Spatial Peripheral Interaction Techniques for Viewing and Manipulating Off-Screen Digital Content

by

Erik Paluka

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Science in Computer Science

Faculty of Science

University of Ontario Institute of Technology

August 2015

CERTIFICATE OF APPROVAL

Submitted by Erik Paluka

In partial fulfillment of the requirements for the degree of

Master of Science in **Computer Science**

Date of Defence: August 6, 2015

Thesis title: Spatial Peripheral Interaction Techniques for Viewing and Manipulating Off-Screen Digital Content

The undersigned certify that the student has presented his thesis, that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate through an oral examination. They recommend this thesis to the Office of Graduate Studies for acceptance.

Examining Committee:

____________________________________ Dr. Andrew Hogue
Chair of Examining Committee

____________________________________ Dr. Christopher Collins
Research Supervisor

____________________________________ Dr. Jeremy Bradbury
Examining Committee Member

____________________________________ Dr. Pejman Mirza-Babaei
External Examiner

□ As research supervisor for the above student, the thesis was rendered acceptable without revisions.

____________________________________ Dr. Christopher Collins

□ As research supervisor for the above student, I read and approved the changes required by the final examiners and recommend the thesis for acceptance:

____________________________________ Dr. Christopher Collins

Acknowledgements

I would like to thank Dr. Christopher Collins and the members of the Visualization for Information Analysis Lab (vialab) at the University of Ontario Institute of Technology for their support over the past few years.

Abstract

When an information space is larger than the display, it is typical for interfaces to only support interacting with content that is rendered within its viewport. To support interacting with off-screen content, our work explores the design and evaluation of several spatial off-screen exploration techniques that make use of the interaction space around the display. These include *Paper Distortion*, *Dynamic Distortion*, *Dynamic Peephole Inset*, *Spatial Panning*, and *Point2Pan*. We also contribute a formalized descriptive framework of the off-screen interaction space that divides the around-device space into interaction volumes and analyzes them based on different factors. This framework guided the design of an off-screen interaction system, called Off-Screen Desktop, which implemented our spatial techniques using consumer-level motion sensing hardware. To enable a more detailed analysis of spatial interaction systems, we also developed a web-based visualization system, called SpatialVis, that visualizes log data over a video screen capture of the associated user interface.

Keywords: around-device, off-screen, spatial user interface, spatial interaction, human-computer interaction

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the beginning of the personal computer revolution, interacting with digital content has, for the most part, revolved around physically manipulating indirect input devices (e.g. pointing devices, analog sticks). The problem with this type of human-computer interaction is that it is not natural when compared to how humans interact in the physical world. In large part, we tend to not use an external device to grasp an object; we use our hands to directly perform the action. If we can leverage these interaction skills that we have honed since childbirth instead of creating and using new unfamiliar ones, the communication between man and machine will become more intuitive and natural, and have a lower learning curve. Direct touch technology and tangible user interface objects have helped to bridge this gap; however indirect input devices that require physical contact are still ubiquitous in the computing world. With the emergence of consumer grade depth and motion sensors in the past few years, the established input paradigm is yet again being challenged. People are now able to use free-hand and whole-body gestures and positioning to manipulate digital information without the need for markers. This motion sensing ability combined with future robust object recognition algorithms and spatial feedback mechanisms will help to further eliminate the barrier between bits and atoms [91], thus possibly enabling a more natural human experience.

This interaction evolution brings with it many opportunities and challenges. Standard interfaces are constrained to two types of input: the movement of a single cursor, and the state of buttons. When using gesture-based spatial interaction, there are two types of input as well: movement of objects and/or organisms, and the state of those objects and/or organisms. The difference is in the myriad of states, movements and positions that these objects, organism or parts thereof can be in. Also, most desktop user interface designs in today's age are based on the WIMP paradigm (windows, icons, menus, pointers) and rely on the accuracy of pointing devices to allow their GUI elements to have small input spaces. Usability issues occur when spatial interaction techniques are used to interact with these interfaces. Since the human body is constantly moving when not supported by an object or surface, the cursor tends to move in and out of the GUI elements' small input spaces when using a finger, hand or arm as input. Positional averaging, larger interaction spaces for elements and better transfer functions may ameliorate the small input space problem, but an interface designed with these new types of interactions in mind would make for a better user experience.

Currently, the most popular way to enable spatial interaction on desktop and touch-enabled computers is to use a hand or finger as a controller for the pointer. This creates interesting opportunities for blending direct-touch and spatial interaction. For example, the user can select an object by directly touching the screen and maintain said selection even as they lift their finger off of the screen. In another example, the user could translate the object by sliding their finger off of the edge of the screen and then into mid-air. Although, in popular operating systems, the information space is coupled with the display space; therefore the object would stop translating when it reached the edge of the screen. What if the information space was no longer bounded to the size of the display? What if digital content could be moved off-screen in the same plane as the display (XY plane) as if the display was larger than it actually is? Although this space would not contain direct visual feedback, spatial interaction could be used to interact with content within it. For 2D information spaces, one can imagine this as if the information space extended past the boundaries of the display while still supporting touch interaction. One could interact with specific off-screen content by performing some type of mid-air gesture at the physical location that is associated with the interested content.

In this work, we present the design of different spatial off-screen interaction techniques, and their implementation and evaluation within the Off-Screen Desktop system. Guided by our formalized descriptive framework of different around-device interaction spaces, our techniques were designed to support the exploration of off-screen information spaces, as well as directly interacting with off-screen content. Instantiations of these designs were then implemented using the Off-Screen Desktop system to facilitate our exploration of spatial off-screen interaction and related phenomena. This multimodal system employed consumer-grade motion sensing hardware to allow us to understand how off-screen interaction would occur in the real world. This led to the improvement of our designs and the creation of different techniques to deal with poor recognition and other challenges inherent in the wild. To provide another means of improving spatial interaction designs, our work also dealt with the development of SpatialVis. Our web-based system visually encodes logged spatial interaction data to facilitate its analysis and how the user interface is affected. SpatialVis accomplishes this by temporally and positionally mapping different visualizations of the interaction data with a video screen capture of the user interface.

1.1 Motivation

To take advantage of the advanced visual system of humans, it is common for computing devices to employ photonic technology as a feedback mechanism for the user. Although beneficial, computer displays tend to be smaller than our field of vision especially with respect to mobile devices. Therefore, when the information space is larger than the display, the screen turns into something similar to a peephole, where only a small portion of the space can be seen. This also limits user interaction with the information space since it is common practice to only allow content to be interacted with if it resides on-screen. To always allow content to be interacted with (e.g., widgets), popular operating systems and applications constrain the information space to the size of the screen. Albeit, this causes the display space on many systems to suffer from what we call the *graphical clutter problem* (see Figure 1.1). In standard GUIs, the screen is often cluttered with elements that are not always needed for the user's current task (taskbar, icons, ribbons, etc.). These elements may potentially be distracting, as well as reducing the space allocated for the content and tools needed for the task. Standard solutions include minimizing visual elements, such as windows, to a designated storage space on the screen. However, this does not take advantage of human capabilities to work efficiently with custom layouts of information spaces [143]. The graphical clutter problem is synonymous to the *screen real estate problem* where information spaces are larger than the display space and distortion, compression, or clipping techniques are required to fit content on-screen [28].

To reduce graphical clutter and optimize the display space, we propose to use the off-screen area at the periphery (edges) of the display, combined with spatial interaction techniques to interact within this space. One might imagine this idea as if the movement of the cursor (pointer) is no longer limited by the boundaries of the screen; essentially extending the interactable information space. If the limits of movement for digital content are decoupled from the display space boundaries, then the aforementioned visual pollution can be mitigated through the use of the surrounding off-screen areas. It also allows designers to possibly create richer user experiences by

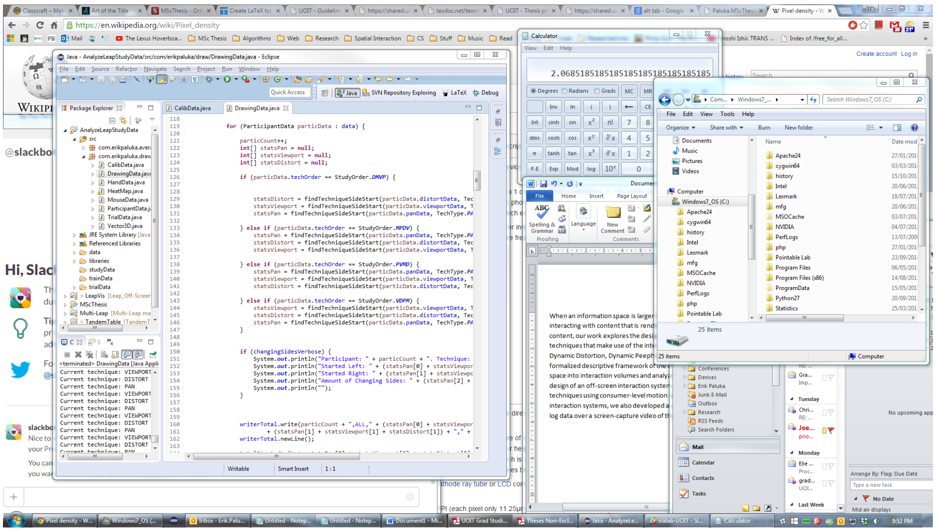


Fig. 1.1 The above figure shows how systems are subject to the *graphical clutter problem* when their information space is constrained to the size of the display. Having multiple windows and other user interface elements on-screen simultaneously can be distracting and cause cognitive overload.

using off-screen areas for other applications and use cases. Digital content could be placed off-screen, as well as automatically appear there, such as when an application is invoked or an email is received. This space could also be partitioned to create areas for different purposes, such as for user-defined storage (e.g. files, windows, directories, application specific content, and toolboxes), or notifications and incoming feed content (e.g. newsfeed, emails, chat messages).

In addition to the display periphery being used for storage and object placement, this space could be used as an area to perform spatially defined interactions to invoke system-wide and application-specific operations, such as changing the volume by vertically moving one's hand at the right side of the display. Another example is tossing a file into an off-screen section on the right side of the display to delete it, and tossing it into another section to copy it to network storage. This would further reduce the need for on-screen widgets, as well as reduce the amount of time required for triggering operations through direct manipulation.

Since the physical space around the screen is employed to store content, people are able to take advantage of their spatial cognition capabilities to create custom layouts and mental maps of the virtual environment [143]. Furthermore, by having users place their hands beside the display to interact with off-screen content or to invoke commands, they also benefit from their sense of proprioception reinforcing their spatial memory (e.g., spatial relationship between items) [169]. For example, when storing an object off-screen on the right side of the display, placing one's hand in the actual physical location where the object is stored reinforces one's knowledge of the object's location in the information space. The benefits of proprioception increases when users have a greater opportunity to develop an associated neural representation [53], making off-screen areas that are always associated with a specific purpose within an application or system advantageous. Users' spatial memory is also aided by being able to use the display as a spatial reference point. When trying to retrieve the object from the last example, one is able to reduce the size of the information space that needs to be searched due to knowledge that the item was stored on a particular side of the display. Spatial memory of an element's horizontal and vertical distance from the display (e.g., top far right corner) can also be used to quickly access the item, especially since people develop an accurate memory for the locations of frequently accessed interface elements [147]. To facilitate this and further strengthen users' spatial memory, landmarks and other visual cues can be embedded in the interface [143]. Additionally, the display makes it easier for users to spatially interact with the virtual environment since gestures are performed relative to a real-world object instead of in empty free-space [79].

We believe that these ideas would work well with many systems, but we find that spatial interaction techniques would fit particularly well with touch surfaces as the movement from touching a surface to performing mid-air gestures is much more fluid than it would be with a mouse or other input device. Furthermore, when these surfaces are quite large, spatial gestures would enable quick triggering of operations without requiring the user to change locations to touch the relevant element(s) such as a tool panel. On the other end, small surfaces (e.g. tablets and smartphones) would benefit the most from reduced graphical clutter and expanded virtual screen space as their screen real estate is quite low.

1.2 Contributions

To the community at large, our work contains four contributions. Our first contribution includes a formalized descriptive framework of the off-screen interaction space that divides the around-device space into interaction volumes. Second, we contribute the designs of our spatial off-screen exploration techniques that enable one to view and interact with off-screen content. Our third contribution consists of a visualization technique called SpatialVis for studying logged spatial interaction data and how the user interface is affected. This technique involves temporally and positionally mapping logged spatial interaction data to a video screen capture of the associated user interface. Our final contribution includes the study results from a comparative evaluation between three of our spatial off-screen exploration techniques and traditional mouse panning.

1.3 Organization

This document is structured as follows. Work related to around-device interaction, interacting with off-screen objects, and visualizing off-screen content is discussed in Chapter 2. In Chapter 3, we describe a formalized descriptive framework that analyzes the around-device interaction spaces with respect to different device types, gestures, and content. The designs of our spatial off-screen exploration techniques are then discussed in Chapter 4. Chapter 5 presents the design and implementation of an off-screen interaction system called Off-Screen Desktop. Chapter 6 discusses our web-based visualization system, called SpatialVis, that facilitates spatial interaction analysis. Chapter 7 describes a quantitative study that was performed to comparatively evaluate three of our spatial off-screen exploration techniques and mouse panning. This same Chapter contains the study results, along with a discussion of the findings. We discuss how our work can be extended, as well as other additional ideas relating to off-screen interaction in Chapter 9. A summary of the document is then presented in Chapter 10.

Chapter 2

Related Work

In the following sections, we will discuss previous related work that deal with around-device interaction, interacting with off-screen objects, as well as visualizing off-screen content.

2.1 Around-Device Interaction

The standard way of interacting with computers has been with the use of indirect pointing and text entry devices (e.g., mouse and keyboard). With the rise in popularity of smartphones and tablets, there has been a shift to interfaces where one's fingers are used to directly manipulate objects on-screen. These devices have been describe as being more "natural" than traditional input techniques [49] and can support multiple simultaneous points of input [64]. The sides or back of these systems can also be leveraged to expand the interaction space (e.g., [149]). A problem with these techniques is that the input space is constrained to the size of the device. To get past this limitation, researchers have worked on using the space above, below, and around the device as additional input areas.

2.1.1 Above and In Front of the Device

In terms of mobile devices researchers have attached infrared proximity sensors to support above the device interaction as seen in the work by Kratz and Rohs [114]. In *GaussSense*, Liang et al. used a 2 mm thick magnetic sensing board attached to the back of a smartphone to be able to detect when a specially designed stylus is hovering above the device [124]. Their system is also able to detect different styli, as well as pressure and tilt levels. The *MagPen* system took a similar approach to enable

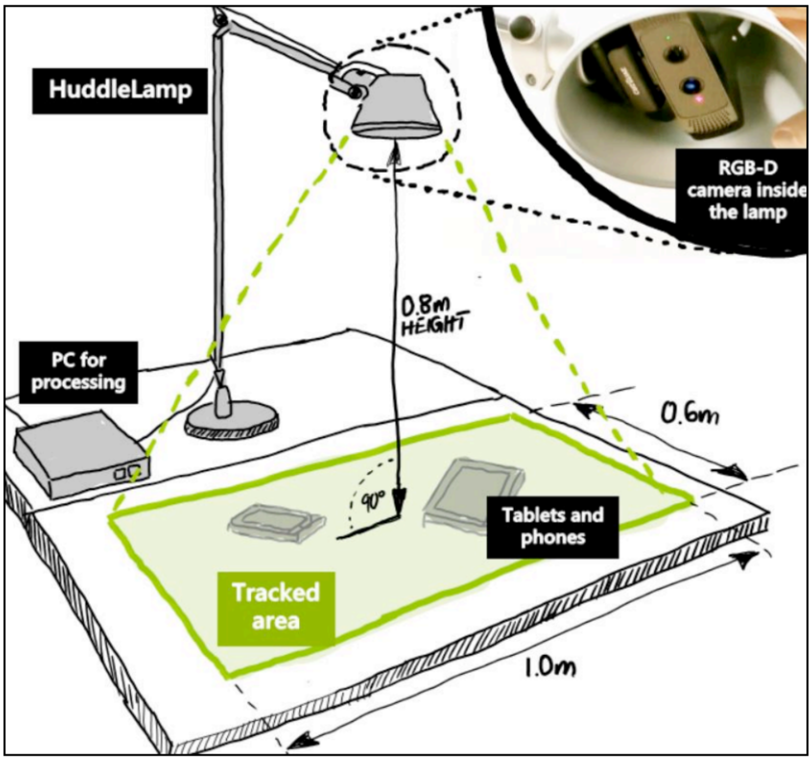


Fig. 2.1 Screenshot image of the © HuddleLamp system by Rädle et al. [138]. HuddleLamp supports ad-hoc around-the-table collaboration and cross-device interaction by using a RGB-D camera above a surface.

a subset of these interaction techniques on commercially available smartphones [86]. Additionally, their setup can detect when a spinning stylus gesture is performed near the device. To aid the design of causal interactions above a device, Pohl and Murray-Smith built and studied a prototype that supports different levels of user engagement [134]. This allowed them to determine that users' level of engagement changes based on the difficulty of tasks. This further gives credit to the idea of being able to complete a secondary task (e.g., dismissing a call on a smartphone with a gesture above the device) without requiring one to fully remove attention from a primary task (e.g., talking with a friend) [136]. In *HuddleLamp*, dynamic displays are able to be created by adding or removing multiple mobile devices with the ability to detect interactions above, on, and between them [138] (see Figure 2.1).

When working with a desktop or laptop computer, people tend to rest their hands on or near the keyboard when typing. Instead of requiring one to switch input devices to control the on-screen cursor, the space above the keyboard could be utilized. In this regard, Taylor et al. took a mechanical keyboard and embedded a low resolution matrix of infrared proximity sensors between the keys [158]. This enabled their system to detect gestures above and on the keyboard with the use of a random decision forest classifier.

With respect to interaction above tables, Wilson used a depth camera to integrate physical objects placed on a table into the digital world, as well as to support spatial gestures above the table [166]. This system used a projector and a non-digital table, but enabling interactions above digital tables is also feasible (e.g., [43, 137, 157]). Hilliges et al. explored ways to allow digital objects to be "picked up" in a more intuitive manner on interactive tables [76]. They created two different systems that could sense the height of hands above the surface with the use of sensors below it. To aid users, the systems also included virtual shadows as a natural feedback mechanism for representing users' hands in the virtual world. A similar project used a projection screen that can electronically switch between clear and diffuse states quicker than the human eye can perceive [92]. This allowed for mid-air gesture detection as well as projection and interactions on separate mobile projection surfaces.

Unlike digital tables, television and computer displays tend to be vertical and face the user. Instead of using direct touch, the space between the user and the screen could be used as an interaction space. In this regard, Hirsch et al. created the *BiDi Screen* which is a transformed LCD screen that supports both direct multi-touch surface interaction, as well as mid-air gestures in front of the screen [80]. These interactions are also supported by the *DepthTouch* system, in addition to tracking a user's head to provide a correct perspective view of a 3D virtual environment [11]. Molyneaux et al. took a different approach by combining a handheld projector, inertial measurement unit and an infrared camera to enable shadow based interactions [130]. When an object or hand is placed in front of the projector, it casts a shadow onto the projected scene. The infrared camera is then used to infer the location of the hand to enable the shadows to interact with the virtual world. The *RetroDepth* system employed infrared-based technology as well, but differs in the sense that it is able to detect many different interaction modalities including touch, pressure, styli, objects, and spatial interaction [111]. It is able to accomplish this by simply using off-the-shelf retro-reflective material, diffuse infrared LEDs and two infrared cameras. To be able to interact with distant displays in the context of pervasive computing, Rateau et al. studied the idea of allowing people to define their own dynamic spatial interaction areas [141]. Using the insights gained from their study, the researchers also created a prototype that allowed multiple users to control and share a cursor on multiple displays at a distance.

All of these devices have employed a visual feedback mechanism, but the work by Gustafson et al. has shown that it is not required [61]. They studied the concept of *Imaginary Interfaces* which are screen-less devices that support spatial interaction without instrumentation of the hands. By using a wearable camera for input, their system allows one to interact in a user-defined space with the help of their "visuospatial memory".

2.1.2 Below and Behind the Device

With the advantages of depth cameras, it is only a matter of time before they are miniaturized and embedded into commercially available mobile devices. To enable precise rotation of 3D content using this setup, Kratz et al. created a prototype that is able to detect spatial gestures behind the device [115]. Their system was empirically shown to be faster than when using a touch-based virtual trackball. In the work by Hwang et al., only the RGB camera embedded in commercially available smartphones was required to perform behind the device interaction [88]. They accomplish this by determining the colour of a user's fingernail which changes when the finger is stretched straight or pressure is exerted upon it.

In the *HoloDesk* system, people are able to reach behind a beam splitter to be able to interact with a 3D virtual environment [77]. With the use of an RGB camera that tracks the user's head, the system is able to display the correct perspective of the scene's contents. A Kinect motion sensor is integrated as well to enable the use of hands and physical objects to interact with the digital world. The *MixFab* system also employs this setup with the goal of reducing the amount of effort required when designing 3D models for personal fabrication (e.g., 3D printing) [164]. Leithinger et al.'s work on the *Sublimate* system used a beam splitter in a similar fashion to allow for the manipulation of an actuated shape display that was augmented with head-tracked stereoscopic graphics [122]. Instead of using a half-silvored mirror for a beam splitter like the previous projects, *SpaceTop* employed a transparent LCD to be able to view the physical world behind it as well as digital content [120]. The system is setup similar to a traditional desktop configuration except with the keyboard behind the display to allow for the easy transition from spatial to keyboard input.

2.1.3 At the Sides of the Device

To explore using desktop computers in conjunction with multi-touch tables, Bi et al. created the *Magic Desk* [13]. This system integrated an interactive table into a standard desktop configuration and displayed interactive content around the keyboard and mouse. Although interaction was constrained to the plane of the table's surface, they were able to augment traditional desktop experiences. Using a similar multi-touch table and desktop configuration, Hausen et al. studied the use of freehand, touch and graspable interaction at the right side of the keyboard and mouse [72]. This was done in the context of transferring attention and interactions into the periphery when confronted with a secondary task with the goal of reducing mental effort. Similarly, Hausen et al. evaluated the same peripheral interaction modalities with an eight week in-situ deployment [71]. Their results indicate that all three modalities can be successfully used for peripheral interaction in terms of vision and attention, and can also reduce cognitive load. Work has also been done on exploring the design of peripheral ambient calendars that maintain awareness of events while at the same time minimizing disruptions [69]. This idea was realized with a prototype that displays a calendar visualization beside a keyboard on a table and allows only a few simple spatial gestures for interaction.

To add touch input capabilities to any surface that a device lies upon, Xiao et al. developed a sensing approach called *Toffee* [168]. This technique employs acoustic time differences of arrival correlation with data from four vibro-acoustic piezo sensors that are attached to the bottom of a phone for example. This allows the system to detect when a user taps on the surface around this phone. In another project, Butler et al. attached infrared proximity sensors to a mobile device to enable finger presence and position detection in the space at the sides of the device [24]. When placed on a flat surface, this allowed for single and multi-touch interaction to be used in conjunction with the device's screen input space. Pohl and Rohs explored repurposing the space and objects around a mobile device as ways to perform casual interactions and called these objects around-device devices [135]. To support this idea, the researchers ran two different studies to determine the typical locations of mobile devices and what objects tend to be near them, as well as what objects people would employ for ten different types of interactions. In one of the prototypes made for the *PalmSpace* project, researchers attached a depth camera to a mobile device to enable spatial interaction at one side [115]. It was found that users' perception of overall effort and required force for their around-device technique was significantly lower than a touch-based interaction technique. Instead of using a depth camera, the *Surround-See* project uses an omni-directional lens mounted on a smartphone's front-facing camera [171]. This allows their system to recognize objects, user activities and the environment at the sides of the device.

2.1.4 Above, Below and Around the Device

To support interacting with very small devices, Harrison and Hudson created the *Abracadabra* input technique which uses a magnetometer in a device and a magnet attached to a stylus or finger [65]. To study their approach, they created a custom smartwatch with a 1.5 inch display and a dual axis magnetometer. It was found that it allowed for the precise selection of radial targets that are as small as 16°, as well as targets that are smaller than the tip of a finger. Similarly, other projects have employed the magnetometers that are already embedded in commercially available smartphones to support interactions around the device [85, 109], as well as for text entry [108], user authentication [107], and music playing [106]. Hemmert et al. created a mobile phone prototype that changes its form based on the proximity of a user and its relationship with him [74]. With the use of servo motors, the system is able to react with five different types of postures: affection, attention, ignorance, anxiousness and aversion. Other researchers have developed a set of around-device interaction techniques for performing multi-scale navigation on mobile devices [96]. Their study showed that spatial interaction performance can be as good as touch, and that constraining the around-device interaction space can negatively affect the user experience.

While not supporting gestures in their current prototype, Avrahami et al. developed the *Portico* system which enables the detection of objects around and on tablet computers [3]. With the use of two cameras that are attached to the tablet, the system is able to detect interactions in an area that is six times the size of the screen. Instead of relying on a fixed display, Wilson and Benko created the *LightSpace* system which allows interactions on, above, and between everyday surfaces and environments [167]. It is able to accomplish this with the use of projectors and depth cameras that are calibrated to 3D real world coordinates.

2.2 Interacting With Off-Screen Objects

When the information space is so large, it is common that objects of interest lie outside of the viewport. The standard way to bring this content on-screen for interaction is to use the zoom and pan techniques as seen in the zoomable graphical interface system called *Pad++* [9]. One can also employ the peephole technique by moving a spatially aware display around a statically situated information space to reveal off-screen content (e.g., [27, 48, 172]). To maintain the current view of the workspace and interact with off-screen content, Irani et al. created the *hop* technique where a "laser beam" is used to create on-screen proxies of the off-screen objects that lie in close proximity to the beam's path [90]. These proxies appear close to the on-screen cursor and allow one to either select the object of interest or jump to its off-screen location. This technique was empirically shown to be significantly faster at selecting off-screen objects than the zooming and panning techniques. The *Vacuum* technique employs a similar approach with a dynamic arc for accessing items on a large display, but could also be used for off-screen interaction if paired with an off-screen visualization technique [12]. To facilitate selection and navigation in a network with off-screen or distant nodes, Moscovich et al. developed the *Bring & Go* technique [131]. When a node is selected, all of the nodes that are connected to it are translated into the viewport while still preserving their spatial relationships. The user can then select one of these nodes to move to its location within the network. In *EdgeSplit*, the borders of the viewport are used to visualize off-screen objects using proxies [84]. These regions are also partitioned to provide each proxy with a larger input space to facilitate the selection of off-screen content. In a study comparing this technique to a similar one based on voronoi partitioning, as well as a selectable version of the *Wedge* technique [60], *EdgeSplit* was found to be significantly faster than both for two different selection tasks.

The aforementioned work have all kept the cursor on-screen, but one can also select off-screen content directly using spatial gestures in the space around a device. To support this type of interaction, Ens et al. created two performance models for different styles of "direct off-screen pointing" when it is employed on a fixed plane around a device and assisted with visual cues of off-screen objects [46]. These models were validated with an empirical study which also showed that assisted direct off-screen pointing can be up to four times slower than on-screen pointing. In addition, they found that performance time and accuracy is dependent on the visualization type employed, and that using coarse grained selection instead of fine grained can significantly improve performance time. It is important to note that their study setup used an interactive surface which emulated a mobile device and its surrounding space. Therefore, their system included haptic feedback, where in our system, there is none. Similar to our work, Hausen et al. created a system that allows users to place content off-screen onto a table [70]. Called the *Unadorned Desk*, this system used a Kinect motion sensor to detect when people hover over or touch a table with their hand which are then used to select items stored on the desk. The researchers ran two studies and concluded that users are capable of interacting with a small number of virtual items on the desk even when visual feedback is not present. Instead of using a physical object, Hasan et al. used the plane defined by a mobile device as an storage and interaction space in their *AD-Binning* system [67]. This work is most similar to ours as our interaction

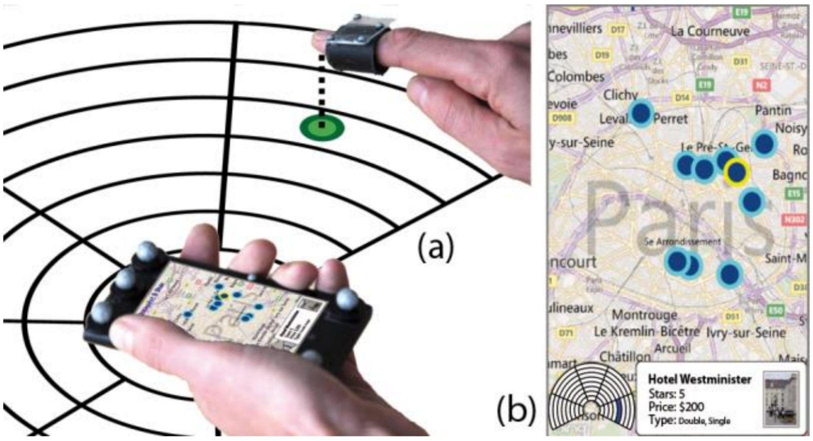


Fig. 2.2 Screenshot image of the © AD-Binning mobile interface by Hasan et al. which used a high-end motion capture system to enable off-screen interaction [67].

and storage space is a mid-air plane as well, but defined by a vertical monitor. In *AD-Binning*, people can store and retrieve content from mid-air at a max distance of 40cm in a circular layout (see Figure 2.2). This space was divided into five sectors to facilitate spatial recall which were then further discretized using three different methods to provide better user control. These methods were uniform discretization where each bin is of equal size, distance dependant discretization where bins farther away from the device are larger, and fisheye discretization where the active and adjacent bins grow in size. In their work, the researchers also studied the use of six different selection techniques for off-screen interaction. These included tapping the front (Tap) or back (BackTap) of the device, dwelling in mid-air (Dwell), quickly raising (LiftOff) or lowering (Pierce) the pointing finger in mid-air, and performing a down-up motion (DownUp) in mid-air. DownUp had the lowest error rate, but worst performance time, whereas Tap had the second lowest error rate and fastest performance time. Work has also been done on comparing direct off-screen pointing, peephole, and standard on-screen flick and pinch gestures in a map navigation and exploration scenario using a mobile device [68]. Results from a study indicated that the spatially-aware navigation techniques (direct off-screen pointing and peephole) were 30% faster than the standard on-screen gestures (flick and pinch) for navigating between workspace locations.

2.3 Visualizing Off-Screen Elements

A typical way of viewing off-screen content is to extract and only display the most important data from the information space or change its format to allow larger content to fit on small screens (e.g., [14, 26, 32, 159]). Another option is to include multiple views which can also support comparing and relating distant objects [161]. However, without the use of data transformation or extraction, it is imperative that the locations of this content be communicated to the user in the first place since it lies outside of the visual feedback space. Without such communication, information spaces will suffer from *"desert fog"*; resulting in users not knowing the existence or position of off-screen objects [100]. When the existence of off-screen objects is known, visualizations employed to indicate their position have been shown to improve search performance [75].

2.3.1 Visual Cues

Arrows are a popular way to indicate the locations of off-screen content or to provide navigation cues (e.g. [33, 44, 148]). The problem with this type of visualization is that it only conveys direction and not distance. This can be mitigated by scaling or stretching the arrows to encode distance, but without spatial awareness of the off-screen space, estimating the exact locations of objects can be difficult [21]. In contrast, modern scrollbars provide information about the relative size of the off-screen area, as well as direction and distance cues of on-screen and off-screen elements with the use of trough marks as seen in the Google Chrome web browser when performing a word search.

An advantage of scrollbars is that they tend to have good space efficiency which is important when overlaying dense information spaces with additional control elements or visual cues. In this regard, Zellweger et al. designed a family of space efficient fisheye techniques, called *City Lights*, to visualize off-screen content [173]. Residing along the borders of windows, these compact contextual views are able to communicate awareness, identification, navigation, and interaction. Baudisch and Rosenholtz employed these design ideas to the create the *Halo* visualization technique which surrounds each off-screen object with a ring that extends into the inner edge of the display [7]. Since each ring is centred on the location of an off-screen object, users will be able to infer the distance and direction of the object due to the gestalt perception law of closure [162]. Distance is also encoded using transparency with closer items being more opaque. One disadvantage to this technique is that it is susceptible to overlapping halos resulting in reduced readability. To avoid this problem, Gustafson et al. employed a wedge, instead of a ring, that rotates to avoid overlap [60]. Each wedge is an acute isosceles triangle with its tip located near the off-screen object, and the other two corners of the wedge appearing on-screen.

A problem with these techniques is that they result in too much clutter when dealing with a large number of dynamic off-screen moving objects. To mitigate this, Gustafson and Irani designed *EdgeRadar* [62] which displays scaled down versions of off-screen objects in overlay regions at the edges of the display. This work has also been extended by Jo et al. to work in 3D environments [95]. Another project worked on reducing clutter by extending the original *Halo* technique with aggregation abilities. In *HaloDot*, neighbouring rings are aggregated into a single ring with textual information to inform the user of the number of objects within each ring's region [54]. This technique also draws a small circle on each ring to help users infer the direction of each off-screen object. In the case of aggregated rings, the small circle represents either the midpoint of the objects or the most relevant object.

The aforementioned techniques are advantageous due to their minimalist design, yet they do not include detailed information concerning each object's surrounding space. *Dynamic Insets* solves this problem by displaying insets for each off-screen object which includes the object itself and its surrounding area [52]. The direction of each off-screen object is encoded by the position of its inset at the edge of the screen. The authors also explored using textual information, border colour, transparency, and size to encode distance information. Other projects have also dealt with visualizing off-screen spatial data and uncertainty (e.g., [93]), and bar graphs and scatter plots (e.g., [51]).

2.3.2 Overview + Detail

Another way of showing users information that lies outside of the viewport is with overview plus detail (O+D) presentation. These interfaces consist of two separate views where one is used to present the user with an overview of the information space while the other view provides detail at a more granular level [34]. The O+D technique is typically used in map-based applications and contains a viewfinder in the overview to communicate to the user the location of the detail view in the information space. This technique is also better suited to be used in conjunction with larger screen sizes [20], and works best when the overview contains detailed information as opposed to just a wireframe [22]. Furthermore, users have been shown to prefer its inclusion when performing navigation tasks due to its ability to support navigation and keeping track of positions on maps [83]. Another place where O+D views are used quite often is in text editing software where scrollbars are augmented with a compressed version of the entire text file (e.g., [94, 128]). Cox et al. took a different approach and overlaid the overview onto the detailed view with the use of transparency to reduce occlusion problems [38]. This allowed both views to be the full size of the viewport and proved to be useful. When examining dynamic data in an O+D interface, it is hard to keep track of objects that move in and out of the detail view. To solve this problem, Ion

Fig. 2.3 Screenshot image of the © Canyon information visualization technique by Ion et al. [89]. This technique attaches a small inset to the detailed view when an object moves beyond its boundaries.

et al. created the *Canyon* visualization technique which, for each object of interest that has moved outside the detail view, attaches a small inset to the boundary of the view [89]. The space between the boundary and the object is folded into the screen to provide spatial awareness while at the same time conserving space (see Figure 2.3).

When comparing different techniques, O+D has been shown to be more effective than Wedge and scaled arrows when the task requires knowledge of the spatial configuration of off-screen objects since users can view this configuration in the overview [19]. People also performed better and experienced less mental workload while using an O+D interface when compared against the Wedge technique in dynamic scenarios with a large number of off-screen objects [23]. Although, in the end, choosing the most effective off-screen visualization technique for one's system depends on the tasks that the users will perform, as well as the number and distribution of off-screen objects [55].

2.3.3 Focus + Context

When dealing with information systems, a common task for knowledge workers is to relate multiple items in their full context. This becomes problematic when at a required detail or zoom level, some of these objects are outside of the current viewport. In this regard, different focus plus context techniques have been developed to address this problem [123]. Instead of showing detailed information of all objects and pushing some of them outside the viewport, detailed information is only shown for a subset at points of foci [146]. The *Bifocal Display* is an early version of these techniques and involves a central region of focus that is surrounded by a compressed version of the rest of the information space [153]. The *Perspective Wall* takes a similar path by taking a two-dimensional layout and folding it into a three-dimensional visualization [125]. As with the *Bifocal Display*, the *Perspective Wall* is made up of three panels with the central one providing detail and the other two providing context. With *Magic Lens* filters, users have more freedom when specifying regions for which they would like to see more detail as they can be arbitrarily shaped [155]. The problem with these techniques is that there is a sharp boundary between the detailed and context views.

A class of distortion techniques, called fisheye views, provide detailed information of elements based on a "Degree of Interest" function that is based on the importance of each element and their distance to the point of focus [50]. Visual continuity is achieved by magnifying or providing more detail in the area surrounding a point of focus with a continuous fall-off in magnification towards the edges. Multiple fisheye instances can exist simultaneously in an interface [101] and usually employ geometric transformations as seen in the work by Sarkar and Brown [145], and Carpendale [29]. When using fisheye views for comparing content, a problem occurs when the distance between multiple regions of interest is so large that they cannot fit into the display space at the same time. Elmqvist et al. solved this problem by using a paper folding metaphor to fold the space between different focal points into the depth dimension [45]. The folds are kept visible to support context and distance awareness of the space between the foci. Butscher et al. expanded on this work by integrating direct bimanual manipulation to create and interact with the folded space [25]. Similarly, Shum used the folding technique to allow multiple people to simultaneously navigate a virtual space on a digital table without disrupting each other [151].

Other researchers have employed fisheye techniques for document viewing (e.g., [6, 144]), dealing with small display spaces on mobile devices (e.g., [8]), and visualizing large tables of information (e.g., [140]) as well as large hierarchies (e.g., [73, 117]). Although, when a fisheye lens is enabled and coupled to the cursor, selecting targets in a new point of focus can be difficult since objects appear to move when the lens approaches them. Gutwin created the speed-coupled flattening technique to mitigate this problem and works by dynamically reducing the distortion level based on the cursor velocity and acceleration [63]. In terms of efficiency, focus+context techniques have been shown to help people extract information from large static documents faster than when using overview plus detail and zooming/panning techniques, as well as reducing error rates [5].

2.3.4 Expanding The Display Space

Instead of creating techniques to visualize off-screen content, work has been done on expanding the display space to allow more content to fit on-screen. One way of accomplishing this is by augmenting the space around a computer monitor with the use of a projector as seen in the focus+context prototype by Baudisch et al. [5]. The creators of the *IllumiRoom* system used this method to project visualizations around television screens to enhance traditional films, television shows and games [97]. This system differs from others since it is able create visual experiences based on the appearance and geometry of the surrounding room that lies in the periphery of the display. Projectors have also been used to simulate digital paper to allow content to be moved from a computer monitor onto physical objects that can be bent, folded, and moved around [81].

Instead of using projectors, it is very common to just include additional display monitors. In the work by Rädle, multiple mobile devices are able to be used in conjunction to create differently sized ad hoc display spaces with the ability to perform interactions between devices [138]. In a more static manner, Chen et al. explored dual-display e-book readers and how they can be used to interact with electronic documents [31]. The size of an e-book reader's display space is able to change based on folding and unfolding the device. Similarly, other researchers have explored the design of dynamically resizable displays with continuous resizing capabilities [121]. In the *Xpaaand* project, Khalilbeigi et al. explored the design of a rollable mobile display device and how physically resizing it can be used as input into the system [110]. Likewise, Steimle and Olberding investigated the use of rollout displays in collaborative and social contexts [154].

Chapter 3

Theory of Interaction Spaces

Over the years, companies and researchers have developed many types of display devices with different form factors. A subset of these include smartphones, smartwatches, tablets, laptops, desktop computers, wall displays and interactive tables. To expand the available input space for each of these devices, direct touch on surface locations other than the screen can be employed, as well as the mid-air space around each device. Based on the typical usage and form factor of the devices, theses areas have different advantages and disadvantages that need to be considered when harnessing them as interaction spaces. Therefore, to inform the design of our interaction techniques as well as other designers, we have developed a descriptive framework describing the space of gesture interaction on and around each of the aforementioned devices. The around-device interaction space is further divided into the following volumes: behind, in front, to the left and right, above, and below (see Figure 3.1).

In the following sections, we will discuss the advantages and disadvantages of these volumes in relation to each device, as well as the digital content and spatial gestures that are appropriate for each circumstance based on a number of factors. These factors include the visibility of gestures and the display, the potential of false positive input detection, proximity to the user and the amount of physical effort required to perform gestures.

It is important to note that our framework has the following assumptions:

- We only include information on the following standard computing devices:
  - mobile devices such as smartphones, smartwatches, and tablets
  - laptops
  - desktop computers (Desktop)
  - wall displays

  - interactive tables (Table)

  Therefore, we do not include information on head mounted displays or immaterial/fog displays (e.g., [119, 127]).

- The user is male, has a stature equal to 176 cm, acromial (shoulder) height equal to 144 cm and arm length equal to 79 cm based on anthropometric data of an average person [56]. These measurements along with the size of the devices are used to determine the usability of the different interaction spaces.
- Interactive tables are rectangular, have a size of 81 cm (32 inches) diagonally, are 13 cm thick, and are at mid-height.
- Wall displays are wider than five feet, taller than three feet and have their vertical centre at shoulder height. As the name implies, they are attached to a wall. Technology does exist to enable communication through different materials employed in the construction of walls, but our framework does not consider this.
- Mobile devices are smaller than 28 cm (11 inches) diagonally.
- Desktop computers have only one thin monitor that measures 61 cm (24 inches) diagonally.
- Laptops have a display size of 43 cm (17 inches) diagonally.
- All devices support touch interaction.
- Digital content on all device types is oriented in only one direction.
- Interacting with content within each interaction space relies on direct spatial or touch interaction. Therefore, we do not include information relating to non-hand-based gestures (e.g., eyelid [99] or foot-based input [98]).

3.1 Factors of Analysis

As mentioned, our framework analyzes each interaction space based on a number of factors including the visibility of gestures and the display, the potential of false positive input detection, proximity to the user and the amount of physical effort required to perform gestures. The importance of each factor is discussed in the section below.

3.1.1 Gesture Visibility

When collaborating with other people, it is beneficial if each person's interactions are able to be viewed by all group members. Gesture visibility increases group awareness of others actions and supports employing gesturing as a means of communication, which can increase the overall effectiveness of the group (e.g., [10, 170]). With certain interactions, whether in a group or not, it is important for a user to be able to view their own gestures within an interaction space. This is due to direct visual feedback having a large influence on gesture difficulty and user success. When using the standard mouse, only a few simple interactions are supported (i.e., horizontal and vertical movement, changing the state of buttons and scroll wheel). Since the mouse is an indirect pointing device, these simple interactions are able to be performed effectively without users directly looking at the mouse. This is also true for simple interactions that make use of indirect touch or spatial interaction. Directly looking at the body part controlling the interaction is not required since users can update their interactions by using the display to determine the state of digital information (e.g., cursor position, value of slider widget).

Visual feedback of the body part being used for input becomes important when direct-touch input or complex gestures are employed. With direct-touch input, the interface is only updated (e.g., position of cursor, state of a widget) when a user touches the screen or is hovering close to it. Therefore, when interacting with a specific on-screen object, visual feedback of the finger is needed to continually update its position with respect to the target and make sure that the correct on-screen position is touched. As the amount of visual feedback is reduced, the difficulty of selecting the correct on-screen position increases. This is due to the user having access to less information when determining if and how the interaction needs to be updated. Without any visual feedback, performing the interaction is still possible due to employing proprioception information, but the interaction is much more difficulty and inaccurate.

As mentioned, visual feedback is also important for complex gestures, and becomes more important as the level of complexity increases. Complex gestures typically consist of a continuous interaction with many states, and continuous feedback is often required to correctly perform them. This is especially true when their performance must be updated based on their distance to a physical object (e.g., computer monitor) or the state of dynamic digital information. The aforementioned difficulty is in part caused by interfaces not providing enough information regarding the state of the gesture. Therefore, a user needs to rely on direct visual feedback of the body part(s) performing the gesture to determine if and how to correctly update the interaction. Similar to direct-touch, if the view of a complex gesture is compromised, it is more difficult to determine the state of the gesture and how to correctly continue it to completion. If the gesture was not performed correctly, less visual feedback also makes it harder to determine which aspect(s) of the gesture was incorrectly performed.

3.1.2 Display Visibility

It is important for interaction spaces to support the full visibility of the display. The visual sensory pathway is the standard channel that computing devices employ to communicate digital information to humans. When other pathways are supported as well, visual communication still tends to dominate with the other channels only being used to send a small fraction of the total amount of information that is sent to the user. The popularity and reliance on visual communication is due to humans possessing an advanced visual system. Out of all of the sensory systems, the human brain is able to process the most amount of information at once when using the visual system. This processing ability allows computers to employ a large visual communication bandwidth without hindering the user. When the visibility of the display is compromised, the full size of this bandwidth cannot be taken advantage of. Since most, if not all, digital information is visually communicated to the user, a reduction to this bandwidth has a large effect on the total amount of digital information that a user has access to. Even if successfully communicated, this information might be difficult or impossible to interpret due to occlusion typically causing the fragmentation of information. Therefore, as the amount of occlusion increases, the level of user awareness concerning the interface's current state decreases, with total occlusion of display resulting in a minimal or nonexistent level of awareness. With less information concerning the state of the system, it will be more difficult to successfully interact with the interface, as well as analyze its contents. Occlusion can even make successful interaction highly unlikely, if not impossible, for example, when the related interface element and pointer are occluded. Most computers are unaware when occlusion is hindering their ability to communicate with the user, but automatic detection is possible. Although, only a certain amount of occlusion can be mitigated by automatically changing the interface upon its detection (e.g., [160]). Therefore, it is important for the design of any system that mainly employs photonic technology as a feedback mechanism to reduce the possibility of occlusion and its negative effects on the user.

3.1.3 Detection of False Positive Input

The usability of an interaction space is affected by the occurrence of false positive input within that space. Unintentional input negatively affects the user and the system; thus the usability of an interaction space increases as the probability of detecting unintentional input decreases. Unintentional input is seen as a negative phenomenon since this type of input has the potential to change the interface in an unwanted manner. Its effect on the system can range from an annoyance, for example when the cursor is sometimes erroneously repositioned, to having a large negative effect. This is seen when a continuous amount of unintentional input affects the system to such a degree that interacting with the interface to perform a task becomes almost impossible. When interacting with a system is feasible, unintentional input can also increase the amount of effort and time required to perform a task. This happens when the state of the system (e.g., widgets, text within a document) is unintentional changed and the user must take this effect into account. This potentially increases the total amount of interactions that are required to complete the task. For example, the user must undo the effects of the unintentional input before continuing the task. This is also made worse when reverting an interface to a previous state is not possible or is very difficult. Therefore, as the occurrence of false positive input increases, one's ability to effectively interact with the system is increasingly compromised, which results in an overall decrease in user performance. The amount of unintentional input also affects the perception of an interface with users typically becoming more annoyed, frustrated and dissatisfied with the system as the occurrence of false positive input increases. Before designing a system that makes use of multiple interaction spaces, their effect on unintentional input must be considered. Although highly dependent on the design of supported interactions, the potential for a system to suffer from false positive input typically increases as the number of supported interaction spaces increases. This is due to the total interaction volume becoming larger which increases the chance for a physical object or a section of a body part to be erroneously detected by the system as intentional user input.

3.1.4 Proximity & Physical Effort

The proximity of an interaction space with respect to the user directly influences the amount of physical effort required to perform gestures within the space. Interaction spaces are analyzed based on these factors since they affect the usability of an interaction space. The proximity of an interaction space is important since it affects one's ability to view objects within it and effectively make use of the space's full interaction volume. Since the direct path from a user to an interaction space can be blocked by a physical object (e.g., display monitor), proximity is based on the distance that the user must travel (e.g., the path around the display monitor) in order to interact within the space. When a space is closer to the user, less effort is required to reposition their body to enable the placement of their limb at any 3D position within this space. Even less effort is required when the entire interaction space is within arm's reach, since the user

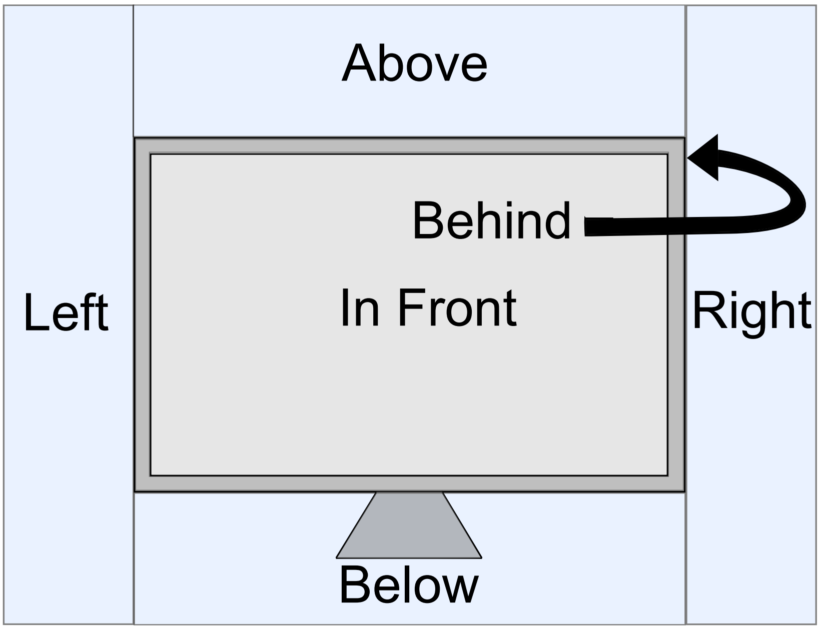


Fig. 3.1 Interaction spaces for a desktop computer display.

would not have to reposition their body at all. Proximity of an interaction space also affects one's ability to view and analyze objects within it. As the distance between an interaction space and a user gets smaller, the difficulty of performing gestures decreases since the user has a greater ability to continually view the state of gestures as they are being performed. Therefore, the amount of physical effort associated with performing any type of gesture within an interaction space typically decreases as the proximity of the space with respect to the user decreases. If an interaction space contains a physical object, the size of the object can greatly affect the amount of effort required to interact within this space. Larger objects increase the amount of occlusion and the distance between the user and any 3D position within the interaction space that is situated behind the object. This increases the difficulty and amount of physical effort required to interact and perform gestures within the space behind the object and is made worse when the object cannot be physically repositioned.

3.2 Analysis of the Interaction Spaces

3.2.1 Behind the Device

Behind the device interaction is defined as interacting within the space behind the display. For digital tables, behind the device interaction involves the 3D space underneath the table. This type of interaction is possible with all device types except for wall displays due to them being affixed to a wall. An advantage of behind the device interaction is that it mitigates occlusion of the display caused by a user's whole hand or individual finger. With regards to direct touch interfaces, it can also reduce the selection ambiguity (*fat finger problem*) resulting from the size and softness of the fingertip [4].

In relation to mobile devices, behind device interaction is relatively easy since a person can move the device closer to their body; thus bringing this space within their arm's range of motion. This is also due to the fact that the distance from the standard interaction space (on-screen interaction) to behind the device is small. Direct touch interaction is feasible [165] as well, except with smartwatches since the wrist blocks the back of the device. An advantage to direct touch input on the back of the device is that people tend to hold their mobile device in the palm of their hands. Therefore, this behaviour enables one to quickly perform direct touch input on the back of the device. Although, it also increases the risk of detecting false positive interaction.

With laptops, desktops and tables, back of the device interaction becomes more difficult. Direct touch and spatial interaction is possible, but increases in difficulty as the form factor of the device increases in size. It is easiest with laptops since one can easily bring the device closer to the body. Since tables and desktops tend to be fixed in place and are large, it makes it difficult to perform gestures behind the device as well as directly touch every part of it. Also, the distance from their typical interaction space (mouse and keyboard for desktops, and direct touch for tables) can be large. Another disadvantage of the desktop is if it is close to a wall, interacting behind the device might be impossible. These aforementioned disadvantages results in the behind the device interaction space being the most difficult interaction space to perform gestures within when using desktops, laptops, or digital tables.

With all device types, the device can semi or fully occlude the gestures performed behind it unless one has a transparent display [120]. Therefore, one does not know the precise location and orientation of the hands and fingers and must rely on proprioception. This can be mitigated if the interface provides visual feedback of hand and finger locations. Mobile devices suffer the least in relation to this problem since the user can move the device around to allow the interactions to be properly viewed. With all devices, if multiple interaction spaces are enabled, reaching behind the device will cause a user to transition through multiple spaces; increasing the risk of interference between these spaces. Although, since users typically do not place their limbs in the

3D space behind a device, the probability of detecting unintentional input in relation to spatial gestures is low.

Due to the difficulty of interacting behind the display among other disadvantages, only content that is seldom used should be contained in this area, unless a mobile device is employed. The space behind a mobile device can support content that needs to be easily accessed. For all device types, only coarse gestures should be used to interact within this space unless the device has a transparent display, or the interface provides feedback of finger and hand locations.

3.2.2 In Front of the Device

In front of the device interaction is defined as interacting within the space in front of the display, and is possible with all device types. With horizontal devices, such as digital tables, in front of the device interaction involves the space above the device. It is the standard space for spatial and touch interaction in general, as well as for direct manipulation techniques, due to displays typically residing on the front of devices. Users' previous experience with this standard enables in front of the device interaction to potentially have the lowest learning curve our of all of the interaction spaces. Therefore, non-expert users have an easier time becoming accustomed to interfaces that make use of this space. The display typically residing on the front of devices also affects how users position their limbs within this space. When holding a mobile device in their hands while it is in use, people tend to not occlude the display with their grip. Also, for other devices, people typically do not rest their bodies or limbs on the display. This behaviour is an attempt by users to reduce or prevent occlusion of visual content and the detection unintentional input. The result of this is the in front of the device interaction space having the lowest associated probability of detecting false positive touch input out of all of the other interaction spaces.

The in front of the device space is the closest space to the user and peripheral input devices are usually located within it. Since interactions can occur in front of the display, the probability of user interactions occluding the display is highest within this space. Although, the opposite is true for gesture visibility. With all device types, limbs are easy to position within this space and are not occluded by the device. Therefore, one is able to fully view their gestures as they perform them. Also, one does not need to move through any other interaction space to get to the front of the device. Systems that make use of spatial and touch interaction should consider employing this space since moving from direct touch to spatial interaction flows well and is immediate. Although due to touch interaction being the standard way of interacting with mobile devices and tables, this space has a high risk of detecting false positive spatial input since the system might mistakenly detect a spatial gesture when the user was solely attempting to touch the display. This problem also occurs with desktops and laptops when intentionally performing touch, keyboard, touchpad, or mouse interaction due to them usually being located within this space. A benefit when using desktops and laptops is that people tend to rest their limbs in the space in front of the display. Therefore moving a limb from rest to interact within this space is quick, fluid and easy, but as before this increases the risk of false positive spatial detection.

Due to the high visibility of gestures and the close proximity to the user, the in front of the device interaction space supports precise and coarse gestures types, and digital information that requires immediate attention of the user or is frequently accessed.

3.2.3 Left and Right of the Device

As the name implies, left and right of the device interaction is defined as interacting within the space on the left and right side of the display. This is possible on all device types, but is more difficult for a couple of them. Due to the large size of tables and wall displays, the sides may be far away from the user; requiring one to move their entire body to interact within this space. With the other devices, this space is close to the user and easy to perform gestures within. For all devices, the left and right sides are at a comfortable interaction height and are close to the standard method of input. If one uses the arm that is closest to each side to interact within that space, the display does not suffer from occlusion. Whereas, crossing limbs (e.g., using the right hand to reach into the interaction volume on the left side) can cause occlusion and might be uncomfortable. This discomfort disappears if one is using a mobile device or laptop due to their small form factors. For all devices, gestures are not occluded by the device and are able to be viewed by the user.

One need not always transition through different interaction spaces (e.g., in front of the device space) to reach the left and right side of the device, but it is possible depending on the starting location of the hand. People tend not to rest or place their limbs within the mid-air area associated with this space; therefore reducing the risk of detecting false positives. Although, since people typically hold their mobile device with their hands touching the sides, this grip can be mistaken for direct touch input. On the other hand, it allows for quick touch input, and unintentional input caused by a user's grip can be rejected using a classifier [112]. Due to the high visibility of gestures and the display, as well as ease of use, the left and right side of the device can support precise and coarse gestures, and should contain content that needs to be easily accessible.

3.2.4 Above the Device

Above the device interaction is defined as interacting within the space directly above the display. When a device is horizontal, such as with a digital table, above the device interaction is defined as interacting within the space beyond the device's edge that is farthest from the user. It is possible with each device type, but can be difficult with tables and wall displays due to their large size. Also when interacting within this space on desktops and wall displays, the hand is held above the shoulder; thus causing fatigue and other negative symptoms known as the *gorilla-arm* effect [78]. False positive recognition of spatial and touch interactions is reduced in this area since people tend not to place their limbs in this space. This even applies for mobile devices since users tend to not grip their phone with their fingers touching the top area [82]. Although, interference between the different spaces is possible since one must move through different interaction spaces to reach above the device. For laptops and desktops, the distance between the keyboard and mouse/touchpad and the area above the device is large, but tolerable for quick interactions. For all devices, interactions above the display can also cause the display to become occluded due to one's arm. A benefit of the space is that it is close to the standard input modality of mobile devices as well as other touch-enabled devices; allowing for fluid movement between the two. Also, gestures performed in this area are visible to the user, as well as potential collaborators. Due to the difficulty of interacting above desktop, wall, and table-based interfaces, this area should only contain content that is seldom used and only support coarse gestures when employing these device types. With laptop and mobile devices, content that needs to be easily accessible can be stored in this space and should support precise and coarse gestures.

3.2.5 Below the Device

Below the device interaction is defined as interacting within the space below the display. When a device is horizontal, such as with a digital table, below the device interaction is defined as interacting within the space beyond the device's edge that is closest to the user. It is possible on each device type, except for laptops due to the keyboard. Interacting within this space does not cause occlusion of the display and gestures are visible to the user. With all device types, it is easy to move into this space due to it being close to the user. Another advantage is that this space is close to each of their standard interaction modalities. With desktop computers, this space is also close to the typically location of peripheral input devices. People tend to keep their hands at their sides when standing at rest; therefore moving into the space below a wall display from a resting position is quick and easy. Although, interacting below a wall display becomes more difficult and uncomfortable when crouching is performed. For all devices, depending on the starting point of the hand, one might need to cross different interaction spaces to reach below the device. With desktops, the below the device interaction space tends to be small height-wise due to the table that the display is resting on and has a low risk associated with unintentional input detection. Although, this risk in relation to spatial input increases when the keyboard and mouse are employed for input. When it comes to mobile devices, depending on how a user holds and positions the device in relation to their body, their grip and body can interfere with direct touch and spatial interaction, which increases the risk of false positive input detection. The same occurs in relation to tables, especially when attempting to interact with a distant object on-screen since the user might place their body against the table's edge. In fact, the below the device space has a very high risk of detecting unintentional input when employed in a digital table system. Due to this space having a high degree of gesture and display visibility and associated ease of use, this space can support precise and coarse gestures and content that can be easily accessed when employed with all device types with a couple exceptions. When used in conjunction with a wall display, this space should only support coarse gestures to minimize the potential of crouching. Also, this space might not be suitable to include at all in a digital table's system design since the user's body can easily trigger unintentional input.

Chapter 4

Design of Off-Screen Exploration Techniques

In the field of information visualization, there are seven basic tasks that visual analysts perform whilst using a visualization application. These are overview, zoom, filter, details-on-demand, relate, history and extract [150]. Depending on the type of data, each of these tasks has advantages and disadvantages to them. In the case of the relate task, which is based on viewing relationships amongst items, it might be tedious to perform such a task if the items being compared cannot occupy the same display space. For example, in a typical map-based application overlaid with population data, it would be tedious to compare different sections of a small town to another town whose location is at a distance from the former. The analyst would have to zoom-in to the first small town to view the data, and then zoom-out to find the second town and zoom-in on its location to view its associated data. The problems that occur in this situation are that the flow of analysis is disrupted by requiring the user to remove the original town from the display space to view the second location, and when comparing both sets of data, the analyst is required to rely on memory. Different visualization techniques can be employed to alleviate this problem (see Section 2.3), but they tend to not provide enough context in terms of the surrounding off-screen and on-screen space.

To overcome this, we employed our descriptive framework (see Chapter 3) to develop a set of spatial interaction techniques that allows one to explore the information space situated outside the display, while making it easier to compare off-screen content with on-screen content. This allows analysts to compare data in a fluid manner that fits better within their flow of analysis. It also enables them to take full advantage of their advanced visual system instead of relying on memory, which is prone to errors.

While the mid-air space around a display is three-dimensional, our designs consider the off-screen information space as being two-dimensional and defined by the plane of the display (see Figure 4.1). A 3D interface was not used since people are slower, less accurate, and require more mental effort to interact with a 3D mid-air interface when compared to a similar 2D one [35]. Employing a 2D information space also reduces other problems associated with spatial and 3D user interfaces such as people having difficulty understanding 3D space (e.g., [79, 156]).

Our techniques include *Paper Distortion*, *Dynamic Distortion*, *Dynamic Peephole Inset*, *Spatial Panning*, and *Point2Pan*. Except for *Point2Pan*, all of these techniques make use of direct spatial interactions (e.g., [46]) to communicate with the system which part of the information space one is interested in viewing. What is different with our techniques is that we geometrically transform part of the information space to bring off-screen content onto the display, but the information space's interaction space remains the same. Therefore, placing one's hand beside the display will allow one to see off-screen content associated with that location, and then one can perform a direct spatial gesture (e.g., tap) at that same location in physical space to interact with this content. Comparison of on-screen and off-screen content is facilitated by bringing off-screen content on-screen while retaining on-screen content, as seen in the *Paper Distortion*, *Dynamic Distortion* and *Dynamic Peephole Inset* techniques. Also, all of the applied geometric transformations can be inversed by just removing one's hand from the spatial interaction space. This facilitates comparison as well as exploration since the user can transform the information space to view off-screen content then quickly invert this transformation to view content that was originally on-screen. Even though the following explanations of our techniques use desktop computers, our techniques can be applied to the myriad of other device types as well.

4.1 Distortion Techniques

We have designed two types of distortion interaction techniques where content on-screen is scaled down to allow off-screen content to fit on-screen. This allows one to view and compare off-screen and on-screen content at the same time. It is important to note that due to the on-screen content becoming distorted, performing comparisons between content does become more difficult. To help mitigate this, we have also developed a distortion technique that takes into account the energy of the information space to minimize distortion of important information. In the following subsections, the

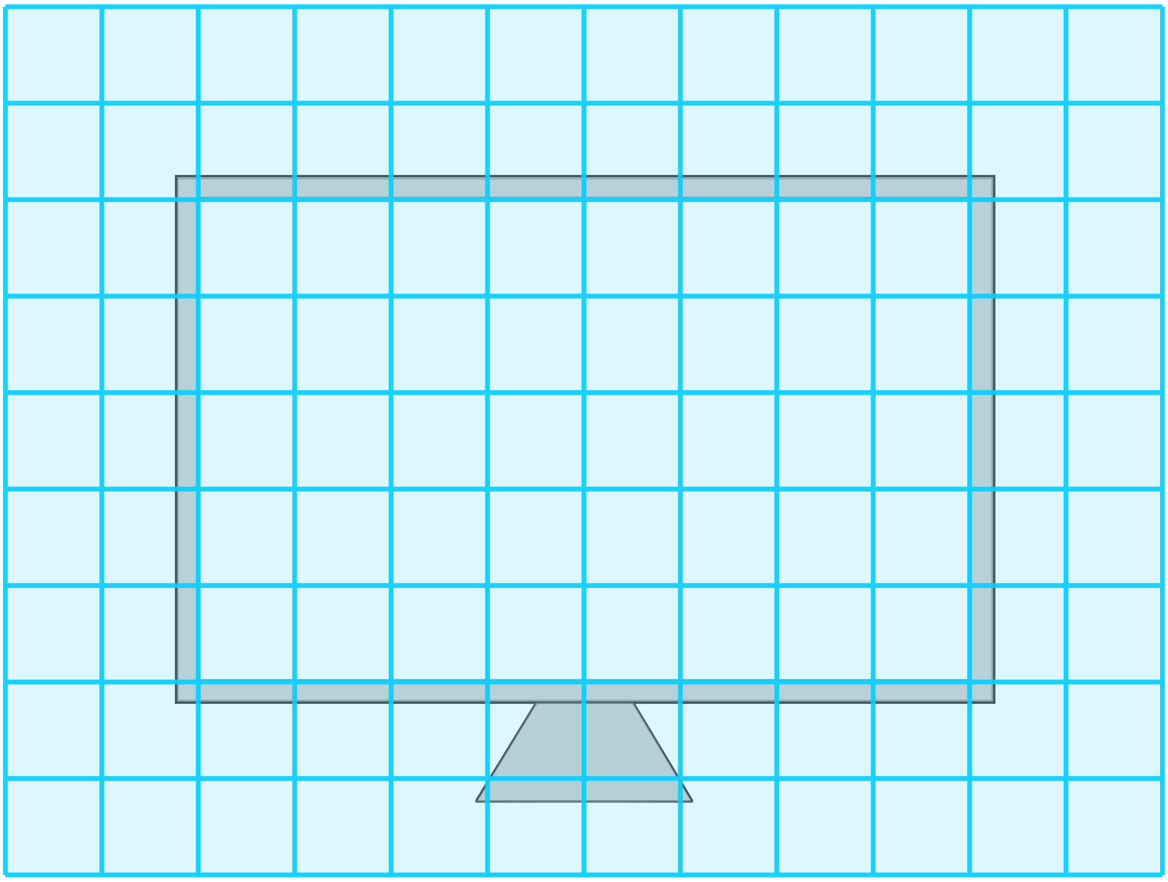


Fig. 4.1 Undistorted information space.

information space is distorted using the same scaling amount for each distorted section, but other techniques can be used as well, such as fisheye.

4.1.1 Paper Distortion

The *Paper Distortion* technique employs a paper pushing metaphor to display off-screen content. If we imagine the 2D information space as a sheet of paper that is larger than the display, the user can push the paper from the side towards the display to bring off-screen content onto the display. This causes the section of paper that is over the screen to crumple (distort); therefore creating enough room for the off-screen content by only scaling down the on-screen content (see Figure 4.2). Instead of distorting all of the content on-screen, the user can touch a location on the display whilst performing the pushing gesture to define a starting point of the distortion. The end point will automatically be the closest on-screen location to the performed push gesture. For example, if one pushes horizontally from the right side and touches the middle of the screen, then only on-screen content that is on the right side of the middle of the screen will become distorted. Similarly, the user has the option to define the end point of the distortion region as well. By touching two locations with one hand and performing the push gesture, only content with those two locations will become distorted. This technique can be performed to push off-screen content from any side or corner onto the screen, and can support multiple off-screen areas (e.g., left and right side) being pushed onto the screen at the same time. When the user removes their hand from the interaction space, this technique can optionally keep the off-screen content on-screen or automatically reset the information space. If off-screen content is kept on-screen, the user can perform the opposite spatial gesture (push the “paper” out from the side of the display) to move this content back to its original off-screen location.

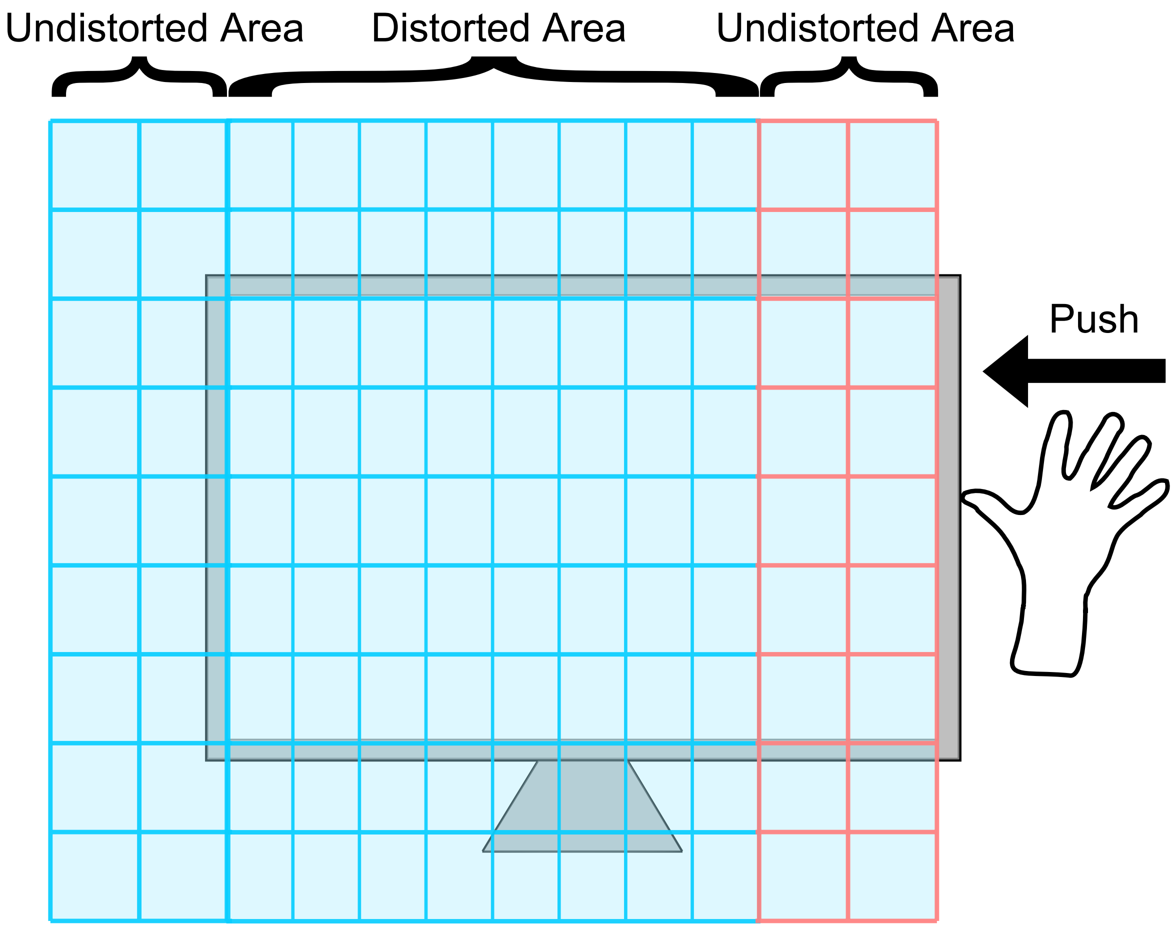


Fig. 4.2 Pushing the off-screen information space into the display causes the on-screen information space to become distorted; thus creating room for the corresponding off-screen content. The section in orange represents the undistorted content that was originally off-screen.

4.1.2 Dynamic Distortion

The *Dynamic Distortion* technique is similar to the *Paper Distortion* technique in regards to distorting the on-screen content to make room for off-screen content. Whereas in *Paper Distortion* a singular swipe gesture is used at the side of the screen to activate the distortion, with *Dynamic Distortion* the user is able to continuously change the amount of distortion by adjusting his hand location in relation to the side of the display. To invoke this technique, the user places their hand in an off-screen area, which causes the system to determine the section of the off-screen information space that is being touched by the hand. A direct 1:1 mapping between the physical space and the information space is employed to accomplish this. The system then updates the interface by distorting the on-screen content to bring the corresponding off-screen information onto the display. By moving one's hand further away from the side of the display, the amount of distortion increases since more of the off-screen information space needs to fit on-screen. To be able to view off-screen content past the corner of the display, the on-screen content is distorted horizontally and vertically as seen in Figure 4.3. This occurs only when a person's hand is above or below the screen, while also being beside it. Figure 4.4 shows the areas of the off-screen space that trigger the different distortions. Above and below the screen trigger vertical distortions, the left and right areas trigger horizontal distortions, and the corners trigger both horizontal and vertical distortions.

4.1.3 Content-Aware Distortion

We have also designed a *Content-Aware Distortion* technique to manipulate how we transform the information space. Similar to image reduction in *Seam Carving* [2], we calculate the energy or importance of each pixel, but instead of removing a seam of low energy pixels, we select these for distortion (see Figure 4.5). Distortions must take place on entire strips or seams of pixels to remove the possibility of shearing the high energy content (see Figure 4.6). To calculate the energy of a pixel or region of pixels, the function can use one or more aspects of the information space, such as the density or importance of the information at each location. Regions with a high amount of energy are only translated, whereas regions with a low amount of energy are scaled down to make room for off-screen content. For example, in a map-based interface, if oceans had low energy, then our *Content-Aware Distortion* technique would only distort the oceans. At the same time, due to continents having high energy, these would only be translated and not distorted (see Figure 4.7). This technique can be combined

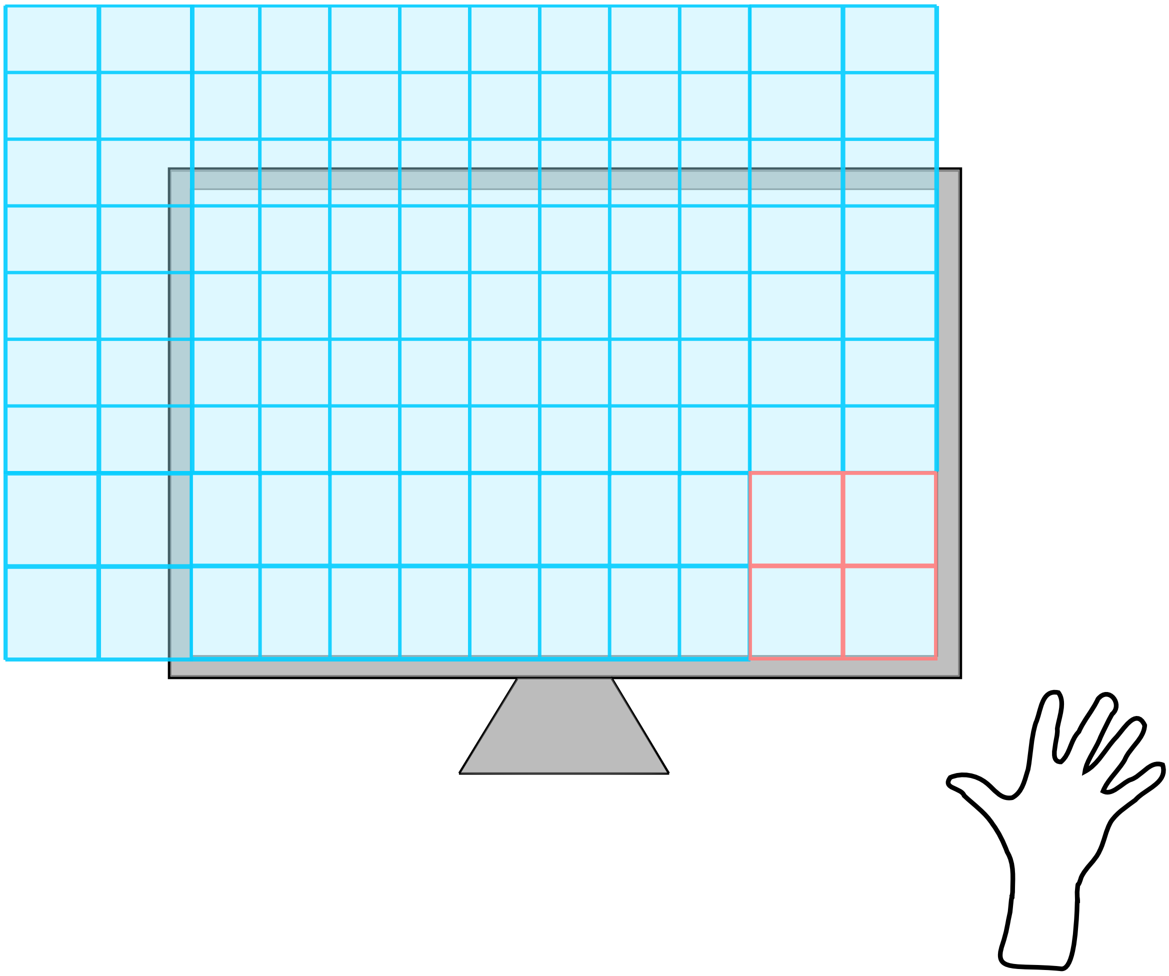


Fig. 4.3 To bring content past the corner of the display onto the screen, the *Dynamic Distortion* technique distorts the information space horizontally and vertically. The section in orange represents the undistorted content in the information space underneath the hand.

with both the *Paper Distortion* and *Dynamic Distortion* techniques to provide the user with minimal loss of important information.

4.2 Dynamic Peephole Inset

The *Dynamic Peephole Inset* takes inspiration from spatially-aware devices employed for peephole navigation (e.g., [129, 132, 139]). Instead of using the position and display of a mobile device, we utilize the position of the hand in the off-screen information space and display the corresponding content in an inset/viewport that is situated on-screen at the edge of the display (see Figure 4.8). This allows one to easily explore

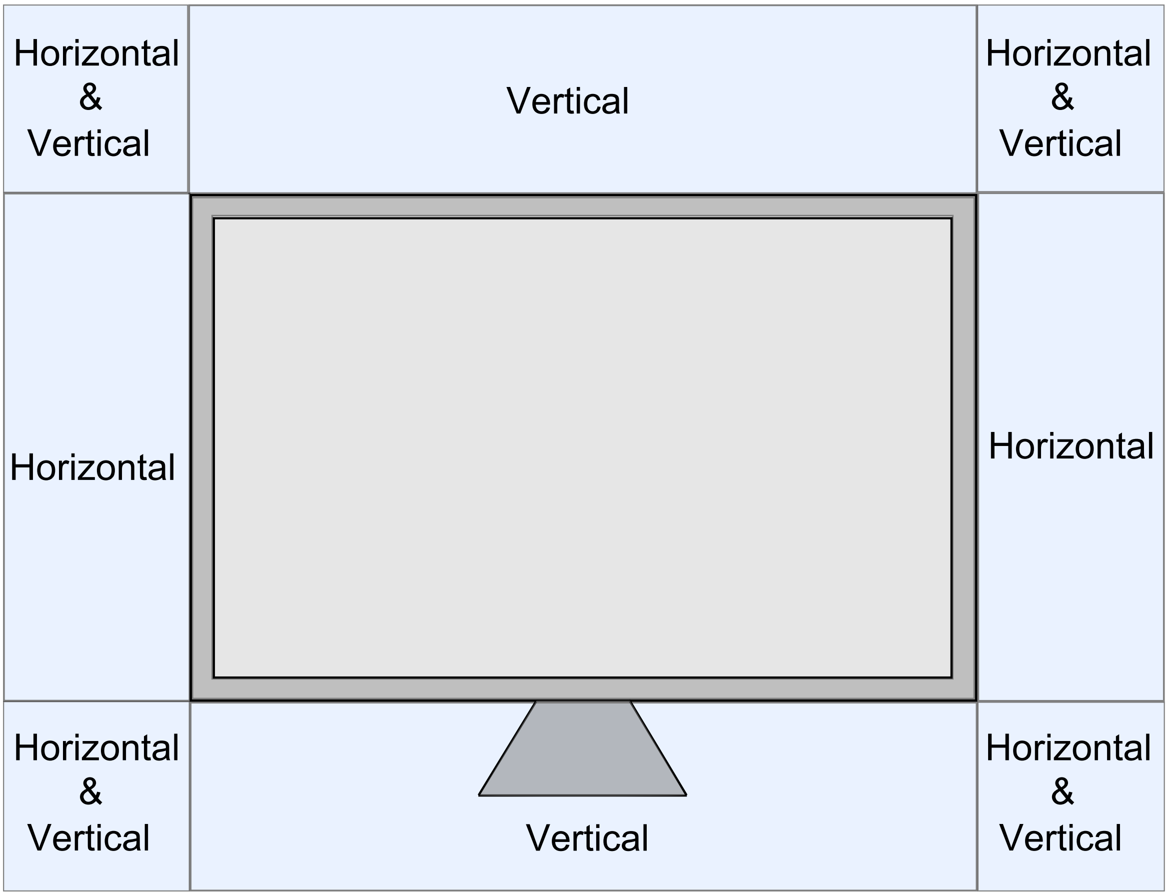


Fig. 4.4 The areas of the off-screen space that trigger the different distortions. Above and below the screen trigger vertical distortions, the left and right areas trigger horizontal distortions, and the corners trigger both horizontal and vertical distortions.

the off-screen information space by moving one's hand around the display. There are many options for the placement of the viewport on-screen. The viewport can be fixed at the closest corner to the hand, the centre of the closest edge, or continuously follow the location of the hand.

Dynamic Peephole Inset supports different mappings between the location of the hand and the corresponding off-screen content shown within the viewport. These mappings include direct, normalized, semi-normalized, normalized based on the side of the display, content-aware, and dynamic. In direct mapping, the content that is displayed within the viewport is the section of the off-screen information space that is directly underneath the hand, as seen in Figure 4.8. Therefore, direct has a 1:1 mapping. In normalized mapping, the off-screen information space is mapped using a predefined interaction space beside the display. This way, entire information spaces of any size can be explored using the available interaction space, simply by reaching

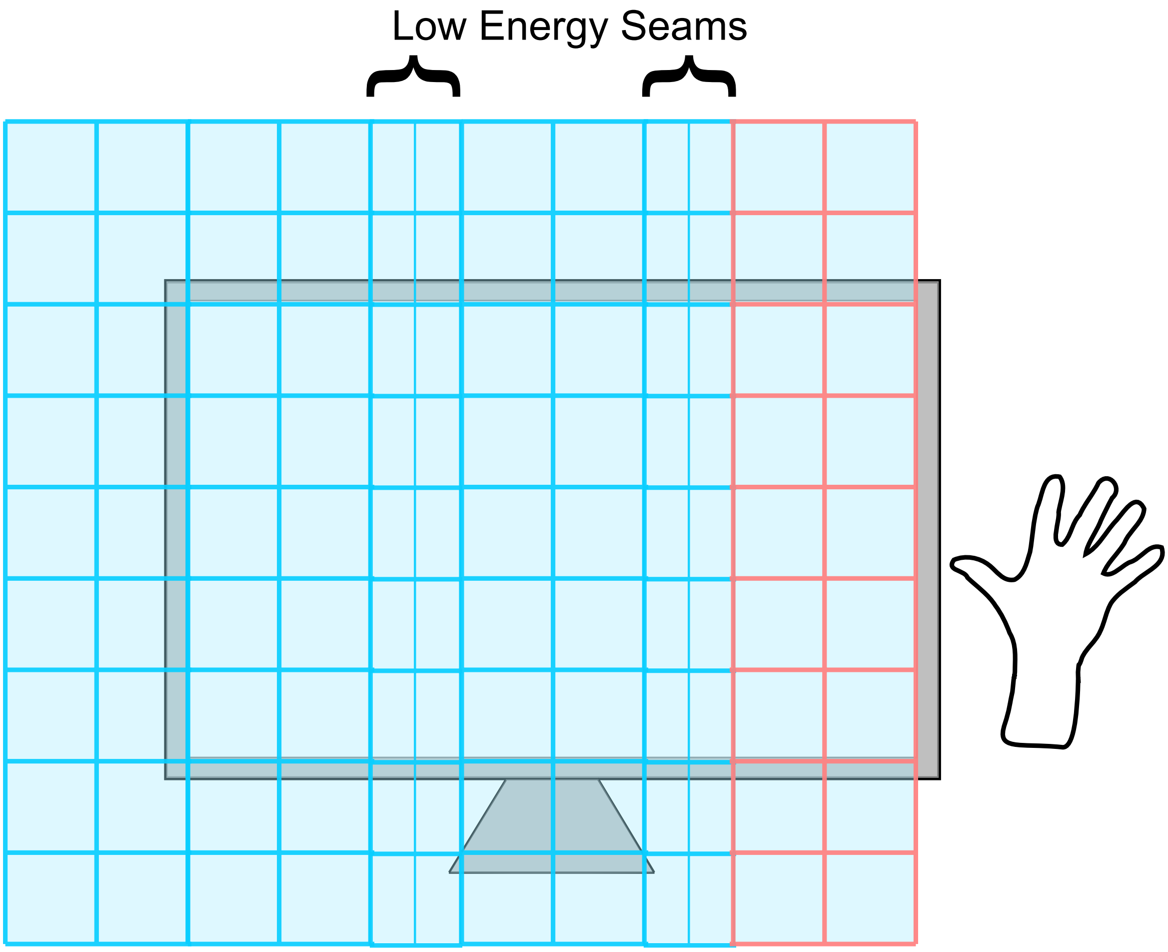


Fig. 4.5 *Content-Aware Distortion* minimize the loss of important information when distorting the on-screen information space to make room for off-screen content. The section in orange represents the undistorted content that was originally off-screen.

out, without clutching or panning the display. However, the input gain increases with larger information spaces, making precise navigation more difficult. The size of the interaction space is usually based on practical limitations such as the motion sensing hardware's field of view. Another option is to base it on anthropometric data (e.g., average length of arm) to increase usability and reduce negative effects such as the gorilla-arm (excessive fatigue) [78].

The semi-normalized mapping depends on which side of the display the hand is located in. If the hand is located on the left or right side, then the vertical location of the hand has a direct 1:1 mapping to the information space, but the horizontal location is normalized using a predefined interaction space. If the hand is located above or below the display, then the reverse is true. The normalized based on the side of the display mapping takes a similar approach where if the hand is on the left or right side

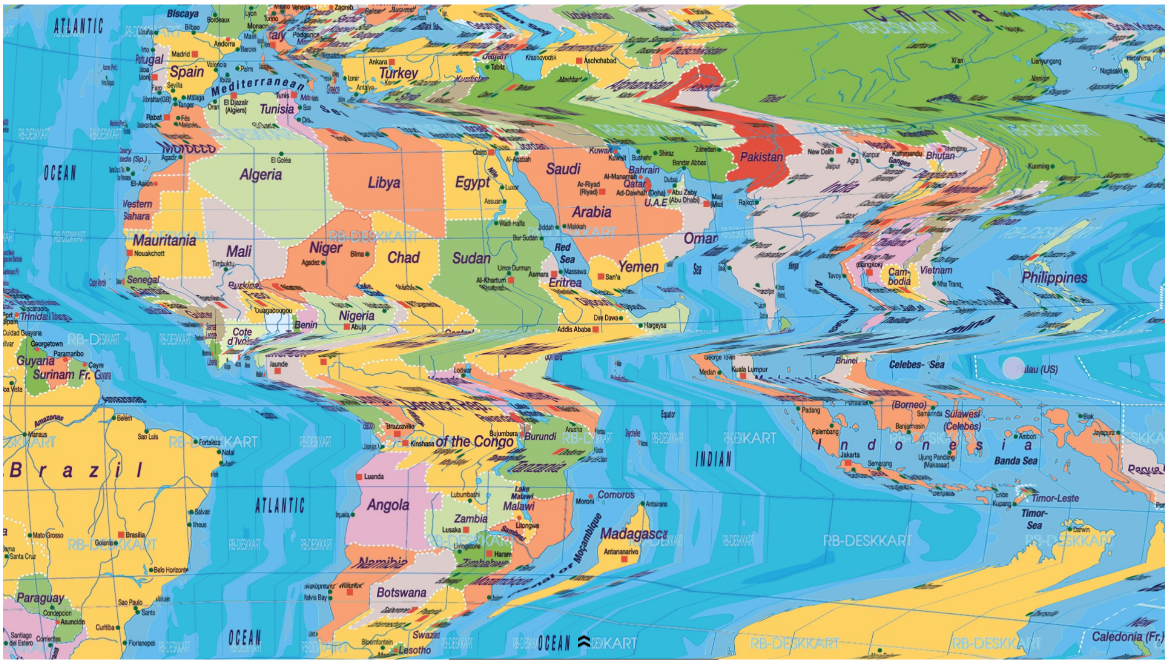


Fig. 4.6 As seen in the above figure, portions of the high energy continents become unintelligible due to shearing when only *subsections* of strips/seams are distorted. To remove the possibility of shearing important information, *Content-Aware Distortion* distorts *entire* strips/seams of pixels. Original map image source: www.welt-atlas.de.

Fig. 4.7 *Content-Aware Distortion* enables off-screen content to be brought on-screen while, at the same time, minimizing the loss of important information. As seen in the above figure, a large section of the Atlantic Ocean is horizontally scaled down (distorted) to less than a pixel wide to be able to bring portions of Russia and the Middle East on-screen. In this example, water is deemed to be low energy (low importance) and landmass is deemed to be high energy (high importance).

of the display and within its vertical boundaries, then there is a vertical 1:1 direct mapping with a horizontal normalized mapping. If the hand goes outside the vertical boundaries of the display, then the vertical mapping becomes normalized as well. If the hand is above or below the display and within its horizontal boundaries, then there is a horizontal 1:1 direct mapping and a vertical normalized mapping. When the hand goes outside the display's horizontal boundaries then the horizontal mapping becomes normalized.

The content-aware mapping is a little different from the aforementioned mapping techniques. It utilizes the same energy function approach as the *Content-Aware Distortion* technique, but instead of distorting the information, the viewport uses a friction metaphor for movement. Information areas with high energy have an associated high friction value and information areas with low energy have a low friction value. Therefore, when one moves their hand past a certain speed threshold, the viewport moves at a higher speed when moving through low energy areas and slows down when a high energy area is reached. The speed threshold is used to allow the user to intentionally investigate low energy areas without skipping over them. This content aware input gain is designed to balance precision as well as speed of access for large information spaces within the constrained interaction space of the sensor and reach limitations. Using the previous ocean example, this mapping would allow one to spend less time traversing the map in search of other continents (areas of interests). In dynamic mapping, the user is able to dynamically change how their hand's XY location is mapped to the information space. By moving one's hand farther away from themselves along the Z-axis (deeper into the plane of interaction), the input gain increases. The opposite motion decreases the input gain. This gives the user precise control over how the information space is explored with the option to change the mapping at any time.

4.3 Spatial Panning

The *Spatial Panning* technique translates (pans) the information space to show off-screen content with the use of two different mappings: direct and normalized. In *Direct Spatial Panning*, the off-screen location of interest is indicated by the location of the user's hand using a direct 1:1 mapping. By directly placing one's hand in the information space that resides off-screen, the system will translate the environment to show on-screen the information space that is located at the position of the user's hand. For example, on the right side of the screen, the vertical panning amount can be calculated based on the distance between the hand and the vertical centre of the screen. Similarly, the horizontal panning amount can be calculated based on the distance between the hand and right side of the screen. As with the normalized mapping in *Dynamic Peephole Inset*, *Normalized Spatial Panning* maps the entire associated

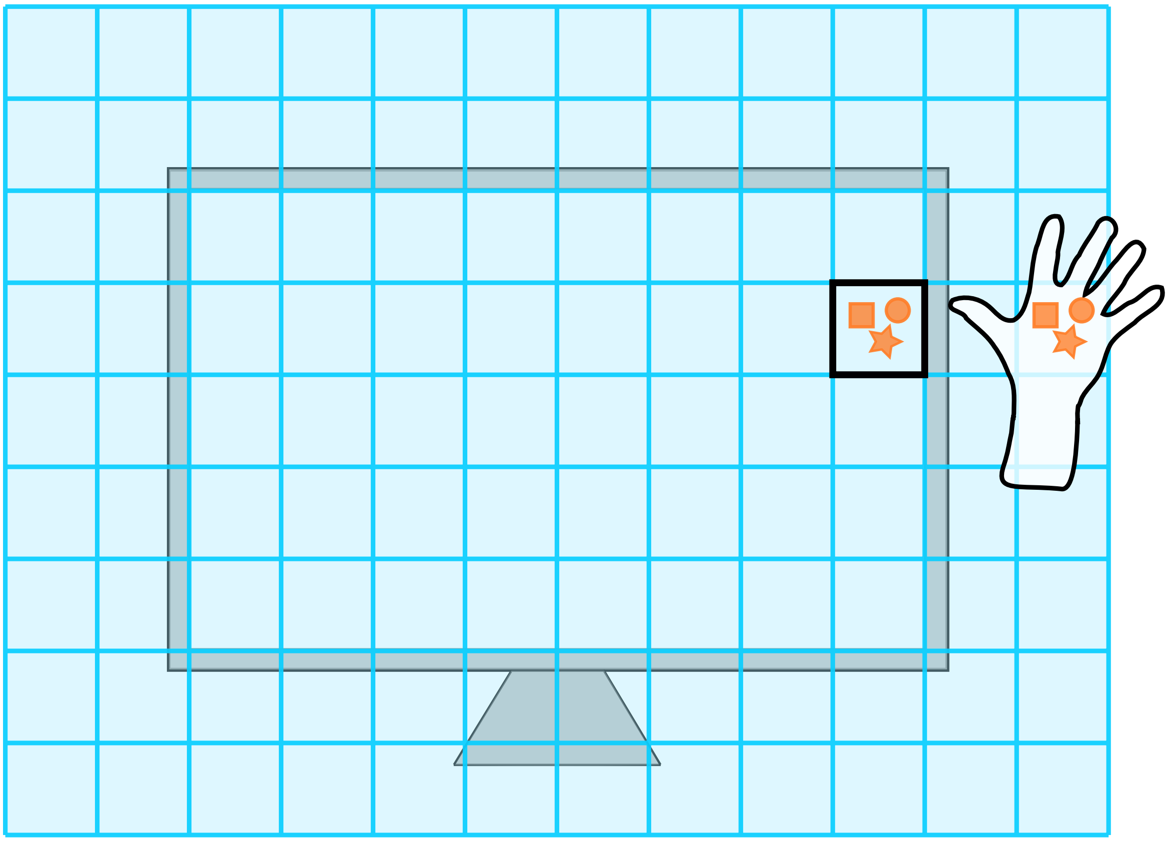


Fig. 4.8 The *Dynamic Peephole Inset* technique displays a viewport on-screen that contains off-screen content situated underneath the user's hand.

off-screen information space to a predefined interaction space beside the display to allow any sized information space to be comfortably explored.

4.4 Point2Pan

Point2Pan is a ray-casting technique similar to the ones designed for interacting with distant objects in 3D environments (e.g., [17]). When a user points to a section of the information space that lies off-screen, the system translates (pans) the information space to show this section on-screen. The user can then manipulate the content using touch or the mouse for example, or end the pointing gesture to translate the information space back to its original location. This allows one to quickly explore the surrounding off-screen area without much physical effort. This can be accomplished by finding the point of intersection between the plane that is defined by the display and the infinite line specified by the direction of the finger.

$$\text{Plane equation:} \quad (p - p_0) \cdot n = 0$$

$$\text{Line equation:} \quad p = dl + l_0$$

Substituting line equation into plane equation and solving for d gives you:

$$d = \frac{(p_0 - l_0) \cdot n}{l \cdot n}$$

Where d is a scalar, p_0 is a point on the plane, l_0 is a point on the line, n is a normal vector of the plane, p is a set of points, and l is a vector in the direction of the line. If $l \cdot n = 0$ then the line and plane are parallel. If $l \cdot n \neq 0$ then there is a single point of intersection, which is given by

$$Point\ of\ Intersection = dl + l_0$$

Chapter 5

Off-Screen Desktop: System for Interacting with Off-Screen Content

Whilst performing serious work, knowledge workers typically make use of desktop computers due to their inclusion of physical keyboards, large screen sizes, and comfortable seating arrangements. Although popular, desktop computing has not changed much in the past 20 years. WIMP-based (windows, icons, menus, pointer) interaction still predominates this computing landscape with the keyboard, mouse, and trackpad as its main input modalities. However, the popularity of employing touch-based laptops and desktop monitors to enhance traditional desktop experiences is growing. We also view off-screen interaction as being able to augment these desktop experiences by providing people with a larger interactable information space. Touch interaction has been employed for interacting with on-screen content, but does not allow one to directly manipulate content that is situated off-screen. To be able to unleash content from the boundaries of the display into the surrounding space and still be able to directly manipulate it requires a different input modality. We believe spatial interaction is the right technique to enable direct off-screen interaction for desktop computing. This had led us to the design of an off-screen interaction system called *Off-Screen Desktop*.

Our system is a multimodal 2D zoomable user interface (ZUI) that enables the manipulation of on-screen and off-screen objects. To accomplish this, we have implemented all of our off-screen exploration techniques with support for spatial, mouse and touch-based selection. These different input modalities were enabled with the use of consumer-grade motion sensing hardware, a touch-enabled monitor, a keyboard, and a mouse (see Figure 5.1). Consumer-grade motion sensing hardware was employed to allow us to understand how off-screen interaction would occur in the wild, along with its inherent challenges (e.g., possibility of poor recognition).

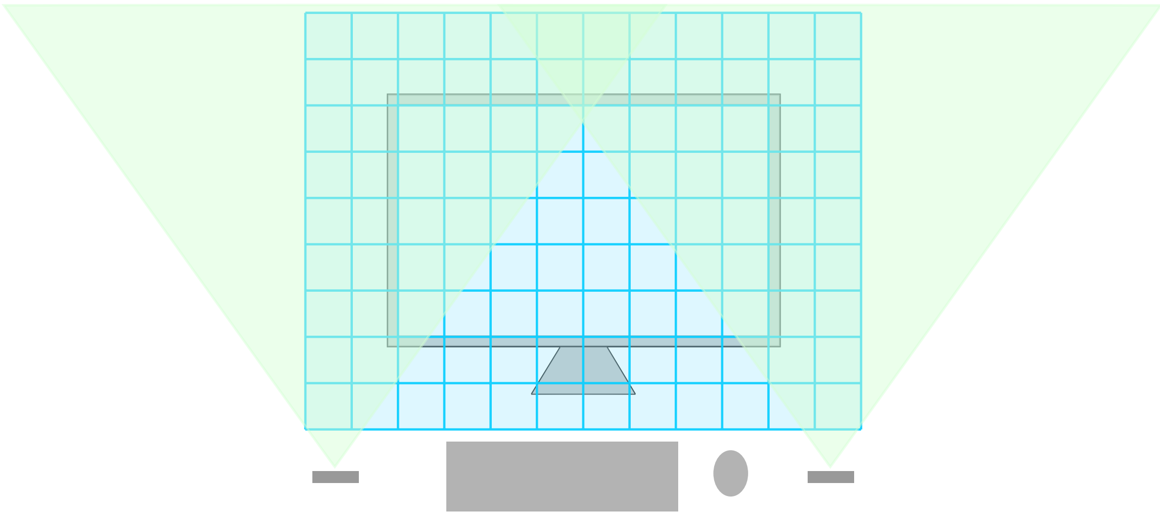


Fig. 5.1 Our system setup consists of a touch-enabled monitor with one motion sensing device at each of its sides, as well as a keyboard and mouse.

Informed from our descriptive framework (see Chapter 3), we only enabled the use of our off-screen exploration techniques on the left and right side of the display. Above, below, in front and behind the display were not supported due various reasons. For the Above the device space, this includes its high potential for causing excessive fatigue in users, and the distance between this space and the keyboard/mouse being large. Below the display was not supported since it has a high risk of unintentional input due to the keyboard/mouse being close by. Also, the vertical height of this space is typically small; therefore it only supports directly interacting with small information spaces. Our off-screen exploration techniques did not make use of the in front of the device interaction space since it does not support directly interacting with a 2D information space that is defined by the plane of the display, and it has a very high risk of unintentional input. Behind the device space does not support the aforementioned direct off-screen interaction as well, and performing gestures within this space is difficult. This is due to the display occluding the view of gestures and the user having to reach around the display to interact within this space. The left and right side were the best choices for supporting off-screen interaction since they are close to the keyboard and mouse, the risk of unintended interaction in this space is low, users are comfortable interacting in this space, and they support direct spatial interaction with a 2D information space that is defined by the plane of the display.

5.1 Visualization of Off-Screen Content

To provide the user with knowledge of off-screen content without requiring one to explore the off-screen information space with spatial, mouse, or touch-based interactions, we have integrated a set of visualization techniques into our system. These include Wedge [60], overview + detail, arrows, EdgeRadar [62], Halo [7], and City Lights [173] (see Figures 5.2-5.7). When the overview + detail visualization is displayed and the information space has been distorted, the visualization's viewfinder will change its size to update what sections of the information space are located on-screen. Since all of the aforementioned visualizations take up screen space, we have also included a spatial gesture that visualizes off-screen content on-demand. By closing one's hand into a fist, all objects that lie off-screen are visualized at the edge of the screen (see Figure 5.8). Performing the opposite action (opening hand) removes the visualization. This allows one to quickly view the state of all off-screen objects, as well as determine the sections of the information space that they lie within.

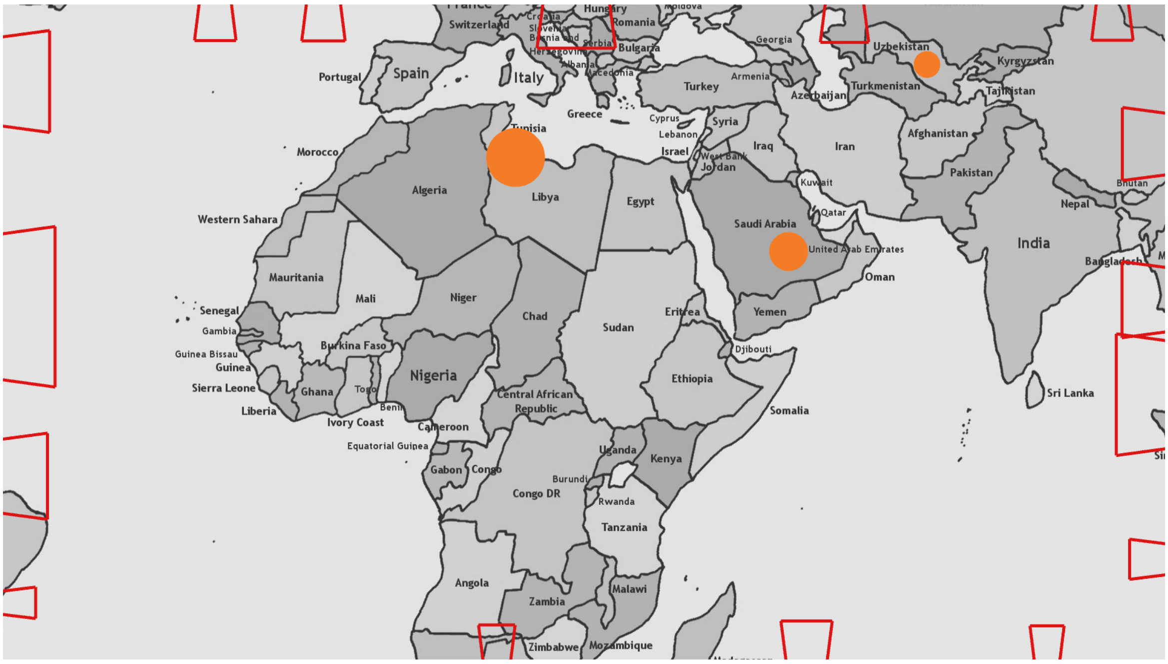


Fig. 5.2 The Wedge technique [60] is employed to visualize the locations of off-screen content. Original map image source: User Ionut Cojocaru on Wikimedia Commons, 2009.

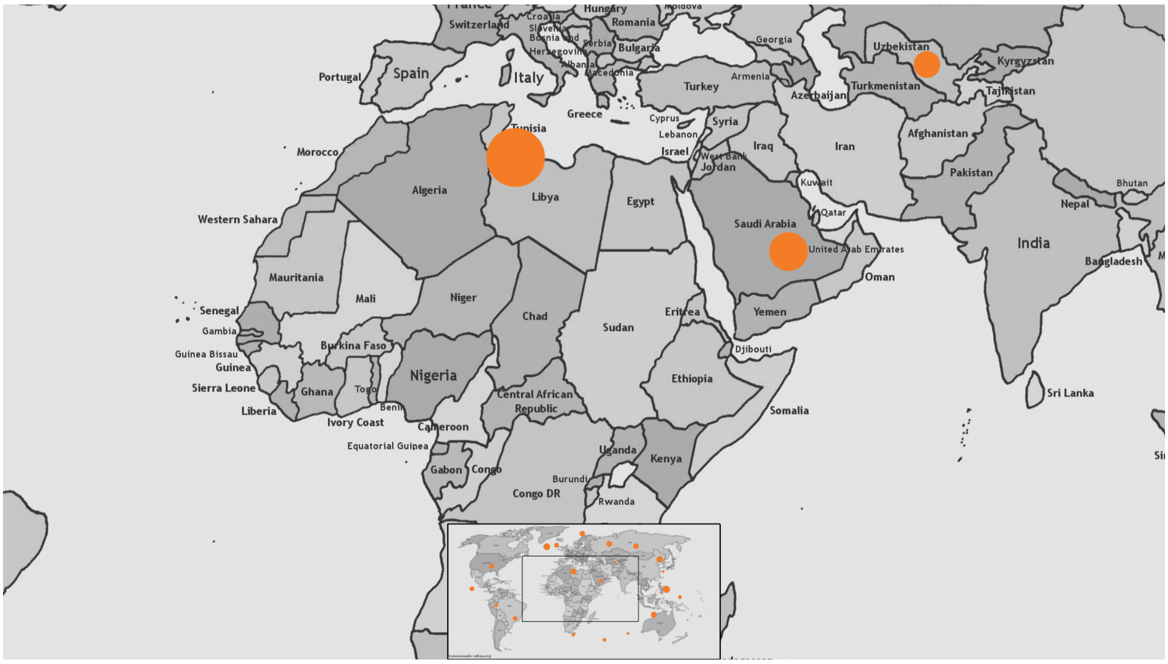


Fig. 5.3 Our system includes an overview + detail visualization. The inset image displays the entire information space with its viewfinder depicting the area that is shown on-screen. Users can translate the information space by moving the viewfinder within the inset or by clicking/touching a location within the inset.

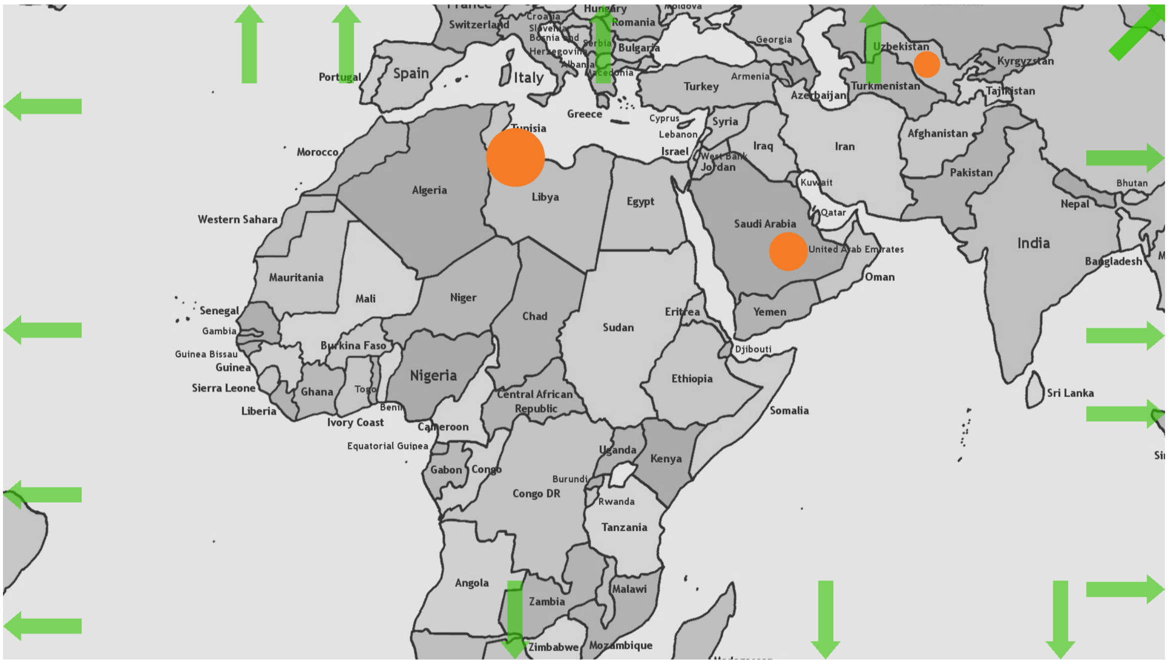


Fig. 5.4 Off-screen objects are represented by semi-transparent arrows at the edge of the screen.

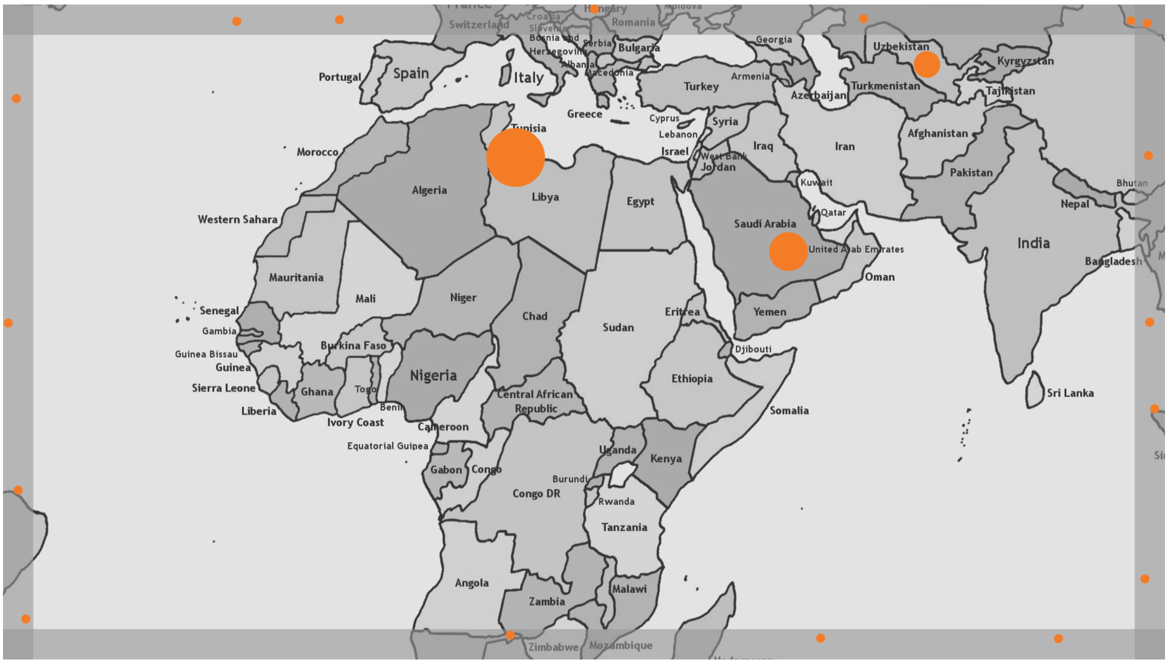


Fig. 5.5 EdgeRadar [62] displays miniaturized versions of the off-screen objects in bands located at the edge of the screen. Each band represents part of the off-screen information space.

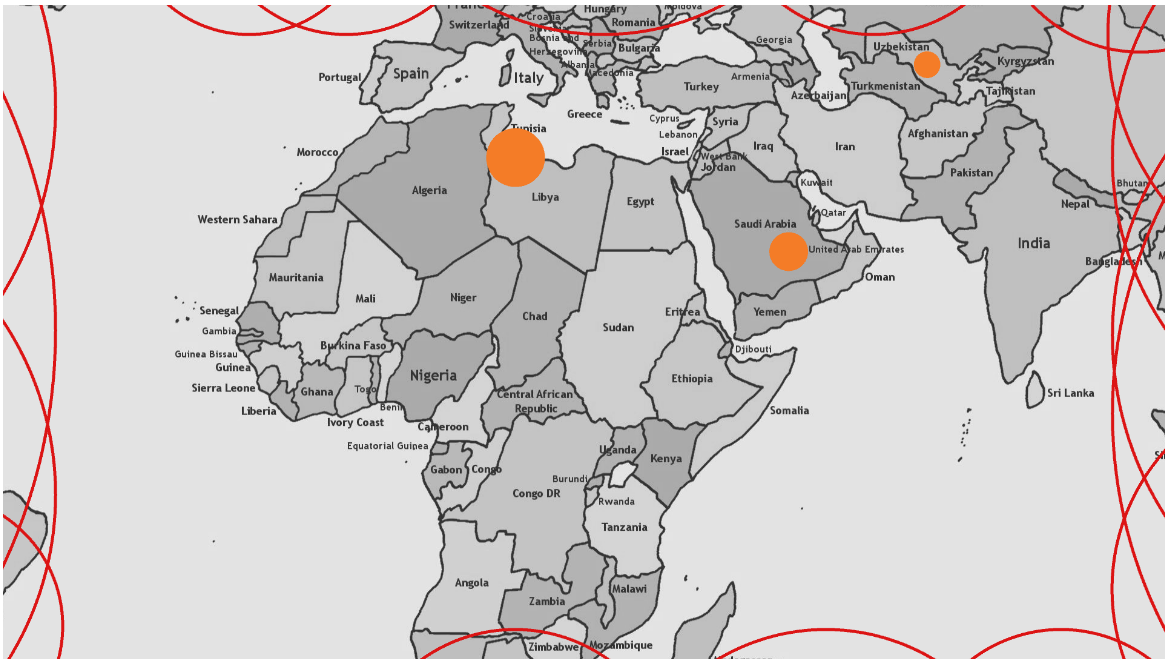


Fig. 5.6 For each off-screen object, Halo [7] displays a circle that is centred at the object's location.

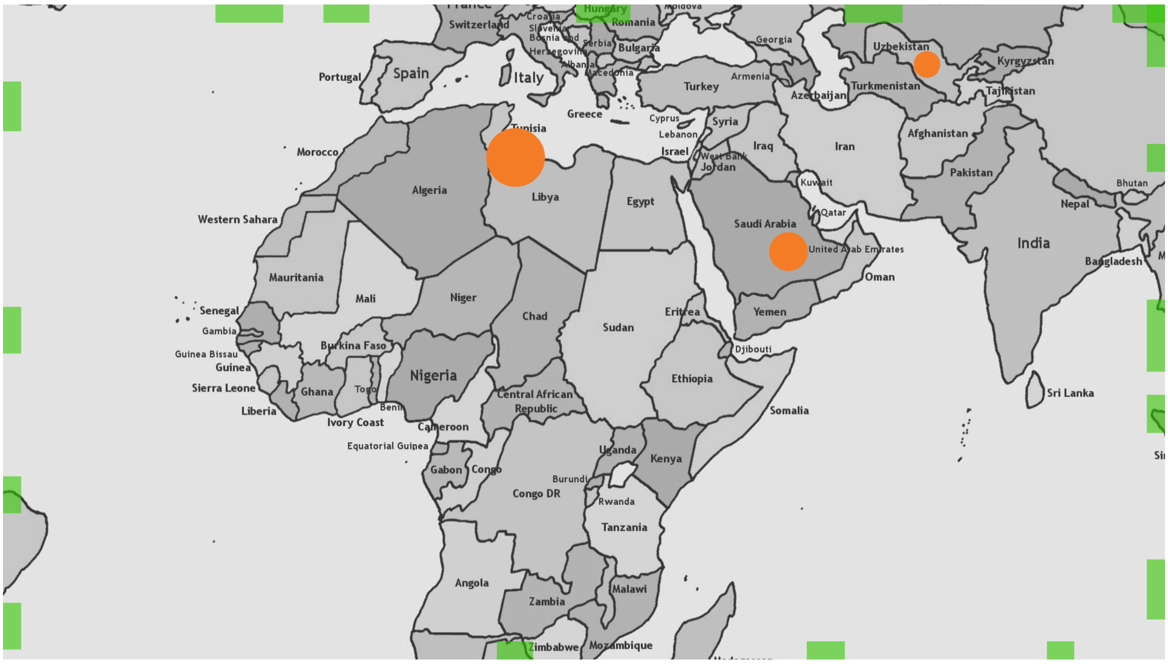


Fig. 5.7 The City Lights visualization [173] draws semi-transparent thick lines at the edge of the display to represent off-screen content. The length of the lines depend on the actual size of the objects.

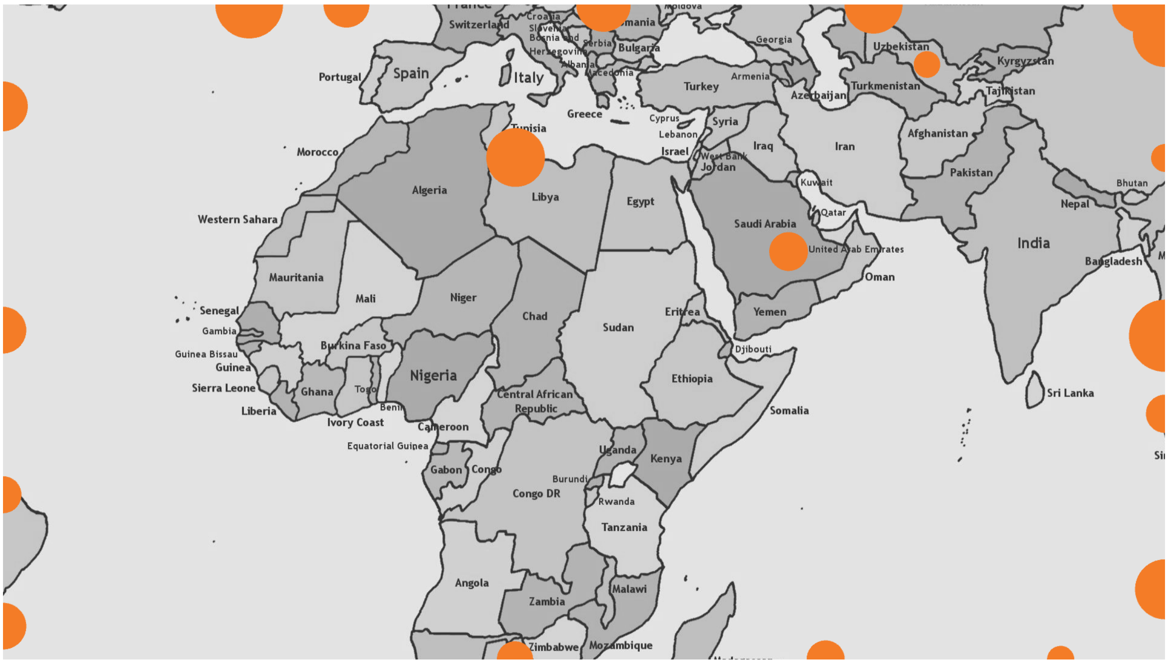


Fig. 5.8 By closing one's hand into a fist, all objects that lie off-screen are visualized at the edge of the screen. This on-demand technique saves screen space since the visualization only occurs when a fist is made.

5.2 Interacting with Off-Screen Content

To enable off-screen interaction, we integrated our off-screen exploration techniques with different selection techniques. Along with mouse and touch-based selection, we also support panning using these modalities. Since our off-screen exploration techniques temporally move content on-screen, we also support a spatial gesture that pans the information space to permanently bring content on-screen. This gesture can be performed by only extending the thumb and pinky and moving the hand. The information space will pan based on the distance moved and the velocity of the gesture. Moving at a faster speed will increase the gain of the interaction; thus allowing one to pan large distances with quick and short movements. The zoom level can also be changed to bring off-screen content on-screen. This can be performed by double clicking on an empty part of the information space with the mouse, using the mouse's scroll wheel, or by performing a spatial gesture. This gesture is performed by surpassing a velocity threshold along the Z-axis with a closed fist.

To select objects, users must position the cursor (pointer) over an object, which will highlight the target, and perform a selection technique. We support mouse clicking, touch, key press (e.g., spacebar), and two spatial gestures that are described in the section below. When selected, these objects can be moved around the information space by using the modality that was employed to select the object (i.e., move mouse, move finger on touchscreen, or move hand in mid-air). If an object is moved into a part of the information space that has been distorted, the object is distorted when released from the cursor. To deselect the object, a similar technique to the one employed for selection must be performed (e.g., releasing the mouse button, removing the finger from the touchscreen). Although our off-screen exploration techniques are only supported in the left and right around-device interaction volumes, we also support spatially selecting objects in front of the display to be able to move on-screen content off-screen with spatial interaction.

When attempting to view non-existent off-screen content that is past a boundary of a finite sized information space, our system displays a semi-transparent red band on the appropriate edge of that information space. This informs the user that they have reached the edge of the information space and that no more content lies past it. With the *Dynamic Peephole Inset*, this red band appears in its viewport.

5.2.1 Spatial Selection

All of our techniques support selecting off-screen objects using spatial gestures except for *Point2Pan* and *Paper Distortion*. *Point2Pan* does not support spatial selection since this off-screen exploration technique translates (pans) the information space based on the direction that the finger tip is pointing. Therefore, performing a gesture with the finger or hand would cause the information space to translate before a selection was made. Our system does not support spatial selection with *Paper Distortion*, although it is possible.

The spatial selection techniques that we support include mid-air tap with a finger and mid-air grab with a whole hand. Once selected, off-screen objects can be moved around the information space by moving one's hand. If the finger-based selection technique was employed to select an object, deselection occurs when the user performs another mid-air tap with their finger. With the spatial grab gesture, deselection occurs when the user opens their hand. To be able to select with these techniques, an on-screen cursor (pointer) is required to inform the user what part of the information space will respond to input. The design of our system's spatial cursor, as well as other aspects related to spatial selection, are described in the sections below.

On-Screen Spatial Pointer

When a user places their hand beside the display, off-screen content is brought on-screen and a purple circular cursor (pointer) representing the hand is displayed within the section of the information space that is mapped to the hand's physical location. The cursor's position in the information space is based on the horizontal and vertical position of the hand with respect to the display, the size of the information space and what geometric transformations have been performed on it, the screen size and its resolution, and which off-screen exploration technique and mapping is being used. For example with direct mapping, if one's hand was on the right side of the display, was vertically centred with the screen and there was a horizontal distance of 95 mm between the centre of one's palm and the rightmost screen pixel, the system can determine what part of the information space the hand would be covering if the screen was large enough to display it. The screen's right side would need to be more than 95 mm wider to view the content at the position of the hand without transforming the information space. If no geometric transformations (e.g., zooming/scaling, distorting) have been performed on the information space, and the information space employs pixels as its internal representation (e.g., an image), the system can determine that the interested off-screen content is around 360 pixels horizontally to the right of the rightmost screen pixel and 540 pixels vertically down from the topmost screen pixel based on the display's ~3.78 pixel per mm and 1920x1080 resolution (see Figure 5.9). Since our techniques cause this section of the off-screen information space to be brought on-screen, the cursor is able to be drawn on-screen at this location within the information space.

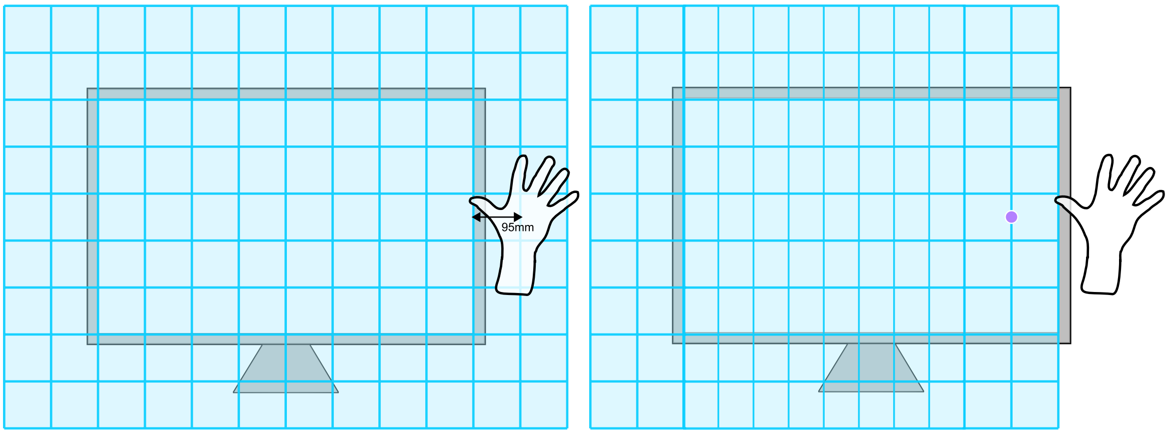


Fig. 5.9 The section of the information space that is directly under the hand in the above left image is around 360 pixels horizontally to the right of the rightmost screen pixel and 540 pixels vertically down from the topmost screen pixel based on the display's ~3.78 pixel per mm and 1920x1080 resolution. As shown in the above right image, the cursor is then drawn within that section of the information space.

To provide the user with more flexibility over the cursor's final position, we change its location based on the horizontal and vertical distance between the index finger and the centre of the user's palm. When the distance is zero, the cursor's position does not change. Moving the index finger higher than the palm and more left will cause the cursor to move up and move left respectively. With the *Dynamic Peephole Inset*, we constrain the cursor to always be within its viewport. When using this technique and the distance between the index finger and palm centre is zero, the cursor is horizontally and vertically centred within its viewport.

Improving Spatial Selection

Performing mid-air selections can be slow, imprecise, and difficult due to the technique being performed in 3D space, especially when no tactile feedback is present. The user can wear a special glove to provide feedback (e.g., [113]), but walk-up-and-use is always preferred. Research into uninstrumented mid-air feedback has resulted in a few interesting techniques. To provide touch feedback, *UltraHaptics* employs localized ultrasonic sound waves to displace air [30], and *Aireal* and *AirWave* utilize vortices of air that collapse upon hitting a person [59, 152]. Force feedback can also be enabled by employing lasers to create plasma in mid-air that create shock waves when touched (e.g., [133]). Since these technologies are still new difficult to acquire, we took a different route to mitigate usability issues. We were inspired by the *Bubble Cursor* [58] and integrated a dynamically sized area cursor within the interaction space to help people perform spatial selections. If the distance from an object to the cursor is under a distance threshold, then the object is highlighted and able to be interacted with. If more than one object are under this distance threshold, the closest object to the cursor is chosen.

To actually select the object with the tap gesture, the user must move their index finger in the Z-axis away from themselves, and surpass a minimum speed threshold as well as a minimum distance threshold. The same is required for deselection, but different threshold values are used. Some people have difficulty performing this selection technique since people tend to not move their finger parallel to the floor when performing a tap or pointing gesture, even when they know that the gesture requires movement along the Z-axis. The end of the finger tends to move along the Y and Z axes (e.g., down and in), and sometimes the X-axis as well. Therefore, it is common that users think they are surpassing the minimum speed threshold, but in actuality, they are not moving fast enough along the required Z-axis. To help users properly perform this technique and select an object, we visualize the Z-axis velocity of the finger performing the mid-air tap gesture. To accomplish this, we change the thickness of the highlighting that is used to inform the user when the cursor is hovering over an object (see Figure 5.10). A higher Z-axis velocity causes the highlighting thickness to increase. This provides feedback to allow users to see if they are performing the gesture wrong (i.e., not moving fast enough along the Z-axis) or if finger movements are not being properly recognized by the motion sensing technology.

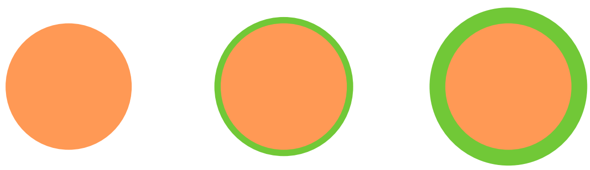


Fig. 5.10 The left and middle images shows how an object becomes highlighted when the cursor hovers over it. To help people perform the mid-air tap selection gesture, we use the thickness of this highlighting to encode the Z-axis velocity of the finger performing this gesture. As shown in the above right image, the user is then able to see whether they are actually moving along the correct axis that is required to perform the gesture.

When performing the mid-air tap gesture, the tendency of the finger to move in multiple axes can also cause the cursor to move outside of the object's interaction space before the tapping gesture was detected. This can make it difficult to actually select an object, especially if it is small. Our spatial interaction design takes this into account and reduces the vertical and horizontal movement of the cursor based on the speed of the hand and index finger along the Z-axis. It effectively "locks" the pointer in place by employing an easily modifiable reduction function. This is accomplished by finding the difference between the tracked entity's (i.e., finger or hand) current and previous location (temporally distant), and multiplying it by a reduction function. Finally, adding this value to the entity's previous location will give you the entity's current "locked" position. The reduction function employed in our system divides a constant (set to one) by the square of the entity's velocity along the z-axis. Therefore, the cursor becomes more horizontally and vertically "locked" as one moves faster along the Z-axis.

If the information space contains a lot of objects, it can be quite easy to mistakenly select one when putting one's hand in the spatial interaction space. To mitigate this, we require hands to stabilize by algorithmically analyzing its spatial and temporal properties. If the hand has just entered the interaction space, it must reduce its speed towards the screen (z-axis) or surpass a time threshold before it can be used to select an object with the spatial tap gesture.

With displays, it is easy to tell when one reaches the edge of the interaction space due to the cursor stopping at the edge of the screen. With off-screen interaction systems, there is no direct visual feedback of the spatial sensor's interaction space. To provide feedback, our system displays yellow coloured bands at the edge of the screen when the user is close to leaving the spatial interaction space (see Figure 5.12). These bands then change to red when one has left the interaction space.

With our off-screen exploration techniques, removing one's hand from the spatial sensor's field of view reverses the geometric transformations that were applied to the information space. Therefore, when one accidentally leaves the spatial interaction space, the user would have to restart the exploration technique. To mitigate this, the user is provided with a few seconds to place their hand back within the interaction space to recapture the recent transformation state. If executed, the aforementioned bands turn from red to green to indicate a successful recapture, and then disappear after a few moments. Although, if the user intentionally removes their hand from the interaction space, he will have to wait a few seconds for the information space to return to its original state, which can be annoying. To reduce this annoyance, our system

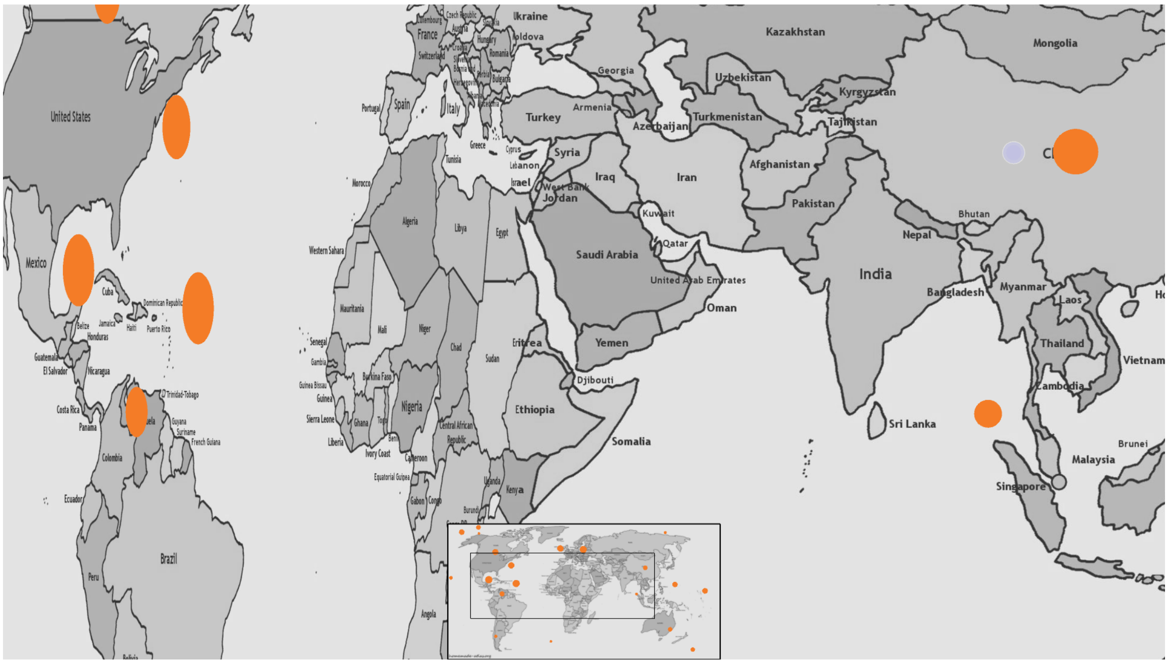


Fig. 5.11 When the *Dynamic Distortion* technique is used to bring off-screen content on-screen, the overview + detail's viewfinder will change its size to update what sections of the information space are located on-screen. The purple circle that is in the top right section of the figure represents the pointer of the user's hand or index finger that is located off-screen on the right side of the display.

determines the speed of the hand when it left the interaction space. If it is above a threshold, then the applied geometric transformations are immediately reversed.

5.2.2 Augmenting *Dynamic Peephole Inset*

Our system augments the *Dynamic Peephole Inset* technique by allowing each hand to have its own viewport. This enables one to explore the off-screen information space from different sides simultaneously with the use of multiple hands. The viewports can also be pinned to the display by performing a forward moving pinch gesture, as if one is placing a thumb tack. Pinning a viewport removes the requirement of keeping one's hand in the off-screen area; thus freeing up the hand to perform another task or for resting. Once pinned, a thumb tack is displayed near its top, and mouse or touch-based interaction techniques can be used to scale the viewport for closer inspection of the contents (see Figure 5.13). To unpin the viewport, the user can perform a backwards moving pinch gesture (opposite of original gesture), or select the thumb tack with the mouse or directly with a finger. When using the *Dynamic Peephole Inset* technique, we also augment the overview + detail visualization with another viewfinder that

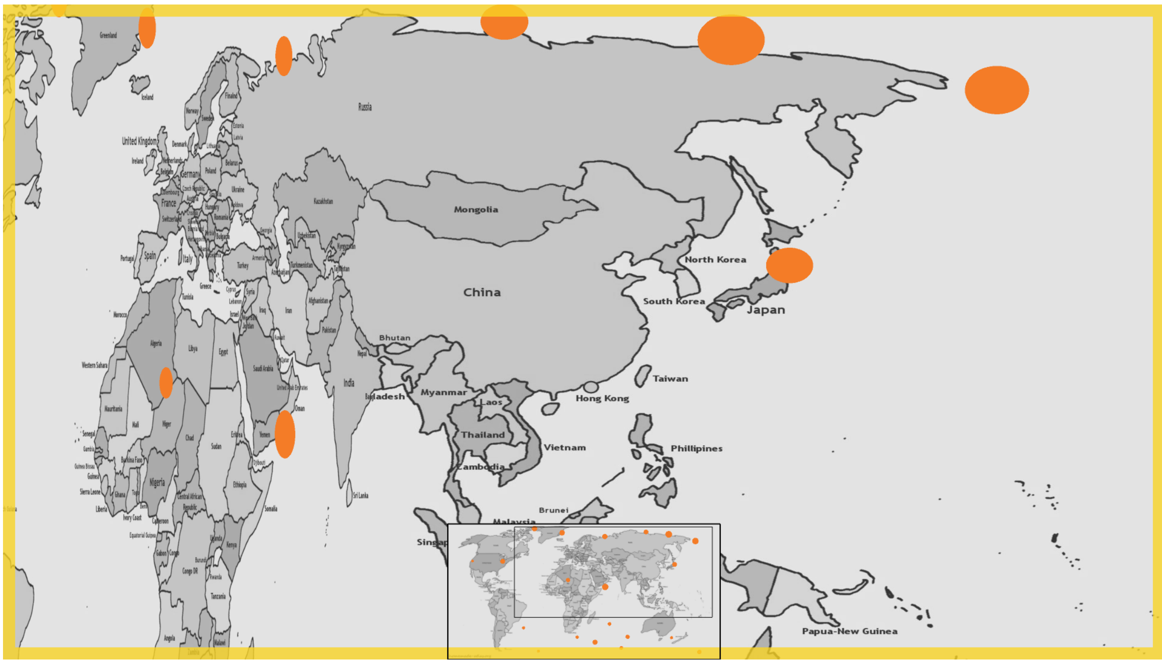


Fig. 5.12 The map is distorted vertically and horizontally to show off-screen content that is situated past the top right corner of the display. The yellow warning bands are used to indicate to the user that he is close to leaving the spatial interaction space.

represents the contents shown inside the exploration technique's viewport. This allows one to better understand the relative location of the off-screen content within the viewport with respect to the entire information space. This is beneficial since the *Dynamic Peephole Inset* does not provide much contextual visual information of the surrounding space due to the viewport only showing a small section of the off-screen information space.

Fig. 5.13 The *Dynamic Peephole Inset* is able to be pinned to the display to free up one's hand for other tasks. Once pinned, the viewport can be dynamically resized using mouse or touch-based interactions. On the right side of the above figure, the viewport has been enlarged for closer inspection of the contents.

5.3 Use Cases

To showcase the applied usage of our off-screen exploration techniques and Off-Screen Desktop, we have developed a number of prototypes as discussed in the following sections.

5.3.1 Desktop

Our desktop prototype augments the standard computer desktop with the ability to have application windows located off-screen. We imagine the off-screen space as being used for storing content such as when a user is working on a project and requires multiple windows open simultaneously. A lot of the time, all of these windows cannot fit on-screen without occluding one another. This can cause frustration and increase mental effort when the user is required to switch between many windows multiple times, since one must search for the correct window from many candidates. The taskbar mitigates this, but can still be frustrating when these windows are from the same application and look similar. To help, the user can store windows off-screen and use their spatial and proprioceptive memory to help them remember the stored location. If one physically places an item on the right side of the display, the aforementioned cognitive processes help one remember which side they stored the content. Therefore, instead of using the task switcher (Alt+Tab) one can use a spatial gesture to directly grab an off-screen window and bring it on-screen. A spatial push gesture could then be used to move the window back to its off-screen location. Also, by using a background image that is full of unique features, memorability of where users have placed windows is likely enhanced. This is due to the person being able to associate features with the content in the application window (e.g., above the roof and left tree as seen in the overview + detail visualization in Figure 5.14).

We also imagine the off-screen space being used for hosting newly downloaded content. For example, a user could have their news or Twitter feed located off-screen and be able to view it only when needed (see Figure 5.14). Having it appear off-screen reduces potential distraction and cognitive load with respect to the user. New emails could arrive off-screen as well, and have an on-screen notification visualization to indicate its presence. Different swipe gestures could then be used to ignore or open that particular email.

The off-screen space could also be divided into sections for different purposes. We call these off-screen 2D areas *portals* and they can be used to perform operations on content that is placed within them. For example, throwing a file into a specific portal could delete the file and throwing it into another portal could automatically send the file to another person or group by email. The 2D areas could be coloured differently along with borders and text to indicate to the user their size, position, and what command is invoked on content that is placed within.

Fig. 5.14 Our desktop prototype is shown here with a before and after sequence of its information space being distorted to view the Twitter feed that is located off-screen. Desktop Background Image Source: User Wowo2008 on Wikimedia Commons, 2014.

5.3.2 Map

We have also developed several map prototypes that showcase some interesting usage scenarios. Map-based interfaces typically employ semantic zooming where the amount of displayed information increases as the zoom level increases. For example, street names and the location of stores is only displayed after surpassing a certain zoom threshold. When route planning, this information can be very useful for the user. If the route is too long to display on the screen, the user can zoom out to show the entire route, but loses the aforementioned information. As seen in Figure 5.15, our off-screen exploration techniques can be utilized to fix this problem by bringing the entire route on-screen. Our other techniques can also be used to view the off-screen section of the route and quickly snap back to the original on-screen section.

Another idea is that instead of having the entire information space at the same zoom level, the on-screen section could be at a higher zoom and detail level than the off-screen sections. This would create a modified overview + detail interface where the on-screen section provides the detail and the off-screen sections provide context as seen in Figure 5.16. Therefore, context is provided on demand and does not take up screen space when not needed.

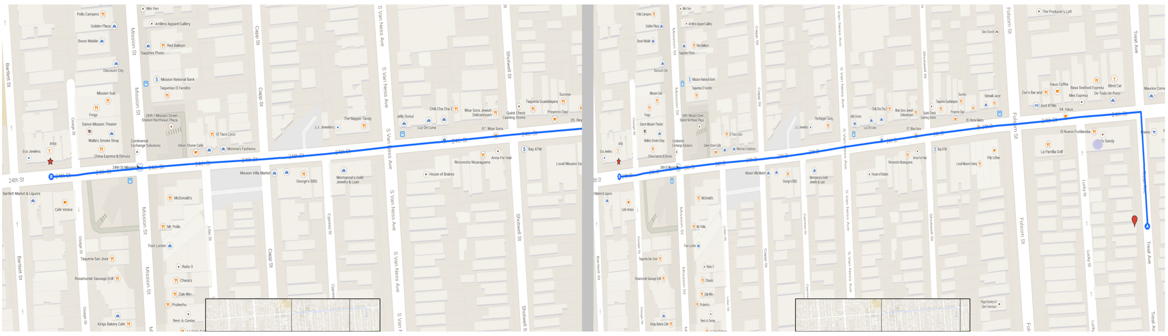


Fig. 5.15 The left image shows a planned route that is too long for the size of the screen at the current zoom level. The right image shows the same route, but employs our off-screen distortion technique to show the entire route without zooming out and losing detailed information.

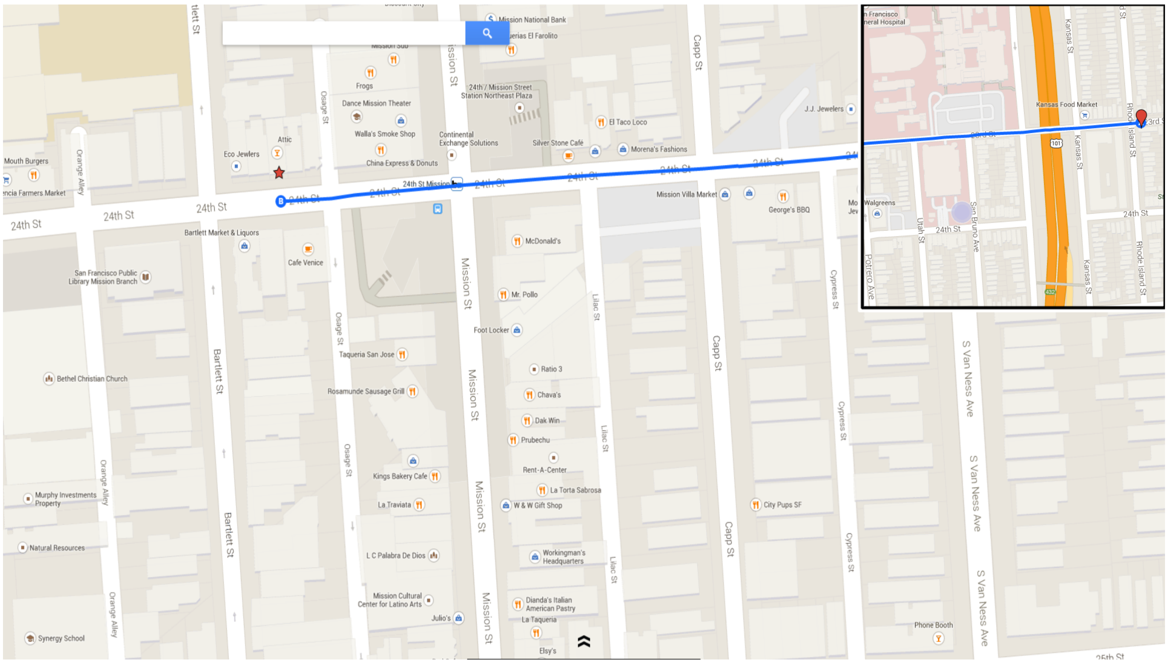
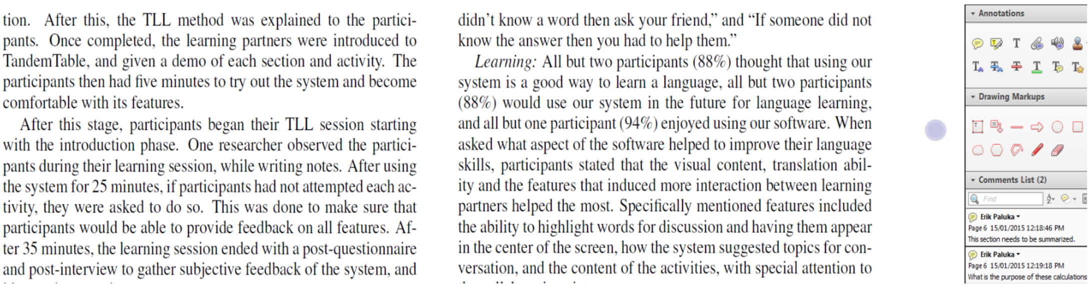


Fig. 5.16 The on-screen section of the information space provides a higher level of detail than the off-screen sections; thus modifying the concept of an overview + detail interface.

5.3.3 Off-Screen Toolbars

In the WIMP paradigm, part of the screen space is usually taken up by toolbars, ribbons and other control widgets. When not in use, these can waste space, cause cognitive overload or be distracting to the user. To mitigate this, the off-screen space could be harnessed to contain these widgets when not in use. As seen in Figure 5.17, the user can then use one of our off-screen exploration techniques to bring this content on-screen. This idea is similar to the use of a swipe down gesture on mobile devices to display the notification panel, and the left swipe gesture on Windows 8 computers to display the Charm Bar. An off-screen interaction system could also make use of a window blind gesture metaphor to display a notification panel or taskbar that is originally located above the display. The user would “pull down the blind” to invoke this operation. Another benefit to this idea is the implicit transience of the widgets being located on-screen. For example, one can gesture to bring a toolbar on-screen, make a selection, then move their hand back to the keyboard to continue working, which causes the toolbar to automatically move back off-screen. The system can either move the toolbar when the selection was made or when the user moved their hand out of the spatial interaction space. If the implicit transience of off-screen widgets is not appropriate for a specific task, the system could also support only moving the toolbar back off-screen when a specific gesture is performed.



tion. After this, the TLL method was explained to the participants. Once completed, the learning partners were introduced to TandemTable, and given a demo of each section and activity. The participants then had five minutes to try out the system and become comfortable with its features.

After this stage, participants began their TLL session starting with the introduction phase. One researcher observed the participants during their learning session, while writing notes. After using the system for 25 minutes, if participants had not attempted each activity, they were asked to do so. This was done to make sure that participants would be able to provide feedback on all features. After 35 minutes, the learning session ended with a post-questionnaire and post-interview to gather subjective feedback of the system, and

didn’t know a word then ask your friend,” and “If someone did not know the answer then you had to help them.”

Learning: All but two participants (88%) thought that using our system is a good way to learn a language, all but two participants (88%) would use our system in the future for language learning, and all but one participant (94%) enjoyed using our software. When asked what aspect of the software helped to improve their language skills, participants stated that the visual content, translation ability and the features that induced more interaction between learning partners helped the most. Specifically mentioned features included the ability to highlight words for discussion and having them appear in the center of the screen, how the system suggested topics for conversation, and the content of the activities, with special attention to

Fig. 5.17 System or application specific toolbars could be located off-screen to save screen space, and be brought on screen only when needed, as shown here by using the *Dynamic Distortion* technique.

5.3.4 System-Wide & Application Specific Commands

Our document scrolling and volume control prototypes show how the off-screen space can also be employed to invoke system-wide and application specific commands. These commands can be invoked with the use of fine grained gestures for tasks that require focus, or the use of coarse grained gestures when more “casual” interactions are needed [134]. Casual interactions are beneficial as they reduce the amount of focus that is required to perform them. This allows one to perform a secondary task (e.g., ignore an email that appeared off-screen) while minimizing the performance loss incurred on the primary task. Our prototypes employ a simple spatial interaction to enable one to read a document while panning its contents, or change the operating system’s volume. Volume and scrolling amounts are calculated based on the vertical position of the hand with respect to the display. We also support document scrolling and muting/unmuting the volume using simple vertical swipe gestures.

Our prototypes support two different mappings when calculating the volume or scrolling amounts based on the vertical position of the hand with respect to the display. In the first, the length of the document or total volume is normalized to the vertical height of the off-screen interaction space beside the display. For the document scrolling prototype, this allows one to pan the document half way through by placing one's hand at a vertical height that is defined by the vertical centre of the display. The second mapping depends on the vertical position of the hand when it enters the off-screen interaction space, and its subsequent vertical velocity (direction and speed) within that space. Moving the hand at a faster speed will increase the interaction's gain. As seen in Figure 5.18, this mapping also makes use of the hand's horizontal position with respect to the display to increase or decrease the scrollbar's scale precision (gain of the interaction). This is similar to the idea of using the space above a digital table to adjust the scale precision of a widget as discussed in *The Continuous Interaction Space* by Marquardt et al. [126]. Another idea is to change the gain of the interaction based on the content that is currently within the display viewport [87]. For example, a document would scroll faster when empty pages appear on screen and slow down when more important or informationally dense pages move into the viewport.

These different mappings and gain control techniques can be employed in other scenarios that can make use of sliders and scrollbar widgets. For example, one could change the volume of the system by vertically moving their hand beside the display, or seek a video the same way. We also see the off-screen areas as being able to support a modified version of Marquardt et al.'s work on discrete interaction layers [126]. Different parallel layers (distances from the screen) could be used to interpret the same gesture differently depending on which layer the gesture was performed in.

Another idea is to map different off-screen interaction areas to different application windows. Consider this example: the interaction area on the right side of the display is mapped to a drawing application, and the left side of the display is mapped to another instance of the same application. The user could then use spatial gestures on the right side of the display to draw within its corresponding application window, and do the same thing on the left side without requiring one to manually change which window is in focus. This also allows one to interact with multiple applications simultaneously. For example, a user could draw in an application window with their left hand while gaining inspiration from a list of photos in another window which is being scrolled with the right hand.

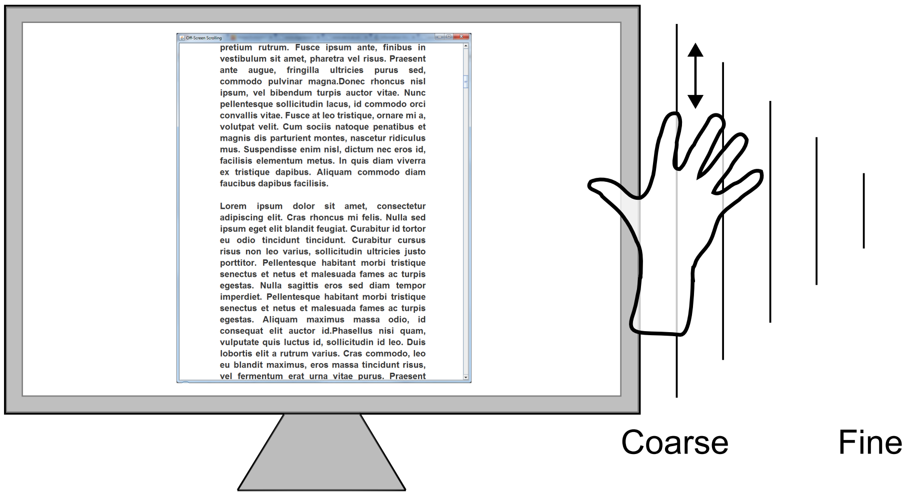


Fig. 5.18 Our document scrolling prototype shows how the off-screen space can be utilized as interactions areas to invoke system-wide or application specific commands. As seen in the above image, the hand's horizontal distance from the display can be used to change the gain of the interaction. For scrolling scenarios, this allows for coarse to fine grained scrolling.

5.3.5 Document Exploration

When reading a document, there are many scenarios where one is required to flip to a different page for a short duration then return to the original page. For example, if a reader comes across a term that was explained on an earlier page, but forgets its definition, the person might flip back to try and find the sentence explaining it. Another example is that due to the structure of documents, figures and tables might not be located on the same page as the text that explains it or refers to it. Therefore, when one comes across the explanation of the figure, the reader might flip to the page containing the figure to better understand it. Also, analyzing and comparing figures on different pages can cause the reader to flip back and forth between pages multiple times. Another scenario is when a reader is exploring or searching a document's contents by making use of its table of contents or index. The reader might flip back and forth between page candidates and the index or table of contents until the content being sought is found or the user is finished exploring.

To aid document exploration and mitigate page flipping, we have created a document exploration prototype. By laying out the document horizontally, the prototype is able to effectively makes use of our off-screen exploration techniques to allow one to view content that is located on different pages (see Figure 5.19). If an earlier explanation of a term requires revisiting or a referenced figure is located on another page, the user can use our techniques to bring that content on-screen by simply placing their hand in the space beside the display. When finished, one can quickly revert back to their original reading location by removing their hand from the spatial interaction space. For comparing multiple figures that are located on pages that a far apart, multiple instantiations of the *Dynamic Peephole Inset* or distortion techniques can be employed to bring them on-screen. Our off-screen exploration techniques also allow one to quickly explore a document (e.g., *Point2Pan*), or keep the table of contents or index on-screen while looking at page candidates.

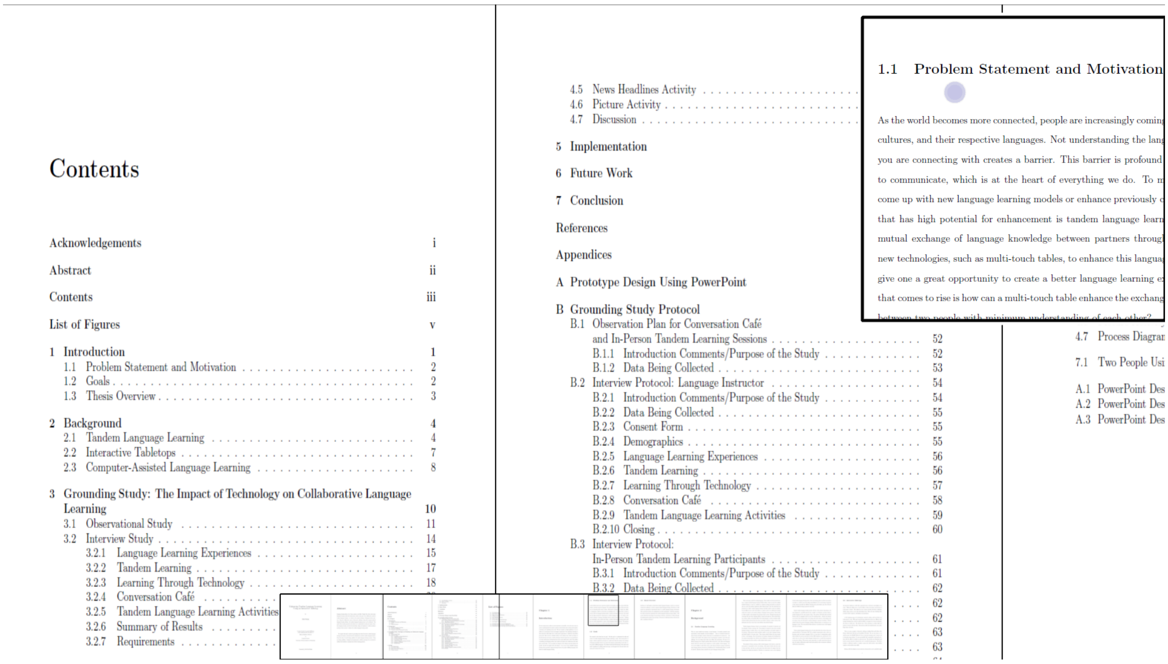


Fig. 5.19 Our document exploration prototype allows one to simultaneously view content that appears on two distant pages.

5.4 Implementation

Our off-screen interaction system was written in the Java programming language. To reduce coding complexity, we employed the use of the Processing library due to it having a simplified syntax and graphics programming model [142]. Different interaction modalities are supported through the integration of a multi-touch-enabled monitor, motion sensing hardware, and a mouse and keyboard (see Figure 5.1). To allow low level control of the touch information, we programmed all touch interactions ourselves based on touch position and time data that was received using the TUIO protocol [102].

5.4.1 Leap Motion Controller: Motion Sensing Hardware

To enable spatial interaction, we employed two Leap Motion controllers (Leap) [118], one for each side of the touch-enabled computer monitor (see Figure 5.1). A Leap is a Universal Serial Bus (USB) peripheral device that contains three infrared (IR) light-emitting diodes (LEDs) and two monochromatic IR cameras [37]. The LEDs emit IR light into the space above the device which reflect off of objects or limbs in that space. This reflected light is then captured by the device's cameras and sent to the computer for analysis. A 3D representation of the captured image data is then constructed to be able to extract tracking information. The interaction space of each Leap is shaped like an inverted pyramid (see Figure 5.20) and has a height of 60 cm (2 feet) with a total volume of 0.2265 cubic metres (eight cubic feet).

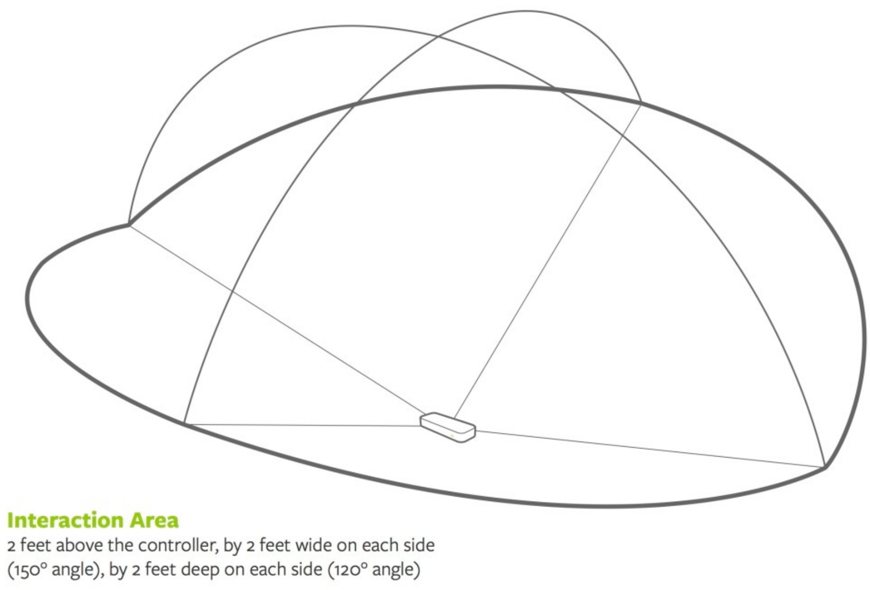


Fig. 5.20 Leap Motion controller's field of view. Image from Leap Motion's blog [37].

This motion sensing device can detect and track multiple hands and fingers, as well as thin cylindrical objects (tools). It also has support for detecting certain gestures including tracing a circle with a finger (Circle), a long linear movement of a hand

and its fingers (Swipe), a downwards finger tap (Key Tap) and a forwards finger tap (Screen Tap). The device also employs a right-handed Cartesian coordinate system with its origin centred directly on top of itself and positive Z-axis values increasing towards the user (see Figure 5.21). With relation to tracked entities, the system can provide developers with velocity and positional data, among many other types of information. As with the touch interactions, we did not use any of the gesture detection functionalities that the Leap software development kit (SDK) provided. We opted to algorithmically create our own so that we had low level control of these interactions.

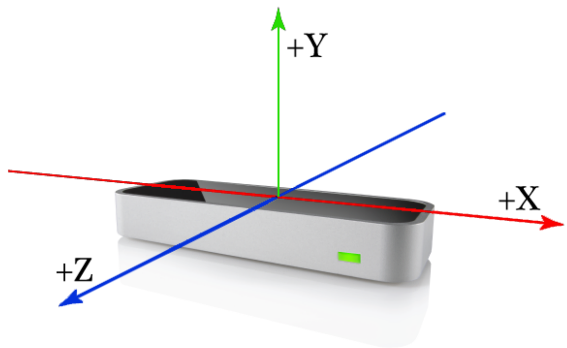


Fig. 5.21 Leap Motion controller's right-handed Cartesian coordinate system. Image from Leap Motion's website.

Using Multiple Leap Motion Controllers on the Same Computer

At the time of this writing, Leap Motion's software does not allow two Leaps to be connected and used simultaneously on the same computer. This caused a problem with the desired setup of our system since our designs included a Leap for each side of the computer monitor. To get around this problem, another computer can be used for capturing the data from the second Leap and sending it over the network to the host computer. Instead of requiring additional hardware, we opted to install the Leap Motion's software on a Linux virtual machine and gave it control of the second Leap. The captured data was then sent to the host operating system using the WebSockets protocol, and a WebSockets client was integrated into our system to be able to receive it. To reduce code complexity and redundancy, we also developed a wrapper to be able to encapsulate the data received over the network and the local Leap data with only one set of data structures.

Calibrating Multiple Motion Sensors for Off-Screen Interaction

Leaps are typically used within a standard desktop computer configuration and do not require calibration if they are centred horizontally with respect to the monitor. This is because the Leap SDK provides functions to automatically map its interaction space to the size of the display. To properly map spatial data in our system, we needed to know the location of each Leap with respect to the display. To accomplish this, we created a calibration mode that contains a touch-enabled target at each corner of the screen and a circular progress bar for feedback (see Figure 5.22). When a person directly touches a target, the system gathers positional data associated with this finger from each Leap. Since each Leap's centre of origin is on top of itself, the distance from the touched corner of the monitor to each Leap can be calculated from this positional data. The user is required to touch the target and hold their finger in this position for ten seconds so that the data can be averaged over time with the aim of increasing the precision of the final result. To determine if a corner of the screen is outside of a Leap's field of view, all the system has to do is query the Leap to see if it detects a hand when a corner is being touched. This calibration technique does contain one caveat: the user must only use one hand at a time with only one finger extended during the calibration process or else the Leaps might gather data from the wrong finger.

To reduce programming complexity, we also used this information to transform the Leap data from its local frame of reference into a global frame of reference (coordinate system). This is accomplished by querying the monitor to automatically determine the size of the display from resolution and pixel density data. The display size and data from the calibration mode are then used to determine the distance of each Leap with respect to the bottom left corner of the monitor. These distance values are then used in mathematical formulae to convert the Leap positional data, as it enters the system, into a global coordinate system.

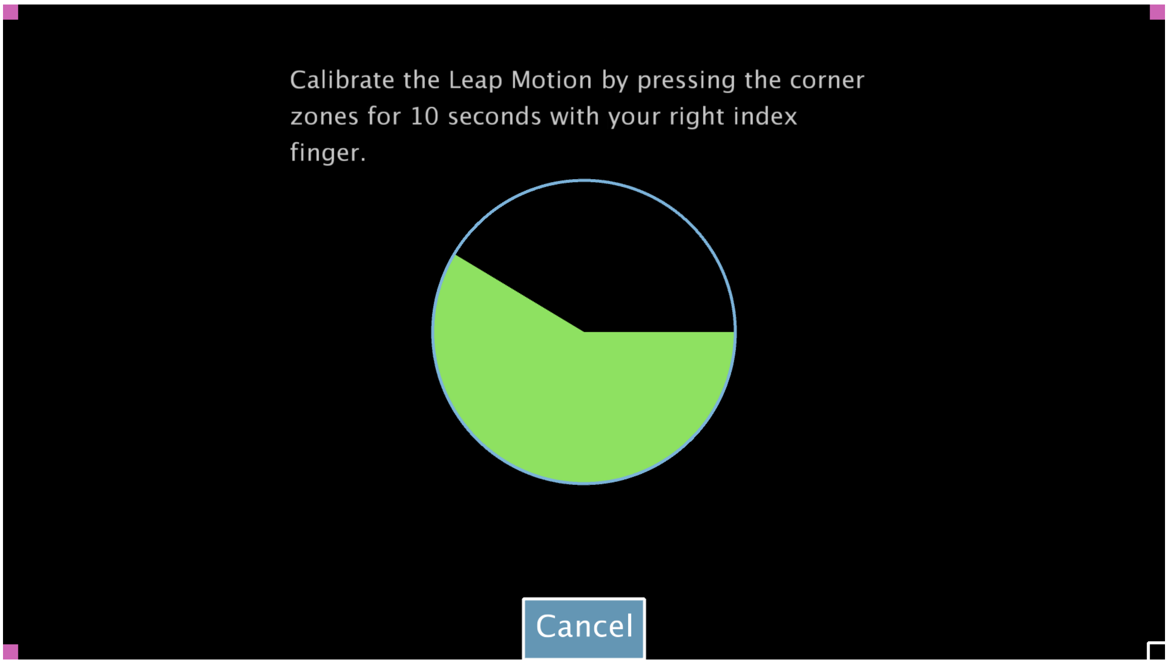


Fig. 5.22 Calibration mode in our system. A finger is touching the bottom right target and its calibration progress is reflected in the centre shape.

Chapter 6

SpatialVis: Visualization of Logged Spatial Interaction Data

An easy way to integrate spatial interaction on a desktop computer is to use spatial gestures to control the pointer. Although, this causes challenges since standard desktop graphical user interfaces (GUI) are designed for precise input devices such as the mouse. A typical virtual object's small display and interaction spaces reflect this, which can lead to problems selecting items as well as other fundamental tasks. To mitigate this problem, a designer can integrate concepts from techniques that facilitate target acquisition (e.g., Bubble Cursor [58]) into the mid-air selection gesture. After doing so and running some personal testing, it is beneficial to test the gesture with people who are not accustomed to it. To gather data for analysis, one can video record and/or observe people as they use the gesture in conjunction with a GUI, as well as administer post-questionnaires and interviews. Log data from the gesture's usage can also be gathered. The problem with this is that, other than the video and observational data, these techniques produce mostly textual data and do not harness the full power of the human visual system; therefore making the analysis difficult.

To mitigate these problems and help designers build better spatial user interfaces, as well as help us study our off-screen interaction system, we developed a web-based application that visualizes logged spatial interaction data. By first uploading a log file and an associated video screen capture of the display, an investigator can employ its features to analyze the 3D interactions and their effects on the graphical user interface. Our system is not meant to replace any other method, but to fit within the investigative process to gain further insight into the related phenomena.

We implemented the application using JavaScript, HTML, and the D3 visualization toolkit [16]. D3 is a JavaScript library for manipulating documents based on data and builds on many concepts from the Protovis project [15]. Our prototype supports the spatial interaction data types provided by the Leap Motion controller (see 5.4.1) and assumes that the controller's interaction space is in front and centred with the display. We also created a modified version of the application to be able to handle interaction spaces at the sides of the display. We did this to visualize data gathered from the study of our off-screen interaction system in order to gain further insight into participant usage patterns (see Chapter 7).

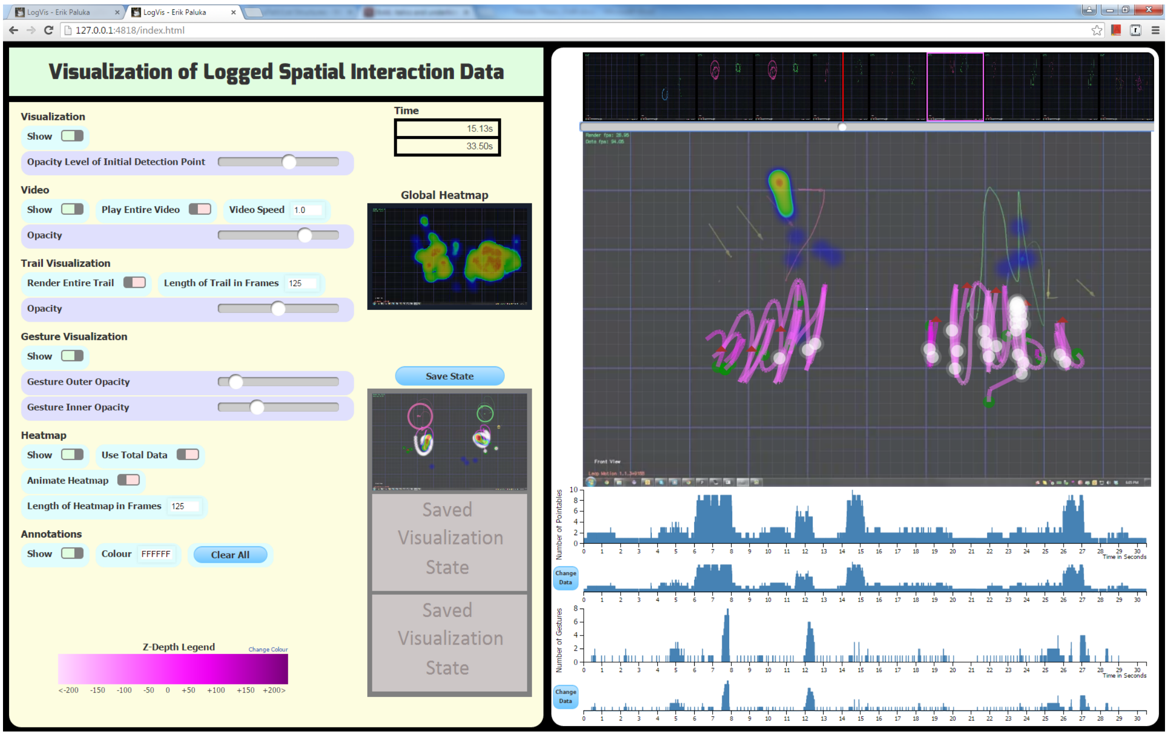


Fig. 6.1 SpatialVis: Web application for visualizing logged spatial interaction data.

6.1 Design

To use our system to analyze a spatial GUI, our Processing application can be used to automatically log all associated spatial interaction data as someone interacts with the spatial interface. A video of this interface must also be recorded, using screen capture software, with a video length equal to the period of time spent interacting with the interface. This allows log data to be mapped to the user interface events that occur in the video. When complete, the designer or investigator can upload the video and log files to our web application, which will then display the video on the right side of the interface with a heatmap and a path visualization overlaid on top of it. The system also includes a timeline situated above the video, graphs of the interaction data underneath the video, and a global heatmap and different controls to the left (see Figure 6.1). When the video is played, the overlaid visualizations display spatial interaction data that is temporally related to the video's current frame.

Going back to the spatial target acquisition example from above, the analyst can use our system in conjunction with observational notes or a video recording of the person performing the gestures. For example, this would allow one to view what data the motion sensing hardware is producing and if that matches up with the gesture that the person is trying to perform. If this analysis was done with logged data that was not visualized, the investigator would have to look through hundreds or thousands of lines of text and would be very tedious.

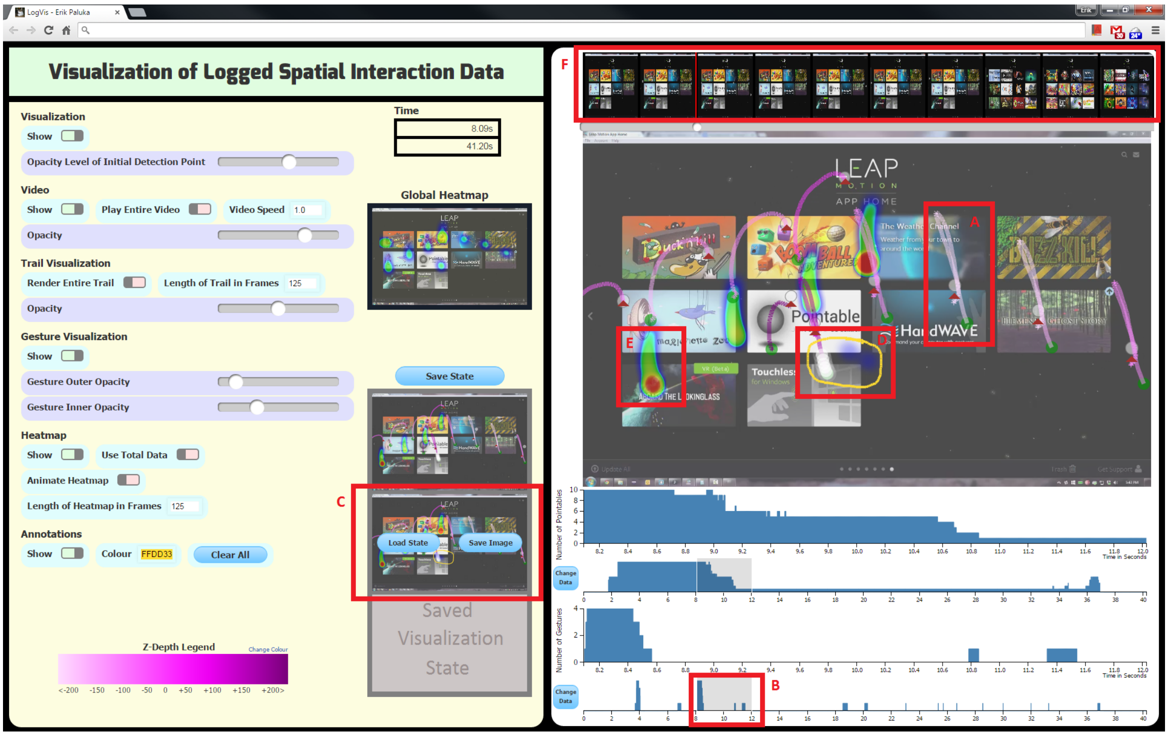


Fig. 6.2 SpatialVis being used by an analyst. (A) Visualizing portion of spatial interaction data.(B) Brushing to show data only associated with 8 to 12 seconds into the video. (C) Saved visualization state. (D) User annotation. (E) Heatmap of data associated with 8 to 12 seconds into the video. (F) Video timeline.

6.1.1 Video Timeline

The timeline is created by dividing the video into ten equally sized sections and using images from the beginning of each video segment to represent each section (see F in

Figure 6.2). When a user hovers over one of the sections, its border changes colour (see purple box at the top of Figure 6.1) and they are then able to select the section to seek the video to the start of it. If a section is selected, then the heatmap will update to show data only from this section's time range. If the video is then played, it will stop playing at the end of the selected section unless a widget on the left side of the interface is toggled. The timeline also contains a slider widget for seeking, while hovering over its handle will cause the play, pause and restart video controls to appear beside it.

6.1.2 Spatial Interaction Graphs

The graphs below the video show spatial interaction information over the length of the video. Their size match the width of the timeline to allow a person to match the current time of the video (slider's handle and vertical line above) with the graph data, as well as to provide awareness of video's current time value. The graphs are also enabled with the brushing and linking [105] techniques. Therefore, if one discovers a time range with an interesting data pattern, the visual complexity of the interface can be reduced to allow the analyst to concentrate on this subset of data. This is accomplished by selecting the data or time range of interest, which will then cause the rest of the data to be filtered out (see B in Figure 6.2). This brushing will then be reflected in the other graph, as well as in the heatmap and path visualization that are overlaid on top of the video. The video is also seeked to the beginning of the time range associated with the brushed data and if played, will stop at the end of this range. If the user is interested in analyzing other spatial interaction data types, they can change the data visualized in each graph from a range of options including pointables (tools and fingers), tools, fingers, hands, all gestures, as well as each individual gesture type.

6.1.3 Video Visualizations

We employed different visualization techniques to visualize each spatial interaction's location with respect to the user interface contained in the video. This was accomplished by overlaying them on top of the video using an orthographic projection mapping. We used a static heatmap to visualize the frequency of gestures that were performed at different locations. Data is selected to be visualized in the heatmap if its associated frame is within a non-sliding window. When the user first loads the required data into the application, the window is the size of the entire video; therefore all of the gesture data is initially visualized. If the video is played or seeked, then the window's starting frame is set to the seeked location or the beginning of the video segment being played. The window's ending frame is then calculated by adding a user-changeable value, contained in a widget, to the starting frame. Although, if the timeline sections or graphs are used to seek the video instead of the time slider, then the window's ending frame is set to either the timeline section's last frame or the last frame associated with the selected graph data. The interface also contains some other widgets that allow the user to set the window's ending frame to always be the last frame in the video, as well as to animate the heatmap over time using the data contained in its window.

We also visualized the path of each pointable (finger or tool) using a semi-transparent path. The pointable's Z-depth is encoded using colour with either a monochromatic or dichromatic divergent colour scheme. The path contains green semi-transparent circles to visualize the location of each pointable when it was first detected by the spatial interaction sensor. To visualize a pointer's spatial location in the current frame, a red semi-transparent triangle is attached to the end of the path. We also affixed a white semi-transparent circle to the path for visualizing the spatial location of different gestures, such as swipe, screen tap and key tap. For example, the white circles in Figure 6.1 show the location of discrete screen tap gestures. The visualization is dynamic since it displays data from a temporal sliding window that starts in the past and ends at the video's current frame. As the video plays, new data is visualized when it enters the sliding window and old data that is no longer inside the sliding window is removed. This aids the analysis process since interaction data would quickly disappear if only the current frame's data was visualized. The path visualization's sliding window is automatically set to a low value, but the user has the ability to change it, such as when one wants to visualize entire paths of all pointers.

In addition to the aforementioned visualizations, our system allows the analyst to create their own visual markings by providing video annotation abilities (see D in Figure 6.2), which can then be used to label the location of interesting events, for example.

6.1.4 Global Heatmap, Controls & Visualization States

Context is important for analysis, therefore we included a global context view of the gesture data with the use of a miniaturized image of the video that is overlaid with a heatmap that visualizes gesture data from the entire video. To further facilitate the analysis process, we also provide the ability to save and load different visualization states (see C in Figure 6.2). The video frame with the overlaid visualizations and user annotations that are associated with a saved visualization state can then be downloaded as an image for sharing and offline analysis. The interface also contains widgets on the left side to allow the investigator to show or hide the heatmap, path visualization, user annotations or the video itself. Opacity levels associated with the video and each component of the path visualization can be modified as well. The interface also contains widgets to allow the video's playback speed and the colour of the annotation brush to be changed.

Chapter 7

Study

To evaluate our off-screen exploration techniques, we conducted a study to compare three of our spatial techniques and standard mouse interaction when used to search for off-screen objects. We employed a common interface that typically deals with off-screen content: a map. Map-based interfaces are a classic example of where the information space can be and generally is larger than the size of the display. Other than changing the zoom level, panning with the mouse is the standard method of interacting with off-screen content in popular map applications (e.g., Google Maps, MapQuest, OpenStreetMap, HERE). Panning with the mouse is used to bring this content on-screen for viewing and to enable the use of the pointer for further interaction.

Our evaluation had a 2x2x4 factor within-subjects study design which included two tasks, two radial distance ranges, and four interaction techniques. In a real-world off-screen system, a person will already know an object's relative location if it was manually placed off-screen. Contrastingly when one's memory fades or when faced with an unfamiliar information space, the user will have to explore the off-screen area to find the content of interest. Therefore, we wanted to see how the different techniques performed when the user had to search for an object in the off-screen information space when its relative location was known and not known. We were also curious to see how the distance of the off-screen object with respect to the screen affected the performance of the techniques in each scenario. The techniques that we compared were panning with the mouse, *Dynamic Distortion*, *Dynamic Peephole Inset* with direct 1:1 mapping, and *Direct Spatial Panning*. *Point2Pan* was not chosen since its current design does not support controlling an on-screen pointer which is required for performing the study tasks. *Paper Distortion* was not chosen due to it being less flexible in terms of controlling what content should be brought on-screen. The two study tasks were search and select-only, which involved selecting the correct off-screen object out of many choices (see Section 7.2 for details). Each off-screen object was a certain distance from the centre of the information space, which gives rise to the two radial distance ranges of *close* and *far* (see Section 7.3).

Since only one section of the off-screen information space can be brought on-screen when using the mouse, we constrained the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques for the study. Only one side of the information space could be distorted at a time when using *Dynamic Distortion*, and the *Dynamic Peephole Inset* technique only supported one viewport. Also, only the left and right side of the display supported spatial interaction. In the following sections, we include details concerning the study to enable its replication. It is very important to do this since reproducibility is one of the main principles of the scientific method, and as a Human-Computer Interaction community, we need to move towards performing and publishing more replication studies to increase our scientific rigour [57].

7.1 Experimental Setup & Equipment

Our study setup included a desktop computer, monitor, keyboard, mouse, and two Leap Motion controllers on a desk. The Leap Motion controllers were placed on the left and right sides of the display to enable spatial interaction in those areas. The display was a Dell S2340T 58 cm (23 inch) multi-touch monitor with 1920x1080 high definition (HD) resolution (~3.78 pixels per mm) at 60Hz and supported 10 simultaneous touch points. The computer was a Lenovo ThinkStation E31 with a 3.30 GHz Intel Core i5-3550 CPU, 6.00 GB of RAM, and a NVIDIA Quadro 2000 graphics card. The installed operating system was the 64-bit Professional version of Microsoft Windows. A Logitech Unifying receiver was used in conjunction with a Logitech wireless keyboard version K520 and a Logitech wireless mouse version M310. Both Leap Motion controllers were consumer versions (not the initially released developer versions) with firmware v1.7.0, and used v2.2.3 tracking software. To run both Leap Motion controllers on the same computer, the virtualization software package called VMware Player version 6.0.1 build-1379776 was used to run a Linux Mint 17 64-bit virtual machine. As shown in Figure 7.1, each Leap Motion controller had a horizontal distance of 35 cm and a depth distance of 7.5 cm from their respective centre to the centre of the display. The vertical distance from the desk to the bottom of the screen was 15 cm.

We used our Off-Screen Desktop system to create the map interface that was used in the study. Although, our system supports zooming, this functionality was disabled. We selected a grayscale map of the Earth (cylindrical map projection) with country

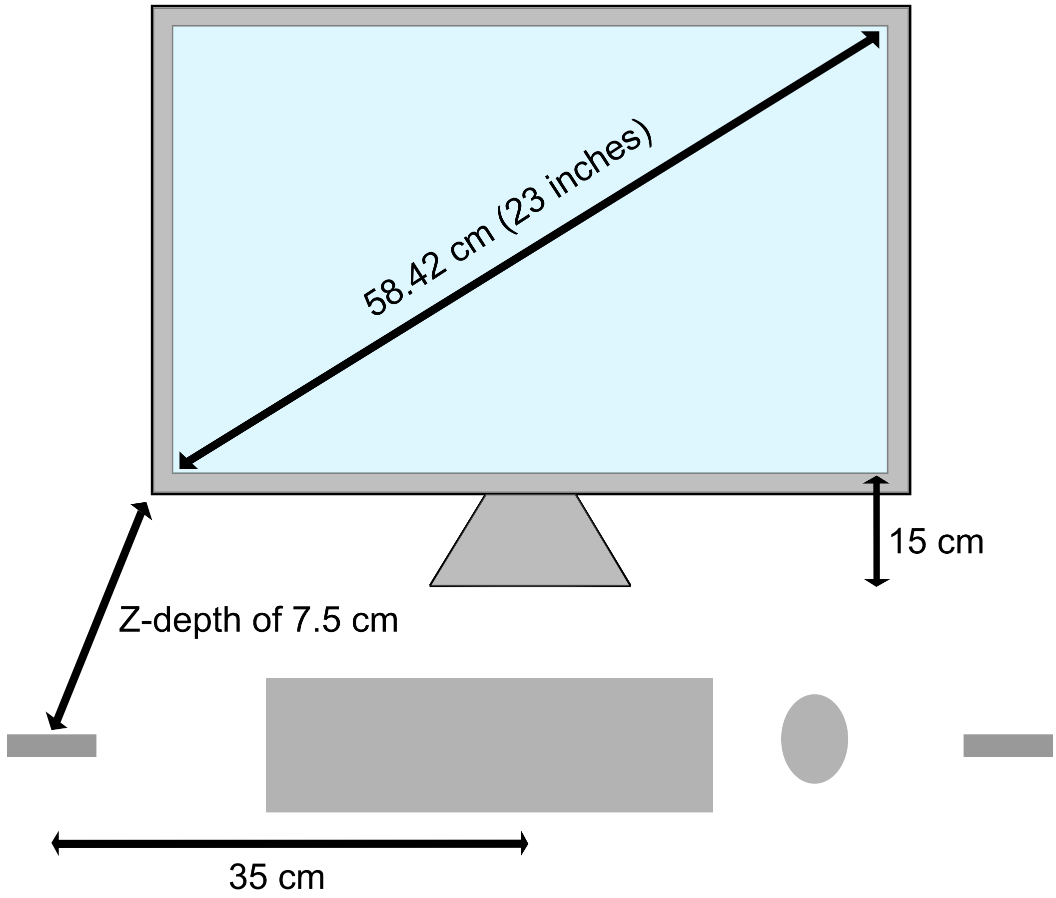


Fig. 7.1 Our study setup consisted of two Leap Motion controllers, a wireless mouse and keyboard, and a multi-touch monitor on a desk. Each Leap Motion controller had a horizontal distance of 35 cm and a depth distance of 7.5 cm from their respective centre to the centre of the display. The vertical distance from the desk to the bottom of the screen was 15 cm.

boundaries and names to define the landscape of the information space (see Figure 7.2). This map had a width of 4500 pixels and a height of 2234 pixels. A map of the Earth was chosen since most people are familiar with it, especially this map projection, which enables people to determine their general location in the information space by just analyzing what continents and countries are visible. The map originally had different colours for each country, but we made it grayscale to reduce the possibility of distraction; thus allowing participants to focus more on analyzing the off-screen targets.

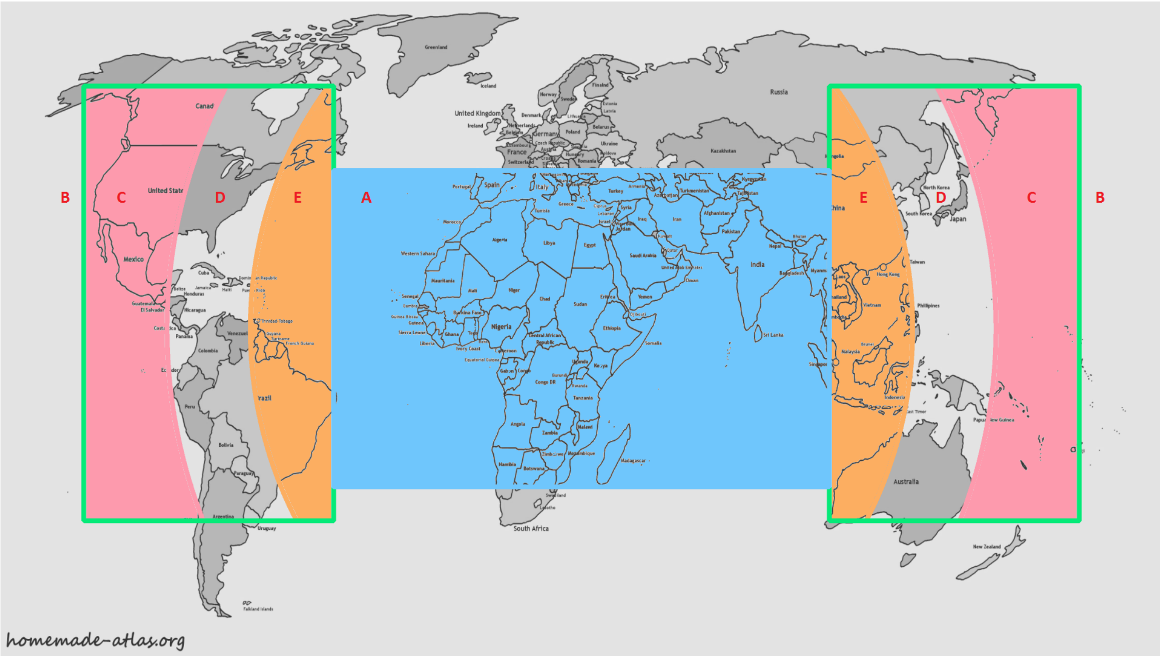


Fig. 7.2 This is a picture of the map that was used in the study, and has been overlaid with coloured lines to depict certain regions. The actual map had a width of 4500 pixels and a height of 2234 pixels. (A) The blue rectangle represents the monitor's screen size of 1920x1080 pixels. This was the section of the map that the user saw at the beginning of each trial. (B) The green rectangles represent the areas where off-screen objects could appear. (C) Within the pink regions defines the *far* radial distance range. (D) The single target in each trial never appeared between the orange and pink regions. (E) Within the orange regions defines the *close* radial distance range.

7.2 Tasks

The study consisted of two tasks, named search and select-only, where participants had to select the correct off-screen object out of 31 possibilities (one target and 30 distractors). In both tasks, each trial started with participants being presented with a copy of the target (reference) in a white square near the centre of the map (see the left image in Figure 7.3). The target reference persisted for the entire duration of each trial at the same location in the information space to allow participants to refer to it when their memory faded. In the search task, participants had to search for the target and select it. In the select-only task, the location of the target was visualized with a diverging dichromatic arrow using blue at the tail and red at its head. The tail was always located at the centre of the screen with its head very close to the target (see Figure 7.3). It is important to note that with the *Dynamic Peephole Inset* technique, two arrows were used since the map is never transformed to bring off-screen content on-screen when employing this technique. Therefore, the interface included the second arrow visualization inside the off-screen viewport (see Figure 7.4). We chose these tasks since we wanted to separate out the direct to target (Fitts' law) type task from an exploration task which may require more extensive use of off-screen interaction techniques. Therefore, generating more data for analysis.

To indicate when the participant could select an object, each object was highlighted when the cursor was within its interaction space. The highlighting employed a grey circle with a thin black border to create a luminance contrast break with surrounding colours [162]. Each object's interaction space was 80 pixels wide and 80 pixels tall. Since people are much more precise and faster when moving a cursor on a 2D surface (mouse) when compared to a 3D space with no tactile feedback (spatial gestures), we integrated an area cursor into our spatial techniques. This effectively doubled each object's interaction space to 160 pixels wide and 160 pixels tall. For all techniques, participants performed a selection by hovering the cursor over the object and pressing the spacebar on the keyboard. We designed it this way because we are comparing a very mature and extensively used input device (mouse) with one that is not commonly used and is technologically immature (spatial interaction). Spatial selection algorithms that make selections quick, easy, and fluid have not been created yet or matured; thus integrating them into the study would be inappropriate and harmful [57]. In addition, a keyboard was used for selection since we are not evaluating selection techniques, but rather techniques for exploring the off-screen space and bringing off-screen content on-screen. The mouse technique employed the standard operating system cursor, and our spatial interaction techniques used a circular cursor that was purple (see top right

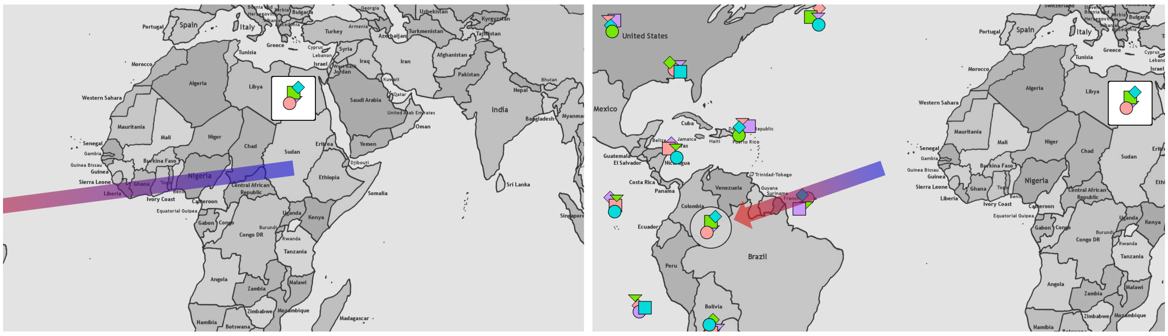


Fig. 7.3 The left image shows the section of the map that is visible at the start of each trial in both tasks, as well as the copy of the target. The arrow visualization is used in the select-only task to indicate the position of the target. The pointer (not shown) is hovering over the target in the right image; causing the target to become highlighted.

section in Figure 7.4). The standard operating system cursor was disabled in the spatial interaction trials.

7.3 Target & Distractors

For each trial, 31 distinct off-screen objects (one target and 30 distractors) were created and randomly placed off-screen. Each of them consisted of a triangle, diamond, circle, and a square that were grouped closely together in a random fashion. For each object, each of the shapes were randomly given one of four luminance controlled (75%) colours. No two shapes within an object could contain the same colour. These colours were 90 degrees from each other on a circular HSL colour wheel. The colours were equiluminous to reduce preattentive processing effects to prevent shapes with certain colours from "popping out" [162]. Their RGB values were RGB(253, 160, 160), RGB(118, 230, 6), RGB(6, 219, 219), and RGB(205, 154, 252). To reduce contrast effects, each shape had a thin black border which created a luminance contrast break with surrounding colours [162].

For each object, each of its shapes had a random position and Z-depth ordering, but was never fully occluded. All of the shapes for an object had to fit within an area that was 80 pixels tall and 80 pixels wide, with a minimum distance of 20 pixels from each other's centroid. The borders of all shapes were two pixels thick. All squares, diamonds and circles were 40 pixels wide and 40 pixels tall (not including borders). All triangles were 40 pixels wide and 20 pixels tall (not including borders). Each of the objects had a minimum distance of 240 pixels from each other's centroid.

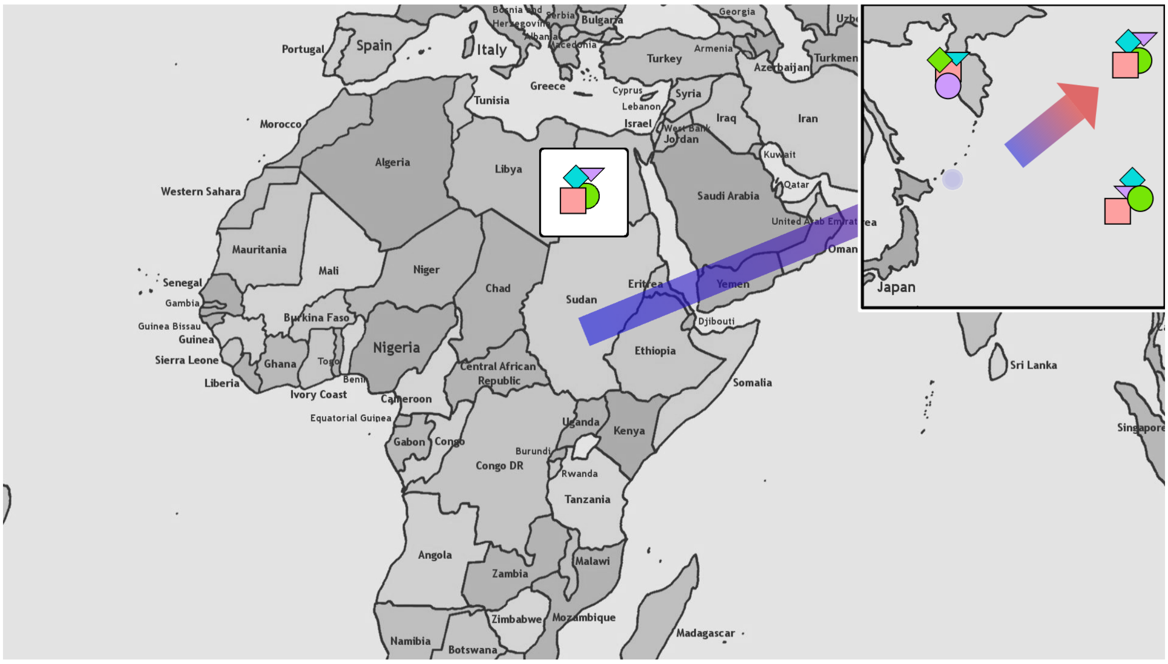


Fig. 7.4 The *Dynamic Peephole Inset* included a second arrow visualization since the map does not transform to bring off-screen content on-screen when using this interaction technique.

At the start of each trial, each object, except for the target, was randomly placed in off-screen areas depicted by the green rectangles in Figure 7.2. Each of these areas did not encompass the full off-screen area in their respective space due to the sensing limitations of the Leap Motion controller. From pilot testing, we concluded that the controllers did not detect spatial interactions reliably enough at the edges of the information space. Detection was possible, but required either lots of experience with the system and/or multiple attempts. Therefore, we excluded the edges of the information space for object placement by determining, with pilot testing, the size of the off-screen areas that provided adequate detection reliability. Each of the off-screen areas depicted by the green rectangles in Figure 7.2 had a width of 968 pixels (322 pixels from edge) and a height of 1484 pixels (288 pixels from top edge and 462 pixels from bottom edge). Although, the top and bottom edges had different heights, these dimensions maximized the amount of off-screen space where objects could be placed and reliably interacted with.

At the start of each trial, the single target was randomly placed within a subsection of the off-screen areas depending on the trial's associated radial distance range from the centre of the map. C and E in Figure 7.2 depict the *far* and *close* distance ranges respectively. The minimum distance of the *far* range and the maximum distance of

	Never used before	Used once or twice	Use a few times a year	Use monthly	Use weekly	Use daily
Computer mouse	0	0	0	1	2	13
Leap Motion controller	13	3	0	0	0	0
Microsoft Kinect	9	5	1	1	0	0
PlayStation Move, Camera, or Eye	9	4	1	1	0	1
Motion capture system (e.g., Vicon, Natural Point)	12	3	1	0	0	0

Table 7.1 Input device usage data for all participants.

the *close* range were calculated based on dividing the width of one side's off-screen area (green rectangle) by three. Adding this value to our screen's width, that was first divided by two, gave us the maximum radius distance for the *close* range. To get the minimum radius distance for the *far* range, this value was added twice instead of only once. It is important to note that participants were not told about the regions for object placement. They were only told that the location of each off-screen object was randomly chosen at the start of each trial. A buffer region between the two distance ranges was employed to make sure that there was a minimum adequate difference in distance between any object in the close distance range and any object in the far distance range. This eliminated the chance of two off-screen objects being right beside each other, yet residing in two different distance ranges.

7.4 Participants

For participants, we recruited 16 undergraduate and graduate students from the University of Ontario Institute of Technology. Nine were male and seven were female with ages ranging from 18 to 27 (M = 21, SD = 2.7). All of them were right handed and received $20 for participating in the study. Through self reporting, all participants were screened to make sure they had normal or corrected-to-normal vision, full use of their arms and hands, and were not colour blind. Three of them were familiar with the idea of interacting with off-screen content and two were familiar with the idea of off-screen pointing. In terms of input device usage, 81% of participants used a computer mouse daily and 81% had never used a Leap Motion controller before. See Table 7.1 for input device usage, which includes data on the computer mouse and different motion sensing systems.

7.5 Procedure

At the start of the study, the participant filled out a questionnaire to gather demographic and device usage data. When finished, the investigator then explained and demonstrated the Leap Motion controller, the system setup, the size of the spatial interaction spaces, and the different interaction techniques employed in the study. Time was also spent to explain the most optimal hand orientations that increase the likelihood of the Leap Motion controller recognizing the spatial gestures. When ready, the participant sat in a chair that did not support reclining nor rotating, and was positioned at a close, but comfortable distance from the desk. The participant then practiced each of the different interaction techniques until they understood how to use them. For each interaction technique (4), the participant performed a training round of six trials with each study task (2), then performed a study round of twelve trials with each study task (2). A radial distance range was randomly chosen for each trial, but balanced to make sure that each round of trials contained the same number of *close* and *far* ranges. This resulted in 48 training trials and 96 study trials for each participant, and 1536 total study trials for all participants.

Each study session lasted 60 to 80 minutes, and ended with a post-questionnaire and a semi-structured post-interview (see Figure 7.5). A balanced Latin Square was employed to counterbalance the different techniques and reduce carryover effects. Task type order was counterbalanced between participants. Task completion time and accuracy were our dependent variables, and user interaction data was logged. Participants were told to be as fast and accurate as possible when performing the different tasks. There was no time limit for each study trial, and participants could rest at any time as long as it was between trials. To gather additional data, a research investigator observed each study session.

Before the start of a round of trials, the name of the interaction technique, the phase (demo, practice, or study), and the task type was shown on a black screen. The participant then had to push the "w" and "\" keys at the same time to proceed. Before the start of each trial, a start button appeared on-screen and the map was centred to have equal off-screen space on the left and right side, as well as equal space above and below the display. To start the trial, the "w" and "\" keys had to be pressed at the same time. The start button then disappeared, and the reference target was displayed in a white square. The participant then performed the required task with the specified interaction technique. Participants only received feedback as to whether they selected the correct off-screen object during training rounds.

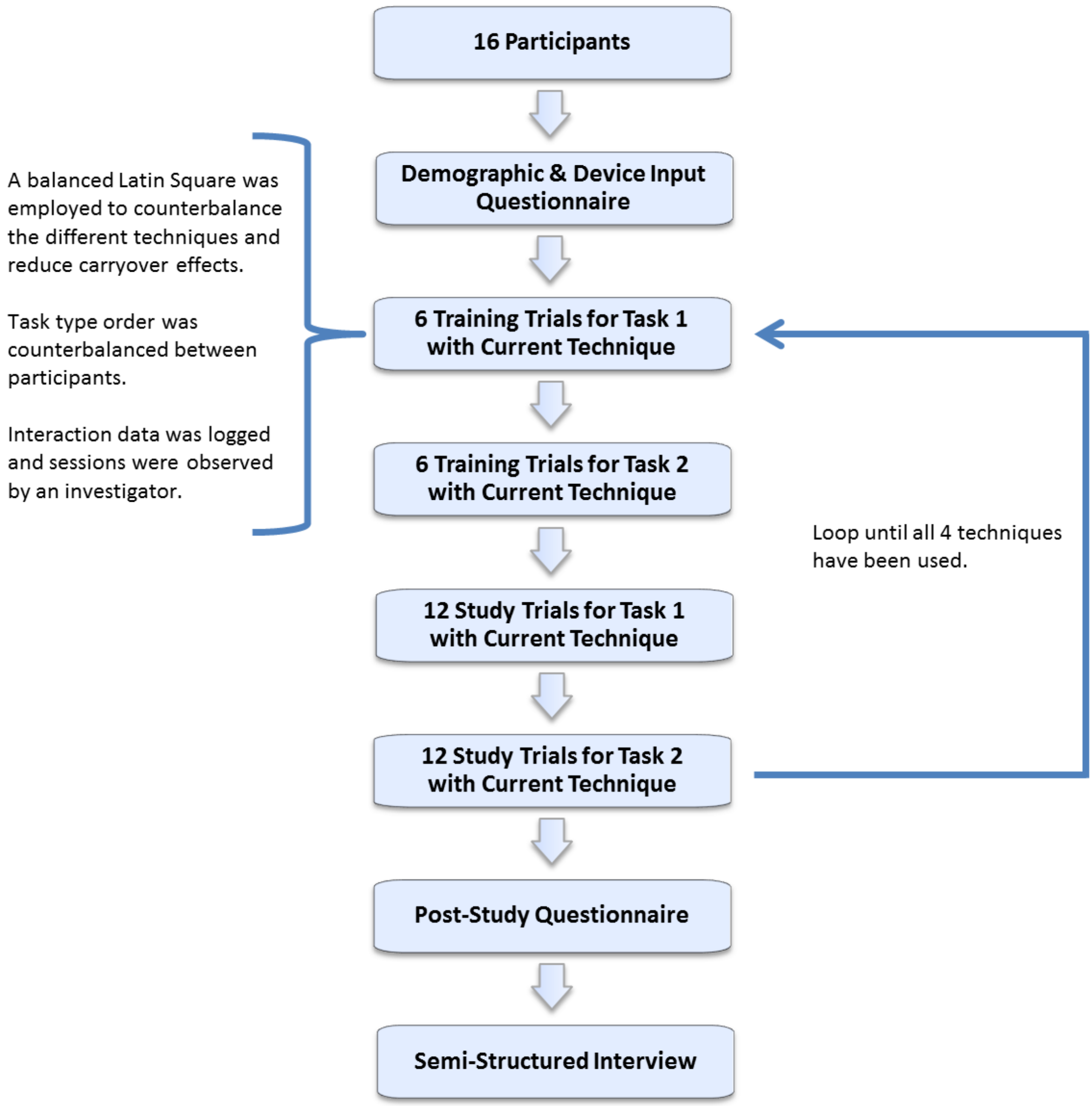


Fig. 7.5 Experimental process diagram.

With spatial interaction, having one's hand inside the motion sensor's field of view at the start of a study trial can bias the results. Likewise, holding a mouse in one's hand can have the same effect. To prevent priming of interactions in our pilot test, we originally requested participants to place both hands on the keyboard before the start of each trial. Unfortunately, participants forgot this request due to the repetitive nature and length of the study. Pushing the aforementioned two keys at the same time required participants to centre their hands on the keyboard at the start of each trial, which enforced our priming prevention method.

7.6 Hypotheses

For the experiment, we propose a number of hypotheses stemming from our experience in designing and using the different off-screen interaction techniques, as well as our knowledge in relation to human-computer interaction.

The mouse input device is widely used and very familiar to most people in technologically advanced societies. The same holds true for its associated panning technique especially in the context of map exploration. Even though spatial interaction has become more popular in the last decade for interacting with game consoles, most people do not have much experience when using them to interact with desktop computers, let alone map applications. Also, the mouse is an indirect pointing device with an interaction gain. This means that it requires less physical movement for navigating an information space when compared to direct interaction techniques, such as our spatial techniques. Therefore, due to this as well as the average person's extensive past experience with the mouse and its panning technique, we believe participants will have the fastest task completion times for both tasks when using this off-screen exploration technique.

H1: Out of all four techniques, the mouse will have the fastest task completion times for both tasks.

Each of the spatial off-screen exploration techniques in our study have been designed to support the comparison of off-screen content with on-screen content. *Direct Spatial Panning* allows one to quickly switch between viewing off-screen content and the original on-screen content by placing and removing one's hand from the the spatial interaction space. *Dynamic Distortion* transforms the information space to be able to bring off-screen content on-screen without removing the original on-screen content, and

Dynamic Peephole Inset creates a small viewport on-screen that displays off-screen content. We believe that by keeping the reference object on-screen, participants will be more likely to visually compare objects than to rely on their memory of the reference object, which can be prone to errors. Therefore, due to *Dynamic Distortion* and *Dynamic Peephole Inset* being able to retain the reference object on-screen at all times, we hypothesize that these two techniques will have a higher accuracy level than the other techniques for the search task. Furthermore, we believe that *Dynamic Peephole Inset* will have a higher accuracy level than *Dynamic Distortion* for the search task since the *Dynamic Distortion* technique distorts the original on-screen content; thus making it harder to compare objects with the reference object.

H2: The *Dynamic Distortion* and *Dynamic Peephole Inset* techniques will have the highest accuracy levels out of all of the techniques in the search task.

H3: The *Dynamic Peephole Inset* will have a higher accuracy level than the *Dynamic Distortion* technique in the search task.

7.7 Results

In the following sections, we will discuss study results from our formal evaluation. These include task completion time, accuracy, and logged interactions, as well as questionnaire and interview data.

7.7.1 Time & Accuracy

As previously mentioned, time (milliseconds) and accuracy were our dependent variables in the evaluation. In the following sections, we will discuss our findings in relation to each of them using a statistical significance level of $\alpha = 0.05$. Since our evaluation uses a repeated measures study design, the likelihood of incorrectly rejecting a true null hypothesis (Type 1 error) is increased. Therefore, where appropriate, we employ the Bonferroni correction method to counteract the problem of multiple comparisons.

Time

We performed three different tests of normality to determine what type of statistical test we could use to analyze the time data. These included Shapiro–Wilk tests, as well

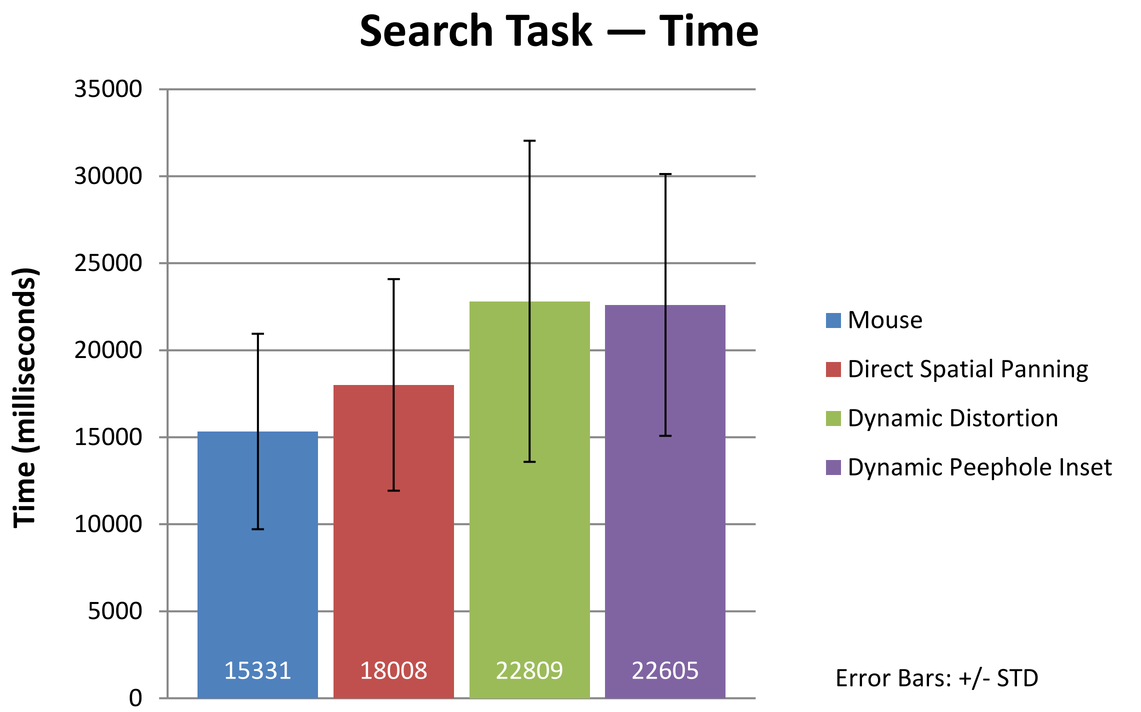


Fig. 7.6 Participant mean task completion time in milliseconds for the search task with the different techniques. Error bars show the standard deviation.

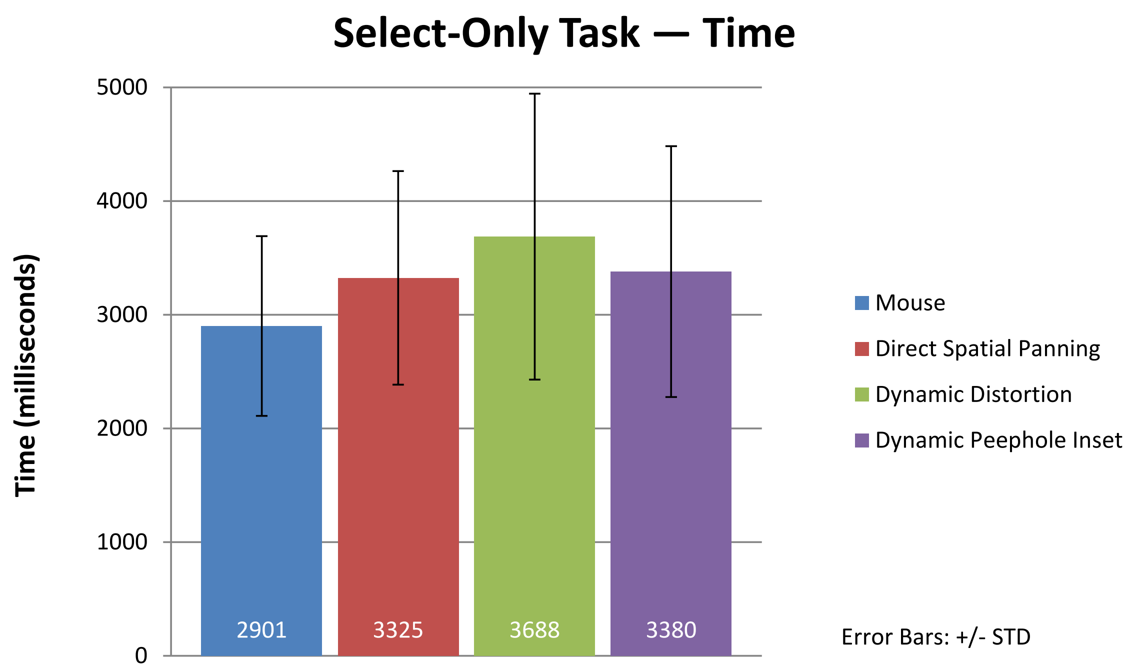


Fig. 7.7 Participant mean task completion time in milliseconds for the select-only task with the different techniques. Error bars show the standard deviation.

3-Way Repeated Measures ANOVA for Time

Technique	*F(3, 45) = 19.861*	***p < .0001***
Task	*F(1, 15) = 197.650*	***p < .0001***
Distance	*F(1, 15) = 0.080*	*p = .781*
Technique * Task	*F(3, 45) = 13.727*	***p < .0001***
Technique * Distance	*F(3, 45) = 0.667*	*p = .577*
Task * Distance	*F(1, 15) = 0.056*	*p = .816*
Technique * Task * Distance	*F(3, 45) = 0.198*	*p = .898*

Table 7.2 Results from a 3-way repeated measures ANOVA with factors *technique* (4 levels), *distance* (2 levels), and *task* (2 levels). Boldface results are statistically significant.

as calculating skewness and kurtosis values. All factor combinations had p-values in the Shapiro-Wilk tests that were above 0.05, and skewness and kurtosis values within the acceptable range of -1.96 to +1.96. This meant that our data was approximately normally distributed and a parametric statistical test can be employed for analysis. We therefore performed a 3-way repeated measures ANOVA with factors *technique* (4 levels), *distance* (2 levels), and *task* (2 levels). The Mauchly's Test of Sphericity was performed and showed that the assumption of sphericity had not been violated — *technique*: $\chi^2(5) = 3.732, p = .590$, *technique * task*: $\chi^2(5) = 3.692, p = .595$, *technique * distance*: $\chi^2(5) = 8.841, p = .116$, *technique * task * distance*: $\chi^2(5) = 7.442, p = .191$.

Analysis showed that there was a significant main effect for *technique* and *task*, and an interaction effect of *technique * task* (see Table 7.2). Post-hoc pairwise comparisons with Bonferroni correction showed that the mouse technique was significantly faster than *Direct Spatial Panning*, *Dynamic Distortion*, and *Dynamic Peephole Inset* (see Table 7.3). Also, *Direct Spatial Panning* was significantly faster than *Dynamic Distortion*, and *Dynamic Peephole Inset*. See Table 7.4 for mean task completion time and confidence interval data for each technique.

A post-hoc pairwise comparison showed a significant difference ($p < .0001$) between *task* types, with participants completing the select-only task significantly faster than the search task. Mean and confidence interval data for each task can be found in Table 7.6. As mentioned earlier, an interaction effect between *technique* and *task* was found. For comparing the different techniques in the search task, post-hoc paired samples t-tests with Bonferroni correction were conducted, which found that participants were significantly faster in the search task when using the mouse technique when compared to the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques (see Table 7.7).

Post-Hoc Pairwise Comparisons of Techniques for Time

Mouse	Direct Spatial Panning	$\boldsymbol{p < .05}$
	Dynamic Distortion	$\boldsymbol{p < .0005}$
	Dynamic Peephole Inset	$\boldsymbol{p < .0005}$
Direct Spatial Panning	Dynamic Distortion	$\boldsymbol{p < .005}$
	Dynamic Peephole Inset	$\boldsymbol{p < .05}$
Dynamic Distortion	Dynamic Peephole Inset	$p > .99$

Table 7.3 Post-hoc pairwise comparisons of techniques with Bonferroni correction. Boldface results are statistically significant. p-values greater than 0.99 are due to Bonferroni correction.

				95% Confidence Interval	
Technique	**Mean**	**Std. Dev.**	**Std. Error**	**Lower Bound**	**Upper Bound**
Mouse	9116	7068	570	7902	10330
Direct Spatial Panning	10666	8313	685	9207	12126
Dynamic Peephole Inset	12993	10723	842	11199	14787
Dynamic Distortion	13248	10827	887	11357	15140

Table 7.4 Mean task completion time and confidence interval data for each technique. Time is in milliseconds.

				95% Confidence Interval	
Radial Distance Range	**Mean**	**Std. Dev.**	**Std. Error**	**Lower Bound**	**Upper Bound**
Close	11438	4570	724	9895	12980
Far	11574	4359	666	10155	12993

Table 7.5 Mean task completion time and confidence interval data for each radial distance range. Time is in milliseconds.

				95% Confidence Interval	
Task	**Mean**	**Std. Dev.**	**Std. Error**	**Lower Bound**	**Upper Bound**
Search	19688	6471	1222	17083	22293
Select-Only	3324	936	186	2926	3721

Table 7.6 Mean task completion time and confidence interval data for each task. Time is in milliseconds.

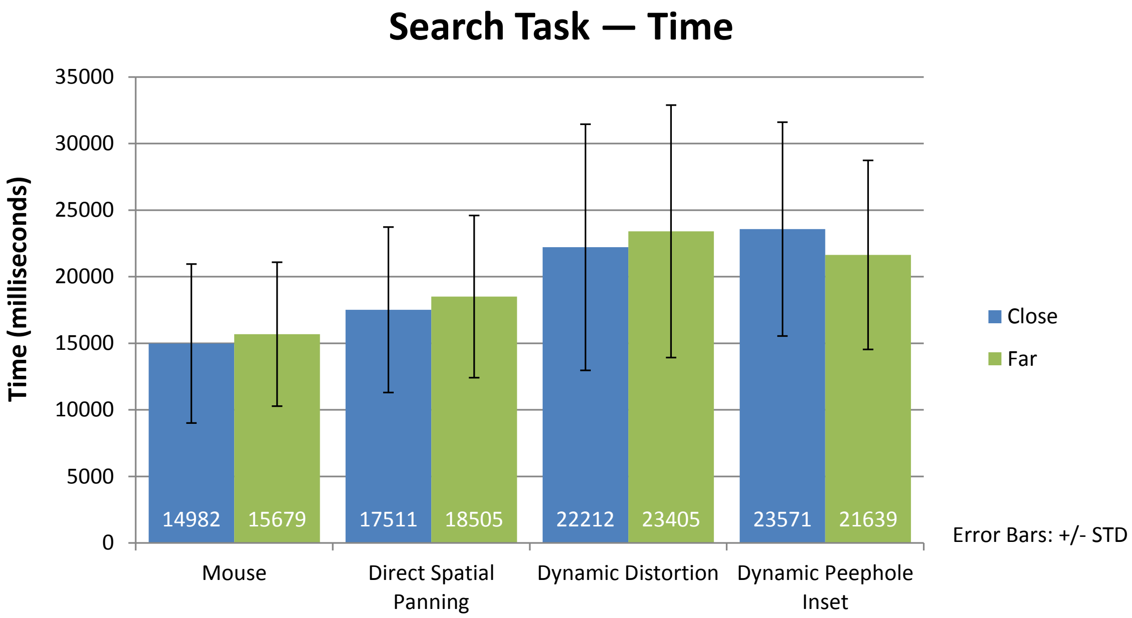


Fig. 7.8 Participant mean task completion time in milliseconds for the search task with the different techniques in relation to the close and far radial distance ranges. Error bars show the standard deviation.

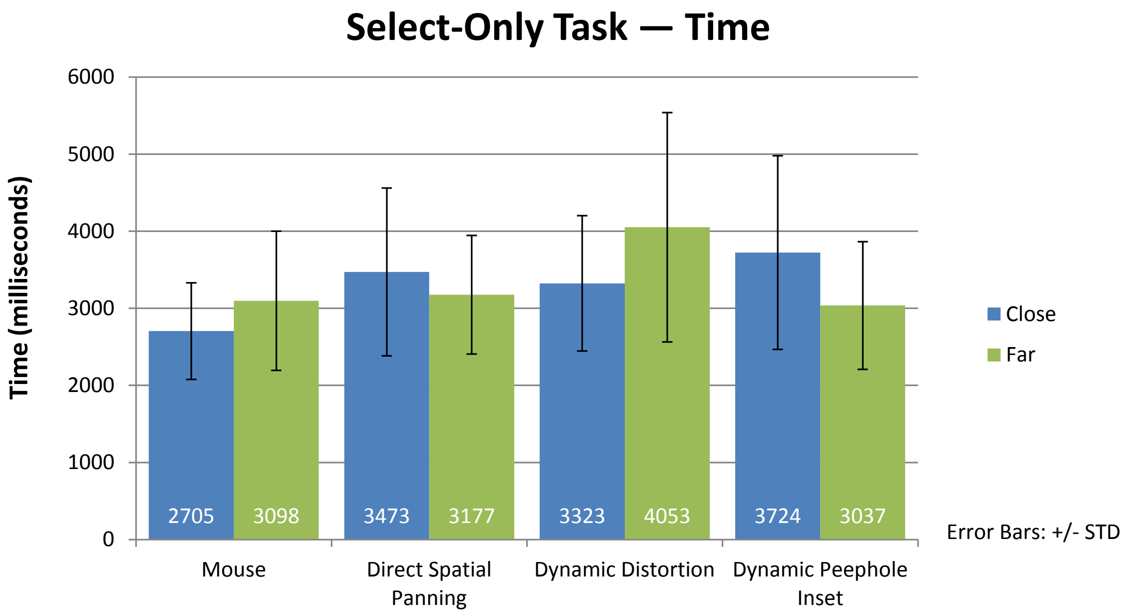


Fig. 7.9 Participant mean task completion time in milliseconds for the select-only task with the different techniques in relation to the close and far radial distance ranges. Error bars show the standard deviation.

t-tests for Technique Time in Search Task

Mouse	Direct Spatial Panning	*t(15) = -2.777*	*p = .085*
	Dynamic Distortion	*t(15) = -5.610*	***p < .0001***
	Dynamic Peephole Inset	*t(15) = -5.157*	***p < .001***
Direct Spatial Panning	Dynamic Distortion	*t(15) = -4.112*	***p < .01***
	Dynamic Peephole Inset	*t(15) = -3.395*	***p < .05***
Dynamic Distortion	Dynamic Peephole Inset	*t(15) = 0.167*	*p > .99*

Table 7.7 Post-hoc paired samples t-tests for technique time in search task with Bonferroni correction. Boldface results are statistically significant. p-values greater than 0.99 are due to Bonferroni correction.

					95% Confidence Interval	
Technique	Task	Mean	Std. Dev.	Std. Error	Lower Bound	Upper Bound
Mouse	Search	15331	4510	1127	12928	17734
	Select-Only	2901	709	177	2523	3279
Direct Spatial Panning	Search	18008	5220	1305	15227	20790
	Select-Only	3325	771	193	2914	3735
Dynamic Distortion	Search	22809	6785	1696	19193	26424
	Select-Only	3688	1103	276	3100	4276
Dynamic Peephole Inset	Search	22605	6285	1571	19256	25954
	Select-Only	3380	1017	254	2839	3922

Table 7.8 Mean task completion time and confidence interval data for each technique in each task. Time is in milliseconds.

					95% Confidence Interval	
Technique	Distance	Mean	Std. Dev.	Std. Error	Lower Bound	Upper Bound
Mouse	Close	8844	3298	736	7274	10413
	Far	9388	3157	708	7879	10898
Direct Spatial Panning	Close	10492	3653	832	8719	12265
	Far	10840	3429	764	9212	12469
Dynamic Distortion	Close	12768	5061	1175	10263	15272
	Far	13729	5482	1262	11040	16418
Dynamic Peephole Inset	Close	13647	4642	1043	11423	15871
	Far	12338	3966	948	10318	14358

Table 7.9 Mean task completion time and confidence interval data for each technique in each radial distance range. Time is in milliseconds.

					95% Confidence Interval	
Task	Distance	Mean	Std. Dev.	Std. Error	Lower Bound	Upper Bound
Search	Close	19569	8102	1362	16666	22472
	Far	19807	7622	1241	17162	22452
Select-Only	Close	3306	1039	193	2896	3717
	Far	3341	1095	203	2908	3774

Table 7.10 Mean task completion time and confidence interval data for each task in each radial distance range. Time is in milliseconds.

Technique	Task	Distance	Mean	Std. Dev.	Std. Error	95% Confidence Interval	
						Lower Bound	Upper Bound
Mouse	Search	Close	14982	5970	1492	11801	18163
		Far	15679	5411	1353	12796	18563
	Select-Only	Close	2705	627	157	2371	3039
		Far	3098	903	226	2616	3579
Direct Spatial Panning	Search	Close	17511	6217	1554	14199	20824
		Far	18505	6089	1522	15260	21749
	Select-Only	Close	3473	1090	272	2892	4053
		Far	3177	770	192	2766	3587
Dynamic Distortion	Search	Close	22212	9244	2311	17286	27138
		Far	23405	9476	2369	18356	28455
	Select-Only	Close	3323	878	220	2855	3791
		Far	4053	1488	372	3260	4845
Dynamic Peephole Inset	Search	Close	23571	8029	2007	19292	27849
		Far	21639	7103	1776	17854	25424
	Select-Only	Close	3724	1256	314	3055	4393
		Far	3037	829	207	2595	3478

Table 7.11 Mean task completion time and confidence interval data for each technique in each task in each radial distance range. Time is in milliseconds.

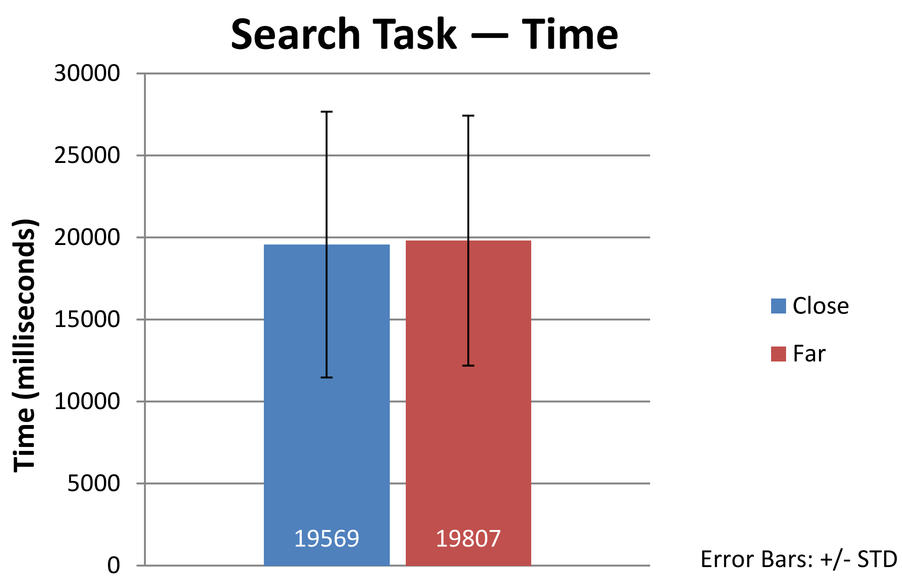


Fig. 7.10 Participant mean task completion time in milliseconds for the search task in the close and far radial distance ranges. Error bars show the standard deviation.

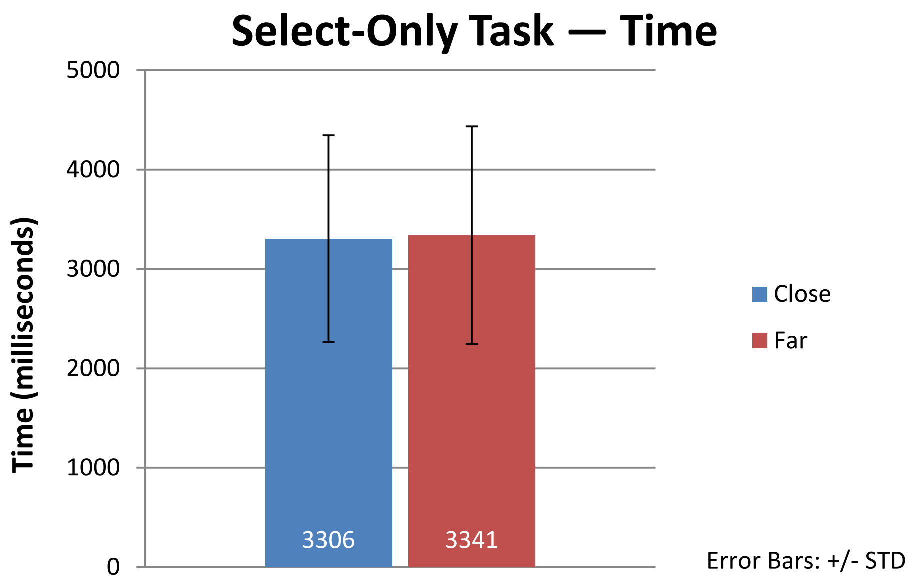


Fig. 7.11 Participant mean task completion time in milliseconds for the select-only task in the close and far radial distance ranges. Error bars show the standard deviation.

t-tests for Technique Time in Select-Only Task

Mouse	Direct Spatial Panning	$t(15) = -3.128$	$\boldsymbol{p < .05}$
	Dynamic Distortion	$t(15) = -3.525$	$\boldsymbol{p < .05}$
	Dynamic Peephole Inset	$t(15) = -2.384$	$p = .18$
Direct Spatial Panning	Dynamic Distortion	$t(15) = -1.536$	$p = .87$
	Dynamic Peephole Inset	$t(15) = -0.278$	$p > .99$
Dynamic Distortion	Dynamic Peephole Inset	$t(15) = 1.111$	$p > .99$

Table 7.12 Post-hoc paired samples t-tests for technique time in select-only task with Bonferroni correction. Boldface results are statistically significant. p-values greater than 0.99 are due to Bonferroni correction.

Similarly, *Direct Spatial Panning* was significantly faster than the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques. For comparing the different techniques in the select-only task, post-hoc paired samples t-tests showed that participants were significantly faster in the select-only task when using the mouse technique compared to the *Direct Spatial Panning* and *Dynamic Distortion* techniques (see Table 7.12). Mean and confidence interval data for each technique in each task can be found in Table 7.8.

Accuracy

In the select-only task where the location of the correct off-screen object was visualized to the participants, the accuracy of all participants in all trials with each technique and distance range was 100%. This pattern was not found in the accuracy data for the search task. Therefore, to determine what type of statistical test could be performed on the accuracy data gathered from the search task in each distance range, we performed three different tests of normality. These included Shapiro–Wilk tests, as well as calculating skewness and kurtosis values, and determining if they lie within the acceptable range of -1.96 to +1.96. All three tests indicated that all of the different factor combinations were not well-modeled by the normal distribution; therefore we could not analyze our data using parametric statistical methods. Thus, we employed a non-parametric statistical test called the Friedman test to determine if there was a significant difference between groups of factors. For post-hoc analysis and for comparing individual factors, another non-parametric statistical test was used, called the Wilcoxon signed-rank test.

For the different techniques (4), the Friedman test showed no statistically significant difference between them in terms of accuracy levels for the search task, $\chi^2(3) = 2.392, p = .495$. No statistically significant difference between the two distance ranges for the search task was found as well, $(Z = -1.434, p = .151)$. We also looked a whether the accuracy levels of individual techniques were affected by the two radial distance ranges (see Table 7.13). A Wilcoxon signed-rank test showed that participants were significantly more accurate with the mouse technique when targets were closer $(M = 0.9792, SD = 0.083)$ to the centre of the screen than when they were farther away $(M = 0.9062, SD = 0.16)$.

Statistical tests were also used to determine if techniques in the search task had significantly different accuracy levels depending on the distance of the off-screen objects from the centre of the screen. We therefore performed a Friedman test with the accuracy levels of the four different techniques in the *close* radial distance range, which showed a significant difference among them, $\chi^2(3) = 10.500, p < .05$. Although, further post-hoc analysis using Wilcoxon signed-rank tests with Bonferroni correction showed

Wilcoxon Tests for Accuracy of Techniques Between Distance Ranges

Mouse — Close	Mouse — Far	$Z = -2.070$	$\boldsymbol{p < .05}$
Direct Spatial Panning — Close	Direct Spatial Panning — Far	$Z = -1.134$	$p = .257$
Dynamic Distortion — Close	Dynamic Distortion — Far	$Z = -0.272$	$p = .785$
Dynamic Peephole Inset — Close	Dynamic Peephole Inset — Far	$Z = -0.412$	$p = .680$

Table 7.13 Wilcoxon signed-rank tests for accuracy of techniques between radial distance ranges for the search task. Boldface results are statistically significant.

Wilcoxon Tests for Accuracy of Techniques in Close Distance Range

Mouse	Direct Spatial Panning	$Z = -1.000$	$p > .99$
	Dynamic Distortion	$Z = -2.121$	$p = .204$
	Dynamic Peephole Inset	$Z = -1.342$	$p > .99$
Direct Spatial Panning	Dynamic Distortion	$Z = -1.890$	$p = .354$
	Dynamic Peephole Inset	$Z = -1.000$	$p > .99$
Dynamic Distortion	Dynamic Peephole Inset	$Z = -1.732$	$p = .498$

Table 7.14 Wilcoxon signed-rank tests with Bonferroni correction for accuracy of techniques in the close radial distance range for the search task. No statistically significant differences were found. p-values greater than 0.99 are due to Bonferroni correction.

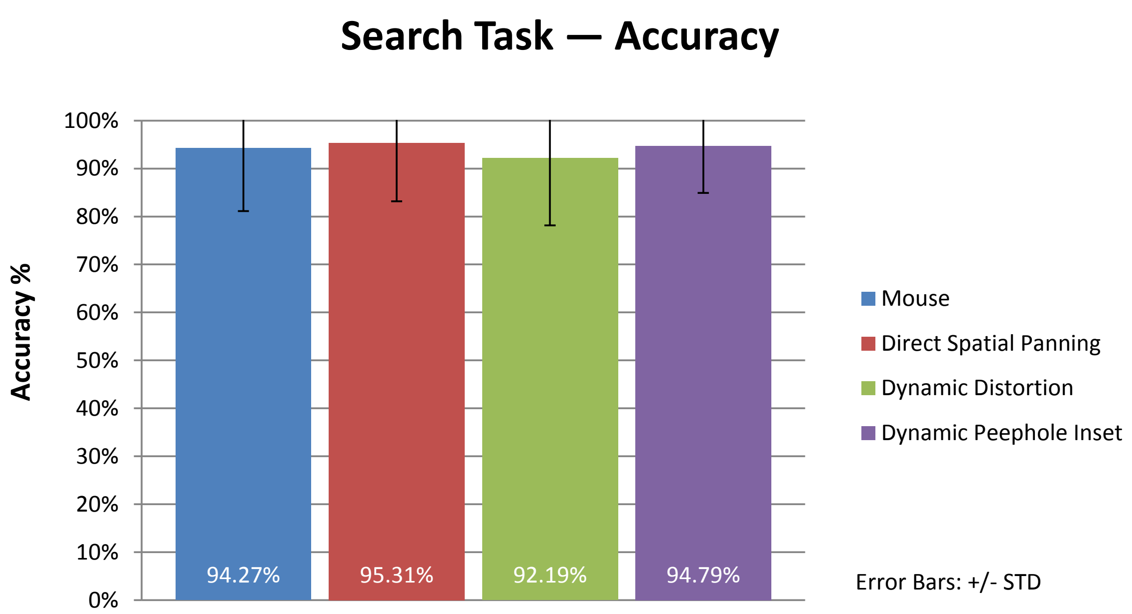


Fig. 7.12 Participant mean accuracy levels in percentages for the search task with the different techniques. Error bars show the standard deviation.

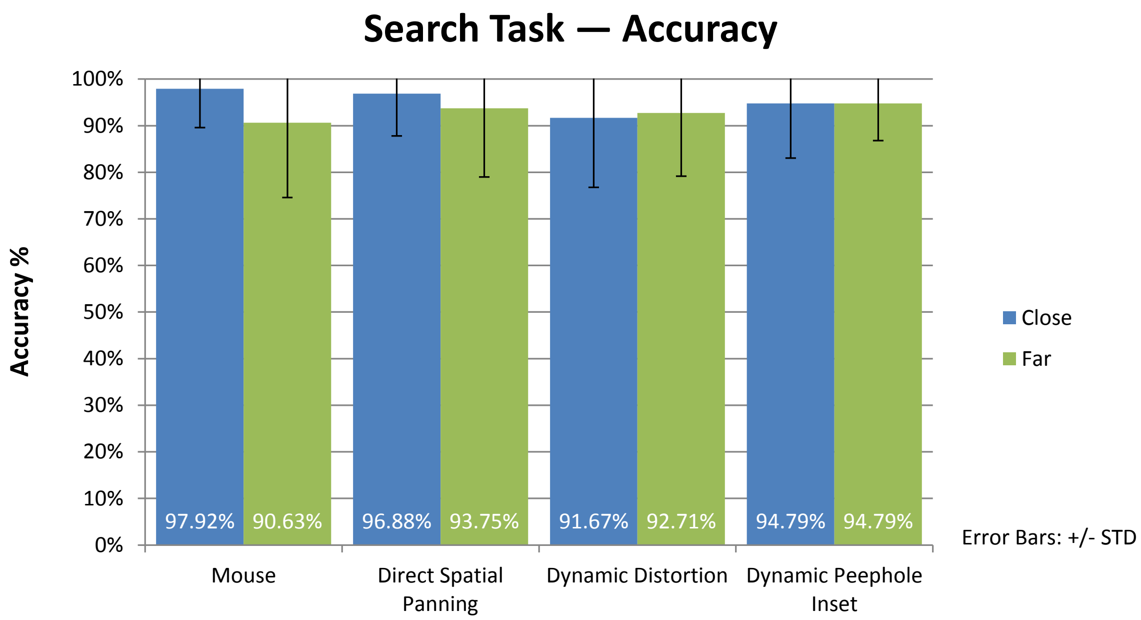


Fig. 7.13 Participant mean accuracy levels in percentages for the search task with the different techniques in relation to the close and far radial distance ranges. Error bars show the standard deviation.

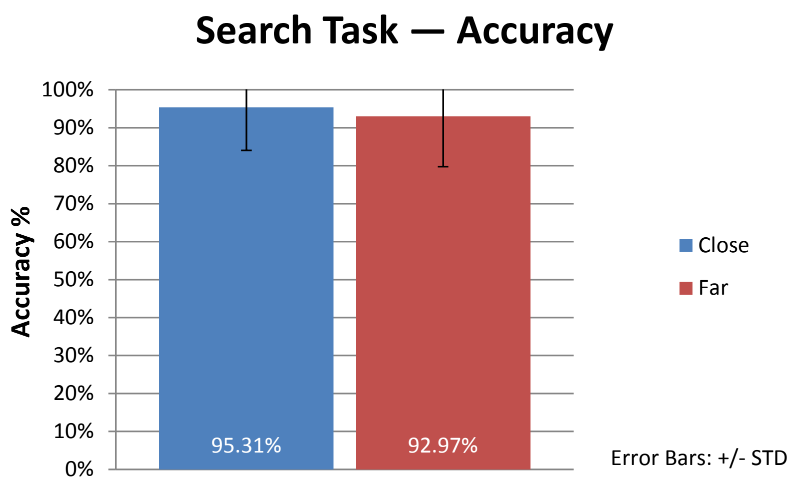


Fig. 7.14 Participant mean accuracy levels in percentages for the search task in the close and far radial distance ranges. Error bars show the standard deviation.

no significant difference between any of the techniques (see Table 7.14). For targets farther away from the screen, a Friedman test of the four different techniques with the *far* radial distance range showed no significant difference among them in relation to accuracy levels, $\chi^2(3) = 1.087, p = .780$.

7.7.2 Qualitative Data

To gather additional data, participants filled out a post-study questionnaire and participated in a semi-structured post-study interview. In the following subsections, we will discuss our findings.

Interview

The semi-structured interview centred on participants' experience in using the different techniques for the two different tasks. For all techniques, participants stated that their strategy for finding off-screen targets was to analyze the on-screen reference object, then explore the surrounding off-screen space. When the reference object was not always situated on-screen, some participants only looked at it again to refresh their memory. Other participants always looked at the reference object before selecting an off-screen object to make sure that they matched. A number of participants stated that they were easily able to memorize what the target looked like; therefore not needing to look at the reference object again after committing it to memory. Most of them stated that they did not search a particular side of the display first on purpose. With relation to the spatial interaction techniques, participants liked how they could envision the physical location that their hand needed to be in to view content at a specific location in the information space.

Most participants stated that they liked the mouse technique due to it being very familiar, and on account of them having lots of experience with this input device type. Participants also found that it supported the most fine-grained control out of all of the techniques. Some participants found the mouse to be slower than the spatial interaction techniques, and did not like how the reference object moved off-screen when exploring the information space. Participants also did not like how the mouse technique required clutching.The *Direct Spatial Panning* was well liked since it was similar to the familiar mouse technique, and participants said that the interaction was easy, fast, natural and fluid. Participants also found this technique to require the least amount of effort to explore the off-screen space. As with the mouse technique, participants did not like how the reference object did not remain on-screen.

For the search task, participants liked how the reference object always stayed on-screen in the *Dynamic Distortion* technique, which helped comparing off-screen objects. The ability to reduce the space between objects and bring a large number of them on-screen simultaneously was liked and made analyzing objects faster for some. Although, some participants found the technique to be difficult due to the continuous distortion effects, "...everything was shrinking and moving.", "Made my brain feel weird." This made analyzing the information space, as well as comparing off-screen objects with the distorted reference object more difficult for some. Some participants also found the technique to be difficult when used to explore and select objects that were in the off-screen corner spaces.

Some participants found that the *Dynamic Peephole Inset* technique was the easiest and fastest technique when used for the select-only task. They stated that they were able to use the arrow visualization to judge the distance of the target from the screen's edge, and immediately "jump" to that location by positioning their hand in the respective physical space. Although, for the search task, participants did not like how their view of the off-screen space was limited and did not provide enough context due to the small size of the viewport. Participants found that this made it harder to find targets since it required them to physically move their hand in all of the off-screen areas to fully explore that space. Some participants stated that this also made it more difficult to determine the off-screen location of the content within the viewport. Although, some participants found that searching for targets was easier due to the viewport reducing the size of information space that needs to be analyzed at each moment in time; thus facilitating concentration. Participants also liked how the reference object always stayed on-screen and was never distorted. An interesting note is that two participants mistakenly thought that the content within the viewport was displayed at a different zoom level than the on-screen content.

Questionnaire

Using a 7-point Likert scale from *Strongly Disagree* to *Strongly Agree*, the questionnaire asked participants questions related to the usability of the different techniques. These included how "easy" and "fast" the different techniques were in each task for finding off-screen targets, and overall how "enjoyable" they were to use. It also asked participants to rank the different techniques in order of preference for finding targets, as well as overall preference. Table 7.15 and Figure 7.15 show the actual questions along with their mean and standard deviation, as well as overall percentages. The questions refer to *Dynamic Distortion* as "distort", *Dynamic Peephole Inset* as "viewport", and *Direct*

Questions	Total Agree	Total Disagree	Mean	STD
Overall, the mouse technique was enjoyable.	10	2	5.38	1.63
Overall, the viewport technique was enjoyable.	11	4	5.00	1.75
Overall, the pan technique was enjoyable.	13	1	5.69	1.30
Overall, the distortion technique was enjoyable.	9	6	4.50	2.07
Finding targets was fast with the mouse technique in the select-only task.	16	0	6.44	0.81
Finding targets was fast with the viewport technique in the select-only task.	13	3	5.75	1.84
Finding targets was fast with the pan technique in the select-only task.	14	0	6.31	1.08
Finding targets was fast with the distort technique in the select-only task.	13	2	5.69	1.58
Finding targets was fast with the mouse technique in the search task.	10	2	5.13	1.63
Finding targets was fast with the viewport technique in the search task.	10	5	4.38	1.86
Finding targets was fast with the pan technique in the search task.	10	4	4.63	1.63
Finding targets was fast with the distort technique in the search task.	7	7	4.25	1.77
Finding targets was easy with the mouse technique in the select-only task.	15	1	6.13	1.20
Finding targets was easy with the viewport technique in the select-only task.	13	3	5.75	1.84
Finding targets was easy with the pan technique in the select-only task.	14	2	6.13	1.54
Finding targets was easy with the distort technique in the select-only task.	13	0	5.88	1.20
Finding targets was easy with the mouse technique in the search task.	13	1	5.88	1.50
Finding targets was easy with the viewport technique in the search task.	9	4	4.69	1.89
Finding targets was easy with the pan technique in the search task.	13	1	5.63	1.20
Finding targets was easy with the distortion technique in the search task.	9	3	4.69	1.40

Table 7.15 Post-study questionnaire data using a 7-point Likert scale. Responses from *Somewhat Agree* to *Strongly Agree* are used to calculate the *Total Agree* score. The same pattern is used to calculate the *Total Disagree* score. The questions refer to *Dynamic Distortion* as “distort”, *Dynamic Peephole Inset* as “viewport”, and *Direct Spatial Panning* as “pan”.

Spatial Panning as “pan”. Table 7.16 and Table 7.17 show the technique ranking results.

7.7.3 Logged Interactions

To gain insight into how our off-screen exploration techniques were employed for searching and moving around the information space, we visualized participants’ movement and positional data in the off-screen space. To accomplish this, we utilized our SpatialVis system, as well as built an ad-hoc Java application to create heatmaps and path visualizations. Using these systems, we were able to view aggregated data over all participants, all techniques, and all trials, as well as data only for each technique, each participant or each trial.

We also looked at which off-screen interaction space (left or right side) participants first used when looking for targets in the search task, as well as how often they changed sides. For all of our off-screen exploration techniques (36 study trials per participant),

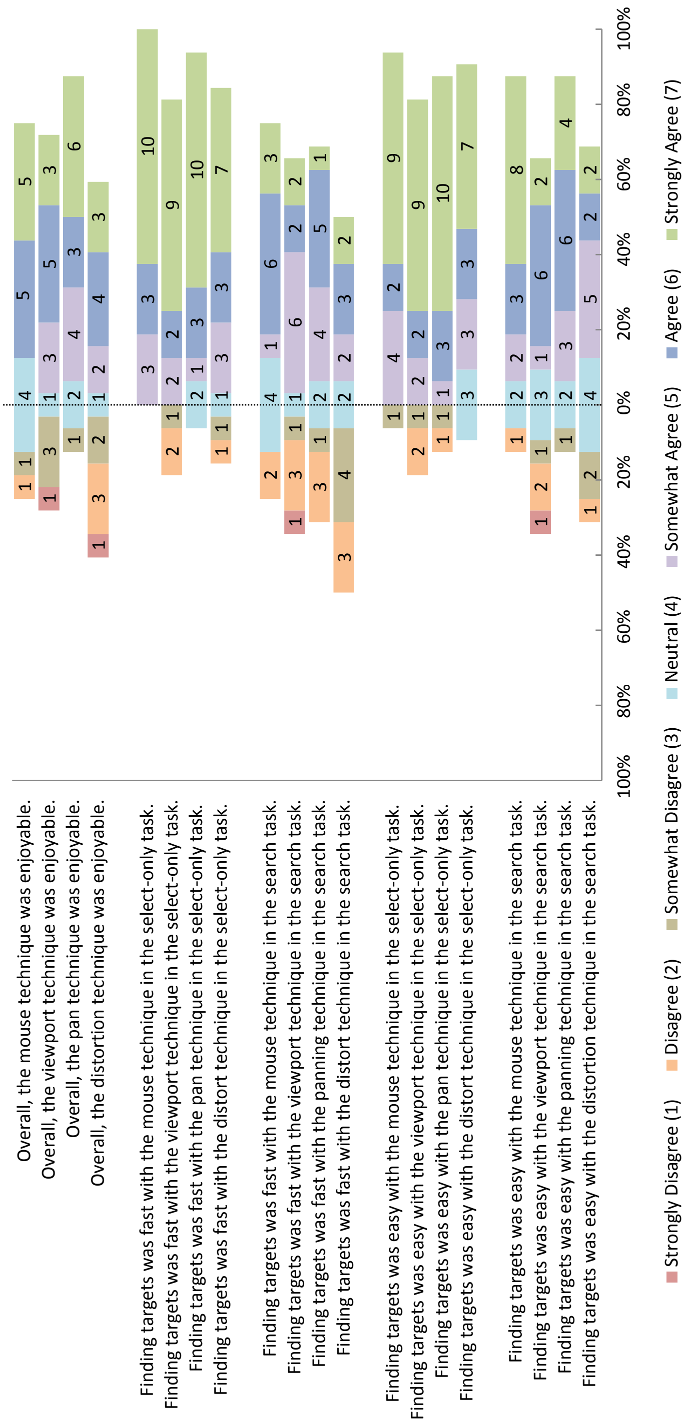


Fig. 7.15 This figure shows participants' answers to the 7-point Likert scale questions in our post-study questionnaire. The number within a bar represents the number of participants who gave that response and its width represents the percentage of participants. The questions refer to *Dynamic Distortion* as "distort", *Dynamic Peephole Inset* as "viewport", and *Direct Spatial Panning* as "pan".

Ranking of Techniques for Finding Targets

Technique	**Mean**	**STD**	**Rank — 4**	**Rank — 3**	**Rank — 2**	**Rank — 1**
Mouse	2.88	1.15	7	2	5	2
Direct Spatial Panning	2.81	0.75	3	7	6	0
Dynamic Distortion	2.25	1.24	3	5	1	7
Dynamic Peephole Inset	2.06	1.18	3	2	4	7

Table 7.16 Participants ranking of the techniques in order of preference for finding targets from 1 (least preferred) to 4 (most preferred).

Ranking of Techniques for Overall Preference

Technique	**Mean**	**STD**	**Rank — 4**	**Rank — 3**	**Rank — 2**	**Rank — 1**
Direct Spatial Panning	2.81	0.91	4	6	5	1
Mouse	2.69	0.95	4	4	7	1
Dynamic Peephole Inset	2.38	1.36	5	3	1	7
Dynamic Distortion	2.13	1.20	3	3	3	7

Table 7.17 Participants ranking of the techniques in overall preference from 1 (least preferred) to 4 (most preferred).

participants searched the left side first on average 17 times and searched the right side first on average 18 times. For those same 36 study trials, participants switched sides on average 23 times, which is an average of 0.65 times per trial. For *Direct Spatial Panning* (12 study trials), participants searched the left side first on average 6.1 times and searched the right side first on average 5.9 times. For those same 12 study trials, participants switched sides on average 8 times, which is an average of 0.67 times per trial. For *Dynamic Distortion* (12 study trials), participants searched the left side first on average 5.7 times and searched the right side first on average 6.3 times. For those same 12 study trials, participants switched sides on average 8.2 times, which is an average of 0.68 times per trial. For *Dynamic Peephole Inset* (12 study trials), participants searched the left side first on average 5.4 times and searched the right side first on average 6.6 times. For those same 12 study trials, participants switched sides on average 7.1 times, which is an average of 0.59 times per trial.

Heatmaps

The following heatmaps visualize the position of participants' palm centre with respect to the information space. We were able to accomplish this due to our off-screen exploration techniques employing a direct 1:1 mapping from the physical space around the screen to the actual information space. The first set of figures displays information gathered from the search task, while the second set displays information from the

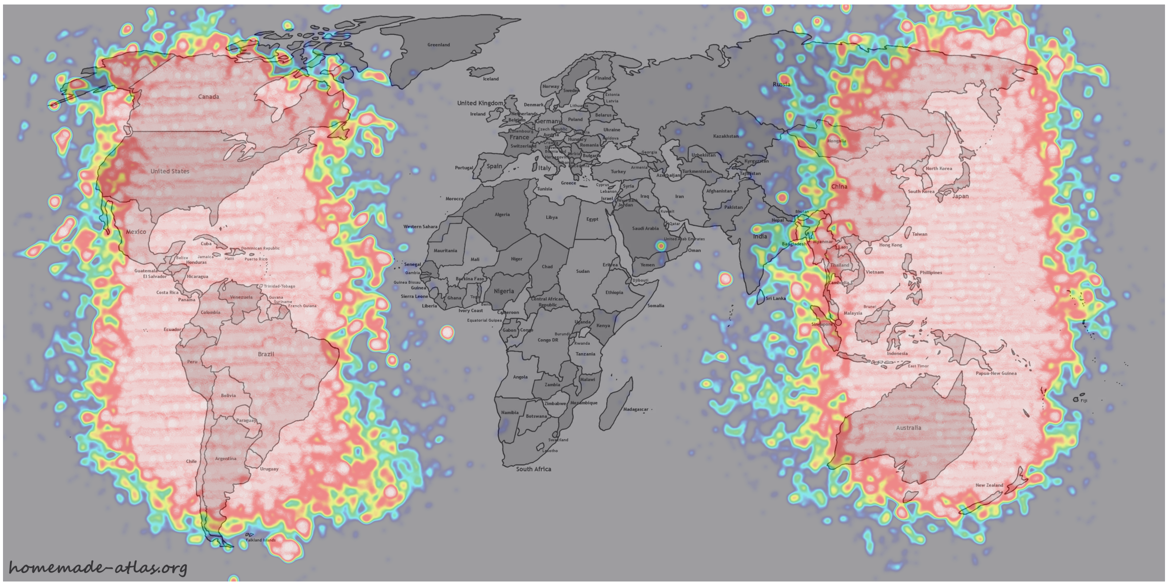


Fig. 7.16 Heatmap showing the position of all participants' hands while searching the off-screen space with all off-screen exploration techniques.

select-only task. We employed a colour scheme of blue to red to white with white areas containing the largest amount of positional data.

Figure 7.16 shows the position of all participants' hands while searching the off-screen space with all off-screen exploration techniques. Similarly, Figure 7.17, Figure 7.18, and Figure 7.19 show the same data, but only for the *Dynamic Distortion* technique, the *Direct Spatial Panning* technique, and the *Dynamic Peephole Inset* technique respectively. Figure 7.20 shows the position of all participants' hands in the off-screen space with all off-screen exploration techniques while performing the select-only task. Similarly, Figure 7.21, Figure 7.22, and Figure 7.23 show the same data, but only for the *Dynamic Distortion* technique, the *Direct Spatial Panning* technique, and the *Dynamic Peephole Inset* technique respectively.

Path Visualizations

To help us understand how participants moved around the information space while using our off-screen exploration techniques, we visualized hand movement data using semi-transparent paths. As with the heatmaps, we were able to accomplish this due to our off-screen exploration techniques employing a direct 1:1 mapping from the physical space around the screen to the actual information space. The colours orange, blue, and purple were used to visualize the *Dynamic Distortion*, *Direct Spatial Panning*, *Dynamic Peephole Inset* techniques respectively. Figure 7.24 shows the movement of all

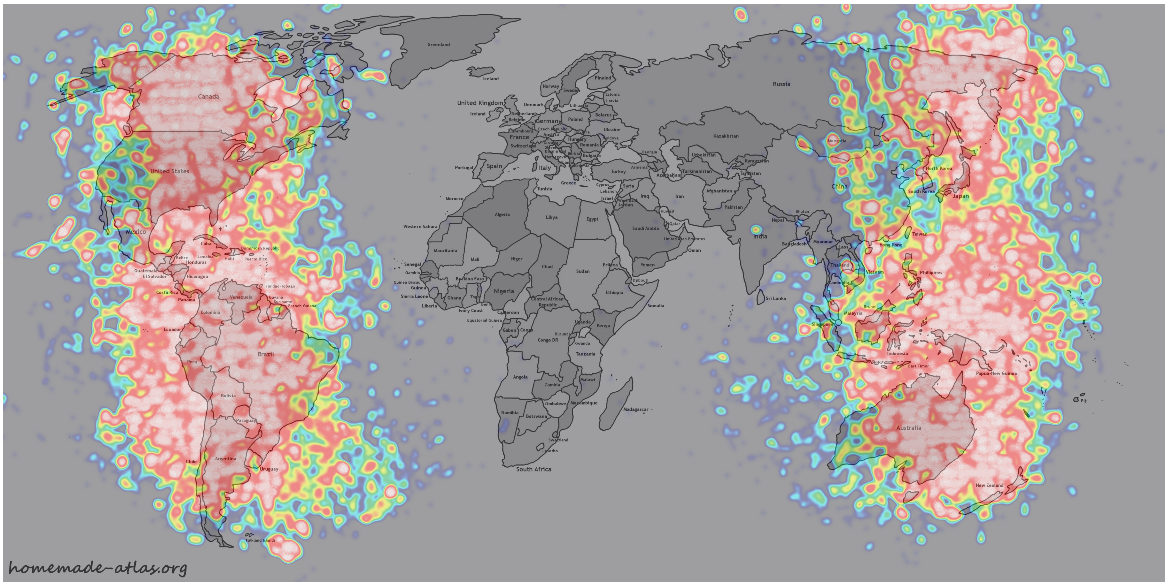


Fig. 7.17 Heatmap showing the position of all participants' hands while searching the off-screen space with the *Dynamic Distortion* technique.

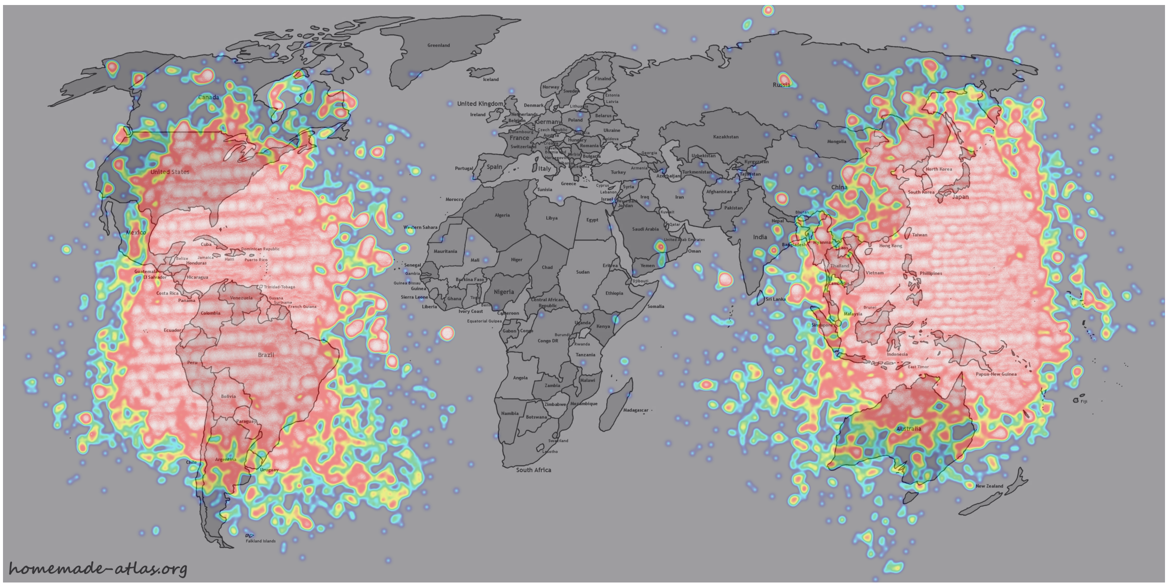


Fig. 7.18 Heatmap showing the position of all participants' hands while searching the off-screen space with the *Direct Spatial Panning* technique.

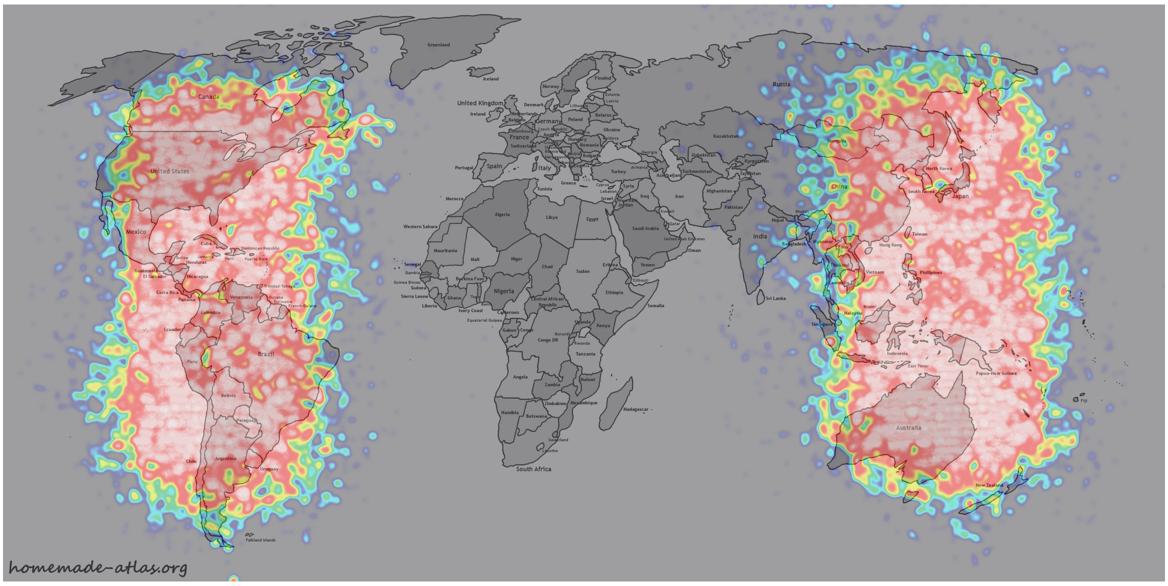


Fig. 7.19 Heatmap showing the position of all participants' hands while searching the off-screen space with the *Dynamic Peephole Inset* technique.

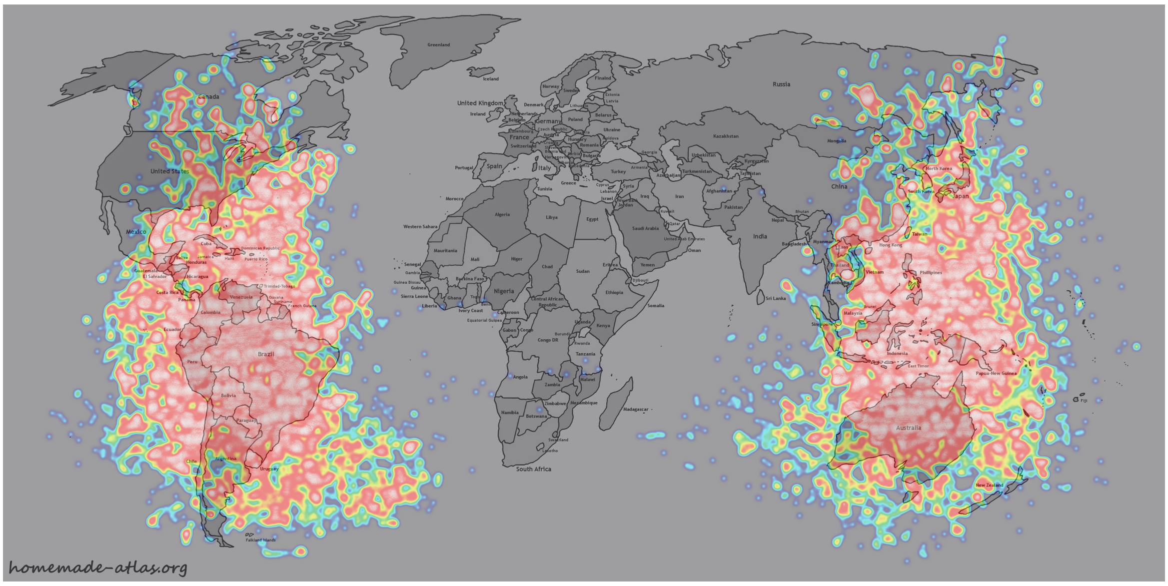


Fig. 7.20 Heatmap showing the position of all participants' hands in the off-screen space with all off-screen exploration techniques while performing the select-only task.

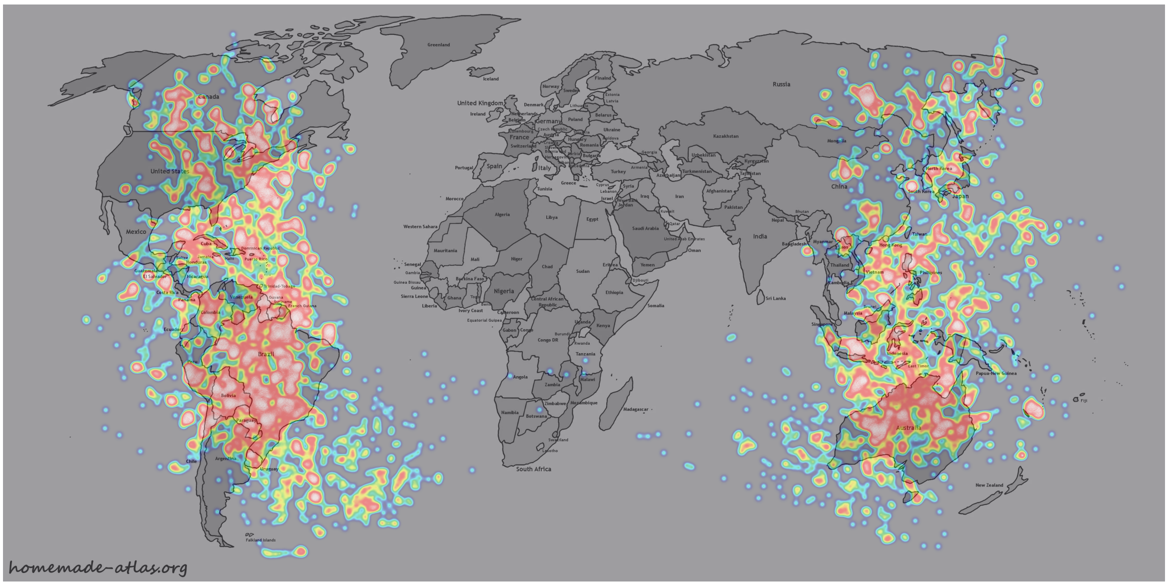


Fig. 7.21 Heatmap showing the position of all participants' hands in the off-screen space with the *Dynamic Distortion* technique while performing the select-only task.

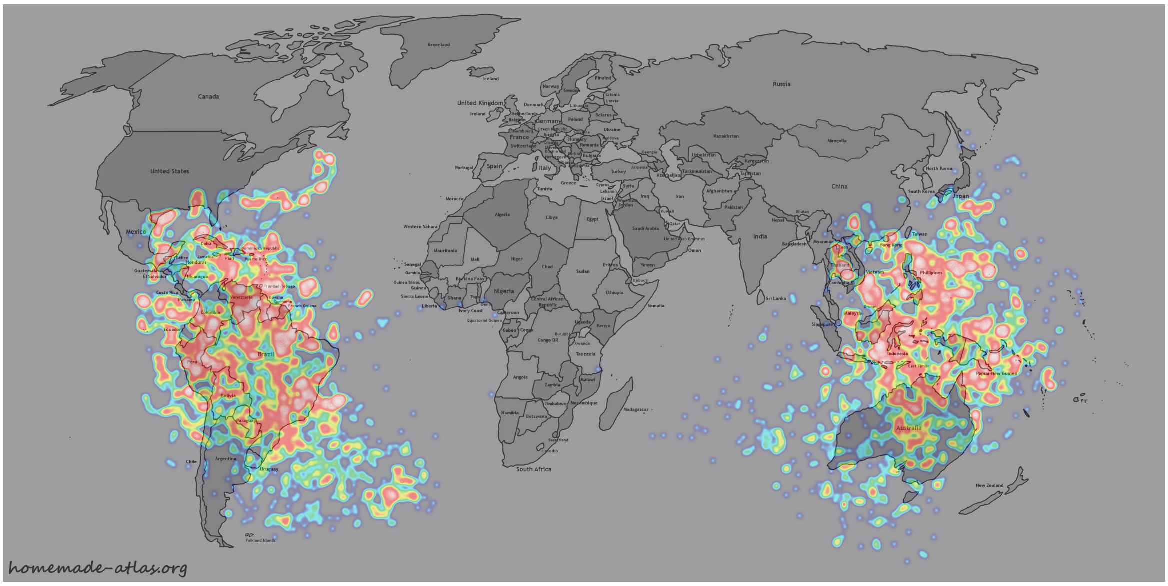


Fig. 7.22 Heatmap showing the position of all participants' hands in the off-screen space with the *Direct Spatial Panning* technique while performing the select-only task.

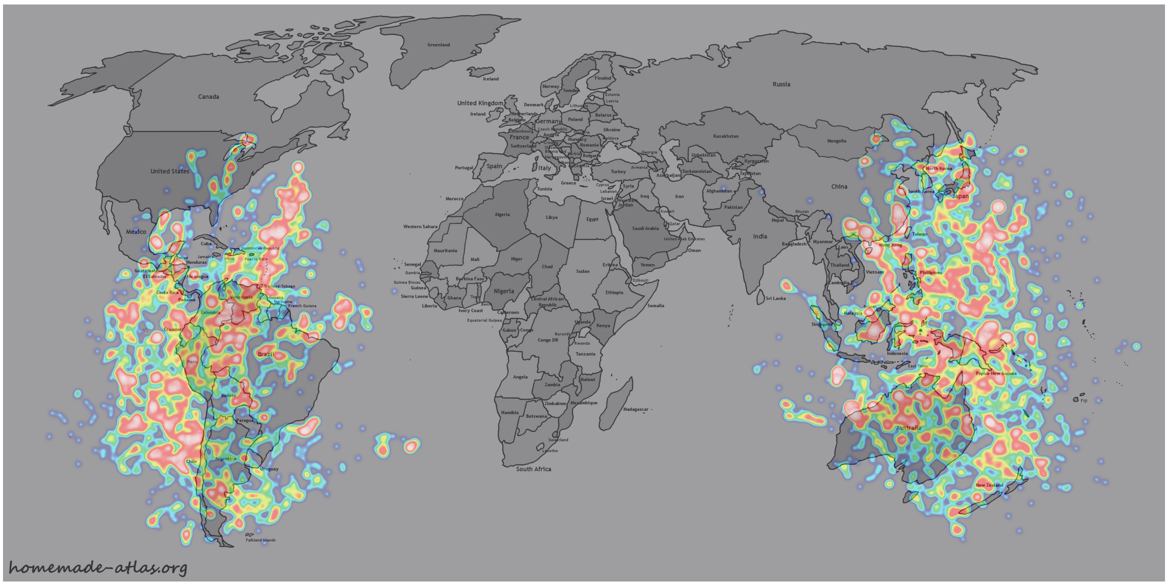


Fig. 7.23 Heatmap showing the position of all participants' hands in the off-screen space with the *Dynamic Peephole Inset* technique while performing the select-only task.

participants' hands while searching the off-screen space with all off-screen exploration techniques. Similarly, Figure 7.25, Figure 7.26, and Figure 7.27 show the same data, but only for the *Dynamic Distortion* technique, the *Direct Spatial Panning* technique, and the *Dynamic Peephole Inset* technique respectively.

Figure 7.28 shows the movement of all participants' hands in the off-screen space with all off-screen exploration techniques while performing the select-only task. Similarly, Figure 7.29, Figure 7.30, and Figure 7.31 show the same data, but only for the *Dynamic Distortion* technique, the *Direct Spatial Panning* technique, and the *Dynamic Peephole Inset* technique respectively.

7.8 Discussion

In this section, we will discuss the results gathered from our formal study. This discussion is partitioned into subsections centring on the different techniques, radial distance ranges, and logged interactions.

7.8.1 Mouse

As expected, participants had the fastest task completion times for both tasks when using the mouse. Statistical analysis showed that the mouse technique was overall

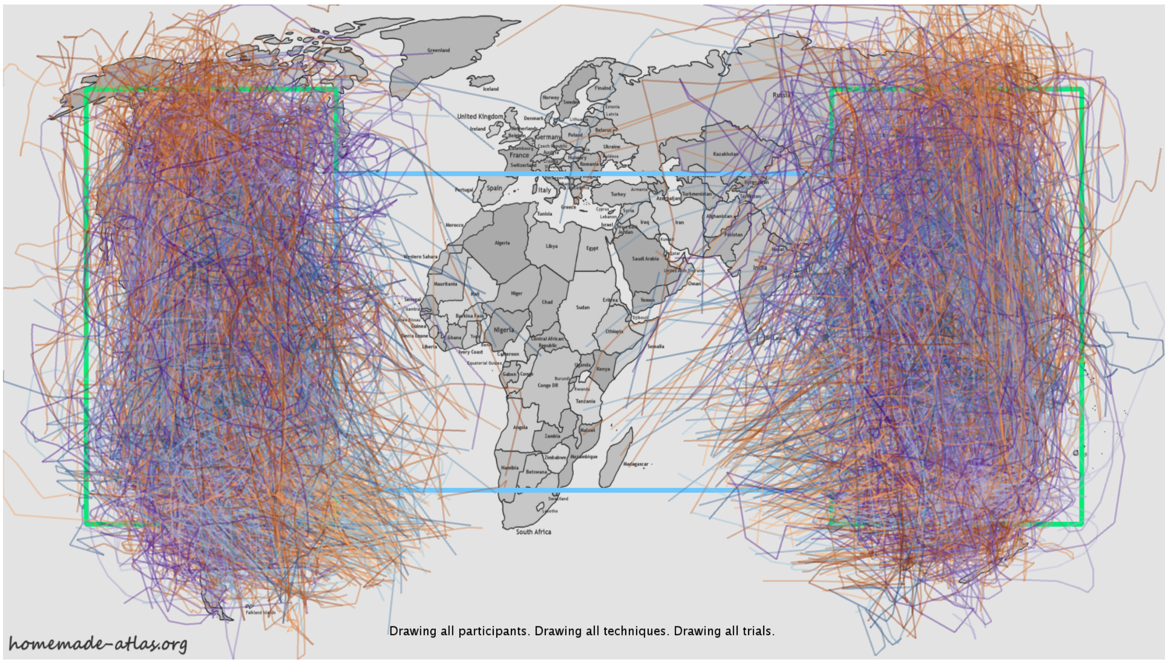


Fig. 7.24 Path visualization showing the movement of all participants' hands while searching the off-screen space with all off-screen exploration techniques.

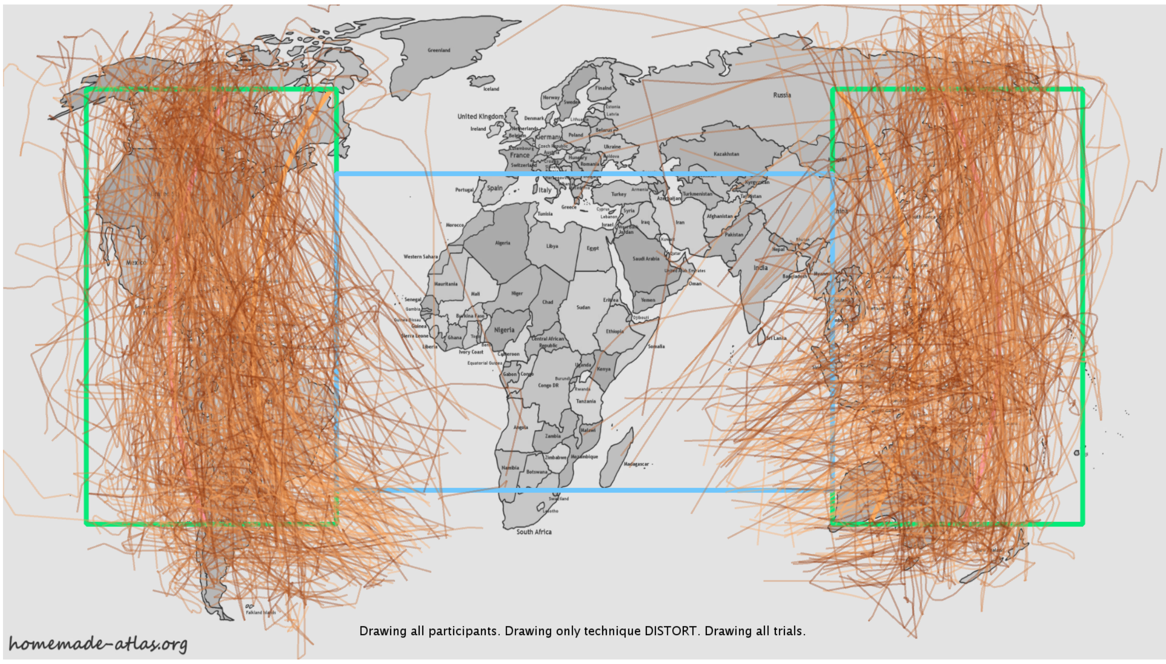


Fig. 7.25 Path visualization showing the movement of all participants' hands while searching the off-screen space with the *Dynamic Distortion* technique.

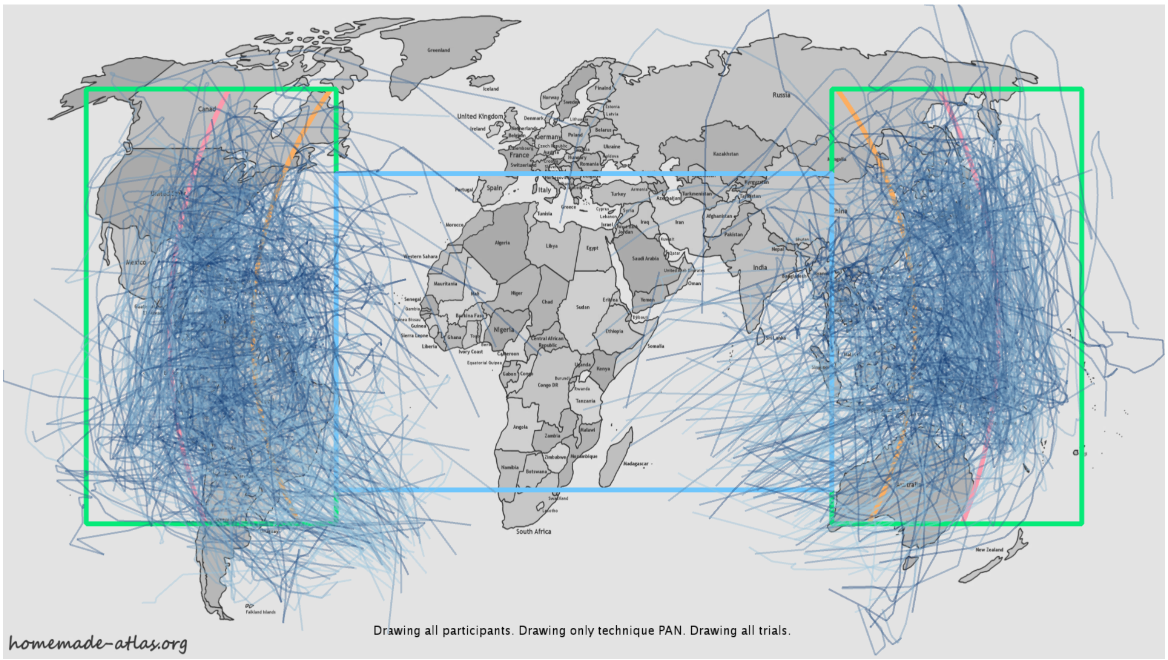


Fig. 7.26 Path visualization showing the movement of all participants' hands while searching the off-screen space with the *Direct Spatial Panning* technique.

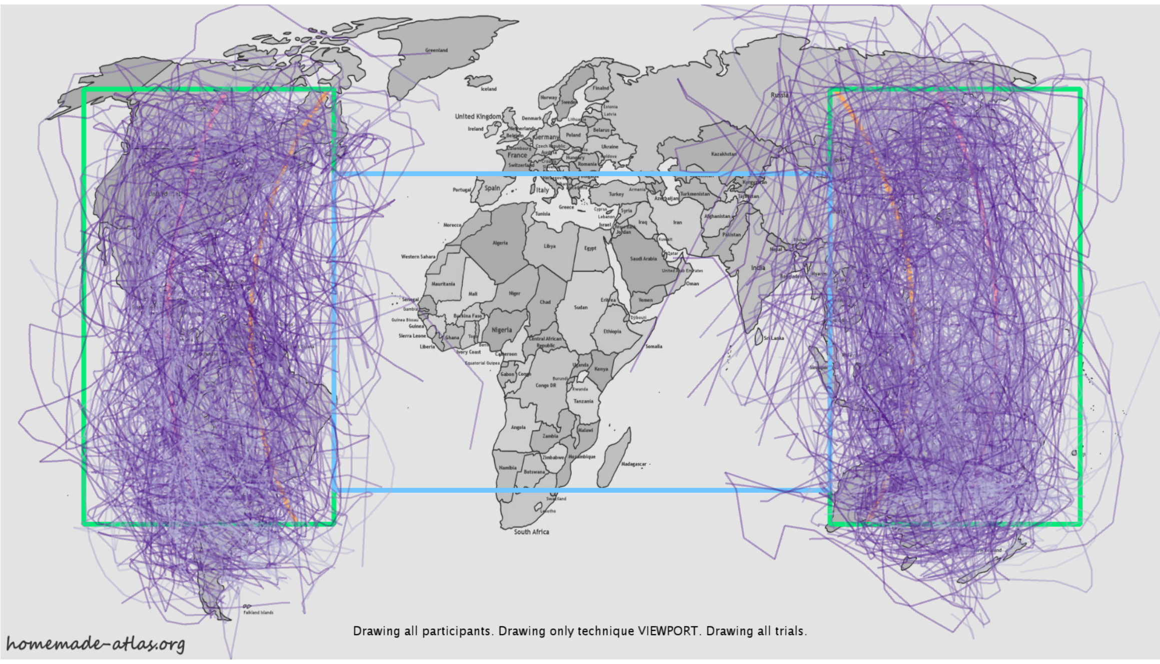


Fig. 7.27 Path visualization showing the movement of all participants' hands while searching the off-screen space with the *Dynamic Peephole Inset* technique.

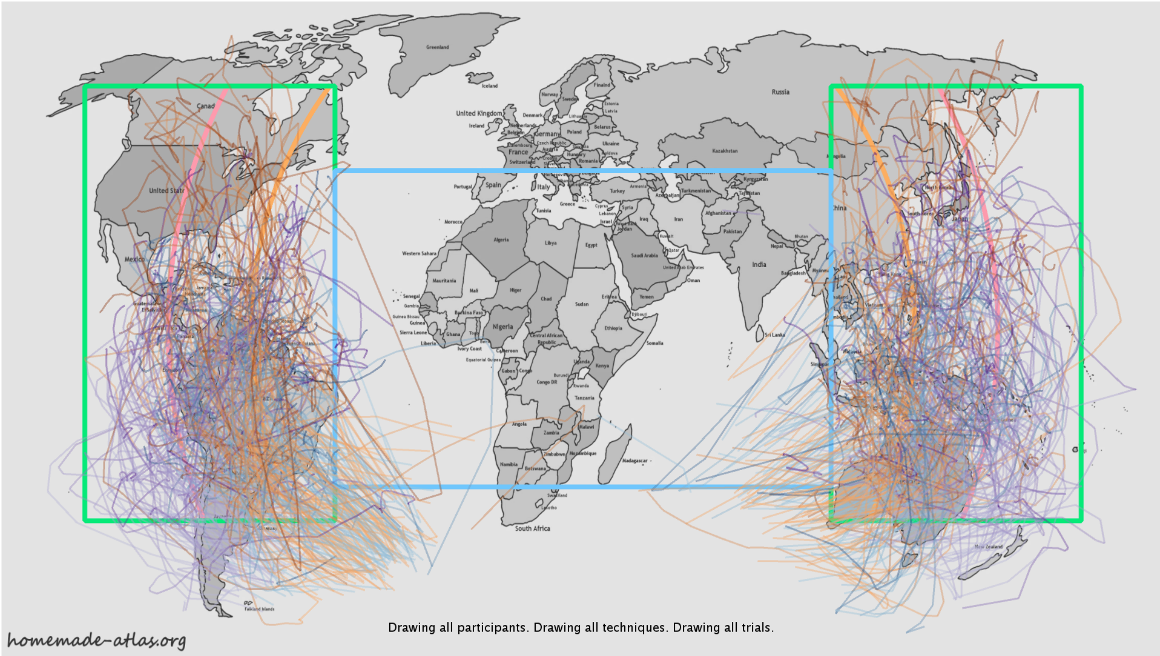


Fig. 7.28 Path visualization showing the movement of all participants' hands in the off-screen space with all off-screen exploration techniques while performing the select-only task.

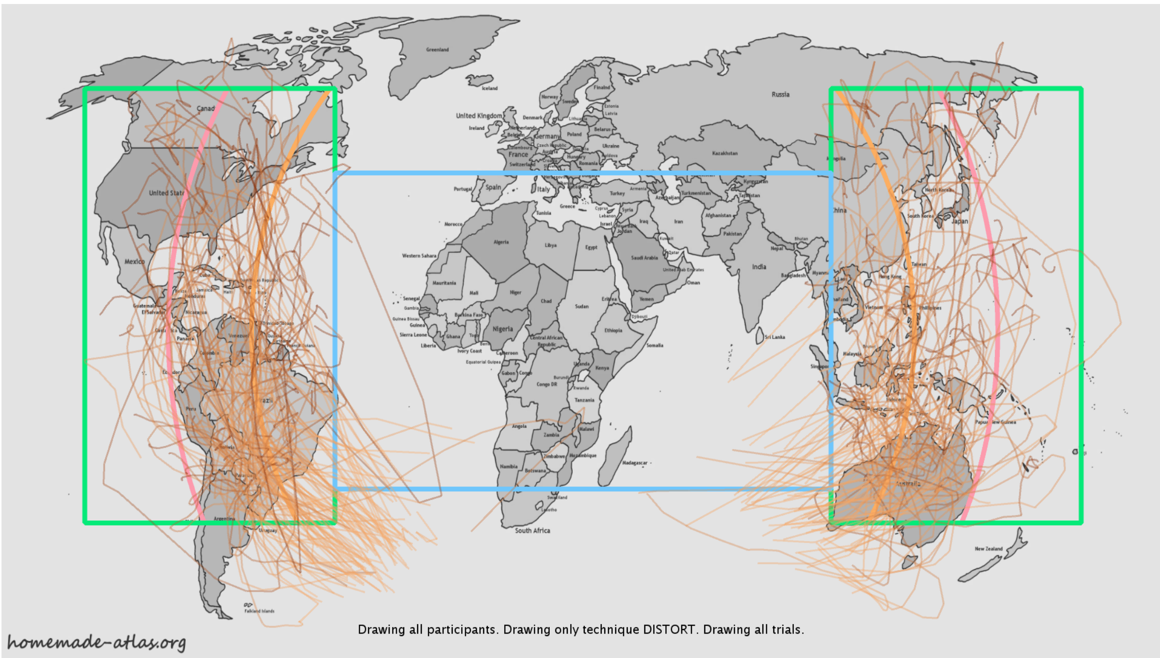


Fig. 7.29 Path visualization showing the movement of all participants' hands in the off-screen space with the *Dynamic Distortion* technique while performing the select-only task.

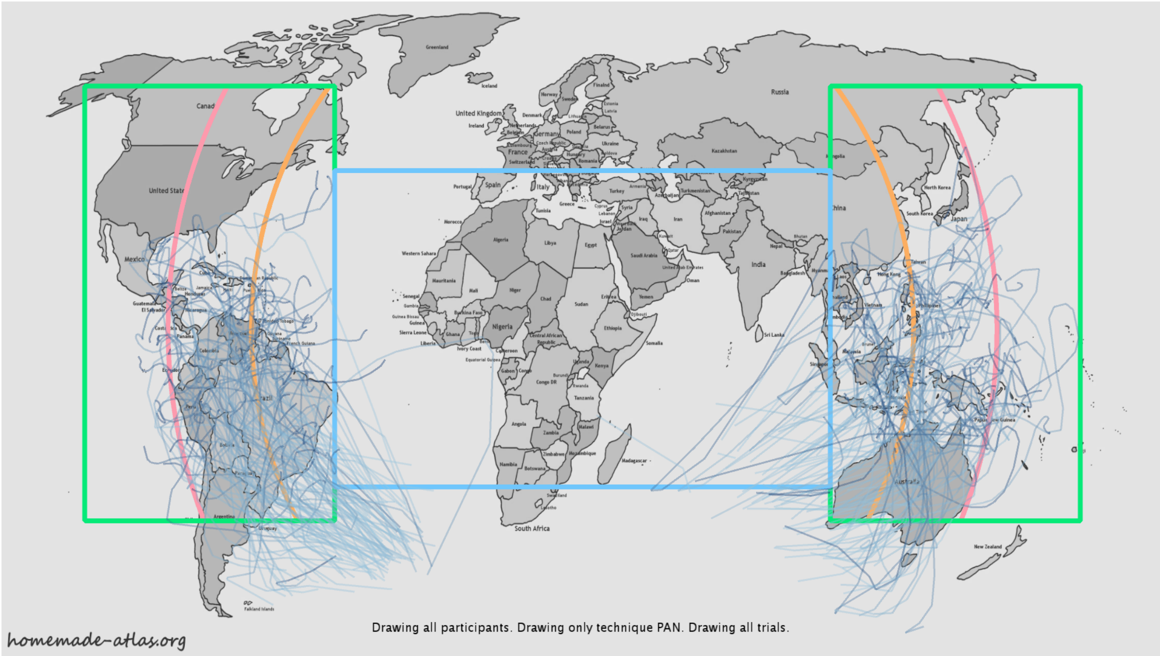


Fig. 7.30 Path visualization showing the movement of all participants' hands in the off-screen space with the *Direct Spatial Panning* technique while performing the select-only task.

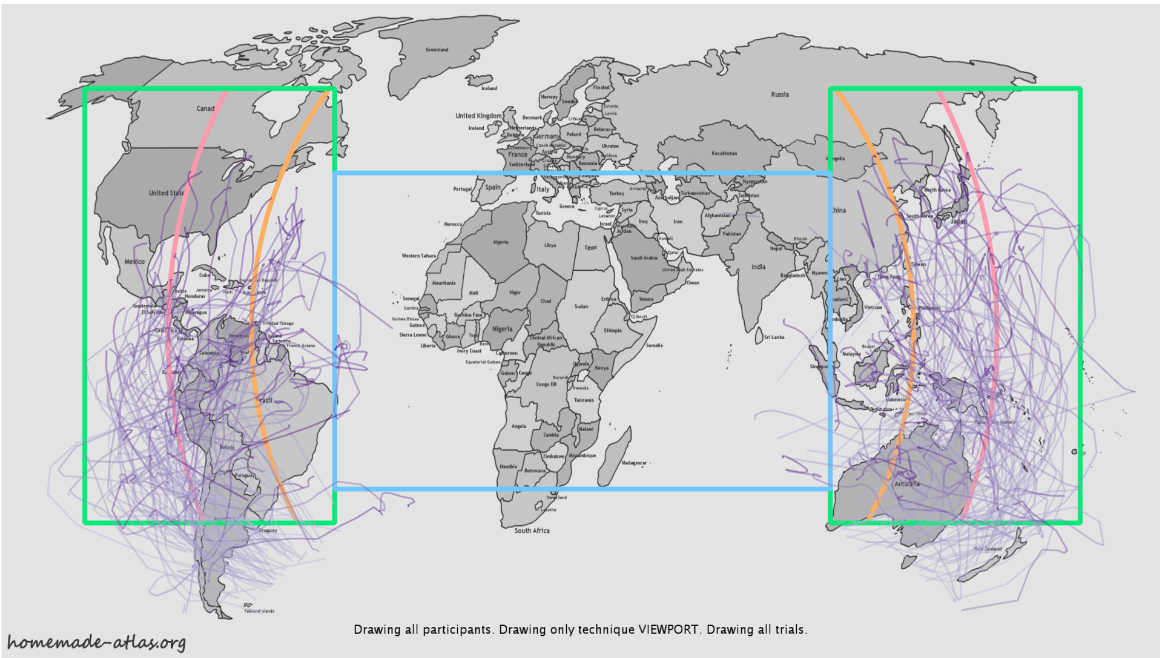


Fig. 7.31 Path visualization showing the movement of all participants' hands in the off-screen space with the *Dynamic Peephole Inset* technique while performing the select-only task.

significantly faster than all of the other techniques, was significantly faster than *Dynamic Distortion* and *Dynamic Peephole Inset* in the search task, and was significantly faster than *Dynamic Distortion* and *Direct Spatial Panning* in the select-only task. We can therefore accept our hypothesis **H1**, which states that out of all four techniques, the mouse will have the fastest task completion times for both tasks. The mouse was also the most preferred technique by participants for finding the correct off-screen object, and received the second highest score in terms overall preference and overall enjoyability. For each task type, the mouse received the highest mean scale rating with respect to its perceived speed and usability ("easiness"). As shown by our hypothesis **H1**, we expected participants to perform the best with the mouse due to it and its associated panning technique being widely used and due to most, if not all, computer users having extensive experience with them. Our participant demographic data illustrates this where 81% of participants indicated that they used a computer mouse every day. In our experience, most people are not used to spatially interacting with their computers. People have experience in performing coarse mid-air gestures (e.g., waving, pointing, etc.), but generally not for controlling anything that requires fined-grained control, such as a computer cursor. For the motion sensing devices listed, participant demographic data showed that 91% of participants had only used devices that support spatial interactions a few times in their lives. For the Leap Motion controller, 81% of participants had never used one before the study and the rest only had a few experiences. Therefore, it was as if we were comparing the performance of a technology (mouse) using expert users, against novice users of another technology (spatial interaction). It takes time to get used to the interactions involved in moving something precisely around in 2D by using a 3D space without force feedback. We also believe that the mouse technique performed the best since it is an indirect pointing device with an interaction gain, and the distance from the keyboard (i.e., trial start position for hands) to the mouse was smaller than the distance to the spatial interaction space (i.e., beside the display). Participants also experienced less fatigue when using the mouse since their hand and wrist could rest on the desk, and only needed to make small movements to explore the entire information space (gain of the indirect interaction) when compared to the spatial techniques.

Although, similar to what has happened with the mouse, we believe that as people gain more experience in using spatial interaction and as the motion sensing technology continues to advance, we will see an increase in people's overall spatial interaction performance and preference. In terms of the future for desktop computers, the mouse will possibly, and most likely, remain the main pointing device, but the inclusion of spatial interaction into systems as an complementary interaction modality will further help to decrease the barrier between human and machine. This is especially true for those with physical disabilities that cannot make use of the mouse. The point that we are trying to make is that an interaction technique, or technology in general for that matter, does not need to perform better than the standard for it to be useful and worthwhile. For example, people are notoriously slower and more error-prone when typing with touch-based virtual keyboards compared to physical keyboards, yet they are accepted and widely used every day. With our spatial off-screen exploration techniques, we are putting forth a vision of new ways to interact with computers, which will hopefully in turn inspire others to improve upon its design until it becomes highly useful in the correct contexts [57].

7.8.2 Spatial Off-Screen Exploration Techniques

The performance results of our spatial off-screen exploration techniques are positive as they show that our techniques perform almost as well as the mouse with very little training. *Direct Spatial Panning* had the highest accuracy levels out of all of the techniques in the search task, and was the fastest spatial technique in both tasks. Statistical analysis found that it was overall significantly faster than the other two spatial techniques, as well as for the search task. Participants also found the *Direct Spatial Panning* technique to be the most enjoyable, and ranked it as the most preferred technique overall. In terms of perceived usability ("easiness") for finding targets, this technique received the highest mean scale rating when used for the select-only task, and the second highest for the search task. For perceived speed in relation to finding targets in both tasks, it also received the second highest mean scale rating. These results indicate that *Direct Spatial Panning* was the best spatial off-screen exploration technique for the study tasks. This is not surprising since the technique is very fluid and similar to the mouse technique. It is therefore the easiest spatial technique for novices to transition to from the standard off-screen interaction technique (mouse) in terms of the cognitive processes involved.

Dynamic Distortion and *Dynamic Peephole Inset* were not found to be significantly faster or more accurate than any other technique. In both tasks, *Dynamic Distortion* had the slowest task completion times with *Dynamic Peephole Inset* being the second slowest technique by doing slightly better. Overall, participants enjoyed the *Dynamic Distortion* technique the least, ranked it last for overall preference and second last for preference in relation to finding targets. For the same qualitative metrics, *Dynamic Peephole Inset*'s scores were second last out of the four techniques, except for participant preference in relation to finding targets where it was ranked last. Participants also perceived the *Dynamic Distortion* technique as being the slowest for finding targets in both tasks, with *Dynamic Peephole Inset* coming in second last. For perceived usability ("easiness") for finding targets, both techniques tied for last place in the search task, with *Dynamic Distortion* beating *Dynamic Peephole Inset* for third place in the select-only task. These results are heavily influenced by task appropriateness, as we believe that *Dynamic Distortion* and *Dynamic Peephole Inset* would be more beneficial in tasks that require glancing at off-screen content, such as looking ahead in a book, or bringing social media content (e.g., a Twitter feed) temporarily onscreen. These two spatial techniques would also be beneficial when the task requires viewing off-screen content and on-screen content simultaneously, such as during the comparison of highly complex information.

The research investigator who observed all of the study sessions noticed that participants tended to have difficulty with the *Dynamic Distortion* technique when trying to select objects in the off-screen corner spaces. This was also explicitly stated by participants in the post-study interview, and included the corner spaces above the display and below the display. Participants could not see off-screen content in those areas until they physically placed their hands in that space, which then caused the information space to become vertically distorted. Unfortunately, the difficulty was caused by the Leap Motion controller being less robust in terms of spatial recognition in the areas close to its cameras as well as vertically far away. We mitigated this problem by reducing the off-screen space where targets could potentially be placed as depicted by the green rectangles in Figure 7.2. Although, the problem still occurred if the user's hand orientation was not well recognized by the Leap Motion controller. Even though the research investigator spent time explaining the optimal hand orientations, as well as the orientations that cannot be recognized by the motion sensing device, a number of participants held their hands in sub-optimal orientations throughout the study. It is important to note that the small stature and arm length of some of the participants made it more difficult for them to optimally place their hand in the space above the display. Therefore, after running the study, the research investigator expected the *Dynamic Distortion* technique to fair the worst, which is unfortunate since the main problems that caused its downfall were not caused by the actual technique. It is possible that *Dynamic Distortion* might still have fared worse than the mouse and *Direct Spatial Panning* techniques even if our system did not experience any recognition problems. Another problem with the Leap Motion controller is that due to the biomechanical properties of the human arm, staying within the motion sensor's field of view is more difficult when making large vertical movements. People tend to pivot at the elbow which causes the hand to move along the Z and Y axes; therefore increasing the possibility of the hand moving outside of the spatial interaction space.

In terms of technique accuracy performance in the search task, no statistically significant difference was found. We therefore must reject **H2**, which states that the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques will have the highest accuracy levels out of all of the techniques in the search task. **H3**, which states that *Dynamic Peephole Inset* will have a higher accuracy level than the *Dynamic Distortion* technique in the search task, must be rejected as well. Although not statistically significant, participants were the least accurate with *Dynamic Distortion* in the search task. This is understandable since some participants stated in the post-study interview that the distortion effects caused by this technique increased the difficulty of visually comparing objects. This was made exacerbated when the reference object was distorted differently than other objects of interest. Also, the entire reference object or some of its important features were easy enough for a number of participants to remember; therefore reducing the need for it to remain on-screen. If they did forget, participants could just bring the reference object back on-screen to refresh their memory. Therefore, situations with a higher degree of object complexity, with a higher amount of similarity between different objects, or with dynamic or time-critical content should be investigated to determine when techniques that enable the retainment of on-screen information are most valuable.

We believe that the performance of our spatial techniques was negatively affected by user fatigue, and how the techniques required one to make larger physical movements than the mouse (indirect pointing device with an interaction gain). Some participants did experience fatigue due to the repetitive nature of the study, and the fact that the spatial techniques required one to make a large movement at the start of each trial. This large movement involved moving from the keyboard to the space beside the display. Fatigue was exacerbated since spatial techniques require one to hold and move their hand in mid-air without any support.

7.8.3 Target Radial Distance Ranges: Close and Far

The radial distance ranges did not play a major factor in relation to accuracy in the search task, except when the mouse technique was used. Participants were significantly more accurate with the mouse when targets were closer to the screen. A similar non significant pattern occurred when all techniques were aggregated in the search task, as well as when comparing the distances with respect to the *Direct Spatial Panning* technique. A possible explanation for the mouse being significantly more accurate with closer targets is that participants might have been more likely to bring the reference object back on-screen when less effort was required. Doing so would allow participants to double check if the object of interest was in fact the correct target. Surprisingly, the *Dynamic Distortion* technique was slightly less accurate when targets were closer to the screen. This could be due to the fact that some participants tended to place their hand farther away from the display in the off-screen corner space at first when searching with this technique. This strategy, which was observed by the research investigator, allowed participants to gain a larger overview of the off-screen space. This could have resulted in the lower accuracy level since closer objects have a higher probability of becoming distorted when one's hand is placed in the off-screen corner space, with this probability increasing as the distance between the hand and the display gets larger.

In terms of task completion time, no significant difference was found between the two radial distance ranges for both task types. This is possibly due to the off-screen space not being large enough for differences in distance to significantly affect performance. Future work should investigate how a larger off-screen space and bigger differences between distance ranges affect off-screen interaction. Participants were slightly faster in both tasks when targets were closer to the screen. This makes sense, especially when the location of the target is known, since a person has to travel a smaller distance in the information space to reach a closer target. Although, interestingly, participants were faster with the *Direct Spatial Panning* and *Dynamic Peephole Inset* techniques when targets were farther away from the screen in the select-only task. Some participants stated in the post-study interview that they were able to use the arrow visualization to judge the distance of the target from the screen's edge, and immediately "jump" to that location. Therefore, this interesting finding might be due to participants overestimating the distance between the target and the side of the display.

In the search task, participants were faster with all of the techniques when targets were closer, except for when the *Dynamic Peephole Inset* was used. Based on our experience of personally using these techniques, as well as observing other people, the discrepancy might be due to the fact that people tend to not place their hand very close to the display when first exploring the off-screen space. Doing so requires more mental effort since one must avoid hitting the actual side of the display. When coupled with the limited view and context of the off-screen space that is provided by the *Dynamic Peephole Inset* technique, closer targets might have had a higher probability of being missed by the participants. To mitigate this, designers can include an overview + detail visualization with an additional viewfinder that represents the location of the off-screen content that is shown by the technique. People could then use this to determine the content's exact position in the information space and make sure that they explore the entire surrounding area.

7.8.4 Logged Interactions

With respect to using the spatial interaction techniques in the search task, participants were found to switch sides on average less than one time per trial. They also started searching the off-screen space pretty evenly with participants searching the right side first a tiny bit more than the left side, even though all of our participants were right handed. This is surprising since we thought participants would start searching for targets, for the most part, by first using their dominant hand and the off-screen space closest to it, whether consciously or unconsciously. Due to the repetitive nature of the study and the fatigue that occurs when repetitively moving one's hand in mid-air, the resulting evenness is possibly due to participants attempting to distribute the physical effort between both arms.

When comparing the heatmaps and path visualizations for both tasks, it makes sense that participants traversed less of the off-screen interaction space in the select-only task. This is due to the fact that the arrow visually indicated the section of the off-screen space that contained the target, whereas participants had no help when exploring for targets in the search task. Therefore, participants were more likely to fully explore the off-screen space in the search task. The heatmaps and path visualizations of the different spatial techniques give an insight into how the different techniques work. The location and movement of participants' hands when using the *Dynamic Peephole Inset* and *Dynamic Distortion* techniques show how one must position their hand in the actual physical off-screen space to view the virtual content that is mapped to that location. This is due to the *Dynamic Peephole Inset*'s small viewport size, and how the *Dynamic Distortion* technique only vertically distorts the information space when one's hand is above or below the display. With the *Direct Spatial Panning* technique, users can take advantage of the fact that this technique translates the entire information space to bring the off-screen content that is associated with the hand's physical location to the centre of the screen. This results in a large section of the information space being brought on-screen whenever a person moves their hand in the off-screen interaction space. Therefore, to see content that resides above the display, this technique does not require the user to physically place their hand in that exact location. Users can move their hand in the off-screen space until the content of interest appears at the edge of the on-screen space. This results in the *Dynamic Peephole Inset* and *Dynamic Distortion* techniques requiring the user to move their hand around the screen more than the *Direct Spatial Panning* technique to explore the same amount of off-screen information. One can view this by comparing the heatmap and path visualizations of the *Direct Spatial Panning* technique with the visualizations of the *Dynamic Peephole Inset* and *Dynamic Distortion* techniques. They shows how less of the off-screen information space was physically traversed when participants used the *Direct Spatial Panning* technique.

Chapter 8

Limitations

There are a number of limitations pertaining to our work. Our Off-Screen Desktop system does not support the interaction areas above or below the display. Due to this, our formal study only examined off-screen interaction on the left and right sides of a desktop monitor. The placement of off-screen objects in our evaluation was further constrained within that space due to the limitations of the Leap Motion controller (i.e., poor recognition in certain areas). Also, participants were not able to fully take advantage of our spatial techniques since most had very little experience with spatial interaction in general, and it was not practical to spend enough time training them until they became proficient.

Another limitation of our study is that our population sample size was not large (16 participants), and it did not accurately represent the population of our country (Canada) or city since they were all university students. Thus, our study suffers from a larger sampling error. Within the human-computer interaction community, this is a common problem. The use of null hypothesis significance testing (NHST) is also standard practice within this field. We acknowledge the controversy surrounding the use of NHST when performing experimental research (e.g., [36, 42, 103, 116]), as well as the growing movement to move away from its use and concentrate more on reporting all data, especially effect sizes and confidence intervals [39–41]. Therefore, some might view our use of NHST when analyzing the quantitative study data as a limitation.

Although it is rarely seen within the human-computer interaction literature, non-neutral questions concerning the different study factors should be counterbalanced by asking each question in a positive as well as a negative fashion (e.g., Was this technique easy? & Was this technique hard?). Each pair of positive/negative questions should also be placed a certain distance apart. The reason behind this is that by asking a question in a positive manner for example, the participant is being primed to answer in a more positive way. When asking participants to rate how much they agreed with certain statements, our 7-point Likert scale did not contain positive and negative versions of the same statement. Therefore, participants could have been biased towards giving the different techniques a more positive result. An option to mitigate the problems associated with developing one's own questionnaire is to use a previously created questionnaire that has been validated [104]. Examples of validated questionnaires include the Positive and Negative Affect Schedule [163], the Self-Assessment Manikin (SAM) [18], and the NASA-TLX (Task Load Index) [66].

Chapter 9

Future Work

Our experience with designing different spatial off-screen exploration techniques, as well as a prototype system that implements their instantiation, has allowed us to gain some insight into how these designs, as well as off-screen interaction in general, can be extended.

9.1 Formal Investigations

The formal evaluation presented in our work helps to elucidate the performance characteristics of our spatial interaction techniques. Yet further investigation is required as there is still much to be learned. A benefit of the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques is that the original on-screen content remains on-screen when viewing off-screen content. Since a number of participants explicitly stated that they were able to memorize the reference object, future work should investigate situations where the retainment of on-screen information is more valuable. These include situations with a higher degree of object complexity, with a higher amount of similarity between different objects, or with dynamic or time-critical content. Another idea is to investigate the benefit of spatial memory and proprioception on the memorability of off-screen storage locations. A study could be conducted where participants are required to find off-screen objects that they have manually placed off-screen compared to when the system forces particular item placements. This can be further expanded by comparing the performance when a free off-screen information space is employed to a sectioned and discretized off-screen information space. Another idea is to look at the effects of including different off-screen visualization techniques.

Other work could look at what effect the background has on the usability of an off-screen interaction system. In our evaluation, a world map helped participants to determine which section of the information space they were looking at. Another idea is to conduct a study where different backgrounds are employed, such as a solid colour background, that is essentially blank of distinct features, with a background with unique features. Additionally, the size of the off-screen information space and display size should be examined to see their effect on off-screen interaction. Future work could also investigate the potential effects of changing the number of objects and their size. Specific object placement in the off-screen information space is another idea, where one could look at whether certain sides (e.g., left, right, top, bottom) are better for certain tasks (e.g., storage) and if handedness plays a role.

Due to the recognition problems with the Leap Motion controller and its effects on the use of our spatial techniques, future work should replicate our study with a more robust recognition system and larger interactable off-screen space to enable further exploration in ways to improve off-screen interaction. An additional idea is to compare standard off-screen exploration techniques with our spatial techniques when combined with spatial selection. To determine which spatial selection technique to support, a study could be conducted to find the best selection techniques that fit well with our spatial off-screen exploration techniques. Lastly, touch interaction should be compared to our off-screen exploration techniques. Unlike mouse panning, touch panning is a direct interaction technique; therefore it is a more fair comparison to our direct spatial techniques. Additionally, our techniques seem to complement touch interaction more than mouse/keyboard interaction since the movement from direct touch to spatial interaction is more fluid, as well as easier and quicker. Popular mobile devices support touch interaction, and moving within their around-device space is quick and easy. Therefore, our spatial techniques might possibly be well suited for touch-enabled mobile devices, which should be investigated.

9.2 3D Information Space

Instead of solely working with a 2D information space, future work could extend our techniques and related system to be able to support a 3D information space. Although, designers must first have good reasons for their use of 3D information space that employs spatial interaction since human performance levels are higher in 2D versions [35]. A simple extension of our techniques would employ the hand's 3D position in the space around the display to determine the X, Y and Z coordinates of the interested off-screen content. Furthermore, *Dynamic Distortion* could allow one to distort the Z-axis in addition to the X and Y axes, which would enable users to bring content with different

Z-depth values on-screen simultaneously. Another interesting idea would be to allow the user to change the camera angle that is used to render off-screen content by rotating their hand in mid-air. If supported in the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques, this would allow one to view on-screen content and content that was originally off-screen at different angles simultaneously.

9.3 Supporting Different Device Types

With respect to our off-screen prototype system, it is designed with the desktop and laptop computer in mind, but we believe that every device type would benefit from being able to interact with off-screen content, especially those with smaller screen sizes. Therefore, future versions could focus on supporting other device types, including mobile devices, digital tables and wall displays. The community at large would benefit from knowing how the different device types, with their limitations and affordances, affect the design of off-screen interaction systems. Also, research has found that when interacting in the space around a mobile device, people are concerned about how other people might react [1]. This is unfortunate for designers that employ the use of off-screen interaction since social acceptance has a large influence on whether a technology or product will be used by people. Therefore, future work should investigate if this self-conscious pattern occurs with all device types or is only linked to specific types, as well as ways to improve overall social acceptance.

An interesting idea for an off-screen mobile system is allowing people to interact with off-screen content from different statically situated information spaces. Spatially aware mobile devices use the peephole technique to view and interact with content that is associated with the actual physical location of the device (e.g., [27, 48, 172]). With a large statically situated information space, these devices require the user to constantly reposition their body (e.g., walking) to fully explore and interact with the virtual content. This is due to the fact that the device's screen must be close enough to the user and at an appropriate viewing angle for its on-screen content to be properly analyzed. Therefore, instead of only mapping virtual content to the location of the device, it would be beneficial if the location of a user's hand could also be used to bring content on-screen, as demonstrated by our spatial techniques. Doing so would allow users to explore a larger section of the information space without moving their body or compromising their view of the screen.

Since we see our spatial off-screen exploration techniques as augmenting existing systems and being used in conjunction with other input devices (e.g., mouse, touch), future work should focus on designing ways to accomplish this vision without hindering the usability of the interaction techniques involved.

9.4 Displaying Off-Screen Content

When dealing with off-screen content, the associated information space is inherently invisible to the user until it is brought on-screen. Although, this does not have to be the case. A mobile device could be used as a peephole to view off-screen content when it is physically positioned in the off-screen space, or when its camera points at this space. Instead of only viewing virtual content, the camera could also be used to provide an augmented reality view of the off-screen information space. By rendering the physical world in the peephole along with the digital world, people would have a better understanding of how the off-screen content is mapped to the physical space beside the display. In a similar manner, a head-mounted display could also be used to enable an augmented view of the entire information space. Instead of requiring the user to hold or wear a device, projectors could be used to display off-screen content as seen in the *IllumiRoom* project [97]. Also, a display that is larger than the main screen could be physically placed behind it to gain the same effect. If either type of peripherally situated displays were used to visualize a 3D information space in conjunction with the main display, they could also be used to display off-screen content that is at a different Z-depth than content on the main screen.

Although beneficial for viewing off-screen content in place, the projector and large display techniques might increase the probability of cognitive overload and users becoming distracted since a larger amount of total information is displayed. The same is true for the other techniques, but is less of a problem because users can easily turn the devices off since they are either holding or wearing them. With respect to reducing the problems that occur when rendering large amounts of information with additional peripherally situated displays, we believe eye-gaze is an appropriate interaction modality. Based on user attention and on-demand information, a display supporting this technique would only show virtual content when the user is directly looking at it. Therefore, when a person is directing their attention on something else, they are implying that the information within the peripheral display is not currently needed which causes it to turn off or render a blank screen. This has the effect of potentially reducing energy consumption, as well as providing users with a technique for retaining privacy in relation to their information. Similarly, off-screen content could also only be shown when a person places their hand in an off-screen interaction space.

Either the entire off-screen information space is displayed or only the subsection that is located near the person's hand.

9.5 Interacting With Off-Screen Content

We think that eye-gaze and eyelid gestures (e.g., [99]) could also be used to explore and interact with the off-screen information space. By using one's eyes to communicate intention with the computer, a user's hands would be free to perform some other task. A simple version of this technique would have a person looking at the off-screen location of content that one is interested in, and automatically move it on-screen by performing an eyelid gesture. This technique, as well as our off-screen exploration techniques, all deal with the information space associated with one computing system. Future work could centre on designing techniques to enable interacting with off-screen content from different devices simultaneously. For example, if a peripherally situated display (e.g., projector, larger display behind the main display) had its own separate information space, off-screen exploration techniques coupled with other spatial interactions could be used to move off-screen content between it and the main display. This would be useful when a wall display display, for example, contains important information and a user wants to have access to it as they move around a building. The user could just walk up to the wall display and bring the interested content onto their mobile device's screen or surrounding space. If one is performing a comparison task with respect to off-screen content, an extension of the *Dynamic Distortion* and *Dynamic Peephole Inset* techniques could be employed to simultaneously bring content from two separate information spaces onto the same screen.

With respect to the placement and position of off-screen content, our prototype system does not make use of any binning discretization techniques. Objects can be placed anywhere in the off-screen interaction space and overlap one another. In the work by Hasan et al., the authors implemented three different radial binning techniques that were designed for mobile devices with 2D information spaces [67]. These techniques divided the off-screen space either uniformly, based on the distance from the device, or by using a hysteresis function (fisheye). Future work could focus on adding binning support to our prototype and investigate how it affects the usability of the system. Another idea is to design a wide range of different discretization techniques and determine which ones are more suitable for each device type. Instead of only dividing the off-screen space in a curvilinear fashion, the design of the different techniques should explore employing a rectilinear grid or ones that make use of other shape types (e.g., triangle, other polygon). Furthermore, the design of discretization techniques for 3D information spaces should be investigated as well. Another idea, which relates to Ens et al.'s work [47], is to analyze the user's physical surroundings and employ different real-world surfaces for storage bins.

In relation to our off-screen exploration techniques, our implementation of *Dynamic Distortion* makes use of scaling transformations to distort the information space. Depending on the placement of the user's hand, sections of content is either scaled uniformly or with different values for each axis. Future work could integrate other transformation techniques (e.g., fisheye, surface folding) [29] or other newly designed off-screen exploration techniques, to provide the user with a larger variety of ways to change the presentation of information, as well as ways to explore it.

Chapter 10

Conclusion

The work presented here focuses on designing different spatial interactions that support exploring and interacting with 2D content located outside of the visual feedback space. This has resulted in the creation of a formalized descriptive framework, the designs of different off-screen exploration techniques, the creation of an off-screen interaction system that supports our off-screen techniques, a quantitative evaluation that compared a subset of our techniques with standard mouse interaction, and a web-based system for visualizing spatial interaction data.

We created our formalized descriptive framework of the off-screen interaction space to help us better understand the space and to gain insight into off-screen interaction. Other people can also use it to inform their designs of related off-screen interaction techniques. Based on the position in relation to the display, our framework divided the around-device space into five separate sections including behind, in front, to the left and right, above, and below. It then provided information regarding the benefits and disadvantages of each of these sections in relation to different device types, gestures and content.

Using the insights provided by our exploration of the around-device space, we designed a number of spatial techniques that support exploring 2D off-screen information. We also designed them to facilitate the comparison of off-screen and on-screen content. These techniques include *Paper Distortion*, *Dynamic Distortion*, *Dynamic Peephole Inset*, *Spatial Panning* and *Point2Pan*. They enable off-screen content of interest to be brought on-screen by employing different transformation strategies that either distort or translate a section of the information space. All of their designs were based heavily on the concept of direct manipulation for determining the location of interested off-screen content. One can imagine this as if the information space continued past the boundaries of the display and supported direct-touch interaction. The location of content is indicated by directly touching the mid-air space that would contain the content if the display was large enough. Therefore, off-screen content that is situated above the on-screen content within the information space is simulated to be physically situated above the screen in the real world. This allows our techniques to employ a direct 1:1 and scaled mapping from the physical world to the digital information space. See Table 10.1 for the advantages and disadvantages of our techniques.

To help us refine, extend, and evaluate the designs of these techniques, we developed an off-screen exploration and interaction system called Off-Screen Desktop. Using our descriptive framework as a guide, the system was designed for desktop and laptop computers and supports spatial, mouse, and touch-based interaction. Low-cost motion sensing hardware was used with different implementations of our off-screen exploration techniques to enable off-screen interaction on the left and right sides of the display. Unlike previous research with off-screen interaction (e.g., [67]), we purposely designed our system to be able to be used in the wild without requiring users to buy expensive hardware. This allowed us to gain a better understanding of the challenges involved in creating consumer-level spatial interaction systems (e.g., smaller spatial interaction space, less robust gesture recognition), and different ways to overcome those challenges.

To continue along this path of exploration and learning with respect to spatial interaction, we also developed a web-based visualization system called SpatialVis. This system enables one to gain further insight into how people are using a spatial interaction system and/or different spatial techniques. By uploading log data and an associated screen-capture of the display, SpatialVis uses different techniques to visualize spatial interaction data and how it in turn affects the user interface. It is meant to help people design better techniques and systems, as well as to help researchers analyze study data.

The development of both Off-Screen Desktop and SpatialVis enabled a more substantial exploration of the design space in relation to our spatial techniques since these systems gave us a platform to study our techniques and visually analyze the resulting participant data. Therefore, to see how well our techniques support off-screen interaction, and to determine how we could improve their designs, we evaluated three of them using a quantitative within-subjects study design. We also evaluated panning with the mouse to allow us to compare our techniques with one of the most widely used off-screen interaction techniques. Specifically, we were interested in knowing how well each of the techniques supported searching for and interacting with content in the off-screen information space. Out of all of our techniques, we only evaluated *Dynamic Distortion*, *Dynamic Peephole Inset*, and *Spatial Panning* since they support direct selection of content and are flexible in terms of defining exactly what content should be brought on-screen. Our study involved four techniques, two tasks, and two distance ranges with task completion time and accuracy level as the dependent variables. In the search and select-only tasks, participants had to use the different techniques to find and select the correct off-screen object that was a certain distance from the screen. 30 other off-screen objects were included as distractors. The only difference between tasks, was that during the select-only task, participants were provided with a visual aid that indicated the target's position in the information space.

Results from the study showed that *Spatial Panning* was overall the most preferred technique. With respect to the other spatial techniques, *Spatial Panning* was overall significantly faster, significantly faster in the search task, and always received the highest usability score. Therefore, *Spatial Panning* performed the best out of all the spatial techniques. *Dynamic Distortion* and *Dynamic Peephole Inset* were the least liked techniques and participants performed the worse when using them. Participants had the best task completion time when they used the mouse technique. The mouse was overall significantly faster than the other techniques, significantly faster than *Dynamic Distortion* and *Dynamic Peephole Inset* in the search task, and was significantly faster than *Dynamic Distortion* and *Direct Spatial Panning* in the select-only task. Target distance from the screen did not have much of an impact, and none of the study factors substantially affected accuracy levels in the search task, with one exception. Participants were significantly more accurate with the mouse when targets were closer to the screen.

The resulting mouse performance was expected due to the mouse being an indirect input device with an interaction gain, where the spatial techniques employed in the study used direct interaction. The average person's extensive use of the mouse panning technique and lack of spatial interaction experience gave the mouse another advantage. Even though the study was skewed in favour of the mouse, we still included it as a base case since it is one of the standard methods of interacting with off-screen content. The performance results showed that our spatial interaction techniques did well without much training when compared to the mouse technique. Although a comparison study was performed, we envision our techniques as complementing the mouse and not replacing it when employed within a desktop environment.

Technique	Advantages	Disadvantages
Paper Distortion	• User is able to select the starting and ending locations for the section of on-screen content that is distorted. • Able to keep a section of the original on-screen content on-screen.	• Cannot dynamically change the amount of off-screen content that is brought on-screen.
Dynamic Distortion	• Able to dynamically adjust the amount of off-screen content that is brought on-screen. • Supports different mappings from the information space to the interaction space. • Original on-screen content remains on-screen. • Able to bring a large amount of information on-screen at once.	• On-screen content becomes distorted, which can make it more difficult to analyze. • Some people find this technique difficult to use when the amount of distortion frequently changes.
Spatial Panning	• Supports fluid interaction. • Resembles standard mouse panning; therefore cognitively transitioning to it is easy for those that are accustomed to mouse interaction.	• Original on-screen content does not always remain on-screen.

Dynamic Peephole Inset	• Original on-screen content remains on-screen.
	• Viewport's small size allows the user to concentrate on a smaller section of the off-screen information space.
	• Multiple sections of the off-screen space can be explored by using multiple hands.
	• Viewport can be pinned on-screen with the ability to change its zoom level by adjusting the size of the viewport.
		• Viewport can occlude a portion of the original on-screen content.
		• Small size of the viewport
		  • reduces the amount of contextual information of the off-screen space.
		  • reduces the amount of total off-screen information that can be viewed at once.
		  • requires more physical movement from the user to explore the off-screen space.

Point2Pan	• Little physical effort is required. • Allows a user to quickly explore an information space.	• Does not support spatial selection with hand that is performing the Point2Pan technique. • Does not take advantage of the benefits from directly interacting with the 2D information space that is defined by the plane of the display (e.g., proprioception). • Original on-screen content does not always remain on-screen.

Table 10.1 The advantages and disadvantages of our spatial off-screen exploration techniques.

References

[1] Ahlström, D., Hasan, K., and Irani, P. (2014). Are you comfortable doing that?: Acceptance studies of around-device gestures in and for public settings. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*, MobileHCI '14, pages 193–202, New York, NY, USA. ACM.

[2] Avidan, S. and Shamir, A. (2007). Seam carving for content-aware image resizing. *ACM Trans. Graph.*, 26(3).

[3] Avrahami, D., Wobbrock, J. O., and Izadi, S. (2011). Portico: Tangible interaction on and around a tablet. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 347–356, New York, NY, USA. ACM.

[4] Baudisch, P. and Chu, G. (2009). Back-of-device interaction allows creating very small touch devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1923–1932, New York, NY, USA. ACM.

[5] Baudisch, P., Good, N., Bellotti, V., and Schraedley, P. (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, pages 259–266, New York, NY, USA. ACM.

[6] Baudisch, P., Lee, B., and Hanna, L. (2004). Fishnet, a fisheye web browser with search term popouts: A comparative evaluation with overview and linear view. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 133–140, New York, NY, USA. ACM.

[7] Baudisch, P. and Rosenholtz, R. (2003). Halo: A technique for visualizing off-screen objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 481–488, New York, NY, USA. ACM.

[8] Bederson, B. B., Clamage, A., Czerwinski, M. P., and Robertson, G. G. (2004). Datelens: A fisheye calendar interface for pdas. *ACM Trans. Comput.-Hum. Interact.*, 11(1):90–119.

[9] Bederson, B. B., Hollan, J. D., Perlin, K., Meyer, J., Bacon, D., and Furnas, G. (1996). Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *Journal of Visual Languages & Computing*, 7(1):3 – 32.

[10] Bekker, M. M., Olson, J. S., and Olson, G. M. (1995). Analysis of gestures in face-to-face design teams provides guidance for how to use groupware in design. In *Proceedings of the 1st Conference on Designing Interactive Systems: Processes, Practices, Methods, & Techniques*, DIS '95, pages 157–166, New York, NY, USA. ACM.

[11] Benko, H. and Wilson, A. (2009). DepthTouch: Using depth-sensing camera to enable freehand interactions on and above the interactive surface. Technical Report MSR-TR-2009-23.

[12] Bezerianos, A. and Balakrishnan, R. (2005). The Vacuum: Facilitating the manipulation of distant objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 361–370, New York, NY, USA. ACM.

[13] Bi, X., Grossman, T., Matejka, J., and Fitzmaurice, G. (2011). Magic Desk: Bringing multi-touch surfaces into desktop work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2511–2520, New York, NY, USA. ACM.

[14] Björk, S., Holmquist, L. E., Redström, J., Bretan, I., Danielsson, R., Karlgren, J., and Franzén, K. (1999). West: A web browser for small terminals. In *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology*, UIST '99, pages 187–196, New York, NY, USA. ACM.

[15] Bostock, M. and Heer, J. (2009). Protovis: A graphical toolkit for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1121–1128.

[16] Bostock, M., Ogievetsky, V., and Heer, J. (2011). D3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309.

[17] Bowman, D. A. and Hodges, L. F. (1997). An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, I3D '97, pages 35–ff., New York, NY, USA. ACM.

[18] Bradley, M. M. and Lang, P. J. (1994). Measuring emotion: The Self-Assessment Manikin and the semantic differential. *Journal of Behavior Therapy and Experimental Psychiatry*, 25(1):49–59.

[19] Burigat, S. and Chittaro, L. (2011). Visualizing references to off-screen content on mobile devices: A comparison of arrows, wedge, and overview+detail. *Interact. Comput.*, 23(2):156–166.

[20] Burigat, S. and Chittaro, L. (2013). On the effectiveness of overview+detail visualization on mobile devices. *Personal Ubiquitous Comput.*, 17(2):371–385.

[21] Burigat, S., Chittaro, L., and Gabrielli, S. (2006). Visualizing locations of off-screen objects on mobile devices: A comparative evaluation of three approaches. In *Proceedings of the 8th Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '06, pages 239–246, New York, NY, USA. ACM.

[22] Burigat, S., Chittaro, L., and Parlato, E. (2008). Map, diagram, and web page navigation on mobile devices: The effectiveness of zoomable user interfaces with overviews. In *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '08, pages 147–156, New York, NY, USA. ACM.

[23] Burigat, S., Chittaro, L., and Vianello, A. (2012). Dynamic visualization of large numbers of off-screen objects on mobile devices: An experimental comparison of wedge and overview+detail. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '12, pages 93–102, New York, NY, USA. ACM.

[24] Butler, A., Izadi, S., and Hodges, S. (2008). Sidesight: Multi-"touch" interaction around small devices. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 201–204, New York, NY, USA. ACM.

[25] Butscher, S., Hornbæk, K., and Reiterer, H. (2014). SpaceFold and PhysicLenses: Simultaneous multifocus navigation on touch surfaces. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pages 209–216, New York, NY, USA. ACM.

[26] Buyukkokten, O., Garcia-Molina, H., Paepcke, A., and Winograd, T. (2000). Power browser: Efficient web browsing for pdas. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 430–437, New York, NY, USA. ACM.

[27] Cao, X., Li, J. J., and Balakrishnan, R. (2008). Peephole pointing: Modeling acquisition of dynamically revealed targets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1699–1708, New York, NY, USA. ACM.

[28] Carpendale, M. S. T. (1999). *A Framework for Elastic Presentation Space*. PhD thesis, Simon Fraser University, Vancouver, BC, Canada.

[29] Carpendale, M. S. T. and Montagnese, C. (2001). A framework for unifying presentation space. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 61–70, New York, NY, USA. ACM.

[30] Carter, T., Seah, S. A., Long, B., Drinkwater, B., and Subramanian, S. (2013). Ultrahaptics: Multi-point mid-air haptic feedback for touch surfaces. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 505–514, New York, NY, USA. ACM.

[31] Chen, N., Guimbretiere, F., Dixon, M., Lewis, C., and Agrawala, M. (2008). Navigation techniques for dual-display e-book readers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1779–1788, New York, NY, USA. ACM.

[32] Chen, Y., Ma, W.-Y., and Zhang, H.-J. (2003). Detecting web page structure for adaptive viewing on small form factor devices. In *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, pages 225–233, New York, NY, USA. ACM.

[33] Chewar, C. M. and McCrickard, D. S. (2002). Dynamic route descriptions: Trade-offs by usage goals and user characteristics. In *Proceedings of the 2nd International Symposium on Smart Graphics*, SMARTGRAPH '02, pages 71–78, New York, NY, USA. ACM.

[34] Cockburn, A., Karlson, A., and Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):2:1–2:31.

[35] Cockburn, A., Quinn, P., Gutwin, C., Ramos, G., and Looser, J. (2011). Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback. *Int. J. Hum.-Comput. Stud.*, 69(6):401–414.

[36] Cohen, J. (1994). The Earth is round ($p < .05$). *American Psychologist*, 49(12):997–1003.

[37] Colgan, A. (2014). How does the Leap Motion controller work? http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/. Accessed: 2015-06-13.

[38] Cox, D. A., Chugh, J. S., Gutwin, C., and Greenberg, S. (1998). The usability of transparent overview layers. In *CHI 98 Conference Summary on Human Factors in Computing Systems*, CHI '98, pages 301–302, New York, NY, USA. ACM.

[39] Cumming, G. (2013). The new statistics: Why and how. *Psychological science*, 25:7—29.

[40] Cumming, G. and Finch, S. (2005). Inference by eye: Confidence intervals and how to read pictures of data. *American Psychologist*, 60(2):170.

[41] Dragicevic, P. (2015). Bad stats: Not what it seems — towards a statistical reform in HCI and visualization. http://www.aviz.fr/badstats/. Accessed: 2015-08-20.

[42] Dragicevic, P., Chevalier, F., and Huot, S. (2014). Running an HCI experiment in multiple parallel universes. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 607–618. ACM.

[43] Echtler, F., Huber, M., and Klinker, G. (2008). Shadow tracking on multi-touch tables. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '08, pages 388–391, New York, NY, USA. ACM.

[44] Ekman, I. and Lankoski, P. (2002). What should it do?: Key isssues in navigation interface design for small screen devices. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '02, pages 622–623, New York, NY, USA. ACM.

[45] Elmqvist, N., Riche, Y., Henry-Riche, N., and Fekete, J.-D. (2010). Mélange: Space folding for visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):468–483.

[46] Ens, B., Ahlström, D., Cockburn, A., and Irani, P. (2011). Characterizing user performance with assisted direct off-screen pointing. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, pages 485–494, New York, NY, USA. ACM.

[47] Ens, B., Ofek, E., Bruce, N., and Irani, P. (2015). Spatial constancy of surface-embedded layouts across multiple environments. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*, SUI '15, pages 65–68, New York, NY, USA. ACM.

[48] Fitzmaurice, G. W. (1993). Situated information spaces and spatially aware palmtop computers. *Commun. ACM*, 36(7):39–49.

[49] Forlines, C., Wigdor, D., Shen, C., and Balakrishnan, R. (2007). Direct-touch vs. mouse input for tabletop displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 647–656, New York, NY, USA. ACM.

[50] Furnas, G. W. (1986). Generalized fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '86, pages 16–23, New York, NY, USA. ACM.

[51] Games, P. S. and Joshi, A. (2013). Visualization of off-screen data on tablets using context-providing bar graphs and scatter plots. volume 9017, pages 90170D–90170D–15.

[52] Ghani, S., Riche, N. H., and Elmqvist, N. (2011). Dynamic insets for context-aware graph navigation. In *Computer Graphics Forum*, volume 30, pages 861–870. Wiley Online Library.

[53] Goble, D. J., Noble, B. C., and Brown, S. H. (2010). Where was my arm again? Memory-based matching of proprioceptive targets is enhanced by increased target presentation time. *Neuroscience Letters*, 481(1):54–58.

[54] Gonçalves, T., Afonso, A. P., Carmo, M. B., and Pombinho, P. (2011). Evaluation of HaloDot: Visualization of relevance of off-screen objects with over cluttering prevention on mobile devices. In *Proceedings of the 13th IFIP TC 13 International Conference on Human-computer Interaction - Volume Part IV*, INTERACT'11, pages 300–308, Berlin, Heidelberg. Springer-Verlag.

[55] Gonçalves, T., Afonso, A. P., Carmo, M. B., and Pombinho, P. (2013). Comparison of off-screen visualization techniques with representation of relevance on mobile devices. In *Proceedings of the 27th International BCS Human Computer Interaction Conference*, BCS-HCI '13, pages 9:1–9:10, Swinton, UK, UK. British Computer Society.

[56] Gordon, C. C., Churchill, T., Clauser, C. E., Bradtmiller, B., McConille, J. T., Tebbetts, I., and Walker, R. A. (1989). 1988 anthropometric survey of U.S. army personnel: Methods and summary statistics. Technical Report NATICK/TR-89/044, U.S. Army Natick Research, Development and Engineering Center, Natick, Massachusetts, USA.

[57] Greenberg, S. and Buxton, B. (2008). Usability evaluation considered harmful (some of the time). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 111–120, New York, NY, USA. ACM.

[58] Grossman, T. and Balakrishnan, R. (2005). The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 281–290, New York, NY, USA. ACM.

[59] Gupta, S., Morris, D., Patel, S. N., and Tan, D. (2013). Airwave: Non-contact haptic feedback using air vortex rings. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 419–428, New York, NY, USA. ACM.

[60] Gustafson, S., Baudisch, P., Gutwin, C., and Irani, P. (2008). Wedge: Clutter-free visualization of off-screen locations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 787–796, New York, NY, USA. ACM.

[61] Gustafson, S., Bierwirth, D., and Baudisch, P. (2010). Imaginary interfaces: Spatial interaction with empty hands and without visual feedback. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 3–12, New York, NY, USA. ACM.

[62] Gustafson, S. G. and Irani, P. P. (2007). Comparing visualizations for tracking off-screen moving targets. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '07, pages 2399–2404, New York, NY, USA. ACM.

[63] Gutwin, C. (2002). Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 267–274, New York, NY, USA. ACM.

[64] Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*, UIST '05, pages 115–118, New York, NY, USA. ACM.

[65] Harrison, C. and Hudson, S. E. (2009). Abracadabra: Wireless, high-precision, and unpowered finger input for very small mobile devices. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 121–124, New York, NY, USA. ACM.

[66] Hart, S. G. and Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. *Advances in psychology*, 52:139–183.

[67] Hasan, K., Ahlström, D., and Irani, P. (2013). Ad-binning: Leveraging around device space for storing, browsing and retrieving mobile device content. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 899–908, New York, NY, USA. ACM.

[68] Hasan, K., Ahlström, D., and Irani, P. P. (2015). Comparing direct off-screen pointing, peephole, and flick & pinch interaction for map navigation. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*, SUI '15, pages 99–102, New York, NY, USA. ACM.

[69] Hausen, D. (2014). *Peripheral Interaction: Exploring the Design Space*. PhD thesis, University of Munich.

[70] Hausen, D., Boring, S., and Greenberg, S. (2013a). The unadorned desk: Exploiting the physical space around a display as an input canvas. In Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., and Winckler, M., editors, *Human-Computer Interaction – INTERACT 2013*, volume 8117 of *Lecture Notes in Computer Science*, pages 140–158. Springer Berlin Heidelberg.

[71] Hausen, D., Richter, H., Hemme, A., and Butz, A. (2013b). Comparing input modalities for peripheral interaction: A case study on peripheral music control. In Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., and Winckler, M., editors, *Human-Computer Interaction – INTERACT 2013*, volume 8119 of *Lecture Notes in Computer Science*, pages 162–179. Springer Berlin Heidelberg.

[72] Hausen, D., Wagner, C., Boring, S., and Butz, A. (2013c). Comparing modalities and feedback for peripheral interaction. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 1263–1268, New York, NY, USA. ACM.

[73] Heer, J. and Card, S. K. (2004). DOITrees revisited: Scalable, space-constrained visualization of hierarchical data. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 421–424, New York, NY, USA. ACM.

[74] Hemmert, F., Löwe, M., Wohlauf, A., and Joost, G. (2013). Animate mobiles: Proxemically reactive posture actuation as a means of relational interaction with mobile phones. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, TEI '13, pages 267–270, New York, NY, USA. ACM.

[75] Henze, N. and Boll, S. (2010). Evaluation of an off-screen visualization for magic lens and dynamic peephole interfaces. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, pages 191–194, New York, NY, USA. ACM.

[76] Hilliges, O., Izadi, S., Wilson, A. D., Hodges, S., Garcia-Mendoza, A., and Butz, A. (2009). Interactions in the air: Adding further depth to interactive tabletops. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 139–148, New York, NY, USA. ACM.

[77] Hilliges, O., Kim, D., Izadi, S., Weiss, M., and Wilson, A. (2012). HoloDesk: Direct 3D interactions with a situated see-through display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 2421–2430, New York, NY, USA. ACM.

[78] Hincapié-Ramos, J. D., Guo, X., Moghadasian, P., and Irani, P. (2014). Consumed endurance: A metric to quantify arm fatigue of mid-air interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1063–1072, New York, NY, USA. ACM.

[79] Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. (1994). A survey of design issues in spatial input. In *Proceedings of the 7th Annual ACM Symposium on User Interface Software and Technology*, UIST '94, pages 213–222, New York, NY, USA. ACM.

[80] Hirsch, M., Lanman, D., Holtzman, H., and Raskar, R. (2009). BiDi Screen: A thin, depth-sensing LCD for 3D interaction using light fields. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 159:1–159:9, New York, NY, USA. ACM.

[81] Holman, D., Vertegaal, R., Altosaar, M., Troje, N., and Johns, D. (2005). Paper windows: Interaction techniques for digital paper. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 591–599, New York, NY, USA. ACM.

[82] Hoober, S. (2013). How do users really hold mobile devices? http://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php. Accessed: 2015-06-16.

[83] Hornbæk, K., Bederson, B. B., and Plaisant, C. (2002). Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Trans. Comput.-Hum. Interact.*, 9(4):362–389.

[84] Hossain, Z., Hasan, K., Liang, H.-N., and Irani, P. (2012). Edgesplit: Facilitating the selection of off-screen objects. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '12, pages 79–82, New York, NY, USA. ACM.

[85] Hwang, S., Ahn, M., and Wohn, K.-y. (2013a). MagGetz: Customizable Passive Tangible Controllers on and Around Conventional Mobile Devices. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 411–416, New York, NY, USA. ACM.

[86] Hwang, S., Bianchi, A., Ahn, M., and Wohn, K. (2013b). Magpen: Magnetically driven pen interactions on and around conventional smartphones. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '13, pages 412–415, New York, NY, USA. ACM.

[87] Hwang, S., Gim, J., Yoo, J., and Bianchi, A. (2015). Contextual drag: Context-based dynamic friction for dragging interaction. In *CHI '15 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 167–167, New York, NY, USA. ACM.

[88] Hwang, S., Kim, D., Leigh, S.-w., and Wohn, K.-y. (2013c). Nailsense: Fingertip force as a new input modality. In *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13 Adjunct, pages 63–64, New York, NY, USA. ACM.

[89] Ion, A., Chang, Y.-L. B., Haller, M., Hancock, M., and Scott, S. D. (2013). Canyon: Providing location awareness of multiple moving objects in a detail view on large displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 3149–3158, New York, NY, USA. ACM.

[90] Irani, P., Gutwin, C., and Yang, X. D. (2006). Improving selection of off-screen targets with hopping. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 299–308, New York, NY, USA. ACM.

[91] Ishii, H. and Ullmer, B. (1997). Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 234–241, New York, NY, USA. ACM.

[92] Izadi, S., Hodges, S., Taylor, S., Rosenfeld, D., Villar, N., Butler, A., and Westhues, J. (2008). Going beyond the display: A surface technology with an electronically switchable diffuser. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 269–278, New York, NY, USA. ACM.

[93] Jäckle, D., Senaratne, H., Buchmüller, J., and Keim, D. A. (2015). Integrated spatial uncertainty visualization using off-screen aggregation. In Bertini, E. and Roberts, J. C., editors, *EuroVis Workshop on Visual Analytics (EuroVA)*. The Eurographics Association.

[94] Jerding, D. F. and Stasko, J. T. (1998). The information mural: A technique for displaying and navigating large information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 4(3):257–271.

[95] Jo, H., Hwang, S., Park, H., and Ryu, J.-h. (2011). Aroundplot: Focus + context interface for off-screen objects in 3D environments. *Computers & Graphics*, 35(4):841–853.

[96] Jones, B., Sodhi, R., Forsyth, D., Bailey, B., and Maciocci, G. (2012). Around device interaction for multiscale navigation. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '12, pages 83–92, New York, NY, USA. ACM.

[97] Jones, B. R., Benko, H., Ofek, E., and Wilson, A. D. (2013). Illumiroom: Peripheral projected illusions for interactive experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 869–878, New York, NY, USA. ACM.

[98] Jota, R., Lopes, P., Wigdor, D., and Jorge, J. (2014). Let's kick it: How to stop wasting the bottom third of your large screen display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1411–1414, New York, NY, USA. ACM.

[99] Jota, R. and Wigdor, D. (2015). Palpebrae superioris: Exploring the design space of eyelid gestures. In *Proceedings of the 41st Graphics Interface Conference*, GI '15, pages 273–280, Toronto, Ont., Canada. Canadian Information Processing Society.

[100] Jul, S. and Furnas, G. W. (1998). Critical zones in desert fog: Aids to multiscale navigation. In *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, UIST '98, pages 97–106, New York, NY, USA. ACM.

[101] Kadmon, N. and Shlomi, E. (1978). A polyfocal projection for statistical surfaces. *The Cartographic Journal*, 15(1):36–41.

[102] Kaltenbrunner, M. (2009). reacTIVision and TUIO: A tangible tabletop toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 9–16, New York, NY, USA. ACM.

[103] Kaptein, M. and Robertson, J. (2012). Rethinking statistical analysis methods for CHI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1105–1114, New York, NY, USA. ACM.

[104] Kazi, A. M. and Khalid, W. (2012). Questionnaire designing and validation. *Journal of the Pakistan Medical Association*, 62(5):514.

[105] Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8.

[106] Ketabdar, H., Haji-Abolhassani, A., Yuksel, K., Kehribar, I., and Ercil, A. (2011). Digital music performance for mobile devices based on magnetic interaction. *IEEE Transactions on Instrumentation and Measurement*, 60(11):3524–3533.

[107] Ketabdar, H., Moghadam, P., Naderi, B., and Roshandel, M. (2012). Magnetic signatures in air for mobile devices. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services Companion*, MobileHCI '12, pages 185–188, New York, NY, USA. ACM.

[108] Ketabdar, H., Roshandel, M., and Yüksel, K. A. (2010a). MagiWrite: Towards touchless digit entry using 3D space around mobile devices. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, pages 443–446, New York, NY, USA. ACM.

[109] Ketabdar, H., Roshandel, M., and Yüksel, K. A. (2010b). Towards using embedded magnetic field sensor for around mobile device 3D interaction. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, pages 153–156, New York, NY, USA. ACM.

[110] Khalilbeigi, M., Lissermann, R., Mühlhäuser, M., and Steimle, J. (2011). Xpaaand: Interaction techniques for rollable displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2729–2732, New York, NY, USA. ACM.

[111] Kim, D., Izadi, S., Dostal, J., Rhemann, C., Keskin, C., Zach, C., Shotton, J., Large, T., Bathiche, S., Nießner, M., Butler, D. A., Fanello, S., and Pradeep, V. (2014). RetroDepth: 3D silhouette sensing for high-precision input on and above physical surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1377–1386, New York, NY, USA. ACM.

[112] Kim, K.-E., Chang, W., Cho, S.-J., Shim, J., Lee, H., Park, J., Lee, Y., and Kim, S. (2006). Hand grip pattern recognition for mobile user interfaces. In *Proceedings of the 18th Conference on Innovative Applications of Artificial Intelligence - Volume 2*, IAAI'06, pages 1789–1794. AAAI Press.

[113] Koch, K., Walker, T., Ji, Y., Gravesmill, K., Kwok, J., and Garcia, M. (2015). Hands Omni haptic glove. http://news.rice.edu/2015/04/22/gamers-feel-the-glove-from-rice-engineers-2/. Accessed: 2015-06-27.

[114] Kratz, S. and Rohs, M. (2009). Hoverflow: Expanding the design space of around-device interaction. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '09, pages 4:1–4:8, New York, NY, USA. ACM.

[115] Kratz, S., Rohs, M., Guse, D., Müller, J., Bailly, G., and Nischt, M. (2012). PalmSpace: Continuous around-device gestures vs. multitouch for 3D rotation tasks on mobile devices. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 181–188, New York, NY, USA. ACM.

[116] Lambdin, C. (2012). Significance tests as sorcery: Science is empirical—significance tests are not. *Theory & Psychology*, 22(1):67–90.

[117] Lamping, J., Rao, R., and Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 401–408, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

[118] Leap Motion (2015). https://www.leapmotion.com/. Accessed: 2015-06-13.

[119] Lee, C., DiVerdi, S., and Hollerer, T. (2009). Depth-fused 3D imagery on an immaterial display. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):20–33.

[120] Lee, J., Olwal, A., Ishii, H., and Boulanger, C. (2013). SpaceTop: Integrating 2D and spatial 3D interactions in a see-through desktop environment. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 189–192, New York, NY, USA. ACM.

[121] Lee, J. C., Hudson, S. E., and Tse, E. (2008). Foldable interactive displays. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 287–290, New York, NY, USA. ACM.

[122] Leithinger, D., Follmer, S., Olwal, A., Luescher, S., Hogge, A., Lee, J., and Ishii, H. (2013). Sublimate: State-changing virtual and physical rendering to augment interaction with shape displays. In *Proceedings of the SIGCHI Conference on Human*

Factors in Computing Systems, CHI '13, pages 1441–1450, New York, NY, USA. ACM.

[123] Leung, Y. K. and Apperley, M. D. (1994). A review and taxonomy of distortion-oriented presentation techniques. *ACM Trans. Comput.-Hum. Interact.*, 1(2):126–160.

[124] Liang, R.-H., Cheng, K.-Y., Su, C.-H., Weng, C.-T., Chen, B.-Y., and Yang, D.-N. (2012). Gausssense: Attachable stylus sensing using magnetic sensor grid. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 319–326, New York, NY, USA. ACM.

[125] Mackinlay, J. D., Robertson, G. G., and Card, S. K. (1991). The perspective wall: Detail and context smoothly integrated. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 173–176, New York, NY, USA. ACM.

[126] Marquardt, N., Jota, R., Greenberg, S., and Jorge, J. (2011). The continuous interaction space: Interaction techniques unifying touch and gesture on and above a digital surface. In Campos, P., Graham, N., Jorge, J., Nunes, N., Palanque, P., and Winckler, M., editors, *Human-Computer Interaction – INTERACT 2011*, volume 6948 of *Lecture Notes in Computer Science*, pages 461–476. Springer Berlin Heidelberg.

[127] Martinez Plasencia, D., Joyce, E., and Subramanian, S. (2014). Mistable: Reach-through personal screens for tabletops. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 3493–3502, New York, NY, USA. ACM.

[128] McCrickard, D. S. and Catrambone, R. (1999). Beyond the scrollbar: An evolution and evaluation of alternative navigation techniques. In *Proceedings of the IEEE Symposium on Visual Languages*, VL '99, pages 270–277, Washington, DC, USA. IEEE Computer Society.

[129] Mehra, S., Werkhoven, P., and Worring, M. (2006). Navigating on handheld displays: Dynamic versus static peephole navigation. *ACM Trans. Comput.-Hum. Interact.*, 13(4):448–457.

[130] Molyneaux, D., Izadi, S., Kim, D., Hilliges, O., Hodges, S., Cao, X., Butler, A., and Gellersen, H. (2012). Interactive environment-aware handheld projectors for pervasive computing spaces. In *Proceedings of the 10th International Conference on Pervasive Computing*, Pervasive'12, pages 197–215, Berlin, Heidelberg. Springer-Verlag.

[131] Moscovich, T., Chevalier, F., Henry, N., Pietriga, E., and Fekete, J.-D. (2009). Topology-aware navigation in large networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2319–2328, New York, NY, USA. ACM.

[132] Müller, J., Rädle, R., Jetter, H.-C., and Reiterer, H. (2015). An experimental comparison of vertical and horizontal dynamic peephole navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '15, pages 1523–1526, New York, NY, USA. ACM.

[133] Ochiai, Y., Kumagai, K., Hoshi, T., Rekimoto, J., Hasegawa, S., and Hayasaki, Y. (2015). Fairy lights in femtoseconds: Aerial and volumetric graphics rendered by focused femtosecond laser combined with computational holographic fields. *arXiv preprint arXiv:1506.06668*.

[134] Pohl, H. and Murray-Smith, R. (2013). Focused and casual interactions: Allowing users to vary their level of engagement. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2223–2232, New York, NY, USA. ACM.

[135] Pohl, H. and Rohs, M. (2014). Around-device devices: My coffee mug is a volume dial. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*, MobileHCI '14, pages 81–90, New York, NY, USA. ACM.

[136] Pohl, H., Rohs, M., and Murray-Smith, R. (2014). Casual interaction: Scaling fidelity for low-engagement interactions. In *Workshop on Peripheral Interaction: Shaping the Research and Design Space at CHI 2014*.

[137] Pyryeskin, D., Hancock, M., and Hoey, J. (2011). Extending interactions into hoverspace using reflected light. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '11, pages 262–263, New York, NY, USA. ACM.

[138] Rädle, R., Jetter, H.-C., Marquardt, N., Reiterer, H., and Rogers, Y. (2014a). Huddlelamp: Spatially-aware mobile displays for ad-hoc around-the-table collaboration. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, ITS '14, pages 45–54, New York, NY, USA. ACM.

[139] Rädle, R., Jetter, H.-C., Müller, J., and Reiterer, H. (2014b). Bigger is not always better: Display size, performance, and task load during peephole map navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 4127–4136, New York, NY, USA. ACM.

[140] Rao, R. and Card, S. K. (1994). The table lens: Merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 318–322, New York, NY, USA. ACM.

[141] Rateau, H., Grisoni, L., and De Araujo, B. (2014). Mimetic interaction spaces: Controlling distant displays in pervasive environments. In *Proceedings of the 19th International Conference on Intelligent User Interfaces*, IUI '14, pages 89–94, New York, NY, USA. ACM.

[142] Reas, C. and Fry, B. (2006). Processing: programming for the media arts. *AI & SOCIETY*, 20(4):526–538.

[143] Robertson, G., Czerwinski, M., Larson, K., Robbins, D. C., Thiel, D., and van Dantzich, M. (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, pages 153–162, New York, NY, USA. ACM.

[144] Robertson, G. G. and Mackinlay, J. D. (1993). The document lens. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*, UIST '93, pages 101–108, New York, NY, USA. ACM.

[145] Sarkar, M. and Brown, M. H. (1992). Graphical fisheye views of graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, pages 83–91, New York, NY, USA. ACM.

[146] Sarkar, M., Snibbe, S. S., Tversky, O. J., and Reiss, S. P. (1993). Stretching the rubber sheet: A metaphor for viewing large layouts on small screens. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology*, UIST '93, pages 81–91, New York, NY, USA. ACM.

[147] Scarr, J., Cockburn, A., and Gutwin, C. (2012). Supporting and exploiting spatial memory in user interfaces. *Interaction*, 6(1):1–84.

[148] Schinke, T., Henze, N., and Boll, S. (2010). Visualization of off-screen objects in mobile augmented reality. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '10, pages 313–316, New York, NY, USA. ACM.

[149] Seipp, K. and Devlin, K. (2014). Backpat: Improving one-handed touchscreen operation by patting the back of the device. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pages 555–558, New York, NY, USA. ACM.

[150] Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, VL '96, pages 336–343, Washington, DC, USA. IEEE Computer Society.

[151] Shum, J. (2014). Waves: A collaborative navigation technique for large interactive surfaces. Master's thesis, University of Waterloo.

[152] Sodhi, R., Poupyrev, I., Glisson, M., and Israr, A. (2013). Aireal: Interactive tactile experiences in free air. *ACM Trans. Graph.*, 32(4):134:1–134:10.

[153] Spence, R. and Apperley, M. (1982). Data base navigation: an office environment for the professional. *Behaviour & Information Technology*, 1(1):43–54.

[154] Steimle, J. and Olberding, S. (2012). When mobile phones expand into handheld tabletops. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 271–280, New York, NY, USA. ACM.

[155] Stone, M. C., Fishkin, K., and Bier, E. A. (1994). The movable filter as a user interface tool. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 306–312, New York, NY, USA. ACM.

[156] Stuerzlinger, W. and Wingrave, C. A. (2011). The value of constraints for 3D user interfaces. In Brunnett, G., Coquillart, S., and Welch, G., editors, *Virtual Realities*, pages 203–223. Springer Vienna.

[157] Takeoka, Y., Miyaki, T., and Rekimoto, J. (2010). Z-touch: An infrastructure for 3D gesture interaction in the proximity of tabletop surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 91–94, New York, NY, USA. ACM.

[158] Taylor, S., Keskin, C., Hilliges, O., Izadi, S., and Helmes, J. (2014). Type-hover-swipe in 96 bytes: A motion sensing mechanical keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1695–1704, New York, NY, USA. ACM.

[159] Trevor, J., Hilbert, D. M., Schilit, B. N., and Koh, T. K. (2001). From desktop to phonetop: A UI for web interaction on very small devices. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, UIST '01, pages 121–130, New York, NY, USA. ACM.

[160] Vogel, D., Cudmore, M., Casiez, G., Balakrishnan, R., and Keliher, L. (2009). Hand occlusion with tablet-sized direct pen input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 557–566, New York, NY, USA. ACM.

[161] Wang Baldonado, M. Q., Woodruff, A., and Kuchinsky, A. (2000). Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pages 110–119, New York, NY, USA. ACM.

[162] Ware, C. (2004). *Information Visualization: Perception For Design*. Elsevier, 2 edition.

[163] Watson, D., Clark, L. A., and Tellegen, A. (1988). Development and validation of brief measures of positive and negative affect: the PANAS scales. *Journal of personality and social psychology*, 54(6):1063.

[164] Weichel, C., Lau, M., Kim, D., Villar, N., and Gellersen, H. W. (2014). Mixfab: A mixed-reality environment for personal fabrication. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 3855–3864, New York, NY, USA. ACM.

[165] Wigdor, D., Forlines, C., Baudisch, P., Barnwell, J., and Shen, C. (2007). Lucid touch: A see-through mobile device. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 269–278, New York, NY, USA. ACM.

[166] Wilson, A. D. (2007). Depth-sensing video cameras for 3D tangible tabletop interaction. In *Proceedings of the second annual IEEE international Workshop on Horizontal Interactive Human-Computer Systems*, TABLETOP '07, pages 201–204. IEEE.

[167] Wilson, A. D. and Benko, H. (2010). Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 273–282, New York, NY, USA. ACM.

[168] Xiao, R., Lew, G., Marsanico, J., Hariharan, D., Hudson, S., and Harrison, C. (2014). Toffee: Enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*, MobileHCI '14, pages 67–76, New York, NY, USA. ACM.

[169] Yamamoto, N. and Shelton, A. L. (2005). Visual and proprioceptive representations in spatial memory. *Memory & Cognition*, 33(1):140–150.

[170] Yamashita, N., Kaji, K., Kuzuoka, H., and Hirata, K. (2011). Improving visibility of remote gestures in distributed tabletop collaboration. In *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work*, CSCW '11, pages 95–104, New York, NY, USA. ACM.

[171] Yang, X.-D., Hasan, K., Bruce, N., and Irani, P. (2013). Surround-see: Enabling peripheral vision on smartphones during active use. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 291–300, New York, NY, USA. ACM.

[172] Yee, K.-P. (2003). Peephole displays: Pen interaction on spatially aware handheld computers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 1–8, New York, NY, USA. ACM.

[173] Zellweger, P. T., Mackinlay, J. D., Good, L., Stefik, M., and Baudisch, P. (2003). City lights: Contextual views in minimal space. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 838–839, New York, NY, USA. ACM.

Appendix A

Study Materials

UNIVERSITY OF ONTARIO INSTITUTE OF TECHNOLOGY

RESEARCH ETHICS BOARD
OFFICE OF RESEARCH SERVICES

Date: March 5th, 2015

To: Erik Paluka (Graduate Student PI) and Christopher Collins (Supervisor)

From: Bill Goodman, REB Chair

REB File #: 14-075

Project Title: Evaluation of Interaction Techniques for Off-Screen Digital Content

DECISION: APPROVED
CURRENT EXPIRY: March 1st, 2016

NOTE: Notwithstanding this approval, you are required to obtain/submit, to UOIT's Research Ethics Board, any relevant approvals/permissions required, prior to commencement of this project.

The University of Ontario, Institute of Technology Research Ethics Board (REB) has reviewed and approved the above research proposal. This application has been reviewed to ensure compliance with the Tri-Council Policy Statement: Ethical Conduct for Research Involving Humans (TCPS2 (2014)) and the UOIT Research Ethics Policy and Procedures.

Please note that the (REB) requires that you adhere to the protocol as last reviewed and approved by the REB. Always quote your REB file number on all future correspondence.

CONTINUING REVIEW REQUIREMENTS:

- **Renewal Request Form:** All approved projects are subject to an **annual** renewal process. Projects must be renewed or closed by the expiry date indicated above ("Current Expiry"). **Projects that are not renewed within 30 days of the expiry date will be automatically suspended by the REB; and projects that are not renewed within 60 days of the expiry date will be automatically closed by the REB.** Once your file has been formally closed, a new submission will be required to open a new file.

- **Change Request Form:** any changes or modifications (i.e. adding a Co-PI or a change in methodology) must be approved by the REB through the completion of a change request form before implemented.

- **Adverse or unexpected Events Form:** events must be reported to the REB within 72 hours after the event occurred with an indication of how these events affect (in the view of the Principal Investigator) the safety of the participants and the continuation of the protocol. (I.e. un-anticipated or un-mitigated physical, social or psychological harm to a participant).

- **Research Project Completion Form:** must be completed when the research study has completed.

All Forms can be found at http://research.uoit.ca/faculty/policies-procedures-forms.php.

REB Chair Dr. Bill Goodman, FBIT bill.goodman@uoit.ca	Ethics and Compliance Officer compliance@uoit.ca

University of Ontario, Institute of Technology
2000 Simcoe Street North, Oshawa ON, L1H 7K4
PHONE: (905) 721-8668, ext. 3693
Version: Jan. 2015

Fig. A.1 Research Ethics Board (REB): Letter of approval for the research study.