also available as a toolkit[1]. The related Mano-a-Mano system uses similar technology to support two-person interactions by taking advantage of the different fields of view of two people facing each other [5]. These projects have contributed significantly to the goal of making FCD and similar environments possible for a wider range of application developers. ASPECTA's approach is related in its goal to extend display space of a common room, but differs in its focus to make the implementation hardware inexpensive (e.g., through a single projector rather than multiple ones), to avoid the use of spatial sensors (these can be difficult to integrate in working and living environments and are not free of problems, e.g., with reflective surfaces), and to simplify application development such that systems are not dependent on 3D sensing.

The Ubi Displays project, developed at Lancaster University, which is now also a commercial venture[2] enables the creation of interactive display surfaces out of everyday surfaces and objects through projection and Microsoft Kinect technology. This project focuses on enhancing every day spaces and surfaces through input and projection; in our work, we instead focus on large-scale coverage of a room display space.

Finally, our own Ubiquitous Cursor research prototype is a precursor of ASPECTA which uses the same kind of hardware setup with the specific purpose of providing visual feedback for indirect input (e.g., mouse input) in what has been termed "displayless" space [30]. Although the ASPECTA toolkit inherits the hardware approach based on a hemispherical mirror, it is not constrained to it and it provides an API and configuration algorithm to enable the implementation of other, more generic, applications.

[1] http://research.microsoft.com/en-us/projects/roomalivetoolkit/

[2] https://www.heinventions.com/ubi-displays

Other Partial Projection Systems

An alternative type of projection-based environments are those built with steerable projectors, which allow for high resolution images to be projected where needed. Notable examples are the Everywhere Displays system [34, 35, 36] and the more recent Beamatron [51]. These systems are different from ours in that they do not simultaneously cover all space (i.e., although most surfaces can be projected on, the projection surface at any given time depends on where the projector beam is pointed), and depend on custom hardware and software such as servo-motors, movable mirrors and servo-motor control and calibration software.

A recent related system is BaseLase [29]. This system is specifically geared towards projecting content on the floor, and also uses mirror technology that can project a fast-moving laser on a large floor area from a centrally located cabinet. Other research has also demonstrated the creation of interactive floors (e.g., [8]). Although ASPECTA enables creation of floor projection systems (see Examples 1 and 2 in the "Existing ASPECTA-supported applications" section), our focus is to provide software that simplifies the creation of applications that project on multiple surfaces, not just the floor.

Multi-Display User Environments

Another approach to creating immersive user interfaces is to integrate multi-monitor, projected displays or mobile displays into a Multi-Display Environment (MDE) where an interface is not bound to a single screen or device. Some MDEs might approach significant coverage of the space around the user [18, 42, 48], and some are even designed to simulate seamless visuals across different displays [31]. Middleware systems, frameworks, and APIs have been developed to support the creation of these spaces (e.g., Gaia [44], iRos [18], Sakurai et al.'s middleware [47] and ZOIL [17]). There are fundamental differences between fragmented MDEs and full coverage displays [37], including effects on the attention transition between displays [38]. Although the toolkit that we present is designed to also support multi-display scenarios, the architecture of both types of systems differs significantly: ASPECTA does not support the distributed use of multiple displays from multiple devices but instead provides access to separate surfaces from a single display as a service to client applications.

FCD USES AND OPPORTUNITIES

The opportunities for using full-coverage displays arise from the characteristics of three main aspects of the environment: the display technology, the human user, and the room in which the system is deployed.

FCD Characteristics

Full-coverage displays differ substantially from other types of displays. First, they have a large physical size, generally the full area of the interior of the room, which is larger than even most wall displays; in addition, unlike other displays, FCDs cover the entire interior surface of the room. Second, the expanded area of an FCD means that individual pixels may be large. The curved mirror and varying distances to room surfaces also means that different display areas might have different pixel sizes. Third, projection-based FCDs project on top of the objects in the room. This implies that there will be shadows behind objects and that the color of the projected light may change based on the object or surface, but also that the light on these objects can be used as an augmentation. Finally, even with bright projectors, the images produced by single-projector FCDs are typically low intensity, making color and detail more difficult to judge.

Human Characteristics

Human abilities and limitations in visual attention, perception, and memory also affect the ways in which FCDs can be used. First, a person's visual attention is usually directed at the area of highest acuity (the fovea) and redirected sequentially through eye movements to areas of interest. This means that people must shift their gaze around an FCD to see and use different parts of the content. However, people can also use peripheral vision to detect motion and change; and since peripheral perception extends almost 180 degrees horizontally and about 100 degrees vertically, FCDs can have a large area of perceptibility overall. Second, humans have strong abilities to remember a very large number of locations in space, both based on landmarks and absolute locations. Studies have shown that people can remember many locations (e.g., users were able to remember more than 100 locations with the Data Mountain system [43]). This means that if content in an FCD remains stable, people can use spatial memory as a retrieval mechanism. Finally, users typically set up work spaces in ways that lead to stable patterns of behavior [24]. For example, people tend to operate in particular places within a room, tend to look frequently at a small number of locations, and tend to move along set paths that are determined by the tasks to be carried out and the layout of the room. These patterns can be leveraged by designers of FCDs.

Room and Workspace Characteristics

The layouts and arrangements of room environments also provide several characteristics that affect the use of FCDs. First, a large proportion of the surface area in an office (walls, ceiling, and floor) usually does not have a particular display function, and can potentially be used as a projection surface for an FCD. Second, many real-world spaces such as offices or meeting rooms have static organizations of furniture, devices (such as printers), and objects (such as pictures or clocks on the wall). These spatial layouts can be used strategically in designing an FCD; in addition, the static organization of the room and the objects in it provide many visual and geometric landmarks that may aid memorability. Third, many of the objects in work spaces are meaningful to the user, and some have relationships to digital work (e.g., a user might have pictures of their children on the wall, and also have the children as contacts in a messaging application). If objects have fixed (or trackable) locations, FCD applications can use knowledge of these semantics.

Design Opportunities for FCDs

The characteristics described above provide a broad design space for FCDs. Some parts of this space have been discussed in large-display literature and previous work on projector-based systems for example, increasing the amount of space

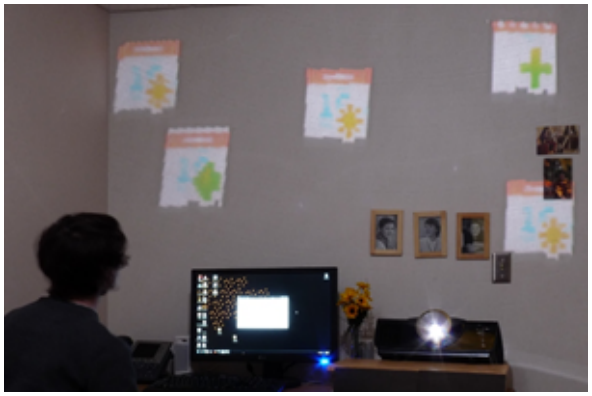

Figure 2. A mockup of calendar reminders moving from the periphery to the focus area.

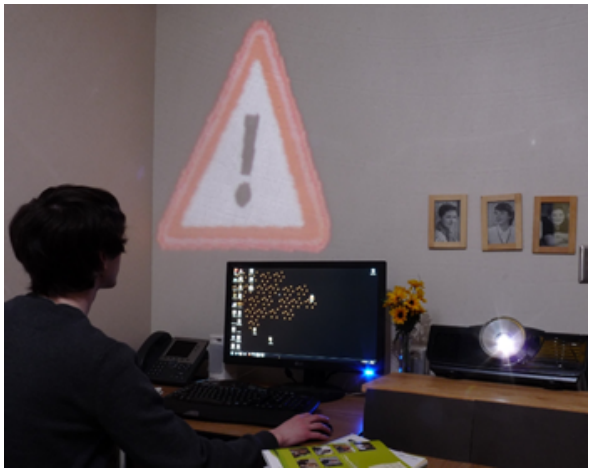

Figure 3. A mockup of an obvious notification.

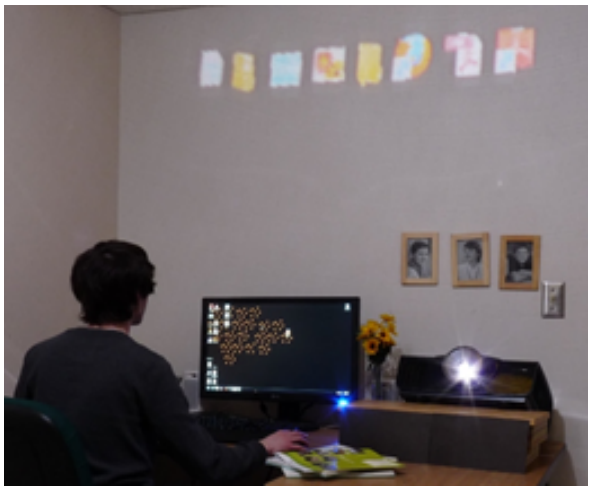

Figure 4. A mockup of storing files on the walls.

available for working with documents (e.g. [41]), using a peripheral-vision display for context (e.g., Focus plus Context displays [3]; OMNI [6]), or adding game content to a room-based display (e.g., IllumiRoom [20]). In addition to these previous explorations, we have identified three ways in which the characteristics of FCDs can be used to generate novel interfaces and applications.

Leveraging Perceptual Weight

FCDs can take advantage of the capabilities of the human vision system to present information with a wide range of perceptual weight and noticeability. There are three main zones for FCDs: outside the periphery, periphery, and focus. The peripheral zone of an FCD provides a natural location for notifications because there is a natural mapping between urgency of the notification and perceptual weight. The perceptual weight decreases with the distance from the user's focus; therefore, less urgent notifications could be placed further from the focus area. In addition, human sensitivity to motion and change in the periphery provides additional dimensions along which a designer can manipulate perceptual weight. As an example (based on work with smaller displays by Birnholtz and colleagues [6]), Figure 2 shows a mockup of calendar reminders that gradually move into focus as they approach in time.

At the other end of the noticeability scale, the large size of FCDs also offer possibilities for being highly obvious. These presentations would make use of a human's sensitivity to change and motion in the visual periphery, and of the full coverage of the display which provides visual information regardless of the user's view direction. As a notification becomes more important, additional techniques can be used to make sure that the item is noticed (e.g., the highly obvious alert shown in the mockup of Figure 3).

Leveraging Size and Location

The large display area of an FCD means that designers can change the way in which space is used in the display. In a computer monitor there is insufficient space to show everything at once, so windows must be stacked (i.e., 2.5D), minimized, or arranged into multiple virtual desktops. In contrast, an FCD display can use different areas of the room for different display purposes, and can leverage human spatial memory as a way to help users remember the context of different displays.

With an FCD display, people can use the room as a storage space, and can put files and objects into particular locations. Although this approach is not a replacement for hierarchical file interfaces, it does provide a rich environment for fast access to important or frequently-used items. Users already use spatial arrangements to organize icons on virtual desktops or launchers such as the OSX Dock [33]; but the size and coverage of an FCD display, and the familiarity of the room, provide additional landmarks and memory cues for spatial-memory-based storage. An example of an FCD being used for icon storage can be seen in Figure 4.

Second, full-room coverage means that FCDs can easily display ambient visualizations. Ambient displays make use of the physical environment as an interface to digital information [52], and typically provide general awareness of a digital

Figure 5. A mockup of an ambient presentation of weather information.

information source. In general, the goal of an ambient display is to provide general awareness and understanding of the information source, goals that are well suited to the capabilities of FCDs. For example, the mockup in Figure 5 shows a persistent weather display that is located near the office's coat hooks.

Leveraging Overlap Between Digital and Physical Spaces
Projection-based FCDs can layer pixels on the objects in the room (e.g., bookshelves, pictures, equipment) as well as on the room surfaces themselves. This differs from smart paint approaches that would only display behind objects, rather than on top of them. The ability to project onto objects, combined with an FCD's ability to map digital data onto the room's geometry, opens up several possibilities for augmenting objects in the real world (as long as object locations are known, and stable). This idea has also been explored with AR devices [27] and steerable projectors [10].

FCDs can present information relevant to a real-world object by projecting directly onto that object. For example, an FCD could highlight items at appropriate points during the digital work, or enable digital search of physical objects [11]. The mockup in Figure 6 shows an example system that highlights family pictures on the wall when there are new messages from that person. Similarly, FCDs could visualize the digital connections between real-world objects: for example, indicating whether a laptop is connected to a printer by drawing a line between them, or showing ways that monitors are stitched together in a multi-display environment [53].

THE ASPECTA TOOLKIT

This section describes the design, architecture and API of ASPECTA as well as the typical hardware configuration of applications built with the toolkit.

Design Goals

We designed the ASPECTA toolkit to fill a gap in the available technologies and tools for building FCDs - our objective is to provide broad access to inexpensive full-coverage systems. We summarize our high-level design rationale in three main goals:

DG1: Allow the use of simple, inexpensive, and readily available hardware for implementation of FCDs.

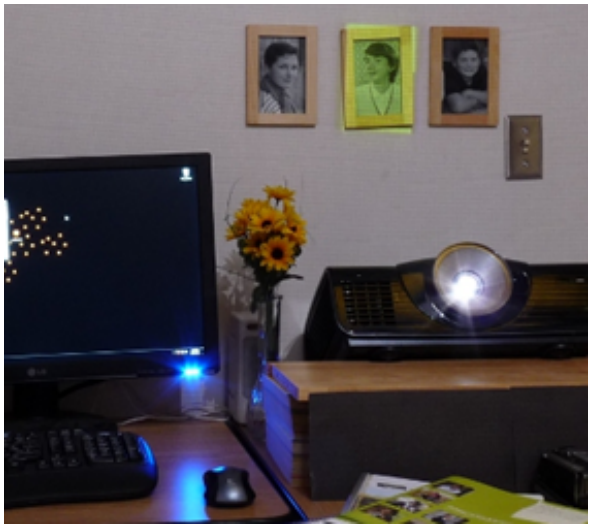

Figure 6. A mockup of a message indicator from a family member.

DG2: Provide a simple method and tool for programming full room display environments.

DG3: Simplify the implementation of FCD applications that integrate multiple users and multiple devices.

The level of success in achieving these goals is addressed through a set of application examples and an evaluation with programmers described in the final sections of the paper.

Architecture

The toolkit is composed of three main elements: the server software, which provides the main display service, holds the room model, and performs the graphical transformations; the APIs, which expose the server functionality to the clients; and the configuration tool, which enables the manual calibration of the system to the geometry of the room. Additionally, we discuss the architecture of the client applications that make use of the service and the types of display hardware that it supports. Figure 7 summarizes the relationships between the different parts of an ASPECTA-powered working system.

Server
The server forms the core of the ASPECTA system and implements a display service that clients can access through two separate APIs. We chose a client-server architecture because it is easy to understand and is familiar to programmers (DG2), and because the use of a single resource by multiple devices (DG3) lends itself naturally this model.

The server accomplishes two main functions: it maintains the model of the room (room geometry and all displayed objects) and it applies the geometrical transformations needed for the graphical output.

The server typically resides in a dedicated machine that is connected to the display hardware, but this machine can also run the client and configuration software if desired. The server software is implemented in Python 2.7 and uses py-OpenGL for rendering.

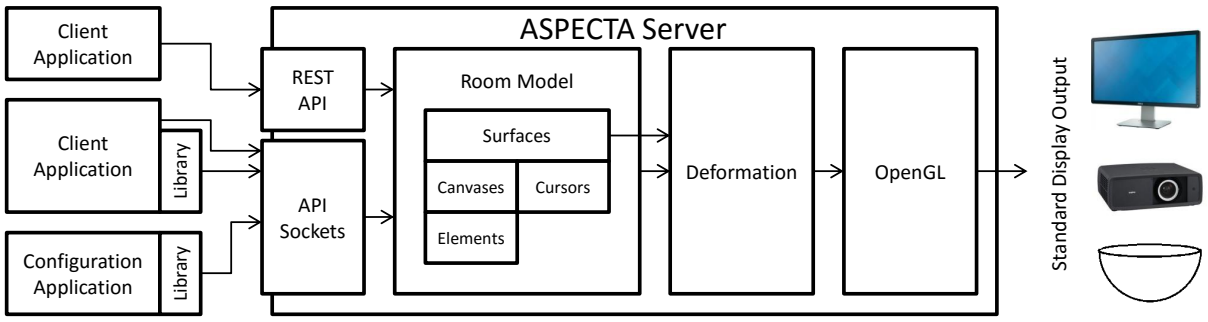

Figure 7. Architectural diagram of the ASPECTA system.

APIs and Clients

ASPECTA exposes functionality to clients through two network interfaces. Clients run applications that connect to the server in one of two ways, through a TCP socket or through a RESTful web API. The RESTful calls take the shape of POST and GET requests and all calls send a JSON string that includes the name of the called function and the call's parameters. Many calls return object references, requested variables, or error codes.

There are three main types of calls: drawing calls, room configuration calls, and query calls. Drawing calls allow clients to draw, move or delete graphical objects on the different surfaces of a room (circles, rectangles, cursors, text and images). Room configuration calls set or change the geometrical configuration of the space or create the canvases where objects are drawn (see the Room Model and Configuration section below). Query calls provide access to room configuration and graphical object information stored in the server.[3]

Although the sockets API can be accessed directly by client applications using the appropriate JSON payload format, we also provide an additional Python client API library that facilitates the construction and transmission of well-formed JSON calls for those programming in Python. The REST interface provides the same functionality as the sockets API but is more convenient for web-based client application development and might appeal more to programmers that favor web paradigms (DG2, DG3).

The APIs expose the functionality required to display content around a room in an organized way, but do not impose unnecessary constraints on the architecture of the client applications. Client applications can themselves implement sophisticated services with their own network and communication architectures.

Room Model

The server uses a single hierarchical room model as the central data structure. The object structure exposed through the API is directly derived from the room model and is designed to be as simple and accessible as possible.

[3]Available API calls can be seen in the ASPECTA GitHub documentation:
https://github.com/uoscompsci/ASPECTA-Client/wiki/API-Calls

The Room Model is a room layout with any number of surfaces (e.g., walls) and the two-dimensional spatial coordinates that enable their appropriate deformation. Each of these surfaces may contain canvases (parts of surfaces that can be drawn on) and cursors (which traverse surfaces). Finally, each canvas can contain any number of graphical elements (a superclass of visible objects such as polygons, lines, textures/images and text). Each object, surface, canvas, cursor and graphical element is directly addressable through a system ID reference that is also returned to clients as a handler. In addition to their ID, each object also stores the ID of the client which created it and an application name. This facilitates access to categories of objects. (e.g. pertaining to a single user or application)

Graphical Transformations and Configuration

Our primary design goal requires a flexible graphical approach to provide deformation compensation for a wide variety of optical projection geometries. The server provides a rendering pipeline that applies manually-configurable visual transformations that pre-deform the graphical output of the video card so that objects rendered on the room look geometrically correct.

The core element of the deformation pipeline is a 2D to 2D transformation of graphical textures carried out in real time for each of the defined surfaces. This is a standard graphical operation provided by OpenGL. It takes a 2D grid of 2D points from a texture, and maps it to a different 2D grid of 2D points which already has the transformations applied, all in real time. The harder problem is how to determine the 2D grid that the texture is being mapped to so that the deformation applied achieves a consistent undistorted appearance after the image has been projected.

Our approach is to allow manual interactive configuration of the deformation on top of the room's projection itself. This can be understood as a manual calibration process in which the person calibrating the system to the room for the first time uses a cursor controlled by a mouse to tell the system the 2D locations in the untransformed space that correspond to the boundaries of the different surfaces of the room.

Depending on the geometry of the optical pipeline (which is typically a complex curved deformation see Figure 8), the movement of the cursor will not be linear. This can make the

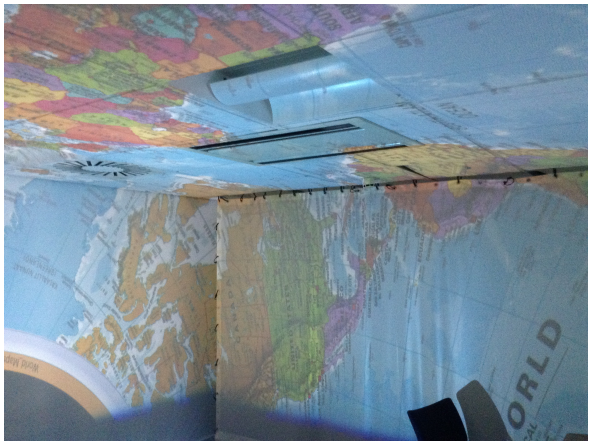

Figure 8. How a projected world map appears without corrective transformations.

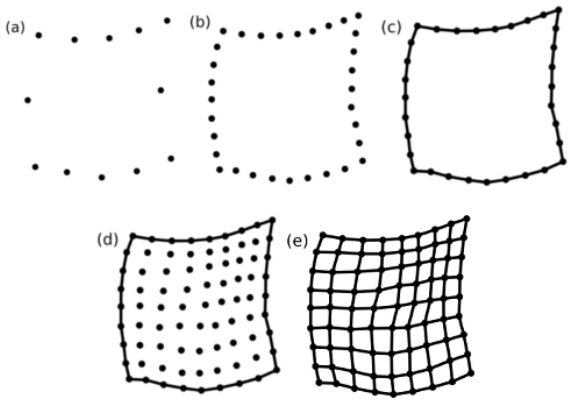

Figure 9. Diagram of the transformation process. (a) The user defines corner points and waypoints using the room configuration program. (b) Bezier calculations are used to infer further points. (c) Lines are added between the points to display a curve. (d) Coon's Patch infers the central points of the mesh. (e) Surface texture is mapped onto the mesh.

control of the cursor during the configuration phase difficult. Early on we realized that the main difficulty is the rotation of the input-output mapping (e.g., moving the mouse forward can make the cursor move down a wall), rather than other geometric anomalies (such as the non-linear control-display gains in different parts of the room). To facilitate control of the cursor of the room we provide a way to rotate the cursor and its axes with the mouse wheel. Although a typical room configuration will require several rotations depending on the area of the room that is being configured, movement and wheel adjustment of the rotation is sufficient to provide nimble movement of the cursor around the projected space during configuration.

The configuration is implemented through a special client, separate from the server, which calls the configuration API. We chose to implement the configuration process as an application instead of as part of the server for two reasons: it enables the creation of alternative configuration software, including software that might use structured light approaches or other sensor-based mapping of the environment (e.g., [19, 22]), and it allows users to run the configuration tool from any computer and input device.

The configuration process essentially provides a point-and-click interface for the creation of multiple surfaces that sets the parameters of the deformation algorithm. This process is described in more detail in the next section. The out-come of the configuration process is the sets of 2D display coordinates corresponding to the boundaries of each real-world surface. These are stored in the room model. The boundary points are interpolated into a full grid through a 2D version of the Coon's Patch[4]:

$$Q(u,v) = q_0(wv)(1-u) + q_1(wv)u + P_0(wu)(1-v) + P_1(wu)v - ((1-u)(1-v)P_{00} + u(1-v)P_{01} + (1-u)vP_{10} + uvP_{11})$$

[4] http://www.maplesoft.com/applications/view.aspx?SID=100369&view=html&L=G

Here the u and v parameters represent the horizontal and vertical distances from the origin in the warped patch ($0 \leq u,v \leq 1$). q_0 and q_1 are the vertical Bezier curve equations and P_0 and P_1 are the horizontal Bezier curve equations. P_{00}, P_{01}, P_{10} and P_{11} are the four corner points of the mesh.

An image of the stages involved in the transformation process can be seen in Figure 9.

Display and Hardware

Our goal of enabling cheap and easy creation of full coverage systems requires a flexible approach that supports a wide range of hardware. ASPECTA was inspired by the creation of inexpensive full-coverage and almost-full-coverage displays that beam a regular projector onto a hemispherical mirror (Figure 10). The use of a mirror instead of a fisheye lens reduces the overall hardware costs by up to two orders of magnitude (mirrors are much cheaper than lenses, see also Bourke [7]), and simplifies the creation of software (a single display output is easier to program and to manage than a combination of multiple displays, which have to be coordinated and calibrated for differences in color, size, and projection position [19, 22]). We discuss advantages and disadvantages of this approach below.

The flexibility provided by the manual configuration process enables any position of the projector and the sphere and makes the software agnostic to the physical setup before the configuration process. The positions of the mirror and the projector can be chosen to maximize resolution in specific areas, to minimize shadows, or simply to take advantage of the room's spatial features.

The flexible approach does not preclude the use of more expensive equipment such as fisheye projectors and lenses - in fact, we have successfully used ASPECTA to quickly configure rooms lit by a spherical projector. Finally, it is possible to use this approach to project onto non-regular shaped surfaces with regular (flat image) projectors, since the configuration

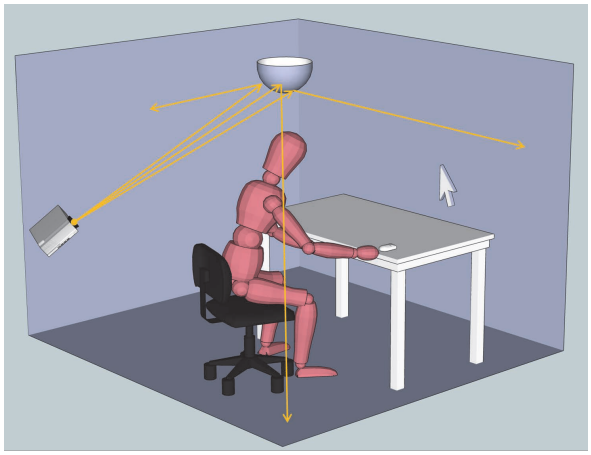

Figure 10. The hardware configuration used in our previous Ubiquitous Cursor project which inspired the development of ASPECTA.

mechanism inherently takes care of off-access projection issues (e.g., keystoning).

Input

The ASPECTA toolkit is purposefully designed to be input agnostic. This choice is deliberate, since input is most often independent from display technology and is highly tied to the specific requirements of the project and the current state of the art. The architectural design does not preclude ASPECTA's integration with third-party input frameworks. Examples already implemented with the toolkit include a standard mouse, a Microsoft Kinect, and an OptiTrack motion tracking system.

BUILDING ASPECTA APPLICATIONS

This section details the process of obtaining the necessary hardware, configuring ASPECTA for a room, and then using the APIs to create a client application. We use a simple application example to introduce the usage of the ASPECTA API: a brainstorming application that allows the creation and display of sticky notes with text on four walls of a room.

Hardware and Software

A suitable room for ASPECTA deployment has one or more light-colored walls; as discussed above, it is also possible to project overtop other features on the walls, although brightness will be reduced. If the application does not need to cover all walls, one need only ensure that the defined digital surfaces correspond to clear surfaces.

The hardware requirements for an ASPECTA system are simple and inexpensive, involving a commodity projector and a hemispherical (or even quarter-sphere) mirror. A dome security mirror can be purchased for as little as $60[5], and although more expensive versions can increase image fidelity, our tests show that an inexpensive mirror works well for almost all applications. Our experience also suggests that a 450 millimeters (17.7 inches) mirror is a suitable diameter, although other sizes can also be used. This size of mirror allows the placement of the projector at a reasonable distance.

Next, the characteristics of the projector are important, as they determine the brightness and resolution of the images in the ASPECTA application. The first priority is often high luminosity. We recommend at least 5000 lumens for complete simultaneous room coverage, but higher luminosities are necessary depending on the ambient lighting conditions in the room. Projector luminosity has recently increased substantially due to LED and laser technologies, and commodity projectors can be used for most interior spaces without direct sunlight.

High resolution is also beneficial, especially if the application is to display text. The mirror and projector position interact with resolution: if the projector is placed at an extreme oblique angle to the mirror, the distribution of pixels around the room can result in some areas that have high resolution and other areas with low resolution (where pixels are large). In some cases this effect may be desirable: for example, if a particular area is more important (such as a work surface), the placing of the projector and the spherical mirror can maximize resolution in this area. In general, the mirror placement should be closest to the intended high-resolution areas. The surface onto which the hemispherical mirror is attached will generally not be projectable. Therefore, it is important to find a good location for the projector, testing different options to maximize brightness and resolution in the most useful areas of the room.

These constraints are not unduly restrictive, however. In our tests we have regularly set up new rooms with ASPECTA hardware in only a few minutes - for example, by simply hanging the mirror on the wall, and placing the projector on a table to point toward the mirror.

The server should be able to output graphics to the projector, either directly or through a wireless display technology (e.g., Chromecast). Note that the server computer can be a relatively low-powered machine (we use an Intel NUC 1.3GHz quad core PC with 8GB RAM and Intel Graphics 5000). The client used for configuration should have input appropriate for the configuration process. With our current configuration system this means just a mouse and a keyboard.

The server and the room configuration applications[6] are built in Python 2.7 and are easy to run using Ubuntu 14.04, for which we provide a guide to obtaining the required packages.

Configuration

The configuration application runs on a client system (which may double as the server system). A window interface allows the definition of each wall in the room by moving the cursor to each corner and clicking.

When the mouse cursor is not vertical, the scrolling wheel changes its orientation to simplify movement. Once the corners are defined, clicking on each side boundary creates more

[5] All prices in this paper are in US Dollars

[6] https://github.com/uoscompsci/ASPECTA-Client/ https://github.com/uoscompsci/ASPECTA-Server/

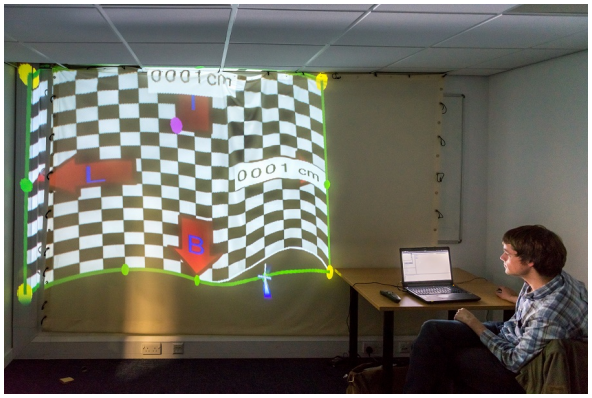

Figure 11. The configuration process of repositioning control points to define the shapes of the wall.

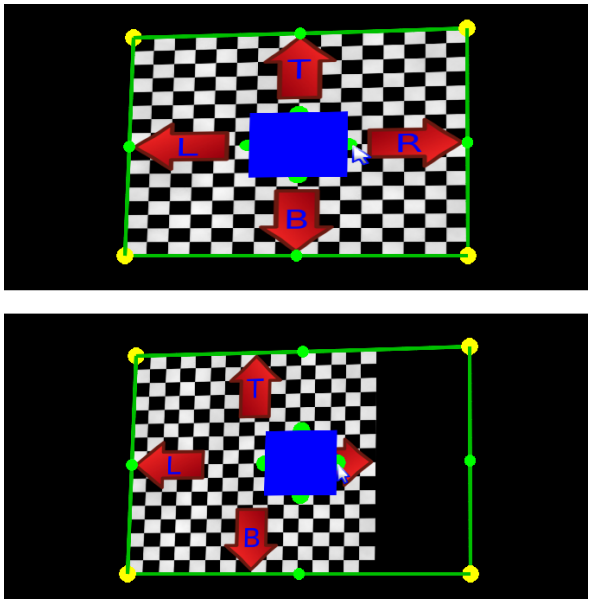

Figure 12. After aspect ratio correction stretching has been resolved while still providing access to the full with of the surface.

control points which can be repositioned to make the boundary match the physical edge of the wall (see Figure 11). This can be done recursively until the boundary edge is sufficiently accurate. To finish the configuration of each wall it is necessary to adjust the aspect ratio by adjusting the pattern inside the wall projection until it looks like a square (instead of a rectangle - see Figure 12). A last optional step allows the configurator to add the physical size of the wall, which allows developers to refer to the projection space in actual physical units rather than pixels (Figure 13). The process is then repeated for any number of room surfaces.

Room configurations can be saved with a name, loaded from a previous session or cleared from the server through a traditional window UI.

This configuration process reflects a low-cost approach that requires no additional hardware. It typically takes 5-10 min-

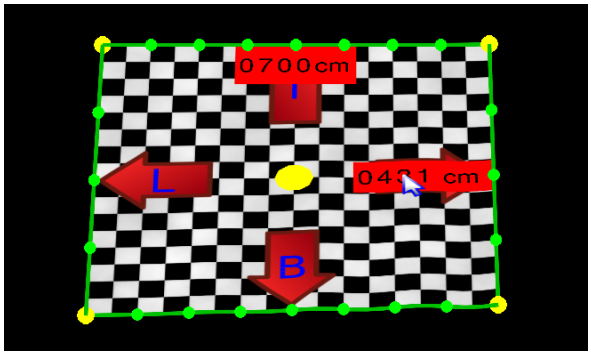

Figure 13. Measurements of the wall assigned through the mouse wheel in real-world units.

utes in the average room. However, since the configuration software is just a client, anyone can build a more sophisticated configuration procedure that uses, for example, structured light sensors.

Application Code

This subsection illustrates the code needed to create a simple application. Assuming that the config.ini files on the client and server have the appropriate server IP address, the first lines of an ASPECTA client initialize the connection and set names for the client and the application:

```
self.sender = messageSender()
self.sender.login("aspecta sticky notes client")
self.sender.setapp("sticky notes")
```

In this application input works with a cursor that moves the notes. There can be many cursors, and each cursor belongs to a specific surface but can be moved to another (this example creates just one cursor in the centre of surface 1).

```
self.maincur = self.sender.newCursor(1, 0.5, 0.5,
    'prop')
```

In this case we want to simply link the movement of the cursor just created to the mouse input. The following code exemplifies how to programmatically move the cursor to the center of the second surface.

```
x = self.sender.getSurfacePixelWidth(1) / 2
y = self.sender.getSurfacePixelHeight(1) / 2
self.sender.relocateCursor(self.mainCur, x, y, '
    pix', 2)
```

The following line uses the shiftCursor() function to move the cursor by an x_distance and y_distance amount, which we can just get from a system input event.

```
self.sender.shiftCursor(self.mainCur, x_distance,
    y_distance)
```

The code below exemplifies the creation of a new sticky note, assuming that position, size and text content are gathered by the software elsewhere. A note is implemented as a canvas with a rectangle that contains text.

```
canvas = self.sender.newCanvas(1, x_pos_note,
    y_pos_note, width_note, height_note, 'pix', ''
    )
self.sender.newRectangle(canvas, x_pos_rect,
    y_pos_rect, width_note, height_note, 'pix',
    (1,1,1,1), outline_width, (1,1,0,1))
text_id = self.sender.newText(canvas, content,
    x_pos_text, y_pos_text, 'pix', font_size,
    typeface, (1,1,1,1))
```

To move the sticky note with all its contents we just call the shiftCanvas() function:

```
self.sender.shiftCanvas(self.canvas, x_distance,
    y_distance, 'pix')
```

EVALUATION

During the design and implementation of ASPECTA we have performed two separate evaluations to assess the usability of the API and toolkit, to detect problems in the design, and to decide which new features to include. After completing the first running version of the system we evaluated the adoption of the toolkit by an external research group for the creation of a commercial display system (Case Study Subsection below). After a new iteration we carried out a formal study of the configuration tool and of the API (User Study Subsection).

Case Study

We obtained ethical approval and consent to evaluate the work of a research group from the University of Calgary who wished to build an application combining an FCD with a Kinect-based tracking system. Their objective was to explore the advantages of providing projected feedback about product locations for a commercial retail space with multiple commercial displays. The researchers used ASPECTA's REST API.

We collected information about the problematic elements of the API during the development of their experimental software, and once the application was completed, we interviewed the main developer to collect qualitative data about the development experience. We integrated the lessons learnt and the suggested improvements during the development of the next iteration of the toolkit.

Findings

The main developer (a graduate student with programming experience but with little exposure to graphical APIs) found the API intuitive. The included documentation helped her get started with the API. She stated that once she was familiar with the functionality, she was able to predict most required function names due to the consistency of the naming scheme. She also reported that the API was "pretty good from a new person's perspective".

At this stage in the development of the toolkit there was not yet a full configuration guide, therefore the configuration process was largely trial and error, providing an opportunity to see how straightforward the configuration process was with minimal guidance. The developer reported that initially she found it relatively difficult to get accustomed to using the mouse to define points on the walls, however once she was familiar with the task it was much simpler. The developer initially missed the configuration feature that enables a user to control the number of waypoints at the side of a wall, therefore they were unable to make the curve fit the wall as closely as they wished to. However, once told, she found that the room configuration program met her needs well and didn't have any suggestions for further improvements. The unawareness of certain features within the room configuration program clearly demonstrated the need for a user guide or a configuration tutorial.

The application made use of a 5000 lumens projector positioned approximately 2.5 meters (8.2 feet) away from a hemispherical mirror. With the hemisphere located far from the wall and the lights on, the projection wasn't as visible as they wished. The solutions that worked best for them were to either turn the lights off or to move the hemisphere closer to the main wall of interest. A similar statement was made about the resolution required. These are general issues of the approach addressed in the Discussion Section along with details of how they can be alleviated.

User Study

We designed a more formal evaluation to test a later version of the API and the configuration application.

Participants

Eight participants were included in this study (1 female and 7 males). We removed the data of an additional (female) one due to significant network infrastructure problems not related to ASPECTA and outside our control. Therefore, the analysis below does not report the results from this ninth participant.

The sessions took place individually and at different times. The participants received an online bookstore voucher in exchange for their time. The study was approved by the local ethics committee and took place in an isolated room with our own hardware. The participants of the study were recruited among the staff and students of the Computer Science department at the University of St Andrews who had some Python experience.

Tasks and Procedure

The main part of the experiment consisted of two tasks. After providing written consent, participants had to use the room configuration program to set up two walls of the room. Then they had to use the Python ASPECTA API library to project yellow sticky notes on either of the walls while allowing the user to drag the notes around either of the walls using a cursor. At the end of the session and between the room configuration and programming tasks they took part in short semi-structured interviews. Figure 14 shows the intended appearance of the resultant program.

Apparatus

For the purposes of this study we set up a server connected to a projector fitted with a fisheye lens. This allowed participants to create configurations that could make use of any combination of the four surrounding walls. The server system ran in Eclipse Juno's PyDev Plugin under Ubuntu 14.04

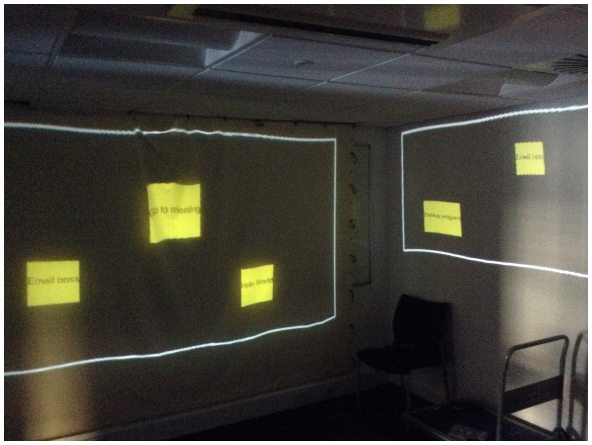

Figure 14. The sticky-note application as created by one of the participants.

Participant Number	Configuration			API	
	Ease	*Intuitive*	*Meet Needs*	*Intuitive*	*Meet Needs*
1	8	8.5	10	9	9
2	9	10	8	10	10
3	8	7	8	9	9
4	9	7	10	10	10
5	6.5	4	9	6	7
6	6	6	8	7	8
7	5	6	8.5	8	9
8	7	6	9	7	8
Average	**7.3**	**6.8**	**8.8**	**8.3**	**8.8**

Table 1. The quantitative results from the user study. Participants rated each aspect out of 10, with 10 being very good and 0 being very poor.

on an Intel Core i5-4250U Intel NUC with 8Gb of RAM and integrated Intel Graphics 5000. The machine for the client was a PC running the same operating system and IDE.

We provided a code template which included basic detection of mouse input (but no handling of these events), and a GUI template (built using tkinter[7]) with a text box and several buttons. This skeleton GUI was provided to avoid testing participants on skills not directly relevant to ASPECTA and to reduce the study times. The working time for the two main tasks was limited to four hours (although all participants finished in less than 3 hours). Participants had access to a wiki page with the documentation of the API, a guide for the room configuration process, and an existing Hello World example. Participants were also allowed occasional queries to the experimenter when they were stuck. In those cases the experimenter only provided pointers to the documentation and did not suggest or help with the ASPECTA code itself.

Results: Configuration Tool

Table 1 summarizes the subjective results for the configuration tool and the API. Many participants reported that the configuration tool was quite easy to use (average score of 7.3/10). However, many stated that they did have to spend some time familiarizing themselves with the procedure. An example is mouse movement control over the walls before any calibration had occurred. One participant reported that they felt that the mouse movement was too fast initially:

> "*That was on the very first experience. I'm like 'Oh! What? Where did it go?' That was confusing.*" [p7]

Participants also felt that using only the mouse to carry out actions during configuration was confusing because they expected to switch configuration modes using the keyboard. Our configuration program avoids the use of the keyboard so that configuration can be done mostly through the use of a pointing device, such as a wireless mouse, without being tied to a desktop computer.

[7] https://wiki.python.org/moin/TkInter

Although the intuitiveness of the configuration program was rated high (average score of 6.8/10) there were many aspects of it where the user guide "needed reading" [p5] as they would be hard to discover otherwise.

Other participants commented that the provided room configuration guide contained too much detail and that it made it a frustrating task to locate the information they wanted. They suggested solutions such as adding a step-by-step list to the guide for creating your first surface after launching the room configuration application, [p4] adding instructions within the application itself to guide you through the wall setup process [p6] or creating a video guide [p1]. Another participant suggested that the cursor modes should be more obvious [p5].

One participant suggested a feature to choose from a list of different curve types instead of just allowing Bezier curves [p6], p7 also felt that it would have been easier to understand if Bezier curves were controllable with standard control points as seen in graphical software, although this would have been difficult to implement due to the unknown required warping at that point in the configuration process.

The subjective scores of the configuration program suggests that the application met the participant's needs (avg 8.8/10). One issue raised in this area was the accuracy of the curves. Several participants found it difficult to achieve perfect alignment.

> "*I was getting a bit fussy with the straightness of the lines and trying to get the waypoints on one either horizontal or vertical line.*" [p2]

Results: API

Participants reported that the API was intuitive to use (average score of 8.3/10). When asked if they had ever guessed the name of a call correctly before using it for the first time all but one answered affirmatively, with three participants explicitly stating that they found the API naming to be consistent across different elements and objects, and three others describing the naming choices as self-explanatory.

> "*Basically every time it was true. It was pretty self-explanatory.*"

When specifically asked whether any previous experience with graphical APIs helped them use ASPECTA, six participants said so, citing Unity, Processing and JavaFX. p2 and p3 stated that their experience hadn't helped.

Several participants missed a call syntax that was more object-oriented and relied less on referencing objects through identifiers. Although we are considering this for the future, the current architecture (with parallel REST and JSON payload calls) makes it hard to integrate a pure object-oriented call style.

When asked whether they thought that the ASPECTA API made their code unnecessarily verbose, seven of the eight participants felt that it did not. One person suggested that some call parameters could take default values and that object orientation would make the code cleaner.

Other issues found by the participants include the lack of more informative error messages, and the inconsistency of the origin of coordinates for the surface and canvas object (starting at the bottom left) with the position coordinates of the canvas objects themselves and other APIs. Participants also suggested new features such as the ability to change the numbers of surfaces after creation, automatically cycling cursors between walls, hit detection facilities, and the API's further integration with gaming development APIs and IDEs.

We asked participants explicitly what applications they could envision that would take advantage of the ASPECTA API. Answers ranged from applications in home environments such as game development, food preparation assistance, music interaction in the shower, party interfaces, creating virtual environments and general artistic exploration. For office environments participants saw value in extending presentations and meetings to be more immersive, in creating more immersive Computer Aided Design scenarios, and having more space for brainstorming. Finally, one participant suggested creating a projected immersive environment for the recording of music videos.

Overall, participants were positive about the toolkit and their prospective use of FCD applications in the future. They indicated that they would use the toolkit and applications if the hardware was sufficiently cheap and readily available. Qualifiers to these answers included sufficient resolution, the existence of an initial set of applications (e.g., creative applications) and concerns about the size of the market and user base that could make developing an ASPECTA application profitable for them.

Existing ASPECTA-Supported Applications

We briefly describe seven examples of use of the ASPECTA toolkit by researchers in the UK and Canada to illustrate the practicality and applicability of the toolkit.

Example 1: Augmenting a Retail Environment

A researcher at the University of Calgary created built a system with a industry design partner which used projected augmentation in a retail environment to improve the experiences of both customers and employees. ASPECTA driven projection was used to create a system to direct customers to products they needed, bring their attention towards ones which they might be interested in, and to indicate coupons being transferred from a wall display to a customer's personal smart device.

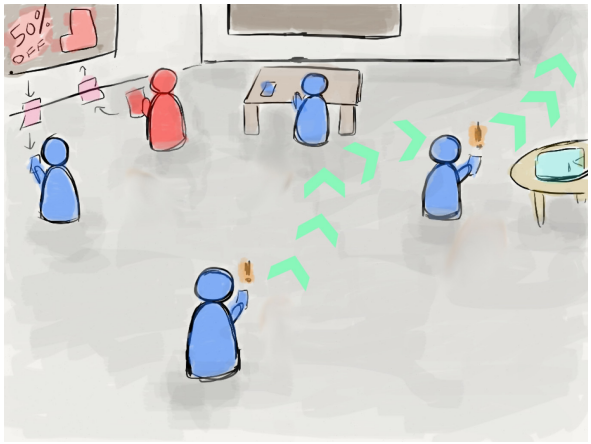

Figure 15. A sketch demonstrating how the created retail environment worked.

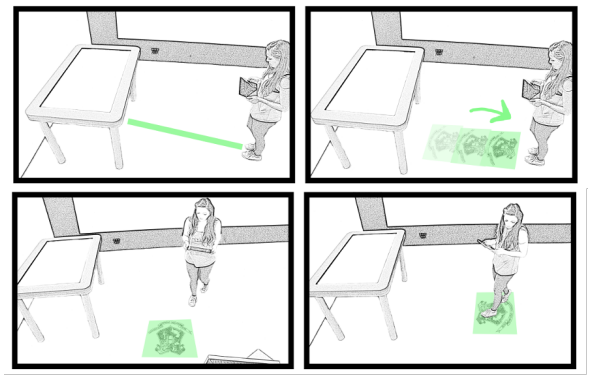

Figure 16. The data transfer and search visualization example.

A sketch of this can be seen in Figure 15.

Example 2: Data Transfer and Search Visualization

An independent research group at the University of Calgary in Canada built a system for a study exploring the value of projection feedback for virtual, projected object discovery tasks in Multiple Display Environments (Figure 16). As separate tasks users were asked to find the source screen, destination screen and location within a room environment of an image file transferred between a tablet an a PC. Different projected visual techniques were used and evaluated to indicate the flow and location of data moving between systems. All projection for this study was carried out using the ASPECTA toolkit alongside a hemispherical mirror and a single projector.

Example 3: Mathematics Game for Children

The same research group from Calgary also developed an educational game to teach children mathematics. Children answered quiz questions by physically standing on the answers which were projected on the floor. The overall aim of the

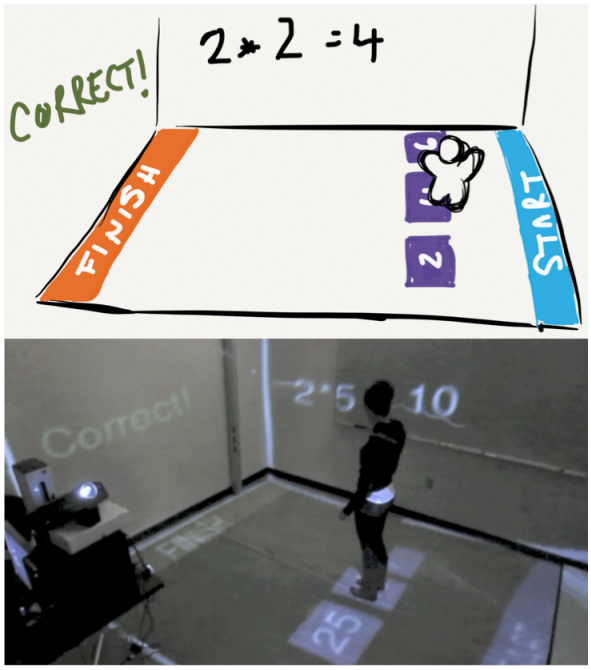

Figure 17. The mathematics game example (sketch above, completed system below).

Figure 18. The icon location recall example.

game was to cross from one end of the room to the other by stepping to the correct answers. For each wrong answer they were forced to take a step back to answer the following question whereas for each correct answer the next question would allow a step forward. All the projection that acts as the graphical interface for this application is projected using the ASPECTA toolkit alongside a projector and hemispherical mirror. Figure 17 shows a sketch and photograph of the system in action.

Example 4: Location Recall in a Room Environment

A research group from the University of Saskatchewan in Canada developed a system to study human recall of icon locations. All projection for this system was carried out using the ASPECTA toolkit and a hemispherical mirror. The system can be seen in Figure 18.

Example 5: A Digital Art Display

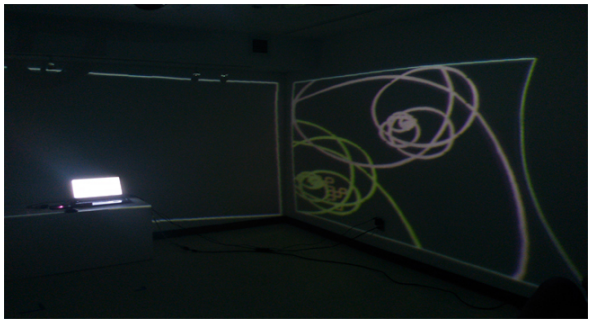

Figure 19. The digital art display that was created using ASPECTA.

A researcher University of Saskatchewan developed a projected art installation which tracks users with a Kinect and, at random intervals, draws either a Fibonacci spiral or dragon curve next to them. It keeps on doing this until a threshold number of draw calls have been made, clears the surfaces and starts again.

Once again, all projection for this system was carried out using the ASPECTA toolkit and a hemispherical mirror and an image of it can be seen in Figure 19.

Example 6: Sticky Note Application

As described in the User Study section, participants built several implementations of an application which allows users to place sticky note style reminders on the walls of a room. In this case the ASPECTA toolkit used a fisheye lens for the projection of the notes and is illustrated in Figure 14.

Example 7: Pointer-Controlled and Perspective Cursor

For the purposes of an upcoming study we have developed two tracking-based cursor control systems. One allows a user to control the cursor using a ray-casting wand tracked with OptiTrack sensors. The other uses the mouse with perspective correction based on the head position of the participant [32].

Example 8: ASPECTA Room Configuration Application

As explained in the Architecture Subsection above, the room configuration application is built using the regular ASPECTA API, and shows that a fairly sophisticated real-time interface is possible with the provided functions and calls. This application has been used with multiple optical projection systems: a hemispherical mirror, a fisheye lens projector and a standard projector projecting over two perpendicular walls simultaneously (see Figure 11).

COMPARISON WITH EXISTING SYSTEMS AND APIS

ASPECTA distinguishes itself from other work in its focus on lowering hardware costs, reducing configuration complexity, and simplifying the programming of full-coverage systems. This section explains how ASPECTA fits within the current landscape of tools and systems, and how it occupies a unique space in that landscape. Table 2 summarizes our qualitative analysis of representative systems according to relevant factors such as the kind of environment that the system can support, the display coverage that it provides, its public availability, and an estimate of its cost. This table can also be useful

	Everyday Environments	*Maximum Coverage*	*Research Availability*	*Open Source*	*Projection Method*	*Multiple Users*	*Freely Available Application Developer API*	*Required Sensing H/W*	*Other Required Hardware*	*Estimated H/W Cost*	*Estimated H/W Cost Breakdown*
ASPECTA	✓	**Full Room** (4 walls + ceiling or floor)	✓ Available	✓	**SP**	✓	✓ (Standard and REST APIs)	**None**	**Security Mirror**	**Low**	**~ $460** (Mirror and projector)
RoomAlive [19]	✓	**Full Room** (4 walls + ceiling and floor)	✓ Available	✓	**MP**	✓	✓	**N Kinects** (1 per projector)	**None**	**High**	**N * ~ $550** (N Projectors and Kinects)
CAVE (CAVELib) [12]	✗ (Special structure required)	**Full Room** (4 walls + ceiling and floor)	✓ Available (Unknown cost)	✗	**MP** (one per surface)	✗ (For perspective corrected VR purposes)	✓ (Open Inventor API)	**None** (Desirable, but not required)	**CAVE structure**	**High**	**~ $2400 + Structure** (6 projectors. High cost structure)
Everywhere Displays Projector [36]	✓	**Partial** (One surface at a time)	✗ Unavailable	✗	**STP**	**Sharing display only**	✓ (Desktop Mirroring)	**Camera**	**Steerable mirror**	**Medium**	**~ $400** (Projector) **+ Camera + Custom Electronic and Mechanical Components**
Ubi Displays [15]	✓	**Partial** (Smaller than room environment)	✓ (~$496 for commercial purposes)	✓	**SP** (Static)	**Within confined area**	✓ (Web app development APIs)	**Kinect**	**None**	**Low**	**~ $550** (Kinect and projector)
Illumiroom [20]	✓	**One Wall**	✗ Unavailable	✗	**SP**	✓	✗	**Kinect**	**None**	**Low**	**~ $550** (Kinect and projector)
BaseLase [29]	✓	**Floor** (5m radius reach)	✗ Unavailable	✗	**CBP** (Laser Pointer based)	✓	✗ (API not freely available)	**7 Depth Cameras** (Asus Xtion Pro)	**6 mirrors** (5 steerable, 1 moulded) **Arduino, sound card, PCBs & frame**	**High**	**~ $1,337** (7 Xtion Pro (Live) cameras) **+ Custom Electronic and Mechanical Components**

SP = Single Projector **MP** = Multiple Projector **STP** = Steerable Projector **CBP** = Custom-Built Projector

Table 2. A comparison of systems related to ASPECTA. Costs of hardware have been estimated using common commercially available products (when possible), e.g. ASUS Xtion Pro Live - $191 [46], Kinect for Xbox One with adapter for Windows - $150 [26, 25]

for designers and programmers trying to decide which platform is best for their particular purposes.

The main motivation of ASPECTA is taking advantage of full or very wide display coverage of a room. Of the systems described in Table 2, both ASPECTA and RoomAlive are suitable for building new FCDs. Although CAVE systems provide better coverage (especially back-projected six-wall CAVEs), these are designed for dedicated environments so they leave out applications supporting most real-life or situated activities. Steerable projector solutions can achieve high coverage as well, but do not cover all spaces simultaneously, which reduces their possible use cases. For example, many of the motivating examples provided earlier in this paper would be cumbersome to implement with Steerable projectors. The other technologies discussed in the table (e.g., UbiDisplays and BaseLase) are generally not practical for providing full coverage.

Many of the compared systems also have significant costs, posing a substantial obstacle for researchers and interactive environment builders. CAVEs are probably the most expensive, even discounting the costs of building dedicated spaces and infrastructure. Some recent work is addressing the built infrastructure cost (i.e., dedicated rooms) by researching pop-up CAVE environments [49], but these still require an investment of many thousands of dollars. Some of the systems such as Illumiroom and Ubi Displays have lower costs if the application focuses on a single plane or reduced coverage angles (and, in case of Ubi Displays, if the application is not for commercial use, which requires an additional paid license). In contrast, all of the costs for ASPECTA systems are low, and the toolkit is freely available we have calculated that a developer in the United States could easily build an ASPECTA system for less than $500.

For other technologies in Table 2, the main obstacle is not cost but the availability of the technology. We are not aware of off-the-shelf steerable display solutions, and systems such as BaseLase are currently custom built. A similar issue is the availability of a software platform that enables the development of applications (e.g., Illumiroom and BaseLase have no toolkit, and Everywhere Displays offers only desktop mirroring). ASPECTA's simple APIs are a major part of its applicability for ordinary developers and researchers. We follow the example of RoomAlive and provide the ASPECTA toolkit as open-source and free software.

Another important parameter in Table 2 is the complexity of configuration, which is directly linked to the sensing that is part of the system. Although ASPECTA requires manual configuration, which might at first appear to require more work than an automated solution, we and our other application users have found configuration to be a lightweight task. It is also important to note that systems with depth sensors or visible-light cameras are far from automatic. For example, the configuration of RoomAlive (which uses one Kinect per projector to establish the shapes, focal lengths, and principal points of the surrounding surfaces) requires that every projector is aimed at a flat surface and has a large object such as a sofa or a few cardboard boxes in front of it for calibration. The current configuration of RoomAlive environments can also require some manual manipulation of XML configuration files. Similarly, despite using depth sensors, Ubi Displays also requires manual configuration (in a similar fashion to ASPECTA) to define surfaces in the projected space.

Finally, it is obviously important for the selection of technology to take into account the type of application being built. ASPECTA and RoomAlive are different in this respect; ASPECTA is geared towards relatively simple applications in 2D while RoomAlive enables sophisticated real-time 3D graphics.

DISCUSSION

Here we discuss the results of the evaluation, the design and value of the toolkit, the technical aspects and technologies available to build FCDs, and the general promise and existing challenges in developing full coverage systems.

The ASPECTA approach to building FCDs

We designed our toolkit to enable a wider audience to take on the challenge of creating full-coverage displays. We believe that there is great research and commercial opportunity in the development of systems that take advantage of spatial and social human behavior through these immersive environments. The future popularity of full coverage displays critically depends on the availability of software and hardware that simplifies their creation and configuration. ASPECTA is part of a wider effort by many research groups that investigates interface alternatives for work, home and play environments. As shown in the previous section and the related work, ASPECTA provides technology and software support for a specific part of the design space - full or near-full-coverage applications that are inexpensive to build.

Open Challenges

Several minor challenges arise from our approach. Spherical projection with mirrors or lenses does not produce uniform distribution of pixels in most room geometries, which means that resolution is not the same across the display space. Additionally, the pixels available through the projector need to be distributed over a larger surface. This affects the available resolution for detailed content: for example, letters smaller than about 5 centimeters are difficult to read in some locations with current projector technologies. This problem will be partially solved with the continued increase in projector resolution, which has increased by an order in magnitude in the last decade. Although resolution limitations might still preclude prototyping of applications that require high-fidelity graphics, we believe that many of the beneficial application ideas do not require high resolution to leverage the spatial ability of interface users.

Brightness is also a limitation because the luminosity of most current projectors is not sufficient to compete with natural sunlight. Projector luminosity will also increase in the future but nevertheless, our approach is already useful in environments with medium to low ambient light, or when subtle indications are sufficient (e.g., highlighting of physical objects already in the room). Other issues such as the appearance of projection on top of existing objects and occlusion are part of the research challenges to be addressed in the near future.

ASPECTA Evaluation

Our evaluation approach for ASPECTA has been multifaceted: first, we studied a specific programming case; second, we designed a semi-controlled experiment with 8 programmers; and third, we have observed several experiences of use with the toolkit from our own group and others. We also provided a qualitative analysis that compares ASPECTA features with other approaches.

Although environmentally valid evaluations of programming tools and APIs are notoriously difficult [13, 54], the evidence that we gathered indicates that the system offers desirable features, that it fills a gap in the existing landscape of tools to build FCDs, and that the API and configuration program's design and implementation enables the creation of FCD applications without much experience and in reasonable time.

Nevertheless, all controlled studies are limited, and ours is no exception: our participants all had some degree of experience, were constrained in terms of time, and had clear instructions in a prepared environment. Although the ultimate evaluation of this work will be the adoption of the toolkit by a broad group of researchers and programmers, our results not only support our effort but also allowed us to refine and improve the design and implementation of ASPECTA's features.

FCD Applications

Some of the most compelling applications of full coverage displays have been demonstrated by earlier work. Systems with high resolution and full awareness of the spatial properties of the space can be used to create novel AR and VR game scenarios [19] and to turn any surface in the room into a digital surface [36].

However, we believe that a large and compelling design space of applications, that can be implemented with low-resolution full coverage displays, is still mostly unexplored. For example, FCDs can expand the digital space of the work desktop into the rest of the room to take advantage of spatial memory for digital files. Similarly, FCDs can leverage the differences in salience of different areas of the room as perceived by the visual system (e.g., front vs periphery) to signal the different urgency of different types of notifications and deadlines. FCDs can provide subtle (calm) hints in the environment about the state of digital systems (e.g., highlighting your daughter's picture frame on the desk if you get an e-mail message from her), and can connect the physical and

digital worlds in meaningful ways such as enabling search of a physical library through a digital search interface (similar to [11]). Further, FCDs enable computer-based control elements in the environment (e.g., dropping a document directly on the printer to print it). We also believe that full-coverage displays provide new ways to glue together multi-display environments, not only through cursor input such as in [53], but through relevant visual connections between related objects that are displayed in separate displays.

CONCLUSIONS AND FUTURE WORK

This paper presents ASPECTA, a toolkit to create full-coverage display applications. The toolkit has been designed to facilitate the implementation of low-cost and low-effort 2D applications that can display content on every wall of a room. We provided an analysis of the opportunities for FCDs as motivation, we described the architecture and implementation of ASPECTA, and we showed how to build a sample FCD application from hardware to code implementation.

We also collected evidence on the use of the toolkit's API and configuration tools through a semi-controlled empirical study with eight programmers and a series of use cases of real applications with multiple research groups. Finally, we analyzed six related technologies to situate the ASPECTA toolkit in the landscape of projector-based interactive interface tools and environments. The results from our studies and analysis indicate that ASPECTA can be a useful tool for furthering research in interactive immersive spaces and for the integration of FCDs in real-world applications.

Plans to extend the ASPECTA toolkit include features to use multiple projectors concurrently, better integration with existing monitors (e.g., screen mirroring), and exploring other curve types for the configuration tool.

ACKNOWLEDGEMENTS

We would like to thank SurfNet (NSERC), and Pufferfish Displays for their support. We are indebted also to the members of the SACHI group from the University of St Andrews. Thanks also go out to all the participants who took part in our user study as well as those who have used ASPECTA in their own research for providing us with information about and images of their software.

REFERENCES

1. Gregory D. Abowd, Anind K. Dey, Robert Orr, and Jason Brotherton. 1997. Context-Awareness in Wearable and Ubiquitous Computing. In *Proceedings of the 1st IEEE International Symposium on Wearable Computers (ISWC '97)*. IEEE Computer Society, Washington, DC, USA, 179–. http://dl.acm.org/citation.cfm?id=851036.856426
2. X. Amatriain, J. Kuchera-Morin, T. Hollerer, and S. T. Pope. 2009. The AlloSphere: Immersive Multimedia for Scientific Discovery and Artistic Exploration. *IEEE MultiMedia* 16, 2 (April 2009), 64–75. DOI: http://dx.doi.org/10.1109/MMUL.2009.35
3. Patrick Baudisch, Nathaniel Good, Victoria Bellotti, and Pamela Schraedley. 2002. Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews, and Zooming. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 259–266. DOI: http://dx.doi.org/10.1145/503376.503423
4. Hrvoje Benko and Andrew D. Wilson. 2010. Multi-point Interactions with Immersive Omnidirectional Visualizations in a Dome. In *ACM International Conference on Interactive Tabletops and Surfaces (ITS '10)*. ACM, New York, NY, USA, 19–28. DOI: http://dx.doi.org/10.1145/1936652.1936657
5. Hrvoje Benko, Andrew D. Wilson, and Federico Zannier. 2014. Dyadic Projected Spatial Augmented Reality. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 645–655. DOI: http://dx.doi.org/10.1145/2642918.2647402
6. Jeremy Birnholtz, Johnathon Schultz, Matthew Lepage, and Carl Gutwin. 2011. A Framework for Supporting Joint Interpersonal Attention in Distributed Groups. In *Proceedings of the 13th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part I (INTERACT'11)*. Springer-Verlag, Berlin, Heidelberg, 295–312. http://dl.acm.org/citation.cfm?id=2042053.2042085
7. Paul Bourke. 2005. Spherical Mirror: A New Approach to Hemispherical Dome Projection. In *Proceedings of the 3rd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia (GRAPHITE '05)*. ACM, New York, NY, USA, 281–284. DOI: http://dx.doi.org/10.1145/1101389.1101445
8. Alan Bränzel, Christian Holz, Daniel Hoffmann, Dominik Schmidt, Marius Knaust, Patrick Lühne, René Meusel, Stephan Richter, and Patrick Baudisch. 2013. GravitySpace: Tracking Users and Their Poses in a Smart Room Using a Pressure-sensing Floor. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 2869–2870. DOI: http://dx.doi.org/10.1145/2468356.2479553
9. Neil Burgess, Eleanor A. Maguire, and John O'Keefe. 2002. The human hippocampus and spatial and episodic memory. (2002).
10. Andreas Butz, Michael Schneider, and Mira Spassova. 2004. *Pervasive Computing: Second International Conference, PERVASIVE 2004, Linz/Vienna, Austria, April 21-23, 2004. Proceedings*. Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter SearchLight – A Lightweight Search Function for Pervasive Environments, 351–356. DOI: http://dx.doi.org/10.1007/978-3-540-24646-6_26
11. D. Crasto, A. Kale, and C. Jaynes. 2005. The Smart Bookshelf: A Study of Camera Projector Scene Augmentation of an Everyday Environment. In

Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on, Vol. 1. 218–225. DOI: http://dx.doi.org/10.1109/ACVMOT.2005.116

12. Carolina Cruz-Neira, Daniel J. Sandin, Thomas A. DeFanti, Robert V. Kenyon, and John C. Hart. 1992. The CAVE: Audio Visual Experience Automatic Virtual Environment. *Commun. ACM* 35, 6 (June 1992), 64–72. DOI: http://dx.doi.org/10.1145/129888.129892
13. Brian Ellis, Jeffrey Stylos, and Brad Myers. 2007. The Factory Pattern in API Design: A Usability Evaluation. In *Proceedings of the 29th International Conference on Software Engineering (ICSE '07)*. IEEE Computer Society, Washington, DC, USA, 302–312. DOI: http://dx.doi.org/10.1109/ICSE.2007.85
14. John Hardy. 2014. *Thesis: Toolkit Support for Interactive Projected Displays*. Ph.D. Dissertation. Lancaster University. http://eprints.lancs.ac.uk/72578/1/Thesis.pdf
15. John Hardy, Carl Ellis, Jason Alexander, and Nigel Davies. 2013. Ubi Displays: A Toolkit for the Rapid Creation of Interactive Projected Displays. *Proceedings of the 2nd International Symposium on Pervasive Displays (PerDis '13)* (2013). http://pervasivedisplays.org/2013/publications/demos/hardy.pdf
16. Tobias Höllerer, JoAnn Kuchera-Morin, and Xavier Amatriain. 2007. The Allosphere: A Large-scale Immersive Surround-view Instrument. In *Proceedings of the 2007 Workshop on Emerging Displays Technologies: Images and Beyond: The Future of Displays and Interacton (EDT '07)*. ACM, New York, NY, USA, Article 3. DOI: http://dx.doi.org/10.1145/1278240.1278243
17. Hans-Christian Jetter, Michael Zöllner, Jens Gerken, and Harald Reiterer. 2012. Design and Implementation of Post-WIMP Distributed User Interfaces with ZOIL. *International Journal of Human-Computer Interaction* 28, 11 (2012), 737–747. DOI: http://dx.doi.org/10.1080/10447318.2012.715539
18. Brad Johanson, Armando Fox, and Terry Winograd. 2002. The Interactive Workspaces Project: Experiences with Ubiquitous Computing Rooms. *IEEE Pervasive Computing* 1, 2 (April 2002), 67–74. DOI: http://dx.doi.org/10.1109/MPRV.2002.1012339
19. Brett Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi, and Lior Shapira. 2014. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. ACM, New York, NY, USA, 637–644. DOI: http://dx.doi.org/10.1145/2642918.2647383
20. Brett R. Jones, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. 2013. IllumiRoom: Peripheral Projected Illusions for Interactive Experiences. In *ACM SIGGRAPH 2013 Emerging Technologies (SIGGRAPH '13)*. ACM, New York, NY, USA, Article 7, 1 pages. DOI: http://dx.doi.org/10.1145/2503368.2503375
21. Joann Kuchera-Morin, Matthew Wright, Graham Wakefield, Charles Roberts, Dennis Adderton, Behzad Sajadi, Tobias Höllerer, and Aditi Majumder. 2014. Technical Section: Immersive Full-surround Multi-user System Design. *Comput. Graph.* 40 (May 2014), 10–21. DOI: http://dx.doi.org/10.1016/j.cag.2013.12.004
22. Alessandro Lai, Alessandro Soro, and Riccardo Scateni. 2010. Interactive Calibration of a Multi-projector System in a Video-wall Multi-touch Environment. In *Adjunct Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. ACM, New York, NY, USA, 437–438. DOI: http://dx.doi.org/10.1145/1866218.1866259
23. Blair MacIntyre, Elizabeth D. Mynatt, Stephen Voida, Klaus M. Hansen, Joe Tullio, and Gregory M. Corso. 2001. Support for Multitasking and Background Awareness Using Interactive Peripheral Displays. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. ACM, New York, NY, USA, 41–50. DOI: http://dx.doi.org/10.1145/502348.502355
24. Thomas W. Malone. 1983. How Do People Organize Their Desks?: Implications for the Design of Office Information Systems. *ACM Trans. Inf. Syst.* 1, 1 (Jan. 1983), 99–112. DOI: http://dx.doi.org/10.1145/357423.357430
25. Microsoft. 2016a. Kinect Adapter for Windows. (2016). http://www.microsoftstore.com/store/msusa/en_US/pdp/Kinect-Adapter-for-Windows/productID.308803600
26. Microsoft. 2016b. Kinect for Xbox One. (2016). http://www.xbox.com/en-US/xbox-one/accessories/kinect-for-xbox-one
27. Paul Milgram, Takemura Haruo, Utsumi Akira, and Kishino Fumio. 1995. Augmented reality: A class of displays on the reality-virtuality continuum. *Telemanipulator and Telepresence Technologies* (1995), 282–292. http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=981543
28. Daniel R. Montello and Martin Raubal. 2013. Functions and applications of spatial cognition. *Handbook of Spatial Cognition* (2013), 249–264. DOI: http://dx.doi.org/10.1037/13936-014
29. Jörg Müller, Dieter Eberle, and Constantin Schmidt. 2015. BaseLase: An Interactive Focus+Context Laser Floor. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3869–3878. DOI: http://dx.doi.org/10.1145/2702123.2702246

30. Miguel A. Nacenta, Regan L. Mandryk, and Carl Gutwin. 2008. Targeting Across Displayless Space. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 777–786. DOI: http://dx.doi.org/10.1145/1357054.1357178

31. Miguel A. Nacenta, Satoshi Sakurai, Tokuo Yamaguchi, Yohei Miki, Yuichi Itoh, Yoshifumi Kitamura, Sriram Subramanian, and Carl Gutwin. 2007. E-conic: A Perspective-aware Interface for Multi-display Environments. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 279–288. DOI: http://dx.doi.org/10.1145/1294211.1294260

32. Miguel A. Nacenta, Samer Sallam, Bernard Champoux, Sriram Subramanian, and Carl Gutwin. 2006. Perspective Cursor: Perspective-based Interaction for Multi-display Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 289–298. DOI: http://dx.doi.org/10.1145/1124772.1124817

33. Bas Ording, Steven P. Jobs, and Donald J. Lindsay. 2008. User interface for providing consolidation and access. (07 10 2008). http://www.google.com.au/patents/US7434177

34. Claudio Pinhanez. 2001a. Using a Steerable Projector and a Camera to Transform Surfaces into Interactive Displays. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems (CHI EA '01)*. ACM, New York, NY, USA, 369–370. DOI: http://dx.doi.org/10.1145/634067.634285

35. Claudio Pinhanez, Rick Kjeldsen, Anthony Levas, Gopal Pingali, Mark Podlaseck, and Noi Sukaviriya. 2003. Applications of steerable projector-camera systems. In *Proceedings of the IEEE International Workshop on Projector-Camera Systems at ICCV 2003*. Citeseer.

36. Claudio S. Pinhanez. 2001b. The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces. In *Proceedings of the 3rd International Conference on Ubiquitous Computing (UbiComp '01)*. Springer-Verlag, London, UK, UK, 315–331. http://dl.acm.org/citation.cfm?id=647987.741324

37. Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. 2012a. The Cost of Display Switching: A Comparison of Mobile, Large Display and Hybrid UI Configurations. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI '12)*. ACM, New York, NY, USA, 99–106. DOI: http://dx.doi.org/10.1145/2254556.2254577

38. Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. 2012b. Factors Influencing Visual Attention Switch in Multi-display User Interfaces: A Survey. In *Proceedings of the 2012 International Symposium on Pervasive Displays (PerDis '12)*. ACM, New York, NY, USA, Article 1, 6 pages. DOI: http://dx.doi.org/10.1145/2307798.2307799

39. R. Raskar, M. S. Brown, Ruigang Yang, Wei-Chao Chen, G. Welch, H. Towles, B. Scales, and H. Fuchs. 1999. Multi-projector displays using camera-based registration. In *Visualization '99. Proceedings*. 161–522. DOI: http://dx.doi.org/10.1109/VISUAL.1999.809883

40. Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. 2003. iLamps: Geometrically Aware and Self-configuring Projectors. In *ACM SIGGRAPH 2003 Papers (SIGGRAPH '03)*. ACM, New York, NY, USA, 809–818. DOI: http://dx.doi.org/10.1145/1201775.882349

41. Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. 1998. The Office of the Future: A Unified Approach to Image-based Modeling and Spatially Immersive Displays. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. ACM, New York, NY, USA, 179–188. DOI: http://dx.doi.org/10.1145/280814.280861

42. Jun Rekimoto and Masanori Saitoh. 1999. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 378–385. DOI: http://dx.doi.org/10.1145/302979.303113

43. George Robertson, Mary Czerwinski, Kevin Larson, Daniel C. Robbins, David Thiel, and Maarten van Dantzich. 1998. Data Mountain: Using Spatial Memory for Document Management. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST '98)*. ACM, New York, NY, USA, 153–162. DOI: http://dx.doi.org/10.1145/288392.288596

44. Manuel Román, Christopher Hess, Renato Cerqueira, Anand Ranganathan, Roy H. Campbell, and Klara Nahrstedt. 2002. Gaia: A Middleware Platform for Active Spaces. *SIGMOBILE Mob. Comput. Commun. Rev.* 6, 4 (Oct. 2002), 65–67. DOI: http://dx.doi.org/10.1145/643550.643558

45. Julian Ryall. 2009. Japanese develop 'television wallpaper'. (2009). http://www.telegraph.co.uk/news/worldnews/asia/japan/4786367/Japanese-develop-television-wallpaper.html

46. Sharif Sakr. 2011. ASUS updates Xtion Pro motion sensor, makes it even more like Kinect. (2011). http://engt.co/1TH8Dgr

47. Satoshi Sakurai, Yuichi Itoh, Yoshifumi Kitamura, Miguel A. Nacenta, Tokuo Yamaguchi, Sriram Subramanian, and Fumio Kishino. 2008. *Interactive Systems. Design, Specification, and Verification: 15th International Workshop, DSV-IS 2008 Kingston,*

Canada, July 16-18, 2008 Revised Papers. Springer Berlin Heidelberg, Berlin, Heidelberg, Chapter A Middleware for Seamless Use of Multiple Displays, 252–266. DOI: http://dx.doi.org/10.1007/978-3-540-70569-7_23

48. Norbert A. Streitz, Jörg Geissler, Torsten Holmer, Shin'ichi Konomi, Christian Müller-Tomfelde, Wolfgang Reischl, Petra Rexroth, Peter Seitz, and Ralf Steinmetz. 1999. i-LAND: An Interactive Landscape for Creativity and Innovation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM, New York, NY, USA, 120–127. DOI: http://dx.doi.org/10.1145/302979.303010

49. W. Stuerzlinger, A. Pavlovych, and D. Nywton. 2015. TIVS: temporary immersive virtual environment at simon fraser university: a non-permanent CAVE. In *Everyday Virtual Reality (WEVR), 2015 IEEE 1st Workshop on*. 23–28. DOI: http://dx.doi.org/10.1109/WEVR.2015.7151691

50. Ivan E. Sutherland. 1965. The Ultimate Display. In *Proceedings of the IFIP Congress*. 506–508.

51. Andrew Wilson, Hrvoje Benko, Shahram Izadi, and Otmar Hilliges. 2012. Steerable Augmented Reality with the Beamatron. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 413–422. DOI: http://dx.doi.org/10.1145/2380116.2380169

52. Craig Wisneski, Hiroshi Ishii, Andrew Dahley, Matt Gorbet, Scott Brave, Brygg Ullmer, and Paul Yarin. 1998. *Cooperative Buildings: Integrating Information, Organization, and Architecture: First International Workshop (CoBuild '98)*. Springer, Berlin, Heidelberg, Chapter Ambient Displays: Turning Architectural Space into an Interface between People and Digital Information, 22–32. DOI: http://dx.doi.org/10.1007/3-540-69706-3_4

53. Robert Xiao, Miguel A. Nacenta, Regan L. Mandryk, Andy Cockburn, and Carl Gutwin. 2011. Ubiquitous Cursor: A Comparison of Direct and Indirect Pointing Feedback in Multi-display Environments. In *Proceedings of Graphics Interface 2011 (GI '11)*. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 135–142. http://dl.acm.org/citation.cfm?id=1992917.1992939

54. M. F. Zibran, F. Z. Eishita, and C. K. Roy. 2011. Useful, But Usable? Factors Affecting the Usability of APIs. In *Reverse Engineering (WCRE), 2011 18th Working Conference on*. 151–155. DOI: http://dx.doi.org/10.1109/WCRE.2011.26