



# SURFNET /

## Designing Digital Surface Applications

*Frank Maurer, (Ed.)*





# SURFNET /

## Designing Digital Surface Applications

*Frank Maurer, (Ed.)*



© All Authors as listed

This book is licensed under a CREATIVE COMMONS License, available for download at <http://dspace.ucalgary.ca/handle/1880/50450>.

ISBN 978-0-88953-387-5 (Softcover)

ISBN 978-0-88953-388-2 (PDF)

**Published by:**



University of Calgary  
2500 University Drive, NW  
Calgary, AB Canada  
T2N 1N4

This book has been published using funds from the Natural Sciences and Engineering Research Council of Canada (NSERC), Strategic Network Grants program.



# **SURFNET /**

## Designing Digital Surface Applications



*Published by:  
NSERC SurfNet  
2016  
University of Calgary*



# CONTENTS



<b>INTRODUCTION</b>	<b>7</b>
<b>HUMANIZING THE DIGITAL INTERFACE</b>	<b>17</b>
Using Social Science Theory to Inspire Surface Design: A Case Study of Proxemic Interactions	26
Effects of Tabletop Embodiments on Coordination	39
Cross-Device Content Transfer in Table-centric Multi-Surface Environments	53
High-Performance Interfaces for Surfaces	80
Transmogrification: Casual Manipulation of Visual Information	95
<b>IMPROVING SOFTWARE TIME TO MARKET</b>	<b>105</b>
Understanding Sketching Practices for Surface Interface Design	112
Designing Tabletop and Surface Applications Using Interactive Prototypes	140
Pairing for Designing Visualizations	155
Constructive Visualization: A New Paradigm to Empower People to Author Visualization	169
An Approach to Automated GUI Testing	192
Agile Product Line Engineering Case Study: Vertical & Horizontal Displays	203



**BUILDING INFRASTRUCTURE FOR DIGITAL SURFACES** **221**

Society of Devices Toolkit and Projected Pixels 229

Filling the Space Between: Augmenting Multi-Surface Environments with Low-Resolution Full-Coverage Displays 241

The Simple Multi-Touch Toolkit 256

**SURFACE APPLICATIONS** **275**

Radiology Image Scrolling 279

Towards At-Home Physiotherapy: Next Generation Teleconferencing and Surface Based Interventions 289

Discouraging Sedentary Behaviors Using Interactive Play 305

OrMiS: Use of a Digital Surface for Simulation-Based Training 313

TableNOC: Touch-Enabled Geo-Temporal Visualization for Network Operations Centers 332

Beyond Efficiency: Intriguing Interaction for Large Displays in Public Spaces 347

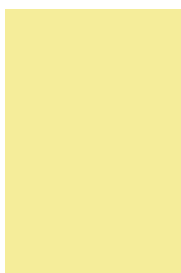
Surface Applications for Security Analysis 374

**BIOGRAPHIES** **402**

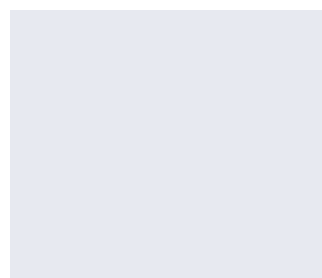
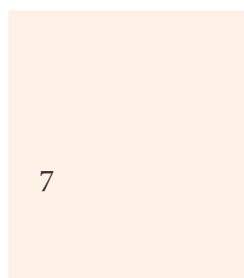
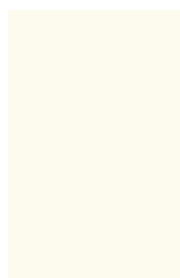
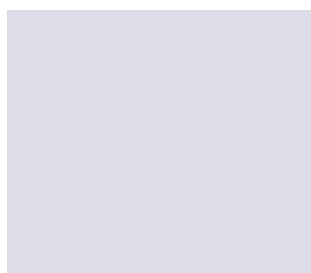
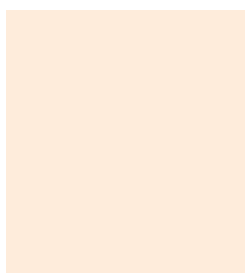
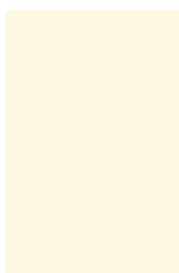
**REFERENCES** **412**

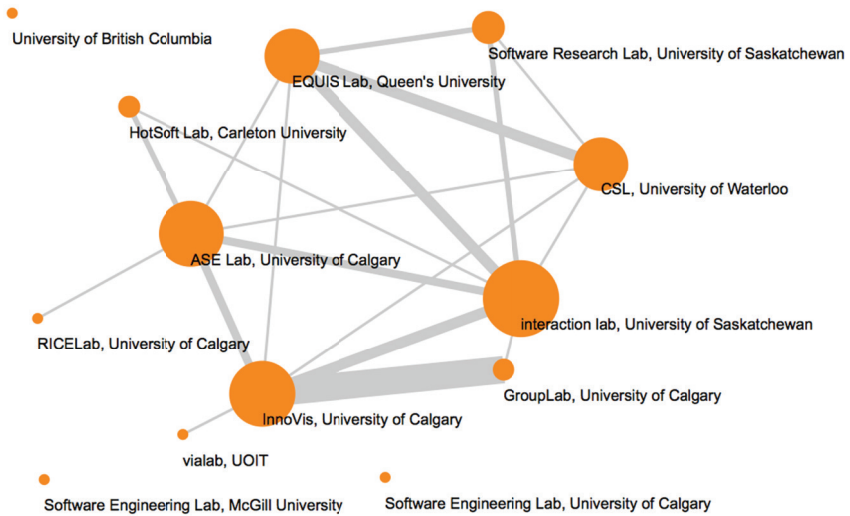






# INTRODUCTION





**Figure 1: Visualization of collaboration amongst SurfNet research labs.**



## INTRODUCTION

*Frank Maurer  
University of Calgary*

### **I**ntroduction

The SurfNet Network was a Canadian research alliance of academic researchers, industry partners, and government collaborators that operated from 2010-2015. The goal of SurfNet was to improve the development, performance, and usability of software applications for surface computing environments: nontraditional digital display surfaces including multi-touch screens, tabletops, and wall-sized displays. Surfaces naturally support group work and collaboration.

Digital surfaces come in various sizes and provide a large number of interaction techniques. Mobile surface hardware was popularized by Apple's iPhone starting in June 2007 and Android (starting with version 1.0 in September 2008). The tablet market was established by Apple's iPad in April 2010. Wall-sized displays form the physically large end of the digital surface spectrum while digital tables like the original 30 inch MS Surface (September 2008) and the Smart Table (2008) add a horizontal form factor to the device ecology. More recently, small form factor surfaces have been added to the mix in the form of smart watches.

The power provided by digital surfaces, however, can only have a substantial impact on businesses and homes when software developers can easily and efficiently create innovative applications for these environments. Our network focused on this missing link by combining specific research projects with a continual focus on actually developing surface applications in collaboration with a large number of industry partners. It is this engineering and software focus that sets SurfNet apart from other groups working on

surface computing.

SurfNet’s mandate was to integrate software engineering (SE) and human-computer interaction (HCI) in support of application engineering for digital surfaces – to identify critical requirements, design new engineering processes, and build new tools for surface-based applications.

SurfNet focused on three research themes driven by the needs of four application areas (Figure 2).

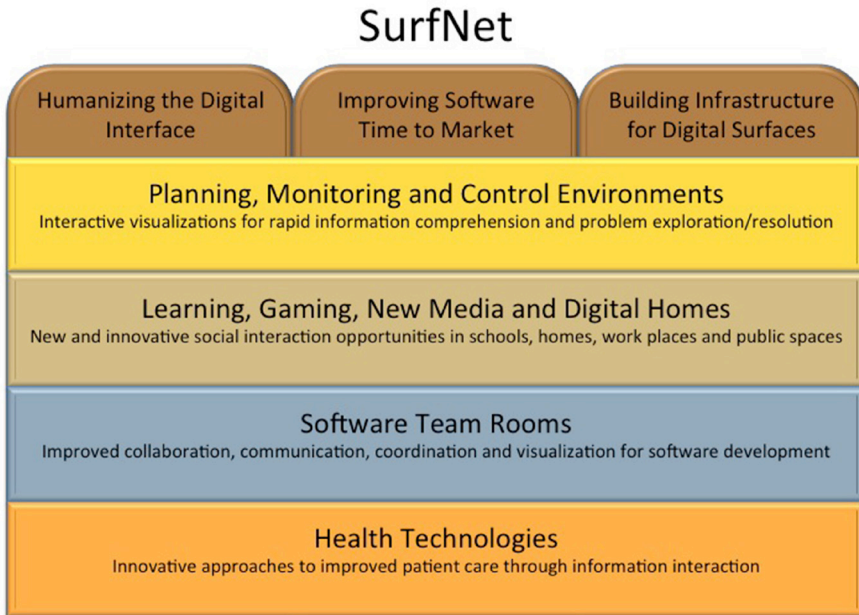


Figure 2. SurfNet Structure.

Theme 1 “Humanizing the Digital Interface” investigated fundamental questions around surface interactions and ways to support everyday practices of the system’s users. Theme 2 “Improving Software Time to Market” focused on agile development processes in support of designing, implementing and testing surface applications. Toolkits and application program interfaces (APIs) that make the life easier for surface application developers were explored in Theme 3 “Building Infrastructure for Digital Surfaces”.

SurfNet’s fundamental research was guided by the needs of industrial applications. Applications also provided test beds and case studies for the research conducted by the SurfNet team. The application areas were developed in collaboration with industry partners, and provided promising vertical markets for digital surfaces. SurfNet researchers were working with industrial partners on the following application areas:

- Planning, Monitoring and Control Environments
- Learning, Gaming, New Media and Digital Homes
- Software Team Rooms
- Health Technologies

The network was structured around collaboration and cross-discipline research, where projects often had overlapping themes. The interaction between labs can be seen in Figure 1.

In this visualization, a node (circle) represents a research lab, and the size of a node represents the total number of research projects from that lab with collaborators from other labs. The larger the node, the more collaboration exists between labs. An edge (line) shows that researchers from the connected labs collaborated on a project. The thicker the edge, the more collaborators exist.

### **Result Summary**

In the end, our overall network results exceeded our initial expectations. This is, in some part, due to the ‘gel’ of the researchers and strong collaborative efforts across and within themes, as well as streamlining administrative reporting to keep researcher efforts focused on research.

SurfNet was an incredibly prolific network that resulted in over 700 publications and presentations. Over its lifetime, the network supported more than 400 students, from the undergrad level to postdocs. Our students moved on to academic positions, research labs in industry and – many – to professional jobs related to their research. Several of them are now working at VizworX, a startup company that was spun out of SurfNet.

Overall, SurfNet was very successful in balancing our research goals with the application needs of our industry partners. We were strategic in our collaborations and able to find significant research value in industrially relevant problems. Although the shift from pure research to application-focused research took some adjustment for those who had not previously engaged strongly with external partners, it has proven an invaluable experience for our network. So much so, that most of our researchers work with several industrial partners, and continue to work on real-world application problems utilizing results from the network.

### **Timeline**

The first discussions about SurfNet occurred in late 2007 during an NSERC workshop whose goal was to establish more research collaboration across Western Canada. Carl Gutwin (Saskatchewan) and I discussed our interests in a stronger collaboration between human computer interaction and software engineering researchers. We both saw the rising tide of digital surfaces and the need for fundamental and problem-driven research on the software side for these—at that point in time—new devices.

To move the initiative forward, we established a cohort of six researchers from Waterloo, Saskatchewan and Calgary and successfully applied for NSERC funding for a series of strategic planning workshops. The first workshop was held in April 2008 and led to the development of a notice of intent for a strategic network, called SurfNet (submitted in Summer 2008). After being selected in Fall 2008 to submit a full proposal, the work really began. The expanded team of twelve researchers came together in November 2008 to refine the concept and plan the grant proposal. Sections were written, reviewed and revised multiple times. Feedback from colleagues and industry partners was gathered and integrated into the draft proposal. I am extremely grateful to my colleagues that spent countless hours on preparing the extensive documents required for a Strategic Network Grant. The proposal was submitted in March 2009. A site visit followed in summer of the same year. To everybody's delight, the grant was approved in October 2009.

The first few months were used to set up the administrative support structure for the network, to establish our SurfNet Advisory Board and to negotiate formal agreements between the seven participating universities and the key industry partners, Smart Technologies and TRTech (now, TRTech).

The network started officially its research operation in March 2010. Work initially focused on single surfaces of all sizes while in the latter half, shifted towards multi-surface environments. Research in SurfNet ended in September 2015, providing time until January 2016 to write a final report and the book that you are currently reading.

### **Book Structure**

This book illustrates the work of SurfNet researchers and their contributions to the state of the art by providing a selection of chapters covering the full spectrum of SurfNet research.

Section 1 presents work from Theme 1 and includes an overview on SurfNet work in this area written by Drs. Carpendale and Scott.

The next section discusses results from Theme 2 and Drs. Biddle and Schneider contributed a summary of this theme.

Drs. Graham and Gutwin summarize research from Theme 3 and Section 3 of this book contains selected work from this theme.

Section 4 presents a set of application-focused papers.

### **Acknowledgments**

I would like to express my deepest appreciation to the Natural Sciences and Engineering Research Council of Canada (NSERC) for supporting our research network and providing long-term predictable funding to SurfNet

researchers and their students. Their investment provided the basis for substantial progress and innovation in SurfNet's research area.

Our network's progress would have been impossible without the strong engagement of numerous industrial partners that collaborated with our researchers and helped focus our work. We are especially grateful for the contributions of Smart Technologies and TRTech. Both organizations were involved with the network from before its inception, helped shape its structure and direction, and provided substantial financial support for our work.

I am deeply grateful to the University of Calgary, Carleton University, McGill University, Queen's University, the University of British Columbia, the University of Saskatchewan, and the University of Waterloo for their substantial and ongoing financial and administrative support to our research network. Their administrative staff was instrumental for the success of the network and are often the unsung heroes in academia.

The research direction of our network was strongly impacted by the external members of our SurfNet Advisory Board. For their willingness to volunteer many hours in support of the network, I want to express my sincere gratitude to Pekka Abrahamson, Andrea Benoit, Hakan Erdogmus, Tony Florio, Bruce G. Gilkes, Michael Haller, Rainer Iraschko, Gerald Morrison, Kori Inkpen Quinn, Mary Beth Rosson, Helen Sharp, Janice Singer, Dave Thomas, and Jennifer van Zelm.

While a strategic network is required to have a principal investigator, its success depends on the dedicated and collaborative work of a group of peers. My co-investigators Robert Biddle, Sheelagh Carpendale, Nick Graham, Saul Greenberg, Carl Gutwin, Joerg Kienzle, Philippe Kruchten, Regan Mandryk, Stacey Scott, Kevin Schneider, and Jonathan Sillito are leading thinkers in their field and the driving force behind our network's success.

A big "thank you" goes to colleagues from all over the world that joined our network over time, providing deep insights during discussions and collaborating with network researchers on specific research projects. Their expertise helped our network accomplish its goals: Scott Bateman, Anastasia Bezerianos, Jeff Boyd, Sonia Chiasson, Christopher Collins, Pierre Dragicevic, Elise Fear, Jean-Daniel Fekete, Mark Hancock, Rashina Hoda, Pourang Irani, Petra Isenberg, Timothy Lethbridge, Angela Martin, Miguel Nacenta, Neil Randall, Ehud Sharlin, Anthony Tang, and Xin Wang.

The real stars in SurfNet were our students and postdocs. Professors in Computer Science realize that most of the work is done by their dedicated team of creative and innovative "highly qualified personnel" while the prof tries to find support for them. Students develop ideas, implement software, run studies and write papers. Sometimes they move mountains, always they

expand knowledge of humanity. SurfNet would not have been possible without their tireless efforts and enthusiasm.

A network cannot be run without the support of dedicated administrative support and I want to express my sincere thanks to Grace Whitehead and Jennifer Harper that served in that role at the early stage of the network. Jeff LaFrenz was key to the success of our network. In his role as Business Development Manager, he tirelessly worked on finding potential industry partners for our network researchers, opening doors and creating connections. His dedication to technology transfer and innovation is proven by his willingness to now serve as the CEO of VizworX, SurfNet's spin-out company.

I cannot express the amount of gratitude that Robin Arseneault deserves from me and the rest of SurfNet. In her role as SurfNet Network Manager, she took on an immense workload on capturing data, reporting, accounting, writing and organizing. She streamlined required administrative processes as much as possible so that researchers could focus on what they do best: conduct research. The hours that she has spent on our final report as well as on designing and copy-editing this book are beyond expectations. Not only is she one of the best organized people that I have ever met, she is also an award winning artist.

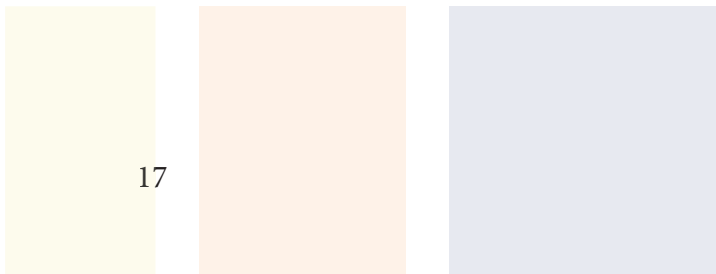








# **HUMANIZING THE DIGITAL INTERFACE**







# HUMANIZING THE DIGITAL INTERFACE

*Sheelagh Carpendale, University of Calgary*  
*Stacey Scott, University of Waterloo*

Focus Areas:

- 1.1 Understanding the Fit Between Surfaces, Humans, & Human Activity
- 1.2 Interacting with a Single Surface
- 1.3 Interacting with Multiple Surfaces
- 1.4 Adapting Interface Concepts to Real-world Settings

## **I**ntroduction

Theme 1 research focused on expanding our understanding of the fundamentals of surface interaction. Interacting with digital surfaces is fundamentally different than interacting with mouse-and-keyboard-based computers like desktops or laptops. This has required new knowledge about how best to design interfaces, interaction techniques, and applications that provide the most effective use of the new interaction capabilities provided by digital surfaces. Moreover, across the course of the SurfNet research program, there has been exciting and rapid changes in the variety of “surface” computing devices, and related interaction modalities available for these surfaces. Small, personal multi-touch surfaces (e.g. smartphones and tablets) have seen wide-scale adoption in Canada and other Western cultures. New consumer hardware (e.g. Microsoft Kinect, Leap Motion) enabled low-cost whole- and part-body interactions with large surfaces, significantly expanding possible interactions on digital walls and tabletops, beyond touch-based interaction. SurfNet research, throughout all three Themes,

played a significant role in extending the capabilities and application use of the emerging input and surface device hardware to enable more extensive interfaces and interactions across a wide variety of application contexts.

As the overarching goal of Theme 1 research was to design, develop and evaluate interaction for surface technologies that supports and participates in, rather than ignores, the everyday-world practices of people, these expanding hardware capabilities usefully expanded the “surfaces” toolbox which to draw from during our design and development activities. As we describe in the following focus area discussions, this expanded toolbox led to innovations, such as multi-surface interfaces that enable the use of smartphones or tablets in conjunction with large digital walls or tabletops during a collaborative analysis task, and large public wall displays that “react” to people as the approach they display (even before they touch the surface).

Important new HCI questions emerged along with this new hardware, such as when is a small surface beneficial?, when is a large surface beneficial?, and how can different surface form factors and interaction modalities best be used together to provide an effective user experience? Theme 1 has adapted to investigate these emerging questions by studying and creating interfaces, interaction techniques, and whole applications for a wide range of surface form factors and interaction capabilities in a variety of different application domain contexts. The network approach to SurfNet research enabled this agile research approach as it required access to a substantial amount of different surface hardware devices and access to a variety of application contexts. This breadth of research, and extensive knowledge gained on the value and limitations of digital surfaces (of all forms) in different contexts, is reflected in the following focus area discussions.

The primary purpose of Focus Area 1.1 was to understand the fit between surfaces, humans, and human activity. Focus Areas 1.2 and 1.3 targeted understanding the advantages and limitations of single and multi-surface set-ups respectively, and how best to leverage their unique advantages to support human activity. Focus Area 1.4 focused on adapting interface concepts to real-world settings, specifically how best to adapt theoretical or idealized interface concepts to particular application areas.

### **1.1 Understanding the Fit between Surfaces, Humans, and Human Activity**

This focus area targeted fundamental research into human activities with and around surfaces. Here we improved our understanding of the relationships between people and all types of surfaces, from the traditional to the digital, from large to small, from single to multiple, in co-located and distributed venues appropriate to our application areas. We closely studied human abilities that are affected by or involved directly with the use of surface technologies.

A significant amount of activity in this focus area explored the use of surfaces—

and increasingly multi-surface set-ups—to support collaborative and social endeavours, especially in co-located settings. For example, contributions were made in understanding the collaborative and cognitive benefits of large tabletop surfaces during group creativity tasks (Hajizadehgashti 2012; Scott et al. 2015); in understanding the cognitive and communicative benefits that large surfaces and multi-surface environments support collaborative sensemaking (i.e. data triaging and analysis) (Wallace et al. 2013; Kuzminykh et al. 2015); and in applying social theories of proximity, body positioning, and territoriality to improve large and multi-surface interactions (Chen et al. 2012; Marquardt et al. 2012a; Marquardt et al. 2012b; Scott 2014).

Facilitated by the recent innovations in input and surface hardware, we used our improved understanding of how people interact with and around surfaces to springboard inventions of new interaction techniques and information presentation methods for surfaces. Important contributions were made in exploiting human proxemics as interaction triggers, for instance, to better engage passersby with large surfaces in public settings (Greenberg et al. 2011; Marquardt et al. 2012a; Marquardt and Greenberg 2012; Marquardt et al. 2012b; Wang et al. 2012; Marquardt 2013; Mostafa et al. 2013). This research has had significant impact both internally within SurfNet and externally among the international surface computing research community: proxemic interactions was the topic of a dedicated invited Dagstuhl seminar workshop in 2013 (Greenberg et al. 2014b) and has contributed to numerous publications by SurfNet (Boring et al. 2014; Brudy et al. 2014; Greenberg et al. 2014a; Mueller et al. 2014; Cheung and Scott 2015a; Cheung and Scott 2015b; Ledo et al. 2015) and external researchers in subsequent years, e.g., (Henrik Soerensen and Kjeldskov 2013; Raedle et al. 2014; Dinger et al. 2015; Jakobsen and Hornbaek 2015; Zhou et al. 2015). Another key contribution in this area was the extensive exploration of novel information visualization techniques for large and multi-surface set-ups to facilitate both individual and collaborative analysis and decision-making around large and/or complex data sets (Anslow et al. 2013; Bhaskar et al. 2014; Huron et al. 2014; Oskamp et al. 2015).

## **1.2 Interacting with a Single Surface**

Our research in this theme focused on interaction issues with single surfaces: developing new input and interaction techniques; creating effective visualizations and feedback for surface interactions; generating new interfaces that promote individual and group information organization and sharing; and, exploring the interaction issues that stem from displaying information on, and interacting with, different surface form factors including horizontal, vertical, small, and very large.

Early research outcomes in the area of interface and interaction design from SurfNet and other surface computing researchers has allowed development of more advanced surface software applications designed to address real-world tasks. This shift exposed the need for more sophisticated and nuanced surface interfaces and interactions that better supported complex

task and social interactions. SurfNet adapted to meet this need. Over the past few years significant research activity focused on designing more effective feedback and awareness mechanisms to improve the usability of surface applications across a variety of different surface form factors. For example, several of our projects focused on developing interfaces that more proactively respond to people's interactions on and around the surface to help teach novice users what the system has to offer and how to effectively use the system, particularly in the case of large surfaces installed in public settings (Seto et al. 2012; Hinrichs et al. 2013; Cheung and Scott 2015a; Cheung and Scott 2015b). Contributions were also made in designing interface elements that help people understand and maintain awareness of automated system changes during ongoing collaborative tabletop activities (Wallace et al. 2012; Chang et al. 2014), and in using tactile feedback to help mediate group coordination when using virtual embodiments (e.g. virtual arms that allow for extended reach at a large surface) during tabletop collaboration (Doucette et al. 2013).

Research in this focus area also investigated the use of large surfaces as a collaboration tool, beyond their task-specific application features. Consistent observations by SurfNet researchers have revealed that when groups gather around a large wall or tabletop surface, they often want to "draw" over the task interface to help strategize, coordinate, or communicate about the task at hand. SurfNet researchers developed various mechanisms to support such abstracted "communication" interactions, including providing annotation capabilities directly into a task application (Bortolaso et al. 2014), providing an additional "add-on" program that interfaces with other software applications to provide common collaboration tools, including an annotation layer over the application software (Simonyi 2015), and visualizing above-the-table gestures in a tabletop interface to better contextualize any communication gestures made during group work (Genest and Gutwin 2012; Genest et al. 2013).

Finally, recent SurfNet work also included projects to improve individual interaction with small surfaces such as smartphones. For instance, contributions were made in improving command selection on smartphones using knowledge of ergonomics and common device grip behaviour (Gutwin et al. 2015), and in developing improved CAPTCHA interaction (a common computer security method) optimized for multi-touch smartphone use (Reynaga et al. 2015).

### **1.3 Interacting with Multiple Surfaces**

While single surfaces provide many advantages for supporting groups, each surface form factor (e.g., large, small, horizontal, vertical) has benefits and limitations. By combining multiple surfaces together we can take advantage of the specific properties of each surface type, thus enabling interfaces that are more efficient and powerful than the sum of their parts. The increased variety of surface form factors and interaction capabilities, along with the greater commercial availability of surface devices, led to a



substantial growth in research on multi-surface environments (MSEs) within SurfNet. This MSE research was also enabled by early research outcomes on single surfaces that addressed many of the basic device-specific challenges: with a stronger understanding of how to design for individual surfaces, we were better able to focus on more complex multi-surface interfaces and interactions.

Our extensive investigations on MSEs over the past few years also revealed just how challenging designing effective multi-surface interfaces and interactions can be: different surface devices have different interaction affordances and capabilities that must be combined in meaningful and usable ways. As MSEs are still relatively rare in practice, there remains a lack of design intuition about what does and does not work in given application contexts. Despite these complexities, we made significant contributions in this area, and were leaders in the international surface computing and HCI fields in the development of novel MSE interfaces and interaction techniques, evidenced by recent workshops and tutorials led by SurfNet researchers on these topics (Marquardt 2013; Anslow et al. 2014; Greenberg et al. 2014b; Isenberg et al. 2015; Scott et al. 2015). Our contributions in this area include examining the benefits and limitations of different device configurations, device form factors, and cross-device interactions during group work in different task contexts (Wallace 2011; Marquardt et al. 2012a; Marquardt et al. 2012b; Wallace et al. 2013; Scott 2014); developing new user and device tracking techniques (Marquardt et al. 2011; Genest et al. 2013; Azazi et al. 2014), and interfaces to leverage those tracking techniques, for instance, to facilitate interconnectivity of devices in a large space (Marquardt et al. 2012a; Chokshi et al. 2014; Scott et al. 2014).

While most multi-surface projects targeted co-located environments, contributions were also made in the area of distributed surfaces. These projects primarily focused on facilitating group communication at remotely connected large surfaces, for example, by displaying arm shadows that indicated a remote collaborator's above-the-table gestures during remote tabletop interactions (Genest and Gutwin 2012; Genest et al. 2013), or utilizing whole-body interaction and large surfaces to build shared virtual scenes that enable active freeplay between friends over a distance (Ledo et al. 2013).

#### **1.4 Adapting Interface Concepts to Real-world Settings**

This focus area—which combined research efforts from focus areas 1.2 and 1.3 into the exploration of possible interactions and interfaces in real-world situations—saw increasing activity over the lifespan of SurfNet. The overarching goal for this focus area was to facilitate the use of SurfNet interface designs in feature-rich surface application interfaces capable of supporting complex human activity in real-world settings. In the past few years, the range of targeted application areas grew increasingly broader. This increased breadth was largely driven by the diversity of interested application partners, demonstrating the wide appeal of surface computing

to real-world partners.

A highly active area of research was applying and adapting surface interfaces and interactions to surface software applications optimized for different usage contexts. Real-world application areas included music and media, gaming, health, command and control, creativity and design, browsing library holdings, air traffic control, computer security, security analysis, data analysis, and geospatial terrain analysis. We gained significant practical knowledge through these projects about utilizing surfaces in real-world settings. For example, there is an important design tradeoff to make between providing “simplistic” interfaces (e.g. visually streamlined, with minimal touch interaction) and providing sufficient accuracy and precision for the task at hand. For instance, in a project focused on supporting simulation training exercises for the Canadian Army, significant design iteration occurred around the design of a touch-based route-planning feature to provide the right mix of simplicity, precision, and utility for end-users (military personnel) who had limited experience with touch devices (Bortolaso et al. 2014).

Another contribution of this focus area was to invent new ways of using surfaces to address real-world problems. For example, many military missions rely on accurate terrain analysis; however, many soldiers do not know how to read traditional two-dimensional (2D) maps containing contour lines (i.e. shaded colours representing slope, relief, elevation, etc.). One project explored a multi-surface system that provided a real-time viewshed (showing the areas of visibility from a certain geographical ground position), a three-dimensional (3D) panoramic view, and a “helicopter” view controlled by an optically tracked tablet (Oskamp et al. 2015).

Significant contributions were also made in designing more effective interfaces and interaction techniques for large displays installed in public settings—a setting where potential users encounter significant social and interaction barriers to using large, especially unfamiliar, surfaces, and thus, tend to avoid using them altogether (Cheung et al. 2014). Across a number of projects, we explored different mechanisms to reduce these barriers and successfully engage passersby in public settings such as lobbies, museums, and libraries (Hinrichs et al. 2011; Thudt et al. 2012; Cheung and Scott 2015b; Thudt et al. 2015).

## **Conclusions**

Over the lifespan of SurfNet, the scope and complexity of projects have significantly increased due in part to the strong basic research outcomes of our early SurfNet efforts and in part to the changing technological landscape in the consumer domain and broader research fields. This foundation enabled us to undertake much more complex surface computing research, especially in the area of multi-surface interfaces and interactions than previously possible. We made substantial progress on our application goals of exploring the potential of surfaces in a wide variety of application

contexts; first in our target application areas, and then much more broadly to other domains such as healthcare, farming, music, computer security, etc., as the success of our early research became known outside of SurfNet and opportunities to work with increasingly diverse application partners arose.



# Using Social Science Theory to Inspire Surface Design: A Case Study of Proxemic Interactions

*Saul Greenberg and Nicolai Marquardt*

## Introduction

Designers of novel surface interaction techniques and applications are influenced by many factors. Some designers follow a mostly iterative approach to system refinement, where they seek to improve existing methods by exposing and solving inefficiencies. Some try to better understand user needs such as through observational studies and by using software engineering techniques to craft requirements analysis. Some base their work around the affordances of technical innovations, where these new technologies expose a plethora of design opportunities that were not previously possible. Some incorporate advances made in other interaction fields to surface design, where methods developed elsewhere are adapted to the surface medium. Some rely on intuitions and personal experiences, where they generate ideas, sift through them, and apply, test and refine what they consider to be the best candidate designs.

Our own approach takes a somewhat different direction: we use social science theory to both guide and inspire our research on surface designs. Our basic premise is that our understanding of human-human interaction can be applied – albeit with some caveats – to human-computer interaction (HCI).

Our design process generally follows five stages. These stages are not purely sequential. All influence one another: they often overlap and may be done in parallel. Earlier stages may be revisited based upon insights garnered in later stages.

*Stage 1. Identify candidate social science theories potentially relevant to surface interaction.* This is by no means straight-forward. There are a plethora of social science theories, and most are of little value to aid design thinking. As well, because these theories explain human-human interaction rather than human-technical interaction, they must be read with a creative eye. This can only work if it is done actively. For every theory considered,

for example, it is useful to ask “what could we do if one or more of the actors in this theory was technology (such as a large display) rather than a person?”. From that question, one can then brainstorm scenarios where the designer could try to apply that theory to a design situation. Of course, this also begs the question of where in social sciences to look. Our own experiences suggests that helpful theories can be found by reading social science texts and primers introducing theories, as these are often written at a level accessible by designers and software technologists. As well, others in the HCI field may have already suggested a link between social theory and technological design.

*Stage 2. Translate that social science theory into a form applicable to technological design.* Social science theories are cast in their own language, with their own jargon, emphasis and interpretation. They target people rather than technology. They are rarely usable by designers ‘out of the box’, simply because they do not address technological innovation or design. They often include detail that cannot be applied to design situations. Consequently, it is important to recast the theory into a form that a designer can use. This could be done, for example, by simplifying the theory into its core concepts, and recasting select portions and details of that theory into a form applicable to the technological setting.

*Stage 3. Quick and dirty prototyping.* It is one thing to know theory, but quite another to understand its ramifications to design. Our approach advocates getting our hands dirty as quick as possible, as we believe this to be the best way to reveal design opportunities afforded by that theory. This means brainstorming ideas (e.g., through sketching), and actually building a variety of simple proof of concept prototypes that can be tried out. By doing so, the designer gains immediate feedback on the utility of the theory. If the prototypes are uninspiring, or are unnatural during use, or do not seem to resonate, then it is likely that the theory is not as applicable to design as predicted. Conversely, if the prototypes generate excitement, feel natural during use, are easily explained to others, and suggest even more prototypes, then it is likely that the theory has considerable potential to design. At the same time, the designer is exposed to the technical challenges of the domain (i.e., software and hardware development), which gives insight into tool development as done in the next step.

*Stage 4. Retrenchment: Building a toolkit for rapid development.* It may be (and often is) the case that applying that theory to actual systems development may require hardware that is not readily available or suitable, and/or that software development is tricky. While it is likely possible that a few prototypes can be built by brute force (stage 3), varying those prototypes can be excessively time-consuming, thus hindering the iterative process. At this point, we advocate retrenchment, where – based on implementation experiences so far – the design team turns to developing a toolkit that will dramatically simplify the programming effort of these systems. This means that concepts that are core to the application of the theory should

be embedded into the system, where a programmer can invoke its features through a few lines in an application programmer's interface (the API). The primary motivation of toolkit development is to allow the designer and programmer to concentrate on the design and iteration of the system rather than its underlying plumbing.

*Stage 5. Robust prototype development and full research applications.* At this point, the designer should have a good understanding of the theory, along with experiences applying it to particular situations. The designer will also have a good toolkit for developing applications within the genre. This is now the time for the designer to pursue developing robust prototypes and applications, including exploring the nuances of interaction techniques. In this final stage, the designer can focus on particular problem areas and nuances within the usual human-computer interaction test/iterate cycle.

In the remainder of this paper, we will use the above stages to introduce our Surfnet project on proxemics interactions, which was built upon the social science theory of proxemics.

### **Stage 1. The Social Science Theory of Proxemic Interactions**

In 1966, anthropologist Edward Hall coined the word 'proxemics', an area of study that identifies the culturally-dependent ways that people use interpersonal distance to understand and mediate their interactions with other people (Hall, 1966). While his theory of proxemics has many aspects to it, its most basic forms define four proxemic zones that characterize how people interpret inter-personal distance. While aspects of these zones are culturally dependent, western culture typically defines distances within these zones as: intimate (~0–1.5'), personal (1.5–4'), social (4'– 12') and public (12'–25'). As these names imply, closer distances lead to increasing expectations of interpersonal engagement and intimacy. In practice, people adjust these distances not only to match their social activities, but to raise defense mechanisms when others intrude into these zones. This is something we understand intuitively, where people often adjust their positions to best fit the dynamics of their interpersonal interactions.

Hall also described how features within the space affect people's interactions. Fixed features include those that mark boundaries (e.g., entrances to a particular type of room), where people tend to organize certain kinds of social activities within these boundaries. Semi-fixed features are entities whose position can affect whether the space tends to bring people together, or move them apart (for example, the arrangement of chairs).

To understand why this theory is relevant, we need to revisit the Ubicomp vision. In 1991, Mark Weiser – recognized as the founder of Ubicomp – saw Ubicomp as technologies that disappear, where they 'weave themselves into the fabric of everyday life until they are indistinguishable from it', where computers are integrated 'seamlessly into the world' (Weiser, 1991). He

envisioned many computers per person, all inter-connected, and all with varying form factors. Significantly, Weiser envisioned the day when devices would know about their location and surroundings, where their behavior and function would depend to some extent on their environmental context (we now call this context-aware computing). As time passed, modern technology is now realizing parts of Weiser's vision, what with the common use of smart phones, tablets, laptops, large digital touch surfaces, and other information appliances. Many devices also exploit location-awareness, where the combination of global positioning systems (GPS) and compass information (location) is used in tandem with knowledge about the physical environment (e.g., nearby businesses and services).

Yet Weiser's vision of seamlessness remains somewhat elusive. For example, consider the digital ecology of the living room in Figure 1. It includes various devices (the digital surface, the information appliances, and the things people carry such as smart phones and tablets). While most devices are networked, actually inter-connecting these devices is painful without extensive knowledge, and requires time to configure and debug. Even when connected, performing tasks across devices is tedious, often requiring complex navigations across interfaces. In practice, this means that – from a person's perspective – the vast majority of devices are blind to the presence of other devices. What makes this even more problematic is that these devices are also blind to the non-computational aspects of the room – the people, other non-digital objects, the room's semi-fixed and fixed features – all which may affect their intended use.

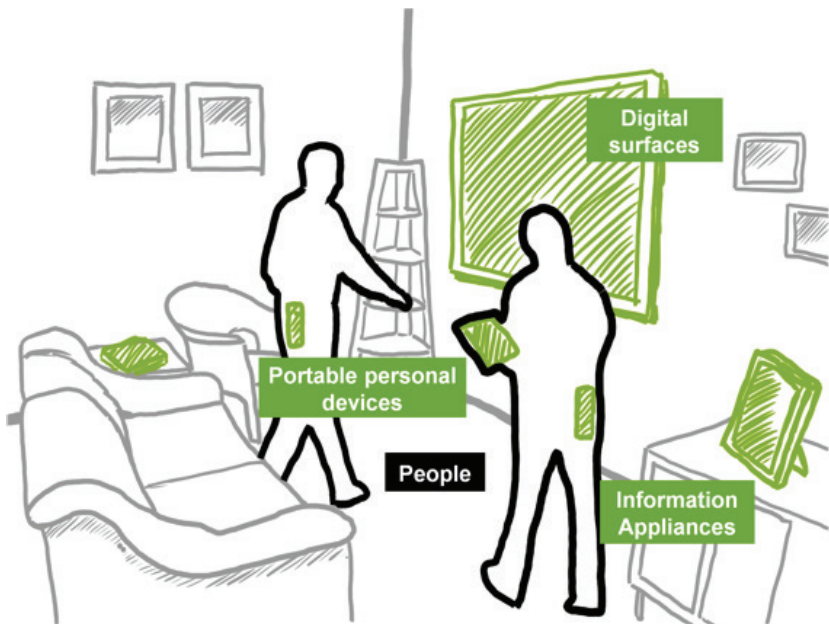


Figure 1: A typical Ubicomp ecology, including a mix of people, digital surfaces, portable personal devices, and information appliances (Ballendat, Marquardt and Greenberg 2010).

This is where we (along with a few others) saw the role of proxemics theory (e.g., see also Vogel and Balakrishnan 2001; Ju et. al. 2008). The main idea is: just as people expect increasing engagement and intimacy as they approach others (as suggested by proxemics theory), so should they naturally expect increasing connectivity and interaction possibilities as they approach devices, and as they bring their devices in close proximity to each other and to other things in the ecology.

## Stage 2. Translating Proxemics Theory to Technological Design

Proxemics theory relies both on people’s ability to sense their environment and others within it, and on people to interpret what they see to adjust their social behaviors. Technology, of course, is much more limited.

We thus had to translate proxemics theory into a form that we could use as our design foundations. The first question was “what should the system be able to sense?” where our constraints were that these sensing capabilities could be something we could operationalize and implement, that is, as proximity measures in the form of variables returned by the system. Our own notion of proxemic dimensions for Ubicomp are characterized in Figure 2 and explained below, where we consider proxemics measures between entities (entities can be people, devices, and/or physical features in the environment). As we will see, each of these dimensions can also vary by fidelity and whether they return discrete or continuous values.

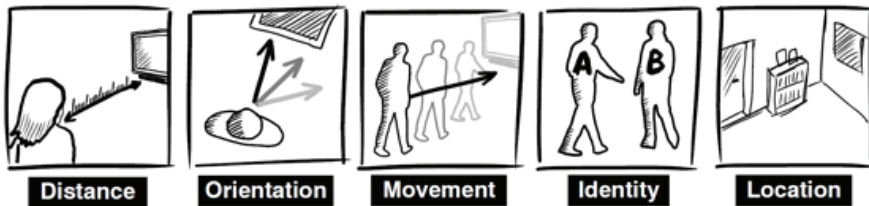


Figure 2. Five proxemics dimensions for Ubicomp (Greenberg et. al. 2011).

*Distance* between entities is a fundamental notion in proxemics theory. We normally think of distance as a continuous measure, such as a value returned between 0 - 6 feet. However, distance can also be discrete, for example, a measure of what zone an entity is in with respect to another entity. In the simplest case, distance can be considered as a binary measure, e.g., one entity is either near or not near to another entity.

*Orientation* between entities is also fundamental in proxemics theory. For example, the ‘social distance’ between two people facing towards vs. away from one another is extremely different, even though the physical distance is identical. Orientation thus captures nuances not provided by distance alone. It too can be continuous (e.g., the pitch/roll/yaw angle of one object relative to another), or discrete (e.g., facing towards, somewhat towards, or away from the other object). Of course, orientation only makes sense if an entity has a ‘front face’ to it.



*Movement* between entities captures the distance and orientation of an entity over time, where different actions can be taken depending on (for example) the speed of motion, and/or whether one entity is moving and turning towards vs. away from another entity. People naturally consider movement as part of the social distance dynamics of proxemics. Technology must be informed about that movement as well.

*Identity* uniquely describes the entity. While proxemic theory is applied to people, we expected we would apply it to a broad range of technical devices as well as physical artifacts within the environment. Thus design requires some degree of entity identification. Identity can range from a detailed measure including exact identity and attributes of that entity, to a less detailed measure such as an entity's type, to a minimal measure that simply discriminates one entity from another.

*Location context* describes the physical context that the entities reside in. People naturally consider location as part of their behaviors, for example, how a couple adjusts their distancing in a family room versus in a public setting such as a store. Yet technology is blind to context unless explicitly informed. Location measures can also capture contextual aspects, such as when an entity crosses a threshold (a fixed feature) marking its presence in a room. Location is important, as the meaning applied to the four other inter-entity measures may depend on the location's context.

While we will not delve into it here, our choice of these particular measures were heavily influenced by our thinking about how proxemics theory could address known challenges in designing Ubicomp systems (Marquardt and Greenberg, 2012). For example, one of the Ubicomp challenges we considered was establishing connections between devices as a consequence of proximity (e.g., a mobile phone and a surface). A simple thought exercise reveals the importance of distance, movement, and orientation to avoid accidental connections: i.e., a person's intension to connect the phone to the surface would be triggered by pointing and moving the phone towards the surface until a particular close distance is reached. Identity is, of course, important for security reasons. Location context is similarly important, for it may allow some people to connect (e.g., an employee using a board room, where the connection re-establishes particular information) but disallows others (e.g., an unescorted visitor).

### **Stage 3. Quick and Dirty Prototyping**

We then developed many quick and dirty prototypes, often using some quite simple technologies. Various examples are described in detail in Marquardt and Greenberg 2015 and in Greenberg et. al. 2011, as well as in many individual research publications. For example, one of our first prototypes used simple off the shelf range finders as a way to control connection and privacy in an always-on media space (Greenberg and Kuzuoka, 1999). The idea was that people would be able to see and hear each other in increasing fidelity as a function of both actor's proximity to their displays.

Our second prototype realized a cartoon actor (a face) on a large surface. Using a few fairly simple proxemics rules, the face would react to people's distance, movement and orientation. For example, its eyes would track the moving person. The face would verbally greet an approaching person, smile as they came closer, frown and get annoyed if they were too close, be sad when they turned away, and so on (Diaz-Marino and Greenberg 2010). We found this application interesting because (a) people with no technical background immediately understood the system's behaviors in terms of how it reacted to their distance, movements, and orientation, and (b) this was in spite of the system following only a few simple proximity-based rules to drive its behavior (it had no artificial intelligence). For our third prototype, we wanted to see what we could do if we added proximity awareness to a traditional presentation tool running on a vertical surface, where the speaker would not have to use a second display or a keyboard. We focused on two specific capabilities: we wanted to make it easier for a speaker to access their speaker notes, and we wanted to make it easier for a speaker to control their slides. For example, when the speaker faced the audience, slides were presented in full. However, if the speaker faced the screen and stood close to one of its sides, speaker notes along with a few navigation controls appeared in the corner closest to the speaker. If the speaker shielded the display from the audience by standing near the middle of the surface, a scrollable deck of slide thumbnails appear, allowing the speaker to rapidly switch to any slide.

These and other applications influenced our thinking about proxemics. They helped solidified our translation of proxemic theory into operational variables (as discussed in the previous stage), and they also influenced our design of the first version of our proximity toolkit (the following stage, discussed next).

#### **Stage 4. Building a Toolkit for Rapid Development**

Building proxemics-aware applications are challenging. While rough measures of distance can be captured by range finders, their accuracy proved less than ideal. Capturing other parameters, such as orientation and directional movement proved even more difficult. Programming raw input streams from these sensors was tedious. Simply put, the technical effort of building these systems meant that we spent more time programming the underlying plumbing, which came at the expense of exploring the design space of proxemics.

We turned to a new goal, where we wanted to simplify the exploration of interaction techniques by supplying fine-grained proxemic information between people, portable devices, large interactive surfaces, and other non-digital objects in a room-sized environment. Our solution was the Proximity Toolkit (Marquardt, Diaz-Marino, Boring and Greenberg 2013). The toolkit offered three key features. First, it facilitated rapid prototyping of proxemic-aware systems by supplying developers with the orientation, distance, motion, identity, and location information between entities, all accessible

via simple-to-program callbacks. Second, it included various tools, such as a visual monitoring tool, that allows developers to visually observe, record and explore proxemic relationships in 3D space, which helped them understand the data being generated by the toolkit before any coding was actually done. Third, its flexible architecture separated sensing hardware from the proxemic data model derived from these sensors, which meant that a variety of sensing technologies can be substituted or combined to derive proxemic information. We initially based our hardware infrastructure on the Vicon Motion Capture system, where the system would return millimeter-accurate data about an entities position in 3D space. However, later versions incorporated other sensing systems, such as the lower-cost Optitrack motion capture system, and the consumer-affordable Microsoft Kinect depth-sensing camera.

Callbacks follow standard programming conventions to track events. For example, consider a simple scenario where a programmer wanted to display information only if a person was facing the display. The callback would be something like:

```
void OnDirectionUpdated (ProximitySpace space, DirectionEventArgs args)
{
    if (args.ATowardsB)
        //Person is facing the display, show content]
    else
        //Hide content
}
```

We developed several versions of the toolkit over a few years. While it took considerable time and effort to do so, the result was well worth it. Programmers with only a brief introduction to the toolkit were able to create proxemics-aware applications almost immediately. More importantly, complex applications could be built, where programmers could concentrate and iterate over the design of particular proxemics-aware systems.

### **Stage 5. Robust Design and Development.**

By this stage, we had developed a solid understanding of proxemics and how it could be applied to the design of systems supporting proxemics interactions. We also had a toolkit that let us actually build, maintain, and iterate through fairly complex proxemics-aware systems. A few examples illustrate what we could do.

*Proxemic Media Player* is a media player that reacts to the proximity of one or more people in a room (Ballendat, Marquardt and Greenberg 2010). Figure 3 illustrates only a few of its functions. At distance (a), the person enters the room. The media player recognizes both the person's identity and entrance, activates the display, shows a short animation, and then displays four large video preview thumbnails held in that person's personal media collection at a size suitable for distance viewing. At distance (b) the person is moving closer to the display. The display responds by showing an

increasing number of his videos by continually shrinking the video preview thumbnails and titles to fit. At distance (c), the person is very close and he can select a video to watch by directly touching its thumbnail, which shows him more about the selected video: a preview that can be played and paused, with detailed title, authors, description and release date. The text is small, but quite readable at this close distance. Finally, at distance (d) the person moves away from the screen to sit on the couch. The system responds by expanding the currently selected video to play in full screen view. When seated at the couch, the person can also point his mobile phone towards the display. The phone is recognized as a pointing device, which in turn can be used to control the media player. If a second person enters the room, the video shrinks slightly to expose the title of the video being played. If that second person then approaches the screen, a description of the video is revealed. When all people leave the room, the video playback stops.

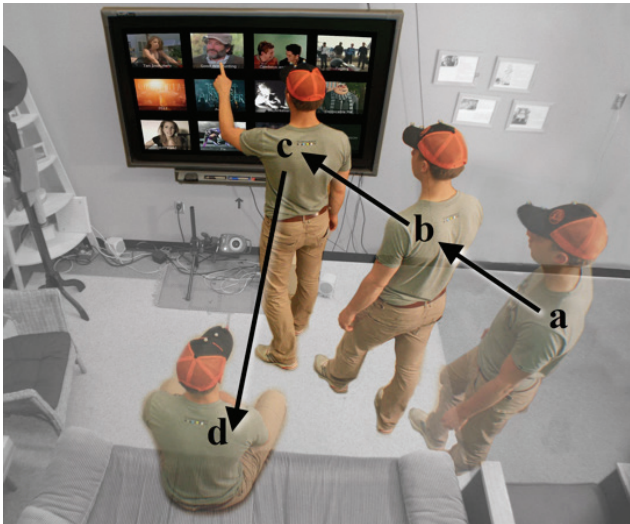


Figure 3. Proxemics Media Player. The position of a person in the room is shown at the top, where letters correspond with what the surface is displaying at those distances (Ballendat, Marquardt and Greenberg 2010).

*The Gradual Engagement design pattern* is a generalizable interaction technique that describes what we believe is a successful way to exploit proximity (Marquardt, Ballendat et al., 2012). The general idea is that we can design devices and interfaces that interpret decreasing distance and increasing mutual orientation between a person and a device within a bounded space as an indication of a person's gradually increasing interest in interacting with that device. The generalized gradual engagement design pattern includes three key phases:

- Phase 1: background information supplied by the system provides awareness to the person about opportunities of potential interest when viewed at a distance;

- Phase 2: the person can gradually act on particular opportunities by viewing and/or exploring its information in more detail simply by approaching it; and
- Phase 3: the person can ultimately engage in action if so desired.

This pattern is directly inspired by the proxemic theory mentioned earlier, and characterises what we thought was the 'best' of how we, and others previously, apply proxemics to Ubicomp design. The intention of this gradual engagement pattern is to characterise how we can facilitate interactions between a person or multiple people and the devices surrounding them by leveraging fine-grained proxemic measurements (e.g., distance, orientation, identity) between all entities. As a design pattern, it helps unifying prior work in Proxemic Interactions, synthesizing essential, generalizable interaction strategies, and providing a common vocabulary for discussing design solutions.

We noticed that many of our early designs incorporated the idea of gradual engagement, for example, the media player, where details of the videos available are revealed as a person approaches the surface, and where interaction techniques are tuned to allow finer interactions (using touch) when the person enters the intimate zone. Furthermore, the Gradual Engagement design pattern also informs and inspires other possible designs, and allows for variations of the pattern applied to different domains. The remaining examples illustrate this broad application of the pattern.

*Gradual Engagement Pattern for Cross-Device Information Exchange.* In this first example, we applied the design pattern to mediate device-to-device operations. In particular, we refined the gradual engagement pattern to ease the information transfer task, where the refined pattern suggests how devices can gradually engage the user by disclosing connectivity and information exchange capabilities as a function of inter-device proximity. That is, as people move and orient their personal device towards other surrounding devices, the interface progressively moves through three stages affording gradual engagement.

1. Awareness of device presence and connectivity is provided, so that a person can understand what other devices are present and whether they can connect with one's own personal device. We leverage knowledge about proxemic relationships between devices to determine when devices connect and how they notify a person about their presence and established connections.
2. Reveal of exchangeable content is provided, so that people know what of their content can be accessed on other devices for information transfer. At this stage, a fundamental technique is progressively revealing a device's available digital content as a function of proximity.
3. Interaction methods for transferring digital content between devices, tuned to particular proxemic relationships and device capabilities, are provided via various strategies.

Each method is tailored to fit naturally within particular situations and contexts. As one part of this pattern, Figure 4 demonstrates the proximity-dependent progressive reveal of digital content stored on personal devices when collaboratively interacting with a large shared interactive whiteboard.

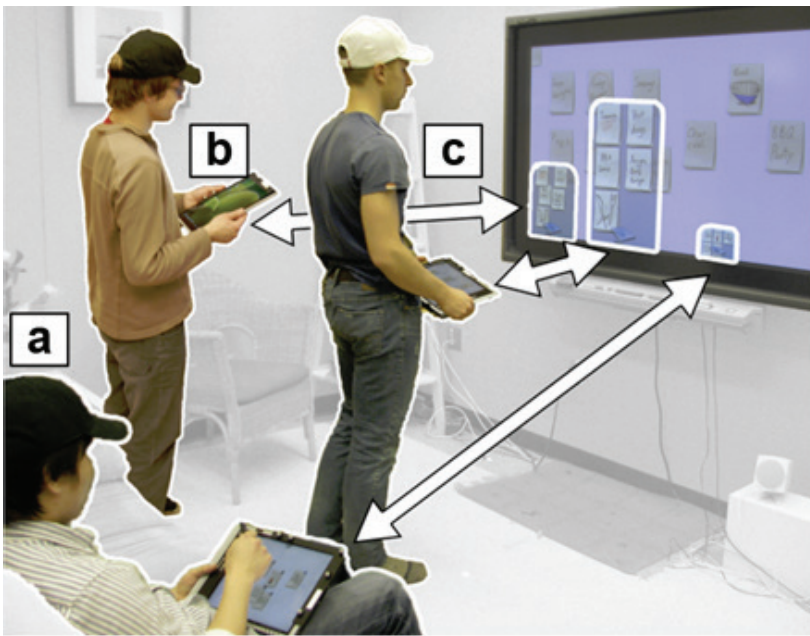


Figure 4. Proximity-dependent progressive reveal of personal device data of multiple users at different distances to the display: (a) minimal awareness of a person sitting further away, (b) larger, visible content of a person moving closer, and (c) large awareness icons of person standing in front of the display (Marquardt, Ballendat et al., 2012).

*Gradual Engagement with Proxemic-Aware Advertisements.* A second application of the design pattern was the Proxemic Peddler that explores how future advertisement displays might try to grab and keep a passer-by's attention (Wang, Boring and Greenberg, 2012). A digital advertisement board – in this case a book-selling display – reacts to the presence, distance, identity, orientation, and movements of a nearby person. The key is to do so in a non-aggressive and non-annoying manner that finds a balance between the advertiser's interest and the passer-by's interest. When no-one appears within its range, it rapidly animates a book list at the bottom, where its motion is an attempt to attract the attention of a passer-by. The animation slows as soon as it detects a passer-by looking towards it (which makes the book list readable and far calmer), as illustrated in Figure 5, upper left. The gradual engagement pattern is then applied, where additional personalised details about preferred books are displayed as the person approaches the display (Figure 5, upper right). If the person momentarily looks away, subtle cues are used to try to re-attract them, such as a slight shaking of the product icon (Figure 5, lower left). If it looks as if the person is about to

leave, it tries to regain their interest by showing different products (Figure 5, lower right). In all cases, it gives up gracefully if it looks like the person is really not interested.



Figure 5. Proxemic Peddler (Miaosen Wang).

*Proxemic-based remote controls leverage Proxemic Interactions in order to mediate the control of appliances in a person's Ubicomp environment (Ledo, Greenberg, Marquardt, Boring 2015). Using a mobile device (e.g., phone or tablet, Figure 6 left) as a personal control device, a person can initially point around the room in order to scan which devices are available. Items coming into view on the display are the ones generally in front of the device. The person can then gradually increase the control of a particular appliance simply by moving closer to it. More details about the appliance's current status and activity are shown on the screen, and the interface reveals further control options to take action. For example, in Figure 6 (right) the progressively revealed stages of a temperature control interface to a physical thermostat are shown, from small icons on the left progressing to detailed graph views of recent activity on the right. In summary, these proxemic-aware controls are an alternate yet complementary way to interact with appliances in people's environments via a mobile device. Through spatial interactions, people are able to discover and select interactive appliances and then progressively view its status and controls as a function of physical proximity. This allows for situated interaction that balances simple and flexible controls, while seamlessly transitioning between different control interfaces.*



Figure 6. (Left) Proxemic-aware remote controls: remote control interface on a tablet computer; (Right) thermostat interface, showing a series of progressively revealed interaction controls on the remote control's screen (Ledo, Greenberg, Marquardt, Boring 2015).

## Summary

This chapter described the five interleaved stages of a research pattern, where its basic premise is to use social science theory to motivate design. Using proxemics theory as a case study, we illustrated how we applied this pattern to co-develop the design notion of Proxemic Interactions along with a toolkit and a broad set of prototype systems.

We are sometimes asked if our work is driven by theory, or whether it is just inspired by theory. The answer is perhaps a bit of both. With theory-driven research, we rely on that theory to frame the behavior of our system-as-actor, where the behavior should correspond (at least to a reasonable extent) to that theory. Similarly, we rely on the theory and its nuances to explain and predict how people will likely respond to our design ideas. However, we do not blindly follow the theory, as we recognize that technology cannot simply be substituted in place of one of the humans. We allow ourselves to go beyond the theory. That is, we use the theory as a starting point to help inspire designs, but are not concerned when our interaction ideas stretch that theory or go beyond what the theory says. We are also open to creating new 'theories' that incorporate technology as one of the actors. For example, our design pattern of gradual engagement is a theoretical variation of proxemics. As such, the gradual engagement pattern offers an interaction technique that can be applied to many technology settings, and that incorporates what we believe are good technological behaviors that are easily understood and beneficial to people.

Design creativity does not have to occur in a vacuum. This chapter offers social science theory a contributor to both the initial design spark and for shaping design alternatives over the course of the design process. Our book "Proxemic Interactions: From Theory to Practice" (Marquardt & Greenberg, 2015) adds considerable detail to what is provided here.





## Effects of Tabletop Embodiments on Coordination

*Carl Gutwin, Andre Doucette, Regan Mandryk, Miguel Nacenta, and David Pinelle*

(Portions of this chapter previously appeared in the following published papers: David Pinelle, Miguel Nacenta, Carl Gutwin, and Tadeusz Stach. 2008. The effects of co-present embodiments on awareness and collaboration in tabletop groupware. In Proceedings of Graphics Interface 2008 (GI '08). Canadian Information Processing Society, Toronto, Ont., Canada, Canada, 1-8; Andre Doucette, Carl Gutwin, Regan L. Mandryk, Miguel Nacenta, and Sunny Sharma. 2013. Sometimes when we touch: how arm embodiments change reaching and collaboration on digital tables. In Proceedings of the 2013 conference on Computer supported cooperative work (CSCW '13). ACM, New York, NY, USA, 193-202.)

### Introduction

Tabletop groupware systems allow co-present collaborators to work together over a shared horizontal display. Tables are a natural site for group work, both because of their ubiquity in the real world, and because their physical characteristics support coordination and communication, such as the face-to-face orientation of people around the table, the central location of work artifacts, and the use of direct touch to manipulate objects on the work surface. Direct touch – where people manipulate objects by touching them with a pen or a fingertip – provides a number of benefits for collaboration. In particular, the use of people's real arms and hands provides obvious awareness information about 'who is working where' on the table, and makes it easy to watch other people work. However, direct touch also has disadvantages: it can be difficult for people to reach objects that are far away; arms and bodies can get in the way of each other, preventing people from working in the same space at the same time; and it can be awkward or uncomfortable to work close to another person. One way to deal with these problems is to use relative rather than direct input techniques – that is, techniques where each person manipulates a cursor rather than touching objects directly. Relative input techniques allow reaching to any part of the table and allow people to work in the same place, but since they do not use people's physical arms, this source of awareness information is lost.

The only awareness information produced by a relative input technique comes from the virtual embodiment of the user (e.g., their cursor) on the

table. This visible representation provides each user with feedback about their own actions, but as a side effect, also provides awareness to other members of the group. Although this is the same mechanism by which real arms convey awareness, virtual embodiments are much less obvious.



Figure 1. Left: large table. Right: artifacts and arm embodiments.

In this chapter, we examine the ways that different virtual embodiments on tabletops affect collaboration. In particular, we look at the implicit coordination that people carry out when they work on tasks in a shared space. One hallmark of physical coordination on tables is that people almost never touch one another, because social norms prevent them from getting close enough for collisions or conflicts. In most situations, it is considered impolite to cross over or under another person's arm while reaching across a table, and it is considered rude to touch or bump into them. This behaviour on tables may stem from people's natural touch avoidance (Anderson and Leibowitz, 1978), which affects our spatial interactions with others, or it may be an attempt to avoid disrupting another person's activities (for example, getting in their way or occluding their view of the workspace).

Whatever the reason, people's unwillingness to carry out movements that result in touch or collision is consistent and predictable. The phenomenon is so reliable that groups use this mechanism to coordinate shared access to tabletop space. For example, laying an arm around an area on the table defines a personal territory and blocks others from taking 'protected' items. Even though the arm presents a minimal barrier, it advertises that others will have to commit an impolite act (crossing) to take items. Similarly, our natural touch avoidance makes us aware of others' movements. This provides a coordination benefit, in that selection and placement conflicts are rare; instead, people negotiate turn-taking. People interacting over digital touch tables can also gain these coordination advantages, because they use physical arms and hands to manipulate objects. However, it is not clear whether these benefits will continue when people use virtual embodiments.

To understand this question, we investigated crossing and touching behaviour on physical and digital tables, using both physical and virtual embodiments. First, we observed dyads carrying out tasks on physical tables, finding that crossing and touches are extremely rare. Second, we recreated the same task on a digital table, but implemented several

different arm embodiments. We considered two issues: whether digital embodiments altered behaviour compared to physical arms, and how visual factors of embodiment design (size, transparency, and realism) affected user behaviour. In addition, we explored whether a dyad's level of intimacy affected their behavior.

Our studies provide five main findings:

- Crossings with physical arms are exceedingly rare, but are common with all types of digital embodiments,
- The size of digital embodiments is the most important factor of visual embodiment design; realism had little to no effect,
- Subjective perceptions of awkwardness and invasion of space differ between physical and digital embodiments,
- Relationship had a strong overall effect on the number of crossings, but did not interact with the other factors,
- Perception of awareness differs for physical and digital embodiments and is also affected by all visual factors.

This work shows the marked difference between behaviour with physical and digital embodiments, suggesting that the sense of touch may be more important in touch avoidance than the visual sight of touching; and we discuss the design implications for supporting space management issues in digital table environments.

### **Touch and Personal Space in Physical and Digital Environments**

Touch is the most intimate interpersonal communication channel. It is "... the most carefully monitored and guarded, the most vigorously proscribed and infrequently used, and the most primitive, immediate and intense of all communicative behaviours" (Thayer 1986, p.24). Touch has many functions: it can demonstrate dominance (Major and Heslin, 1982), increase compliance (Patterson et al., 1986), and even increase tips in a restaurant (Zweigenhaft, 1986). Body-accessibility research has shown that people's comfort level with being touched on different parts of their body depends on who is doing the touching, where the touch occurs, and the type of touch (Jourard, 1966). Results showed that people are comfortable when others touch their arms and hands, regardless of gender (Nguyen et al., 1973) or relationship (Heslin et al., 1983); however, the social norms of table interactions dictate that it is rude to reach over or under another's arm to reach an item. This may be due to people's natural touch avoidance (Anderson and Liebowitz, 1978), or may be due to personal space norms (also called interpersonal distance), which define comfortable minimum distances between people (Hall, 1966). Personal space is moderated by many factors, including age, relationship, culture, and gender (Hayduk, 1983). Although invasions of personal space are generally avoided, research suggests that people can mediate the uncomfortable feelings of an imposed invasion by changing

another behaviour (e.g., not making eye contact in a crowded elevator).

People use metaphors from the physical world when learning digital systems (Carrol and Thomas, 1982), and previous researchers have shown that personal space also exists in digital environments. For example, in immersive virtual environments, people stand farther away from virtual humans that engage them in mutual gaze (Bailenson et al., 2003), the same effect as with fellow humans. People also assign personal space to avatars. For example, people use gaze avoidance to compensate for personal space invasions (Yee et al., 2007), and researchers found that people were uncomfortable with invasions of their avatar’s personal space (Slater and Steed, 2002). Users treat their avatar’s personal space as they would their own; invasions (e.g., standing too close) are uncomfortable (Smith et al., 2000); and people avoid actions that could cause others to be uncomfortable (e.g., walking through another’s avatar) (Slater and Steed, 2002). Physical group spacing is also similar for avatars, forming a circle and facing each other during speech (Smith et al., 2000).

### Observational Study on a Physical Table

To begin our investigation into tabletop reaching behaviour, we carried out an observation-and-interview study of people working with paper artifacts at a physical table. Ten dyads (1 female pair, 6 male pairs, 3 mixed pairs) were recruited from the local university. Participants were instructed to build a haiku (a three-line poem) by arranging words cut from a sheet of paper and placed on the tabletop (Figure 2, left). The two participants built their haikus at the same time, each on a different topic, and assembled the words on the table in front of where they were sitting. Words were scattered around the table and were available to either of the participants; however, the words related to the left participant’s topic were on the right side of the table, and vice versa. Participants had to reach to the other side of the table to retrieve the most appropriate words for their haiku (e.g., see Figure 2, right), which created the potential for many reaching conflicts in a short session.

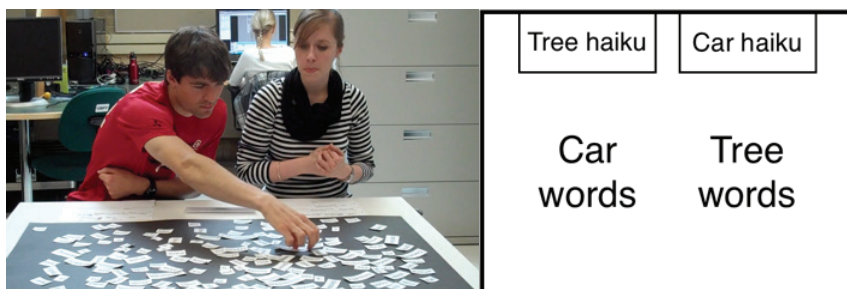


Figure 2. Study setup (left), and word distribution (right).

We observed two clear behaviours in the study – touch avoidance, and territoriality – both of which led to specific kinds of space management strategies on the tabletop. First, it was very clear that people avoided touching the other person’s arm or hand. Over ten sessions, with hundreds

of reaching events, we observed only three crossings (i.e., where one person reached over or under the other person's arm). This behaviour was consistent across all groups, and was verified by watching video recordings of the sessions. In interviews, participants repeatedly stated that it was rude to reach over or under another person's arm, and that they avoided doing so. When we asked the three people who had been crossed how these episodes felt, all three said that they noticed when it happened, and that they felt uncomfortable – the other person was invading their personal space.

Touch avoidance led to two mechanisms for managing table access: implicit coordination, and accommodation. We observed nascent reaching conflicts where both people simultaneously began reaching to the same area; however, these never became selection conflicts (where both people grabbed the same object) as groups used coordination techniques to avoid selection conflicts. People also leaned back slightly when the other person reached in front of them; this subtle behaviour was observed in all groups. People reported that they moved away not because the closeness of the other's arm made them uncomfortable, but because doing so would let the other person work without feeling uncomfortable about reaching into their personal space. This accommodation technique provides a subtle and low-effort means for giving permission to reach into personal space.

The second obvious behaviour that we observed was territoriality (Scott et al., 2004). The main way that territoriality was exhibited in the study was that people immediately adopted the area in front of them as their personal territory. This organization is normal for tabletop work (Scott et al., 2004), and was also encouraged by the setup of the study, because people were told to build their haiku in the area in front of where they were sitting. However, we also manipulated the sense of ownership in the public space of the main table, by reversing the arrangement of topic words (described above).

### **Digital Table Study**

To investigate reaching behaviour in the digital world, we replicated the haiku-building task used in our physical-table study on a digital tabletop. We were interested in two main research questions: how physical and digital embodiments differ in terms of reaching, and whether the visual design of a digital embodiment affects reaching and collaboration. To study how users behave with digital arm embodiments as compared to physical arms, we compared four digital embodiment designs to reaching with a physical arm. We varied three factors of digital embodiment design: size, transparency, and realism. The larger an embodiment (size), the more likely others are to notice it; however, it also occludes more of the workspace. The more transparent an embodiment, the less prominent it is, and the less it might affect a collaborator's actions. Realistic-looking embodiments may cause people to treat them more like digital extensions of a user.

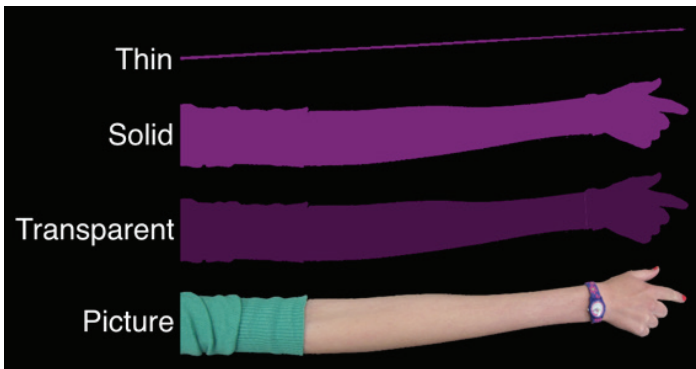


Figure 3. The four arm embodiments.

### Embodiment Conditions

When dyads arrived, we took a picture of each person's right arm to be used as their embodiment. The image was displayed between the cursor location and the right side of their haiku paper for their embodiment (Figure 4). As users reached farther onto the table, the arm image was stretched. By using an image of the participant's arm, shape of the embodiment was kept constant for all conditions. We tested one physical embodiment and four digital embodiments that varied in the previously identified visual factors of embodiment design. People used a mouse to control the cursor location when using digital embodiments.

- Thin: the embodiment image was scaled to 5 pixels wide, and filled in with purple or green to differentiate users.
- Transparent: the unscaled embodiment image (approx. 200 pixels wide; everyone's arm is a different size and shape) was filled in with purple or green and made semi-transparent (set at 60% opacity), so users could see the words through the embodiments.
- Solid: the unscaled embodiment image (approx. 200 pixels wide; everyone's arm is a different size and shape) was filled in with purple or green and was opaque.
- Picture: the unchanged image of the user's arm (same size as the transparent and solid conditions).
- Pens: the physical arm. In this condition, people moved words using direct touch on the tabletop - a cursor appeared below the tip of a pen and the embodiment in this case was their physical arm. Pen (cursor) location was tracked using a Polhemus Liberty tracker, and selection occurred via a button at the tip of the pen controlled by a Phidget interface board. Pens were used instead of a touch table to track hand locations at all times, not just during object selection.



Figure 4. Embodiment occlusion (left), and example of the system (right).

## Results

To answer two main questions on collaboration over digital tables using arm embodiments, we collected a variety of dependent measures, which are grouped into the following three themes of understanding collaboration.

- **Touch Avoidance** – We used the number of crossing events (when the two embodiments cross each other) as an objective measure of the lack of touch avoidance. In addition, we asked participants to rate their feelings of awkwardness when crossing embodiments.
- **Territoriality** – We calculated the number of word pick up and drop events on either side of the table, as well as on their partner's piece of paper, as an objective measure of territorial behaviour. In addition, we asked participants to rate how awkward it felt to reach to the other side of the table, and their feelings of invasions of person space, with each embodiment type. Lastly, we asked them to rate their level of ownership of various interface elements.
- **Awareness** – We asked participants to rate their feelings of awareness of their partner's embodiment table location.

We present analysis for each theme by the factors of visual embodiment design presented in the previous section. Our planned comparison for each factor was: Physicality (Pens to Solid), Size (Thin to Solid), Transparency (Transparent to Solid), and Realism (Picture to Solid). Effects of relationship are included for each theme.

### Touch Avoidance

There was a main effect of embodiment on the number of crossing events ( $F(4,116)=30.02$ ,  $p\approx 0.000$ ,  $\eta^2=0.53$ ). The pairwise comparisons in Table 1 show that there were significant effects of physicality and size on the number of crossings, but not of transparency or realism. Figure 5 shows that physicality was the dominant factor affecting touch avoidance as measured by crossings. Although there was a main effect of relationship on the number of crosses ( $F(2,27)=4.45$ ,  $p=0.021$ ,  $\eta^2=0.25$ ), there was no interaction with embodiment ( $F(8,108)=1.27$ ,  $p>0.05$ ,  $\eta^2=0.09$ ). As Figure 5 shows, Strangers crossed fewer times than Romantics ( $p=0.016$ ), Acquaintances

did not significantly differ from Strangers or Romantics ( $p > 0.05$ ).

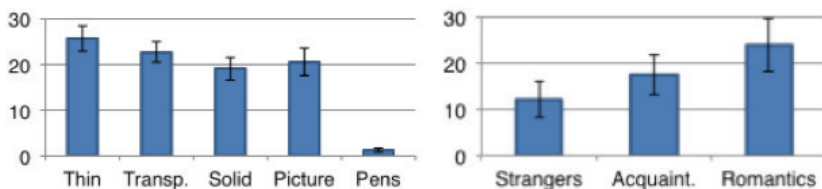


Figure 5. Mean number of crosses by embodiment (left), and by relationship (right).

We asked participants to rate their agreement with the statement: “It felt awkward to cross embodiments with this embodiment”; results are shown in Figure 6. A Friedman test showed a main effect of embodiment on participants’ feelings of awkwardness crossing embodiments ( $n_2 = 58.69$ ,  $p \approx 0.000$ ). As Table 1 shows, there were significant effects of physicality, size, and transparency, but not realism. A Kruskal-Wallis test showed no main effect of relationship on any ratings of awkwardness of crossing embodiments (all  $n_2 < 3.53$ ,  $p > 0.17$ ).

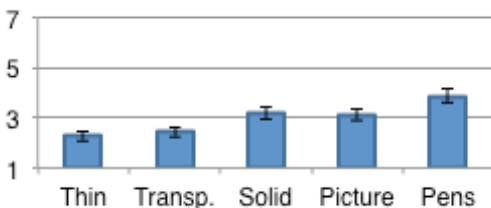


Figure 6. Feelings of awkwardness of crossing embodiments.

### Territoriality

We also collected data on the territorial behaviour of participants. Previous work in territoriality (e.g., Scott et al., 2004) showed that people’s reaching behaviour was mediated by the location of items on the table. There was no main effect of embodiment on the number of words picked up and dropped on either side of the table. On average, people picked up 15.5 words from the ‘other’ side of the table, and 12.0 words from ‘their’ side of the table (recall that the other side of the table contained more relevant words). We interpret this to mean that people grabbed the words they wanted, regardless of location. We also asked participants to rate their agreement with the statements, “I felt like my partner was invading my space” and “I felt like I was invading my partner’s space” (see Figure 7). Friedman tests showed a main effect of embodiment on participants’ feelings of being invaded by their partner ( $n_2 = 52.66$ ,  $p \approx 0.000$ ) and of invading their partner’s space ( $n_2 = 63.69$ ,  $p \approx 0.000$ ). As Table 1 shows, participants felt less awkward invading and being invaded with increased transparency and decreased size. Participants felt more awkward being invaded with a physical embodiment (Pens), but there was no effect of physicality on the feeling



of invading space. Realism did not affect the awkwardness of invading or being invaded. A Kruskal-Wallis test showed no effect of relationship on feelings of being invaded with all embodiments (all  $n_2 < 0.695$ ,  $p > 0.17$ ) except Picture ( $n_2 = 8.00$ ,  $p = 0.018$ ). Acquaintances were different than Strangers and Romantics (both  $p < 0.02$ ). A Kruskal-Wallis test showed no main effect of relationship on the ratings of invading partner's space (all  $n_2 < 2.35$ ,  $p > 0.309$ ).

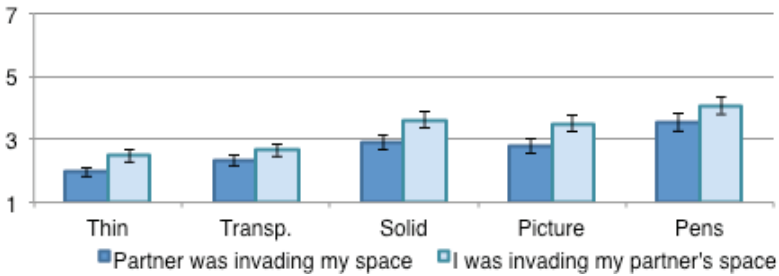


Figure 7. Feelings of being invaded, and of invading partner.

Participants had complete freedom constructing their haikus and we did not specially instruct them on whether they were allowed to reach onto another user's paper. Only 15 of the 30 groups ever accessed words on their partner's paper (3 Strangers, 6 Acquaintances, 6 Romantic couples), and there were large variations in the amount of this activity in the dyads Strangers invaded their partner's paper sparingly (1-2 times), Acquaintances did so more often (1-11 times), and Romantic couples invaded most of all (3-96 times). Half of the groups did not invade their partner's paper; they stated they did not realize that they would be able to do so.

Theme	Measure	Physicality (Pens-Solid)	Size (Thin-Solid)	Transparency (Transparent-Solid)	Realism (Picture-Solid)
Touch avoidance	Number of crosses	Fewer crosses ( $p=0.000$ )	More crosses ( $p=0.016$ )	No difference	No difference
	Feelings of awkwardness	More awkward ( $p=0.017$ )	Less awkward ( $p=0.000$ )	Less awkward ( $p=0.000$ )	No difference
Territoriality	Proportion of events on opposite side	No difference	No difference	No difference	No difference
	Feelings of awkwardness reaching to other side	More awkward ( $p=0.000$ )	Less awkward ( $p=0.001$ )	Less awkward ( $p=0.000$ )	No difference
	Feeling of being invaded	More invaded ( $p=0.021$ )	Less invaded ( $p=0.000$ )	Less invaded ( $p=0.000$ )	No difference
	Feeling of invading partner's space	No difference	Less invading ( $p=0.000$ )	Less invading ( $p=0.000$ )	No difference
Awareness	Feeling of awareness	More aware ( $p=0.018$ )	Less aware ( $p=0.000$ )	Less aware ( $p=0.038$ )	More aware ( $p=0.010$ )

Table 1. Pairwise comparisons showing the effect of each factor as compared to Solid (e.g., Pens had fewer crosses than Solid).

We also asked people to report their level of ownership over table items on a 5-point scale (1="no ownership", 5="complete ownership"). Although people felt more ownership over their paper (mean=4.07) and the words on their paper (3.75) than over their partner's paper (1.97) or words on their partner's paper (2.05), people did not differentiate ownership of words on

the opposite side of the table (2.71) from words on their side of the table (2.9). There were no main effects of embodiment on these ratings.

In addition to finding out how participants behaved with and felt about visual embodiments, we asked two free-text questions about crossing embodiments. We grouped participant responses into categories based on the words used (one response can appear in multiple categories). When responding to the question, "briefly describe why you avoid crossing over (or under) someone's physical arm", people reported because it is rude, impolite, uncomfortable, or awkward (33), it is an invasion of personal space (19), and it causes a performance – occlusion, interruption, and distraction – cost to my partner (19). When responding to the question, "briefly describe how crossing over (or under) someone's physical arm is different than crossing over (or under) someone's digital embodiment", people reported embodiments can't "feel" (26), the embodiment is not "me" or "them" (18), and the embodiments don't have or invade personal space (14).

In addition to clear evidence of touch avoidance (as described above), we also observed instances of implicit coordination and accommodation (e.g., see Figure 4). One coordination policy we observed with the pens was that some people 'planned out' the words they wanted, and then quickly reached out and grabbed the words, making a pile on their paper, and then organizing into sentences. We also observed instances of apologies while reaching with the pens, often when someone reached in front of the other person, or during a turn-taking coordination episode (i.e., the hallway passing effect). This is in contrast to the apologies we observed for the digital embodiments, which were much fewer (e.g., "I can't see, move your arm - oh sorry" or "you stole my word - oh sorry").

When first using a new digital embodiment, people would often poke at each other's embodiment. Two groups referred to this as 'sword fighting'. We also observed people poking at the other's embodiment when they were done building their haiku. In that case, it seemed to be done explicitly to annoy the other person, even though the other person cannot feel it and may not even notice it happening. While searching for words, many people reached out with their embodiments and moved it across the screen as a guide (similar to how a mouse cursor is used to scan a document or website). This happened with all digital embodiments, though was less common with occluding embodiments (Solid and Picture). Users never scanned in the same manner with the physical embodiment.

Another observed behaviour unique to digital embodiments was that people left their embodiment stretched out when they were done interacting with their haiku. This is akin to throwing down your scissors into the center of the table when you are done with them. Most people realized after a few minutes that their embodiments were 'in the way', and moved them. People never threw down their Pens.

## Discussion

The user study shows five main results:

- Physical arm crossings are exceedingly rare (fewer than two per session, on average), but are common with all types of digital embodiments (twenty or more);
- The size of digital embodiments is the most important factor of visual embodiment design; realism had little to no effect;
- Subjective perceptions of awkwardness and invasion of space differ between physical and digital embodiments;
- Relationship had a strong overall effect on the number of crossings, but did not interact with the other factors;
- Perception of awareness differs for physical and digital embodiments and is also affected by all visual factors.

### Differences Between Physical and Digital Embodiments

People rarely crossed physical arms, but had little issue crossing digital embodiments (even when they looked like their own physical arms). The main reasons for this dramatic difference lie in the way people felt about the arms' connection to the real bodies, and the lack of any touch sensation. First, most participants reported that they did not associate the digital embodiments with their own, or their partner's, actual body: several people said that the embodiments were "not me" and "not my partner;" others stated that the digital embodiments did not have personal space. We saw further evidence in the lack of proprioception with the digital embodiments – people often left their digital arms 'laying out on the table,' something that would likely never happen with real arms. Second, participants stated that the digital embodiments cannot "physically touch" or have no sense of feeling, and so the awkwardness of crossing was removed.

These statements clearly imply that people perceive physical touch differently than a visual representation of touch, even if that visual representation is dynamic and realistic. The touch avoidance first seen in the physical-table study appears to be dependent on a true sensation of touch rather than a representation. Although this appears to be a simple finding, it is in part dependent on the fact that representations of arm crossing are not subject to social norms; it is possible, however, that other representations of touch (e.g., 'holding hands' touches) might not be seen as being as neutral as crossing. Nevertheless, in our tabletop systems, the lack of true touch in digital arm embodiments appears to remove most touch-avoidance behavior. This has strong design implications, because people may perform actions in the digital world that they would strongly avoid in the physical world (e.g., crossing over an outstretched arm to steal an item).

### Territoriality

People did not extend their private territories in front of them beyond their

pieces of paper. This may be because we swapped the word locations, which forced people to reach into what otherwise might be the other person's territory. We also did not allow people to create their own territories in the public workspace. The system automatically moved words back to their original location when they were dropped anywhere outside of pieces of paper. Our territoriality results also suggest there is an effect of relationship on territorial behaviour (which has not been reported before). The more intimate the relationship, the more likely people are to invade personal territories. In addition, although people's public workspace territorial behaviour was different than reported in other research, people's subjective responses matched previous work (e.g., people are uncomfortable reaching to the other side of the table (Hornecker et al., 2008)).

### **Occlusion and Digital Embodiment Size**

Although not nearly so strong as the effect of physicality, we also saw an effect of embodiment size on crossings and awareness. Figure 6 and Table 1 show the same trend: the larger an embodiment is, the more aware people feel of their partner, and the less they cross. In addition, increased size was also paired with more feelings of awkwardness reaching to the other side of the table. These effects are likely due to both the increased visual prominence of the larger embodiments, and the increased likelihood that the arm will occlude artifacts on the table and disrupt the partner's activities. Many of the free-text responses stated that people were concerned about disrupting their partner's work, both with physical and digital embodiments. We speculate the cause of the differences in behaviour and subjective results between the digital embodiments was directly related to the level of occlusion caused by that embodiment. The lack of effect for Realism (Picture vs. Solid) provides additional evidence for this hypothesis, because both Picture and Solid occluded the workspace to the same degree.

### **Implications for Design**

There are five issues from this research that designers should consider when developing tabletop systems.

- Touch input (real arms) vs. indirect (digital embodiments). When designing tabletop systems, designers must choose the way that people will interact with the table. In some cases, indirect touch (and digital embodiments) has been proposed as a way to simplify reaching on large tables. Our study shows that this decision can greatly impact the way that people use the system; as a result, designers should think carefully about the ramifications of different choices. For example, designers might use only real-arm touch input when selection conflicts could lead to severe errors; with real touch, people will be more aware of their partner and less likely to come into conflict over the table.
- Visual realism does not reproduce social protocols. The study showed that no purely visual design reproduced the degree of touch avoidance seen with physical arms. This means that designers will not be able to re-introduce

social control mechanisms simply through appearance (although several participants found the picture arms 'creepy', this did not produce additional touch-avoidance). As a result, systems that use digital embodiments may need to build in explicit access control to prevent uncontrolled access.

- Lack of awkwardness could be useful. In some situations, such as fast-paced tasks or games, people may be able to complete their work faster when they do not have to worry about making others uncomfortable. In these cases, designers could choose digital embodiments to allow for comfortable crossings, and narrow embodiments to avoid occlusion. However, this decision also means that actions will be less obvious, decreasing awareness.
- Relationships change behaviour. Reaching and territoriality behaviour is strongly dependent on the relationship of the users. This is important for public installations (e.g., museums), where the system may be used by anyone. Designers who know the relationship of their users can choose embodiments that fit the relationship type – for example, if users are more familiar with one another, access control mechanisms might be required.
- Occlusion is an important factor in embodiment design. Of the visual factors we investigated, size was the only one that had an effect on behaviour. In general, people did not want to disrupt others (this was true even for intimate couples). Transparency is easy to build into arm embodiments, and provides a good combination of visual salience (for awareness), but without occlusion.

### **Directions for Future Research**

Touch avoidance provides people with a natural way of avoiding conflict, but without true touch, alternate means of managing access to the table will be needed. First, access could be controlled at the system level through roles or permissions. Previous CSCW work on explicit roles and access provides the control required and provides solutions to conflicts, but these methods are often too heavyweight to be used in practice. We plan to explore new possibilities for light-weight access controls for tabletops (e.g., touching an object to reserve it for a short time).

Second, new social protocols may appear as people become more experienced with digital embodiments. The changes that we saw may have occurred because people have so little exposure to these techniques. With more experience, groups may develop new coordination methods – for example, they may start to associate digital touching with the negative implications of physical touching, or may develop other mechanisms that do not depend on touch avoidance (e.g., more explicit turn-taking behaviours).

Third, systems could increase the costs of crossing, to try and create embodiments that behave more like physical arms. In the physical world, it takes longer to reach over or around another's physical arm, so we could introduce a performance cost to crossing behaviour (e.g., increasing the C/D ratio during a crossing event). Similarly, because touch avoidance is based

on tactile sensations, it may be possible to reintroduce these sensations (e.g., by vibrating the mouse) when crossings occur. These added costs could cause people to behave with arm embodiments as they do with physical arms, recreating the real-world protocols.

Our results suggest it will be important to know more about systems that allow multiple types of input and embodiment. For example, systems that combine direct and indirect input will have the two embodiments mixed together. We speculate people would have little issue crossing an arm embodiment over a physical arm, but more study is needed. Remote collaboration over distributed tables is another mixed setting: both people interact with direct touch, but are represented remotely via an arm embodiment (Tang et al., 2006).

Our work looked at the change from a physical form to a representational form, and how this changes behaviour. We chose arm embodiments as our representation and touch avoidance as the behaviour. Although we lose touch avoidance with this representation, feelings of awkwardness and invasion are still present, so other protocols may also remain. For example, touching certain parts of another's avatar with your avatar's arm may still be considered rude, even though neither person can "feel" that touch.

## **Conclusions**

In this paper, we presented two studies of tabletop reaching behaviour: a physical table study, demonstrating that people rarely cross arms, and a digital table study, demonstrating the marked difference between reaching with physical and different digital arm embodiments. We showed that the most important factor in the visual design of embodiments is the level of occlusion caused by the embodiment: the lower the occlusion, the less people are aware of each other's actions, the less awkward it is to interact in shared spaces, and the more people cross embodiments. This research is an important step in understanding the differences between physical and digital group interactions, opening up many new questions on what factors tabletop designers should manipulate to ensure that groups are able to work as naturally as they do over physical tables.



## **Cross-Device Content Transfer in Table-Centric Multi-Surface Environments**

*Stacey D. Scott, Guillaume Besacier, Phillip McClelland, Julie Tournet, Nipun Goyal, and Frank Cento*

### **Introduction**

There has been increasing interest in the surface computing community to use small, personal surfaces, such as tablets or smartphones, in conjunction with large surfaces, such as interactive walls and digital tabletops. Combining personal and large surfaces into a functional multi-surface environment (MSE) introduces new design challenges. For example, effective mechanisms are needed for transferring content across different surfaces to allow the most flexible use of the available personal and large surfaces. Significant cross-device transfer research exists in the Human-Computer Interaction (HCI) and Computer-Supported Cooperative Work (CSCW) fields, particularly in the area of multi-device environments (MDEs) (Rekimoto and Saitoh, 1999; Nacenta et al., 2005; Nacenta et al., 2009; Wallace et al., 2009; Wallace, 2011). This research has yielded many useful cross-device transfer techniques (see Nacenta et al. (2009) for a review). Yet, most of these techniques rely on mouse-based, or otherwise device-aided, input capability that is unavailable in touch-based MSEs. For example, a popular cross-device transfer technique is Rekimoto's (1997) PICK-AND-DROP (P&D) technique, which relies on a digital pen to transfer content from one display to another.

To address this limitation, we conducted a series of three studies to systematically investigate how existing cross-device transfer techniques could be applied or adapted for use in touch-based MSEs. These studies focused on cross-device transfer in a tabletop-centric MSE (T-MSE) context, where a small group of people, each with an individual multitouch tablet, were engaged in a joint activity around a multitouch digital tabletop. The first study examined how two popular cross-device transfer techniques (a "virtual portals" technique (explained below) and the aforementioned P&D technique) could be applied (or adapted) to a "current" T-MSE setup. In this T-MSE, the digital tabletop was unable to distinguish between different users interacting with the tabletop—a limitation of most current multitouch digital tabletops. It therefore posed unique challenges for

cross-device transfer during multi-user interactions. The second and third studies continued the investigation of the P&D technique, further evolving its design adaptation in each subsequent study to better optimize its use for T-MSEs and the specific application task context. The latter two studies focused on a “future” T-MSE set-up that was able to differentiate between users interacting with the tabletop. This capability built on new above-the-surface sensing methods from SurfNet (Genest et al., 2013) and the broader surface computing research community (Hilliges et al., 2009; Pyryeskin et al., 2012; Haubner et al., 2013).

In the remainder of the chapter, we provide an overview of existing cross-device transfer mechanisms, and discuss their limitations for touch-based MSEs. Next, we overview the DOMINION game as the application context for the three studies. We then overview our study methodology, which remained relatively fixed across the three studies. Next, we report each study. Full, detailed versions of Studies 1 and 2 have previously appeared in HCI literature (Scott et al., 2014a; Scott et al., 2014b); thus, only select findings are included in their respective study sections. Study 3 is a previously unpublished follow-up study that investigated design limitations of the P&D adaptation explored in Study 2. Finally, we reflect on insights learned from these investigations and their implications for cross-device transfer in T-MSEs.

### **Cross-Device Transfer in Multi-Surface Environments**

(Components of the background presented here were also reported, in full or in part, in earlier publications on Study 1 (Scott et al. 2014a), Study 2 (Scott et al. 2014b).)

Cross-device transfer is an active area of research in MSEs, and the broader area of multi-device environments. Also, to address reach and ergonomic issues related to dragging digital objects over a large distance, single-surface object transfer techniques have been developed that minimize the need for long drag-and-drop actions. This section overviews these single-surface transfer mechanisms first, followed by the mechanisms used to move content across multiple devices. As all three studies explored the Pick-and-Drop (P&D) technique, this mechanism, and its applicability to touch-based T-MSEs, is discussed in detail.

*Object Transfer across Large Surfaces (Within-Device Transfer).* Using direct-touch interaction to drag digital content across a large surface has several known ergonomic issues, including fingertip discomfort due to friction and arm and finger fatigue. Moreover, some locations are difficult to reach. Therefore, drag-and-drop extensions have been developed for moving content across large surfaces, including techniques that move an object onto a distant object (e.g. a folder) or location (Baudisch et al., 2003; Hascoët, 2003; Collomb et al., 2005; Collomb and Hascoët, 2008; Doeweling and Glaubitt, 2010). Techniques have also been developed that leverage the physicality of direct-touch surfaces, such as tossing or flicking interaction gestures that use pseudo-physics to “propel” objects to distant



locations (Scott et al., 2005; Weber et al., 2008; Wilson et al., 2008). The aforementioned P&D technique has also been used to transfer objects from one location to another on pen-based interactive wall and tabletop surfaces (Haller et al., 2010). Further, P&D has been shown to be more efficient than drag-and-drop in these contexts (Rekimoto, 1998). Another approach is to move objects from one surface location to another by using “virtual portals,” where an object placed on a portal (typically a virtual interface container or widget) in one location then appears on an associated portal in another location (Besacier et al., 2007; Voelker et al., 2011). The above single surface transfer techniques, especially those designed for direct-touch environments, provide useful inspiration for touch-based cross-device transfer.

*Object Transfer across Multiple Devices (Cross-Device Transfer).* Existing cross-device transfer techniques broadly fall into three main categories: moving content across contiguous virtual workspaces; moving content via a virtual portal; and moving content via a physical proxy.

Contiguous virtual workspace techniques are based on the physical configuration of displays in the environment. In this approach, displays are connected to a common software architecture that maintains awareness of the physical configuration of the displays (static or dynamic configurations are possible). The display configuration information is then used to provide a contiguous virtual workspace across devices. Thus, moving an object off the edge of one display moves it to the nearest edge of the adjacent display (Rekimoto and Saitoh, 1999; Streitz et al., 2001; Johanson et al., 2002; Hinckley et al., 2004). For example, in PointRight (Johanson et al., 2002), several large screen displays and an interactive tabletop share a single mouse pointer. A static adjacency map, based on the room topology, determines where the pointer moves when it leaves the edge of a screen. In Stitching (Hinckley et al., 2004), an ad-hoc adjacency map is created, with the system inferring the user’s intention to join two adjacent displays when a “stitch” gesture is drawn, starting on one display and ending on a second display. This map can then be used to move digital artefacts between connected tablet computers. Marquardt et al. (2012) propose a similar tablet-to-tablet transfer capability between adjacent tablets, but instead of using a connection gesture they establish the initial ad-hoc connection by tilting one tablet towards the other.

A disadvantage of the contiguous virtual workspace approach for transferring digital objects between a tabletop and a personal surface is the asymmetric size of the displays. The large edges of the tabletop do not map well to the small edges of a tablet or smartphone. The virtual portals technique mentioned above can be used to resolve this issue by providing a dedicated portal area on each device for transferring content (Hinckley et al., 2004; Bachl et al., 2011; Fei et al., 2013). We examined a virtual portals method called BRIDGES in Study 1. The previous two cross-device transfer approaches require people to drag the transferred object to and

from the virtual portal (or display edge) from its origin and to its destination. This can introduce the aforementioned ergonomic issue related to long-distance touch-based dragging. Physical proxy techniques address this intermediary interaction step issue by using a physical object to manage the transfer. They allow for collection and placement of the transferred object directly from its origin to its destination on the respective displays by taking advantage of the three-dimensional space around the displays. This approach involves binding a digital object to a physical object and then moving the physical object to the target display. This typically requires a system-recognized object to facilitate the binding/unbinding process, such as a digital pen (Rekimoto, 1997; Baudisch et al., 2003; Haller et al., 2010; Scott et al., 2014a) or “puck” (Kobayashi et al., 2008). For example, P&D (Rekimoto, 1997) allows someone to “pick up” a digital object at its original location using a digital pen and “drop” it directly at the destination location using the digital pen. This technique evokes the commonly used drag-and-drop concept, and bears strong similarity to the familiar action of lifting and relocating a physical object.

Given the more direct origin-to-destination interaction process, physical proxy techniques like P&D are highly desirable in T-MSEs. They reduce intermediary drag actions across a large tabletop surface, and so, provide more efficient interaction and avoid the ergonomic issue of long distance dragging. Thus, we were highly interested in using P&D in our T-MSE applications. However, the touch-based interaction and the multi-user nature of T-MSEs introduced difficulties for applying P&D in this context; we discuss these issues further below.

*Applying PICK-AND-DROP to Touch-based, Multi-User T-MSEs.* In touch-based surfaces, no digital pen (or other readily available physical object) is available to serve as the proxy for P&D transfer. In our research, we address this by using the user’s hand as the physical proxy between the tabletop and a personal tablet. This allows someone to “pick-up” the object using a menu or gesture on the tabletop, move their hand to their tablet and then “drop” the object by touching the tablet (and vice-versa). However, in a collaborative T-MSE, multiple people may wish to simultaneously transfer content between various devices. In this situation, the system needs to associate the correct picks with the correct drops, which is only possible if the system knows who is doing what in the environment.

Because people often bring and, exclusively use, their own personal devices in a group setting, a reasonable design strategy in a T-MSE context is to associate a specific user with a specific personal surface (e.g., a tablet, smartphone), and to assume that all interactions with that device are made by that person (i.e., the device “owner”). Using this strategy, we can then assume that all picks or drops on a given personal device are performed by the device owner. Knowing who is doing what on the shared tabletop is more challenging. Indeed, most existing tabletop systems cannot distinguish between different users. Thus, automatically associating picks or drops with

a given person is more difficult, and requires some design adaptation of the P&D technique or additional user-identification system capabilities.

In Study 1, we addressed this issue by providing dedicated “personal territories” along the tabletop edge in front of each group member. Any picks or drops conducted in these territories were associated with the “owning” user, enabling simultaneous, multi-user P&D transfers. In Studies 2 and 3, we addressed this issue by augmenting our tabletop with user-identification capabilities, as detailed in the Study Methodology section below.

## Research Approach

Figure 1 summarizes the overall research approach used across the three studies, including the study research questions, the cross-device transfer techniques included in the studies, and the T-MSE environments used in the studies. The figure shows the progression from Study 1’s comparison of two existing cross-device transfer approaches (Bridges virtual portals vs. P&D physical proxy) to Study 2 and 3’s investigation of successive design refinements of a single cross-device transfer approach (P&D) to improve its usability in T-MSEs. Each successive study focused on addressing interaction issues revealed by the previous study. The following section details the specific study methodology that was used in the studies.

	Research Question	Cross-Device Transfer Methods		Tabletop Environment
<b>Study 1</b>	<i>How well do existing transfer methods support T-MSEs?</i>	BRIDGES (virtual portals)	VS (TERR.-ADAPTED) PICK-AND-DROP (physical proxy)	“Today’s Tabletop” (no user-identification) + Personal Tables
<b>Study 2</b>	<i>Does SURFACE GHOSTS feedback improve effectiveness of P&amp;D in T-MSEs?</i>	PICK-AND-DROP W/ SURFACE GHOST feedback	VS PICK-AND-DROP W/O SURFACE GHOST feedback	“Tomorrow’s Tabletop” (with user-identification) + Personal Tables
<b>Study 3</b>	<i>Does adding tablet feedback improve transfer awareness during P&amp;D in T-MSEs?</i>	P&D W/ SURFACE GHOST + TABLET feedback	VS P&D W/ SURFACE GHOST only feedback	“Tomorrow’s Tabletop” (with user-identification) + Personal Tables

Figure 1. Overview of studies conducted to investigate cross-device transfer in T-MSEs.

## Study Methodology

(Components of the methodology presented here were also reported, in full or in part, in the earlier publications on Study 1 (Scott et al. 2014a) and Study 2 (Scott et al. 2014b).

All studies utilized a mixed-methods research methodology that involved quantitative and qualitative study measures. All studies were conducted in the same controlled human-computer interaction laboratory environment at the University of Waterloo.

*Participants.* In all studies, participants were recruited both from the University of Waterloo student and staff population and from local board game stores’ clientele through email lists, social media sites, and posters. To promote natural group behaviour, participants were required to sign-

up with one or two friends, depending on the study, and to have previous experience with the commercial version of the DOMINION game. Table 1 summarizes the participant details for each study.

Study	Number of Participants	Gender of Participants	Age of Participants	Group size	Number of Groups
Study 1	28	23 Male, 5 Female	20-44 (M=27, SD=6.5)	2 people	14
Study 2	18	16 Male, 2 Female	20-38 (M=26, SD=5)	3 people	6
Study 3	24	19 Male, 9 Female	18-59 (M=30, SD=9)	2 people	12

Table 1. Participant details for each study.

*Experimental Design.* Studies 1 and 2 each included only one independent variable in a single factor (Study 1: transfer technique, Study 2: visual feedback) within-subjects study design, with three levels for each factor in each study. Study 3 included two independent variables in a two-factor, 2 (tablet feedback) x 2 (tabletop feedback), mixed methods design where the tablet feedback was a within-subjects factor and tabletop feedback was a between-subjects factor. This more complex study design is further detailed in the main Study 3 section. Table 2 summarizes the study conditions utilized for each study.

		Within-Subject Factors			Between-Subject Factors	
Study 1	Factor:	<i>Cross-device transfer technique</i>			n/a	
	Factor Levels:	TERR.-ADAPTED PICK-AND-DROP (TA-P&D) w/ IMPLICIT CONTROL	TA-P&D w/ EXPLICIT CONTROL	BRIDGES		
Study 2	Factor:	<i>Tabletop feedback during P&amp;D transfer</i>			n/a	
	Factor Levels:	SURFACE GHOSTS w/ IMPLICIT OWNERSHIP	SURFACE GHOSTS w/ EXPLICIT OWNERSHIP	No Feedback		
Study 3	Factor:	<i>Tablet feedback during P&amp;D transfer</i>			<i>Tabletop feedback during P&amp;D Transfer</i>	
	Factor Levels:	TABLET BRIDGE visualization	No Feedback (NO BRIDGE)	n/a	SURFACE GHOSTS w/ IMPLICIT OWN.	SURFACE GHOSTS w/ EXPLICIT OWN.

Table 2. Summary of the main experimental design details used in each study, separated by the respective within- and between-subjects factors.

*Experimental Task.* The DOMINION Game. DOMINION is a 2-4 player medieval themed card game, in which each player builds their own personal deck to utilize during game play by “buying” cards from a bank of shared card decks. Game play in DOMINION typically occurs on a turn-by-turn basis, though players can take some actions during other players’ turns. In a typical turn, a player draws a minimum of five cards from their deck, and then makes several card-based actions (e.g. revealing (i.e. “playing”) one or more cards to “buy” resources, “attacking” other players (i.e., forcing them to discard cards), or discarding unused cards). Players monitor their opponent’s game actions and may alter their game strategy in response to an opponent’s actions.

To facilitate investigation of cross-device transfer in this game, a custom digital tabletop software application of the DOMINION game was developed that incorporated the use of multiple, portable tablets to provide each player a private digital space (Figure 2). In this digital DOMINION game, cards can be freely moved and rotated using direct touch manipulation.

When two cards are moved to the same position, they are automatically stacked into a deck of cards. A card may be drawn from a deck of cards by touching and dragging the top card, while the whole deck can be moved by dragging its border.

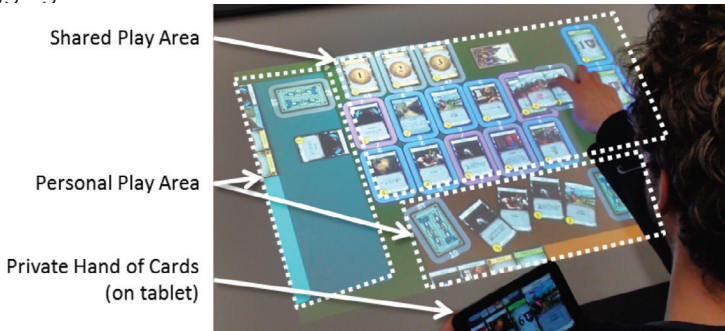


Figure 2. DOMINION digital tabletop system. The Personal Play Area denotes the personal territories used in the BRIDGES condition in Study 1. These colour-delimited areas were omitted in the PICK-AND-DROP conditions in all three studies (from Scott et al. 2014a).

Cards and decks can be flipped via a contextual pie menu invoked by tapping on a card or deck. Decks can also be shuffled with this menu. In both cases, a short animation confirms the action. In the study conditions reported in this chapter that involve PICK-AND-DROP (P&D) transfer, card “picks” were also performed via this menu.

*Equipment and Setting.* Studies 1 and 2 utilized a custom-built infrared laser light plane (LLP) multitouch digital tabletop with a surface size of 90x130 cm and projected display of 1280x800 pixel resolution (see Figure 2). Study 3 utilized an upgraded custom-built multitouch tabletop incorporating a 4K (3840x2160 pixel) resolution 55-inch flat-panel LED display fitted with a PQLabs infrared multitouch frame. In Study 1, participant pairs sat at adjacent sides of the tabletop. In Study 2, the 3-person participant groups sat at three adjacent sides of the tabletop, with the middle player seated at the long side of the rectangular table. In Study 3, participant pairs sat facing each other at the long sides of the tabletop. This change in seating arrangement from Study 1 was made due to the wide screen configuration of the upgraded table and, consequently, the larger disparity in the length between the long and short sides.

In all studies, participants were each provided a 7-inch Galaxy Tab tablet computer. Tablets were preconfigured to be associated with the player’s position at the table to facilitate the cross-device transfer methods under study. Separate laptops were set-up on nearby desks for administration of the study questionnaires. Study questionnaires were administered through the SurveyMonkey® (<http://www.surveymonkey.com>) online data collection service. In all studies, the DOMINION tabletop software application used TUIO multitouch events. In Studies 1 and 2, an infrared camera under the table and the open-source toolkit Community Core Vision (CCV) ([59](http://</a></p></div><div data-bbox=)

ccv.nuigroup.com/) were used to process touch. In Study 3, the PQLabs input frame natively produced TUIO data. Finally, in Studies 2 and 3, user identification of tabletop touches and above-the-table arm movements were obtained using a Microsoft Kinect mounted 1.5m above the digital table and an adapted version of the KinectArm toolkit (Genest et al., 2013), as described in Scott et al., 2014b.

*Procedure.* Participants performed the main study activities together in a group of 2 (Studies 1 and 3) or 3 (Study 2), but completed written forms and questionnaires individually. Upon arriving, participants first completed informed consent forms and a background questionnaire that gathered demographic information and their prior game play experience. They were then given a short demonstration of the experimental hardware systems. Each participant group played three games in a row, one for each study condition. The order of presentation of the three conditions was counterbalanced. In addition, three different sets of ten previously selected Dominion cards were used for the banks of purchasable cards, always presented in the same order to avoid interfering with the counterbalancing of the conditions. Learning effects related to card sets were not anticipated, as all players had previous experience with DOMINION.

Before the first condition, players were given a brief demonstration of the system. In Study 1, each cross-device transfer method was also demonstrated before each condition. In Studies 2 and 3, the P&D technique was only demonstrated at the beginning of the study. Most groups also took 4-5 minutes at the start of each game to read aloud the description of each available card in the bank for the session. After each condition, players completed a post-trial questionnaire about that condition. After the final game and post-trial questionnaire were completed, participants either completed a post-experiment questionnaire (Studies 1 and 2) and/or interview with the researchers (Studies 2 and 3). Finally, participants were thanked and paid for their participation. All three studies were approved by the university's institutional ethics review process.

*Data Collection and Analysis.* In all studies, quantitative and qualitative data were collected and analyzed. Participant interactions with the tabletop and tablets were captured in computer log files. Video data (with audio) and observer notes captured participants' verbal and non-verbal behaviour during the sessions. Background, post-trial, and post-experiment questionnaires included closed- and open-ended questions. All post-trial feedback questions utilized a 7-point Likert-style rating scale to capture participant perceptions and experiences in each condition.

Different qualitative analysis approaches were used across the three studies, characterizing the diminishing exploratory nature, and increasingly hypothesis driven goals of each successive study. In Study 1, the video data and open-ended participant responses underwent an extensive qualitative analysis, including open coding to reveal interaction and communication

patterns, as well as incidents of confusion or frustration and development of flow diagrams to represent emergent interaction strategies (Beyer and Holtzblatt, 1998), to better understand the advantages and disadvantages of each studied cross-device transfer technique (BRIDGES and TA-P&D). Details of full analysis is reported in McClelland (2013); only relevant themes and insights are reported in this chapter. In Studies 2 and 3, the video and open-ended participant responses were reviewed for patterns and emergent themes to provide context and deeper understanding of the quantitative results.

The Likert scale data from the post-trial questionnaires were analyzed using Repeated-Measures Analysis of Variance (RM-ANOVA). To account for the non-independence of group member responses, group was used as a dependent factor by using seating position at the table as the additional repeated measures factor. Thus a 3 (Condition) x 3(or2) (Seating Position) RM-ANOVA was conducted. As seating position was not expected (and was not found) to significantly impact the study measures of interest (e.g. awareness of cards being transferred, awareness of cards being transferred by a partner), we only report the main effects related to Condition in this chapter. An alpha-value of  $\alpha=.05$  was used to determine statistical significance.

## **Study 1**

The goal of Study 1 was to explore the potential of existing cross-device transfer approaches for supporting transfer in a T-MSE. Of the three main approaches discussed above, the contiguous virtual workspace approach was ruled out due to the previously mentioned display size disparity issue in T-MSE settings that can introduce confusion about where objects should be placed or will appear during transfer on the different sized devices. As mentioned, the virtual portals approach resolves this issue by bounding interaction to visible containers in the interface that indicate where object transfer can occur. As the physical proxy approach uses point-to-point transfer, rather than moving objects via the display borders, the display size disparity does not impact its use. Thus, we chose to include a virtual portals technique and a physical proxy technique in the study.

*BRIDGES Interaction Design.* For the virtual portals technique, we implemented a version called BRIDGES, in which a visible container widget was provided along the tabletop edge in front of each user (called the TABLETOP BRIDGE), and along the top edge of each personal tablet (called the TABLET BRIDGE). For the purpose of the study, the location of the BRIDGES were fixed, and the virtual connection between each user's TABLETOP BRIDGE and their TABLET BRIDGE was established during study set-up and fixed throughout the study. This restriction was deemed appropriate due to the nature of the experimental task—a "sit down" card game. In use cases where users are expected to move around the tabletop, the T-MSE could be augmented with proximity or user-tracking sensors to flexibly allow users' TABLETOP BRIDGES to follow them around the

environment, similar to the proximity-based virtual portals technique used by Fei et al. (2013).

In the context of the Dominion game, when a card was transferred to either the TABLET or TABLETOP BRIDGE the top half of the card would appear on the TABLETOP BRIDGE and the bottom half of the card would appear on the TABLET BRIDGE. Once on the BRIDGE, the card can be moved onto the target device by dragging it off the corresponding BRIDGE, moved back to the originating device by dragging it off the originating BRIDGE, or simply left on the BRIDGES.

*Territory-Adapted-Pick-and-Drop (TA-P&D) Interaction Design.* For the physical proxy method, we adapted the P&D method to the “current” T-MSE constraints discussed above in a version called TERRITORY-ADAPTED-P&D (TA-P&D). In TA-P&D, the T-MSE was divided into different spatial territories. A personal territory was provided along the tabletop edge in front of each user, a shared territory covered the rest of the tabletop workspace, and a private territory was provided on each person’s personal surface. Each user’s personal territory was virtually connected to their personal tablet (private territory), and this connection remained fixed throughout the study. A “pick” conducted in a user’s personal territory was associated with that user, which allowed them to subsequently “drop” the transferred object on their personal tablet. Similarly, picking up an object from their personal tablet allowed them to drop the object onto their personal territory, without interfering with others’ tabletop interactions.

Within the context of the DOMINION game, tabletop picks were enabled via a context menu that could be opened by tapping on a card (or deck of cards). (While the use of a context menu for initiating the “pick” action was originally due to technique limitations in implementing a “pick-up” grab gesture in our original hardware and software, it turned out that this approach later allowed for in-game efficiencies that were very popular and often requested by our players, such as multi-card pickup menu options, that would have been very difficult to achieve using gesture interaction.) Successive taps on the menu allowed for multiple cards to be picked up and then transferred together to a different location. Cards being transferred could then be dropped either back on the tabletop by tapping in the user’s personal territory or dropped onto the user’s tablet. Dropping the cards on the tablet required a “swipe-down” gesture from the top of the tablet screen (i.e. a downwards drag action) to avoid interference with card manipulation actions. For convenience, if the tablet interface was empty, the user could tap anywhere on the tablet interface to drop transferred cards. A “swipe-up” gesture on the tablet (i.e. an upwards drag action) initiated a “pick” action from the tablet. Several cards could be transferred together performing multiple successive pick actions on the tablet before tapping on the tabletop.



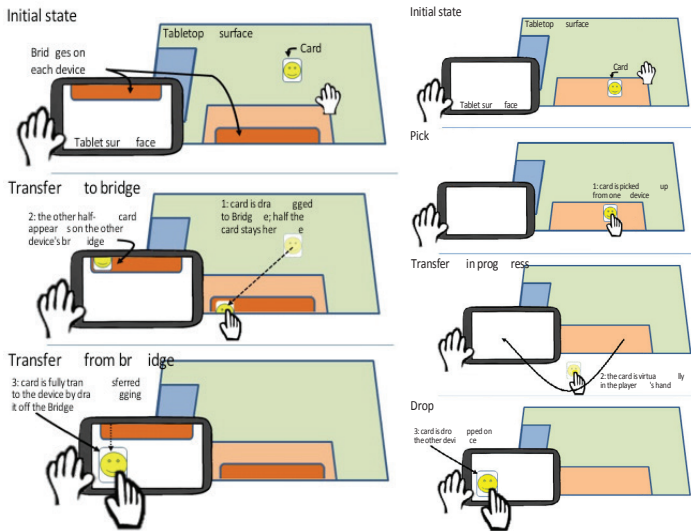


Figure 3 (left). BRIDGES cross-device transfer (Scott et al. 2014a); Figure 4 (right). TERRITORY-ADAPTED-PICK-AND-DROP (TA-P&D) cross-device transfer (Scott et al. 2014a).

Two control mechanisms were implemented to allow a user to temporarily perform pick and drop actions in the shared territory on the tabletop. An IMPLICIT CONTROL mechanism allowed a user to touch and hold any empty spot in that person’s personal territory to extend it to also cover the shared territory, allowing them to temporarily pick or drop cards directly in the shared territory. An EXPLICIT CONTROL mechanism allowed a user to place a digital token labelled, “I Control the Centre” in their personal territory to extend their territory to cover the shared territory, and to allow them to pick or drop cards directly in the shared territory.

*Post-Transfer State.* An important design consideration when implementing a cross-device transfer mechanism within the context of a card game application like DOMINION is the two-sided nature of the game cards (i.e. each card has a back and front side). The simplest approach—to retain a card’s face-up/down state at its originating point when it is transferred—would introduce significant interaction overhead post-transfer. For instance, most cards in the tabletop decks are initially face-down to preserve the secrecy of the card’s value. Yet, users are likely to want all cards on their personal tablet to be face-up, as this space is private from others’ view (unless they chose to disclose the tablet contents). Thus, the common game action of moving five cards from tabletop decks to one’s personal tablet would require significant amounts of tedious turn-over actions after these transfers. Consequently in both the BRIDGES and TA-P&D techniques, cards transferred to a personal tablet were automatically turned face-up, regardless of their originating face-up/down state.

In transfers to the tabletop, the face-up/down state varied by technique. With TA-P&D, a card dropped onto an existing deck was transferred with the face-up/down state of the deck (all cards in a deck had the same face-

up/down state), while a card dropped on an empty workspace area was transferred with a face-up value, to facilitate “playing” the card. With BRIDGES, cards were always transferred from the tablet to the tabletop face-up since the most common player action after such transfer was to “play” a card by revealing its value to other players. An exception to this was if a player had left face-down cards on the BRIDGES after transferring them from the tabletop (we refer to this behaviour later as employing a “partial-transfer” strategy), they could then return it face-down to the tabletop.

*Summary of Main Study Findings.* Analysis of data from the three study conditions (BRIDGES, TA-P&D (EXPLICIT CONTROL) and TA-P&D (IMPLICIT CONTROL)) revealed that, in general, all conditions sufficiently supported card transfers, as evidenced by the, on average, 322 transfers that occurred per game across the study. The results also revealed a lack of clear preference for transfer method across players. Reported preferences differed drastically between groups, and even between players within groups. For example, one player commented that having the BRIDGES widgets “partly on both screens was beautiful and very helpful”, while another player reported that, in the BRIDGES method, having cards appear “in two places [on both the tablet and tabletop] was a little unwieldy”. Similarly conflicting comments were made about the TA-P&D method: One player reported that “Pick up is a much better mechanic [than BRIDGES]”, while another commented that “Picking up cards was NOT intuitive”.

Question	Pick-and-Drop Implicit Control	Pick-and-Drop Explicit Control	Bridges	RM- ANOVA
	Mean (SD)	Mean (SD)	Mean (SD)	
I had fun playing the game.	5.6 (1.4)	5.9 (1.0)	5.9 (0.9)	F(2,26)=1.00, p=.38
When the other player took action, I always understood their motivations for doing so.	5.4 (1.4)	5.4 (1.8)	5.4 (1.6)	F(2,26)=0.02, p=.98
When taking my turn, I was always aware of my play options.	6.1 (1.2)	6.3 (1.0)	6.3 (0.7)	F(2,26)=0.40, p=.68
I was always aware of the other player’s actions.	4.3 (1.6)	4.6 (1.9)	4.4 (1.8)	F(2,26)=0.41, p=.67

Table 3. Average participant ratings on enjoyment and awareness-related post-condition survey questions from Study 1 (1=strongly disagree, 7=strongly agree).

The RM-ANOVA analysis of the post-condition questionnaires similarly revealed no consistent player preference or perceived utility for any single transfer method. Participant ratings were generally positive on enjoyment and awareness-related measures (with mean ratings of 5.4 to 6.3 out of 7), with no significant differences across conditions (see Table 3).

The qualitative analysis shed light on the lack of clear preference between transfer techniques. It revealed that the effectiveness of a given transfer technique was player- and context-dependent. Preliminary analyses revealed that players in the two TA-P&D conditions rarely, if ever, used either the Explicit or Implicit Control methods for picking and dropping cards directly in the shared territory (i.e. most picks/drops were performed in the players’ personal territories). Thus, both TA-P&D conditions were aggregated into a

single TA-P&D condition for the in-depth qualitative analysis. This analysis revealed several key benefits and limitations of each method that impacted their use: the required cognitive and physical effort, and the ability of the method to maintain the privacy and secrecy of transferred data.

Some players found the TA-P&D transfer method more cognitively demanding than the BRIDGES method since the TA-P&D method required players to mentally keep track of which card(s) they had picked up, and were currently holding, during the transfer process. Once a player picked up a card it would disappear—“in the ether”, as reported by one participant—and was no longer visible on either the tabletop or tablet until the corresponding drop action occurred. While both the tablet and tabletop interfaces provided visual feedback in response to pick/drop actions, such as a short animation on the tabletop after a pick occurred, and the hand-of-cards being rearranged on the tablet after a pick/drop action, these interface changes appeared to be too subtle, or were sometimes occluded from the player’s view. In contrast, cards were always visible on the BRIDGES widgets during the transfer process, eliminating any mental burden from players regarding the state of the cards. Consequently, players reported that it was “easier to keep track of cards” with the BRIDGES method.

Despite its cognitive simplicity, BRIDGES required more physical effort than the point-to-point TA-P&D method. In BRIDGES, players had to drag cards across the tabletop to/from the TABLETOP BRIDGE and to drag cards on/off the TABLET BRIDGE during each transfer. Also, multi-card transfers required multiple drag actions to/from the respective BRIDGES. Thus, some players found transferring cards with BRIDGES to be quite tedious, as evidenced by the player comment, “The hand zone [BRIDGES] was super annoying... It just added more clicks to the game.” In contrast, TA-P&D allowed for multiple cards to be picked up at once and then transferred (and dropped) together.

BRIDGES was also found to be less privacy-preserving than TA-P&D. As mentioned in the Post-Transfer State section, all cards transferred from the tablet to the tabletop in BRIDGES arrived face-up on the TABLETOP BRIDGE to simplify post-transfer game actions, which commonly involved “playing” a card (i.e. revealing its value to opponents). However, at the end of each player turn, players discarded unplayed cards onto the player’s discard deck, typically located in their personal territory. In highly competitive games, revealing the value of discarded cards could reveal a player’s game strategy to observant opponents, potentially reducing a player’s competitive advantage. The “partial-transfer” strategy described early was adopted by some high-competitive players to help preserve card secrecy with the BRIDGES method, but this strategy had limitations that made it unusable for non-expert players (see Scott et al. (2014a) for details). In contrast, the TA-P&D method used the drop context to determine the face-up/down state of transferred cards. Thus, the secrecy of the card values dropped onto a face-down deck, such as the discard deck, would be preserved.

In Summary, while the BRIDGES method provided simple, straightforward usability that provided persistent feedback of the transferred cards, it also was less physically efficient and did not preserve the privacy of transferred objects as well as the TERRITORY-ADAPTED PICK-AND-DROP transfer method. As privacy is often an important goal of providing personal surfaces in a multi-surface environment, and efficiency of an interaction method is always an important usability goal for interaction techniques, we chose to investigate the TA-P&D method further in subsequent studies. More specifically, these follow-up studies focused on reducing the cognitive effort required to use this method for content transfer.

### **Improving P&D Transfer with SURFACE GHOSTs Visual Feedback**

A common HCI approach for helping people understand ongoing changes in a computer system is to provide persistent visual feedback related to changes in system state (Smallman and St. John, 2003; Scott et al., 2006; Chang et al., 2014). Study 1 revealed that the brief visual feedback provided after a card was picked up on either the tabletop or tablet was insufficient. During the actual transfer stage, no visual feedback was provided to indicate that cards were being “held” by the user. Thus, if someone became distracted after picking up a card—for instance, by an opponent’s game play actions or an ongoing conversation—they might forget they were holding a card and hence be surprised when the card appeared in the interface when subsequently touching the tabletop or their tablet.

Changing a virtual object’s visual appearance has been previously used to indicate changes in object state. For example, in Rekimoto’s (1997) original P&D implementation, when the digital pen hovered over the target display (within millimetres), the transferred object was displayed with a virtual shadow cast underneath it. This object-with-shadow representation would follow the hovering pen around in the interface until the object was dropped on the display, and then the shadow would disappear, leaving the active object. Similarly, “shadow” or “silhouette” object representations have been used to indicate objects being copied across adjacent tablet devices (Hinckley et al., 2004) and objects being held above the tabletop in a 3-dimensional tabletop workspace (Hilliges et al., 2009). Based on this prior work, we hypothesized that showing a similar visual representation of transferred cards in the interface during the transfer process may help reduce the cognitive effort associated with using our touch-based P&D transfer method. We also felt that providing feedback on who was transferring which cards would further reduce any user confusion in our multi-user setting. So, we designed the SURFACE GHOST object representation to provide visual feedback of cards being transferred with our touch-based P&D transfer method.

In Study 1, players tended to position their tablets directly along the tabletop edge. Thus, cross-device transfer interaction occurred largely over the tabletop surface. Therefore, we hypothesized that displaying visual feedback of transferred objects on the tabletop as the objects are

carried over the tabletop surface should provide (sufficiently) persistent visual feedback during transfer. Accordingly, the SURFACE GHOST visual feedback was designed to appear in the tabletop interface underneath the “owning” user’s hand as it traveled across the tabletop surface between the originating pick location and the target drop location. SURFACE GHOSTs were displayed as semi-transparent, greyscale versions of transferred objects. When multiple objects were being transferred at once, they were stacked together and a counter displayed the total number of transferred objects. Figure 5 illustrates the SURFACE GHOST visual designs for single-object (c) and multi-object (d) transfers in a digital card game.

To accommodate concurrent multi-user card transfers, the SURFACE GHOST design also conveyed ownership of the transferred object(s) through a number of static and dynamic design features. The basic SURFACE GHOST design provided several implicit indications of ownership: upon pick up the SURFACE GHOST object would “fly” (via a brief animation) toward its owner, the SURFACE GHOST object was oriented toward its owner, and it was displayed in real-time beneath the owner’s hand as their hand moved across the tabletop surface. As we were unsure how apparent such ownership information needed to be in DOMINION game setting, we developed two versions of the SURFACE GHOST design. The IMPLICIT OWNERSHIP version provided the above ownership information along with another, still subtle, indication of ownership; a large dark arrow attached to the bottom of the SURFACE GHOST object that “pointed” to the owning user (Figure 5, c and d). The EXPLICIT OWNERSHIP version replaced the black arrow with a more visually salient representation of the owner; a semi-transparent white silhouette of the owner’s arm displayed on the tabletop beneath the user’s physical arm. The SURFACE GHOST object was positioned at the arm silhouette’s hand (Figure 6), indicating that that user was “holding” the card.

To implement either of these SURFACE GHOST designs, it was necessary to move beyond a “current” T-MSE set-up to a “future” T-MSE set-up that provided multi-user identification and above-the-surface tracking. In this enhanced environment, the system was able to keep track of who was transferring which cards. Thus, it was no longer necessary to divide the tabletop into personal and shared territories to facilitate simultaneous multi-user P&D transfers. Hence, in Studies 2 and 3, users could perform pick or drop actions at any location on the tabletop interface. So, we dropped the “Territory-Adapted” aspect of our P&D implementation, and refer to the technique as simply P&D transfer when discussing the method used in Studies 2 and 3 rather than TA-P&D. As user-tracking was limited to the area above the tabletop surface, any pick or drop actions on a tablet were still assumed to belong to the “owning” user.

## **Study 2**

The goal of Study 2 was to determine whether the SURFACE GHOSTs visual feedback reduced the confusion people experienced during the P&D

transfer process, and improved their awareness of cards being transferred. We were also interested in learning whether the SURFACE GHOSTs feedback improved people’s awareness of when other players were transferring cards during the game, thereby improving their collaborative awareness. Given the multi-user nature of our task environment, another goal was to determine how the two different ownership designs (IMPLICIT vs. EXPLICIT) might impact people’s awareness of transferred objects, and overall transfer performance.

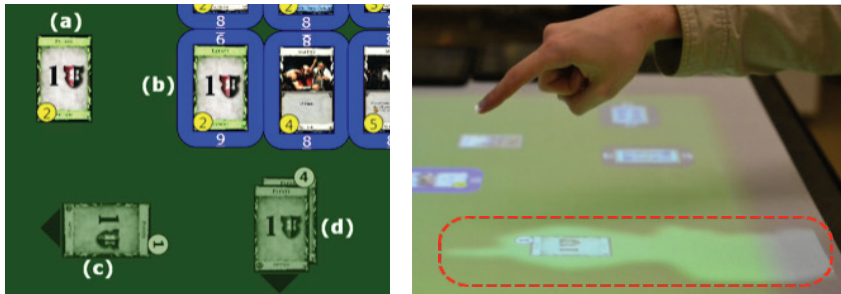


Figure 5 (left). SURFACE GHOSTs in a card game context: (a) a normal card, (b) a deck of cards, (c) a SURFACE GHOST (with IMPLICIT OWNERSHIP feedback) of one card being transferred by the Left Player, and (d) a SURFACE GHOST of multiple cards being transferred by the Bottom Player (from (Scott et al 2014b)).

Figure 6 (right). SURFACE GHOSTs with EXPLICIT OWNERSHIP in a tabletop card game context (from (Scott et al 2014b)).

Following the methodology described above, groups of three participants completed three DOMINION game play sessions using the P&D transfer method with the three different visual feedback conditions: SURFACE GHOSTs with IMPLICIT OWNERSHIP (IMPLICITSG), SURFACE GHOSTs with EXPLICIT OWNERSHIP (EXPLICITSG), and a control condition no feedback (NF).

*Summary of Main Study Findings.* Similar to Study 1, players performed a significant amount of P&D transfers during the study. A total of 4455 P&D transfers occurred across all game sessions. Similar to Study 1, participants’ preferences were evenly split across the three conditions (6 preferred IMPLICITSG, 6 preferred EXPLICITSG, 6 preferred the control (NF)). Despite the fact that a third of the participants preferred the control (NF) condition, the RM-ANOVA analysis of the post-condition questionnaires revealed that both SURFACE GHOST conditions significantly increased reported awareness of transferred cards compared to the control (NF) condition for tabletop-to-tablet transfers. Yet, the analysis revealed that the SURFACE GHOST feedback did not provide the same awareness benefits for card transfers in the opposite direction (tablet-to-tabletop transfers). No differences were found in reported awareness levels between the two SURFACE GHOST conditions in either transfer direction. Similarly, no differences were found in reported awareness levels of card transfers performed by others at the table across all conditions. Table 4 summarizes the reported awareness levels across conditions and the RM-ANOVA results.

Question	No Feedback Mean (SD)	Surface Ghost w/ Implicit Ownership Mean (SD)	Surface Ghost w/ Explicit Ownership Mean (SD)	RM- ANOVA
<i>I was always aware...</i>				
<i>...when I had a card in my hand when moving from the tabletop to my tablet</i>	4.4 (1.9)	5.7 (0.8)	5.9 (1.2)	F(2,10)=8.16, p=.008* *Sig. Contrast: NF vs. SGwImp: p=.028 *Sig. Contrast: NF vs. SGwExp: p=.024
<i>...of how many cards I had in my hand when moving from the tabletop to my tablet</i>	4.6 (1.9)	5.8 (1.2)	5.9 (1.1)	F(2,10)=8.71, p=.006* *Sig. Contrast: NF vs. SGwImp: p=.026 *Sig. Contrast: NF vs. SGwExp: p=.027
<i>...when I had a card in my hand when moving from my tablet to the tabletop</i>	5.5 (1.5)	5.7 (1.0)	5.8 (1.0)	F(2,10)=0.56, p=.59
<i>...of how many cards I had in my hand when moving from my tablet to the tabletop</i>	5.4 (1.2)	6.1 (0.8)	6.0 (1.2)	F(2,10)=8.38, p=.007* *Sig. Contrast: NF vs. SGwImp: p=.007
<i>...of when my partner had cards in transit</i>	3.8 (1.9)	4.4 (1.9)	4.3 (1.8)	F(2,10)=1.56, p=.26

\*significant at  $\alpha=.05$ .

Table 4. Average participant ratings on awareness-related post-condition survey questions Study 2 (1=strongly disagree, 7=strongly agree).

Our results show that both SURFACE GHOST designs were more effective at promoting awareness of transferred objects during transfers originating on the tabletop than transfers originating on the tablet. The qualitative data analysis provided insights on this asymmetric awareness benefit of SURFACE GHOSTs by revealing how participants used this visual feedback. SURFACE GHOSTs were found to support three main aspects of P&D transfer: confirming that a pick or drop worked, keeping track of how many cards were picked up, and confirming that picked up cards went to the right player.

Confirming that an intended pick or drop action succeeded was the most prevalent use of the SURFACE GHOST feedback. Players frequently used the local animation of the SURFACE GHOST object “flying” from the card’s original location toward the owning user to confirm picks. Also, players commonly shifted their hand and wrist positions during pick actions to facilitate viewing the SURFACE GHOST object located under their palm (which was more robustly tracked than their fingertip) or arm silhouette during this pick confirmation process. Similarly, players often double-checked that the SURFACE GHOST feedback disappeared after a drop operation. In the control (NF) condition, the lack of feedback often resulted in participants redoing a whole sequence of actions.

Players also made extensive use of the counter provided in the multi-object SURFACE GHOST design to track how many cards they had picked up during multi-card. In the control (NF) condition, players relied on counters attached to each deck to determine how many cards they had picked up, by tracking how much the number decremented after each pick up. This method was more cognitively demanding, as revealed by Study 1. In contrast, the SURFACE GHOST counter provided the information directly, without mental calculation, and was available if players missed the original pickup actions.

The third main use of SURFACE GHOSTs was to confirm that cards on the tabletop were picked up by the right person. Due to technical limitations of above-the-table tracking, the system’s user identification was occasionally

incorrect when players were interacting in close proximity. When this occurred during pickup, the card(s) would be associated with the wrong user. As part of the pick confirmation behaviour described above, players commonly relied on the local animation of the SURFACE GHOST object to confirm the correct user association. Figure 7 illustrates an example where the SURFACE GHOST object animation helped participants to detect an incorrect association, during simultaneous proximal interactions. If this animation was missed, the various forms of persistent and dynamic feedback provided by SURFACE GHOSTS was also useful: the dynamic movement of the SURFACE GHOST object following a user's hand, and in particular, in the case of EXPLICITSG, the arm silhouette, was reported to be particularly useful in diagnosing inaccurate user identification.

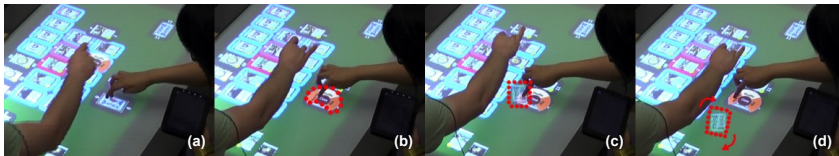


Figure 7. Players using SURFACE GHOST animation to recognize that a picked card went to the wrong player: a) Left Player waits to interact near Right Player's hand b) Right Player's menu (highlight) is oriented toward Left Player due to inaccurate user identification, yet Right Player does not appear to notice, c) Right Player picks up card, and d) the Surface Ghost (highlight) flies toward Left Player's hand. Right Player says, "I am under the impression that you might have my cards". From (Scott et al 2014b)).

The qualitative analysis also revealed that participants unexpectedly appropriated the P&D technique for transferring cards between different tabletop locations, rather than using drag-and-drop transfer. All participants in all conditions exhibited this behaviour, even though they were only shown how to use P&D for cross-device transfers. They spontaneously, often accidentally, discovered this possibility during game play. Distance did not seem to be a main factor for triggering within-tabletop transfers: the same players were observed using drag-and-drop to transfer cards over long distances, and using P&D to transfer cards over very short distances.

In summary, we found that SURFACE GHOSTS feedback successfully promoted transfer awareness during tabletop to the tablet transfers, but was less effective during tablet-to-tabletop transfers. The lack of improved awareness during transfers originating on the tablet was likely caused by the lack of SURFACE GHOSTS feedback during tablet pick and drop actions, due to the positioning of the tablets outside the active tabletop area. The study also revealed that both IMPLICIT and EXPLICIT OWNERSHIP design variations provided sufficient ownership information in most transfer situations, yet the arm silhouettes provided by the EXPLICIT OWNERSHIP design provided better support for coping with common technical issues encountered on multi-touch surfaces, minimizing frustration and improving the overall user experience. The final study focused on increasing transfer awareness during tablet-to-tabletop transfers, thereby improving the overall



cross-device transfer experience.

### **Improving Awareness during Tablet-to-Tabletop P&D Transfers**

The fact that the SURFACE GHOSTS feedback was unavailable during pick operations on the tablet was only a minor issue when transferring a single card: the SURFACE GHOSTS feedback would appear as soon as the user's hand was over the tabletop, and so, visual feedback was available almost immediately after the pick operation. However, during a multi-card pick up sequence that required the user to make repeated pick operations (recall, a tablet pick operation involved dragging a card upwards across the top edge of the tablet using a swipe-up gesture); each successive pick operation would bring the user's hand repeatedly back over the tablet surface (and away from the tabletop surface). Thus, this interaction sequence delayed the appearance of the SURFACE GHOST feedback until the final card had been picked up. Consequently, the user had to rely on (sometimes subtle) changes in the arrangement of cards in the tablet interface to confirm the success of the pick operation, which was easy to miss if the tablet contained a number of visually similar cards.

To address the ineffective feedback on the tablet, we considered various design solutions. We first considered a variant of SURFACE GHOSTS on the tablet, but found it had several drawbacks. The first issue was technical: tracking a user's hand above a tablet—especially when players moved their tablet—was highly challenging and not feasible in our tracking environment. Second, there was limited screen real-estate to display a useful SURFACE GHOST object or arm silhouette. Also, it would likely be obscured from the user's view by their physical hand, or positioned off the display. Thus, we wanted to provide a device-appropriate feedback mechanism that would serve the same purpose as SURFACE GHOSTS on the tabletop: convey which cards, and how many cards were currently being held by the user. A consistent feedback from Study 1 was that the visual feedback provided by the BRIDGES mechanism provided high levels of transfer awareness. Also, the location of the TABLET BRIDGE coincided with the swipe-up and swipe-down gestures for tablet pick and drop actions. Thus for Study 3, we included a modified version of the TABLET BRIDGE visualization (without the BRIDGE transfer functionality). Unlike the split card visualization in Study 1, in Study 3 we displayed miniature versions of entire cards along the top edge of the tablet during transfer (see Figure 8).

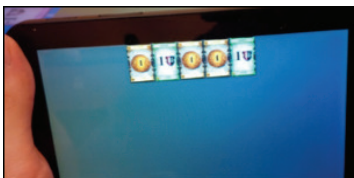


Figure 8. The modified TABLET BRIDGE visualization. When cards are dropped on tablet, miniature cards disappear from TABLET BRIDGE and appear full size in main tablet interface below.

We also made improvements to the SURFACE GHOSTS tabletop feedback and to the overall P&D interaction process to better support the DOMINION task environment. First, we fixed an interaction bug revealed during Study

2 (detailed in Scott et al. (2014b)) that interfered with touch actions over the arm silhouette in the EXPLICITSG condition. We also displayed a second counter on the lower left corner of the SURFACE GHOST multi-card visualization to improve visibility of the counter. Finally, we added an option to pick up 5 cards at once to the context menu to facilitate this frequent Dominion game action. Figure 9 shows the updated SURFACE GHOST designs on the tabletop used in Study 3.



Figure 9. The updated SURFACE GHOST visual feedback and tabletop environment: EXPLICITSG (left), and IMPLICITSG (right).

### Study 3

The primary goal of Study 3 was to determine whether the combination of SURFACE GHOSTS feedback on the tabletop and TABLET BRIDGES feedback on the tablets improved player's overall awareness during P&D transfer, in both transfer directions. A secondary goal of Study 3 was to determine whether our software improvements resolved transfer performance issues observed in the EXPLICITSG condition in Study 2.

To reflect these primary and secondary goals, we modified the study method used in Studies 1 and 3. To address our primary goal of comparing the effectiveness of adding the TABLET BRIDGE feedback, we included a tablet feedback factor with two levels: BRIDGE and NO BRIDGE conditions. To address the secondary goal of assessing the timing performance of the modified EXPLICITSG design, we included a tabletop feedback factor: EXPLICITSG and IMPLICITSG conditions. Due to practical concerns involved with playing full-length DOMINION games in each study condition, we chose to use a mixed within-subjects (tablet feedback) and between-subject (tabletop feedback) experimental design, rather than a fully crossed, within-subjects design, to minimize participant fatigue. Also, as Study 3's main measures related to the tablet feedback factor focused on player's perceived awareness of their own transferred cards, for practical issues, we utilized a participant group of size two (similar to Study 1).

Each group completed three DOMINION game play sessions using the P&D transfer method under three different visual feedback conditions. All groups experienced the EXPLICITSG tabletop feedback both with BRIDGE tablet feedback (EXPLICITSG+B) and with NO BRIDGE tablet feedback (EXPLICITSG+NB), and experienced either the IMPLICITSG tabletop feedback with BRIDGE tablet feedback (IMPLICITSG+B) or with NO BRIDGE tablet feedback (implicitSG+NB). Thus, each group only played one condition with the IMPLICITSG feedback on the tabletop (with or without

the TABLET BRIDGE visualization). Our data analysis of the awareness metrics only included data from the EXPLICITSG conditions to enable more statistically robust repeated-measures analysis of questionnaire responses, while our data analysis of the transfer timing metrics utilized both within- and between-subjects analyses across conditions, as described below, due to the more numerous occurrences of card transfers available from the interaction logs.

*Study Findings.* The data analysis revealed that participants had a strong positive reaction to the addition of the TABLET BRIDGE visualization. Twenty-two out of 24 participants preferred having the BRIDGE feedback on the tablet (18 preferred EXPLICITSG+B; 4 preferred IMPLICITSG+B), while the remaining two preferred the NO BRIDGE conditions (1 preferred IMPLICITSG+NB, 1 preferred EXPLICITSG+NB). According to participant's post-experiment interview comments, the two preferences for the NO BRIDGE condition was influenced by a minor interaction difference between the BRIDGE and NO BRIDGE conditions: the "tap anywhere to drop" convenience feature when the tablet was empty was missing in the BRIDGE conditions due to inherited functionality from the Study 1 BRIDGES transfer method (unfortunately not identified during pilot testing). However, the lack of this feature was not mentioned by most participants, who appeared to prefer using the swipe-down drop gesture. For the remaining few who also commented on this missing feature, their overall preference for the BRIDGE condition appeared to be strongly influenced by the high level of transfer awareness it provided.

The data analysis also revealed that providing the TABLET BRIDGE feedback significantly improved participants reported awareness of transferred cards, in both transfer directions. Also, analysis of the transfer timing data found no differences between EXPLICITSG and IMPLICITSG conditions, suggesting that our software modifications addressed the transfer time performance issues related to the EXPLICITSG design uncovered in Study 2. As the timing investigation was included to validate our software implementation improvements rather than our transfer method interaction concept, timing results are not included here, but are detailed in an online technical report (Scott et al., 2015). We expand on the transfer awareness results below.

*Perceived Awareness of Transferred Cards.* The RM-ANOVA analysis of the post-condition questionnaire responses from the two EXPLICITSG conditions revealed the BRIDGE (EXPLICITSG+B) condition significantly increased reported transfer awareness compared to the NO BRIDGE (EXPLICITSG+NB) condition for both tabletop-to-tablet and tablet-to-tabletop transfers. Table 5 summarizes the reported transfer awareness data and RM-ANOVA results. (Comparing two conditions would normally call for a t-test statistic, but recall from the Methodology section that tabletop position was also included as a main between-subjects factor in all RM-ANOVA analyses across all studies to account for the effect of group. No effect of tabletop position or interaction across main factors was found.)

Question	SURFACE GHOST (Exp) w/o TABLE BRIDGE	SURFACE GHOST (Exp) w/ TABLE BRIDGE	RM- ANOVA
	Mean (SD)	Mean (SD)	
<i>I was always aware...</i>			
<i>...when I had a card in my hand when moving from the tabletop to my tablet</i>	4.7 (1.7)	5.8 (1.0)	$F(1,11)=9.44, p=.011^*$
<i>...of how many cards I had in my hand when moving from the tabletop to my tablet</i>	4.8 (1.6)	5.7 (1.3)	$F(1,11)=6.29, p=.029^*$
<i>...when I had a card in my hand when moving from my tablet to the tabletop</i>	5.1 (1.5)	6.0 (1.0)	$F(1,11)=10.65, p=.008^*$
<i>...of how many cards I had in my hand when moving from my tablet to the tabletop</i>	4.8 (1.9)	5.7 (1.3)	$F(1,11)=8.93, p=.012^*$

\*significant at  $\alpha=.05$ .

Table 5. Average ratings on awareness-related post-condition survey questions (1=strongly disagree, 7=strongly agree).

These results supported our expectation that the BRIDGE condition would better promote transfer awareness than the NO BRIDGE condition for tablet-to-tabletop transfers. Yet, they contracted our expectation that the BRIDGE and NO BRIDGE conditions would provide similar support for transfer awareness for tabletop-to-tablet transfers, given the effectiveness of the SURFACE GHOSTS feedback alone to support transfers in this direction in Study 2. Thus, the BRIDGE condition appeared to effectively promote transfer awareness in both transfer directions. This result was confirmed by the many positive comments participants made regarding the utility of the TABLET BRIDGE in response to the open-ended survey question, "What feature of the tabletop/tablet assisted the game play?", including: "The visualization of cards at the top of the tablet greatly improved my awareness of when I had cards in transit." (P15 EXPLICITSG+B); "You could see the cards on the tablet that were in transit." (P5 EXPLICITSG+B); "The cards appearing on the tablet when in transit was helpful" (P24 EXPLICITSG+B); and, "Not seeing the cards in transit on the tablet was a hindrance." (P11 IMPLICITSG+NB).

Review of the interview, open-ended questionnaire responses, and video data also provided insights on the unexpected positive influence of the TABLET BRIDGE feedback on transfers originating from the tabletop. Participants reported extensive use of the TABLET BRIDGE feedback, when available, during tabletop-to-tablet transfers, as illustrated by the following comments: "The little bar on the tablet at the top to show what cards you took to the tablet [assisted the game play]." (P22 EXPLICITSG+B questionnaire); "Sometimes you thought you picked up 5 cards when really you hadn't, and hav[ing] that additional feedback on the tablet was nice." (G7 interview); and, "In the second game they [cards on top of the tablet screen] disappeared...It was much more clear what you were transferring from the table to your tablet when you had them up at the top." (G1 interview).

The video data revealed several specific benefits of the TABLET BRIDGE feedback during tabletop-to-tablet transfers. During DOMINION game play, players make extensive use of the "personal territory" near them in the tabletop interface. Unlike Study 1 where personal territories were explicitly delimited in the interface, in Studies 2 and 3, players implicitly established these territories, similar to common tabletop usage in other contexts (Scott

and Carpendale, 2010). The consequence of this territorial behaviour is that pick and drop actions often occur near the tabletop edge, commonly causing the SURFACE GHOST visual feedback to be displayed partially outside the interface. Due to poor touch detection near the tabletop edge on the tabletop system used in Studies 1 and 2, the active game play area in the Dominion tabletop application stopped a few centimeters from the edge. However, since the projected area covered the whole surface, the SURFACE GHOSTS object and arm silhouette visual feedback continued to be displayed in the edge area. The upgraded tabletop used in Study 3 provided improved touch detection across the whole surface. So, the active play area was extended directly to the tabletop edge to facilitate easier player's access to game content. An unintended consequence of this change was that the SURFACE GHOSTS feedback was sometimes unavailable during pick/drop actions near the table edge. Participants used the TABLET BRIDGE feedback, when available, to overcome this issue.

The TABLET BRIDGE feedback also helped compensate for the positioning lag of the SURFACE GHOST and arm silhouette caused by necessary image smoothing performed on the imperfect Kinect tracking data. Once participants became familiar with the P&D transfer mechanism, they could perform card transfers very quickly. Thus, sometimes a transfer was almost (or completely) finished before the SURFACE GHOST feedback would appear. In contrast, the TABLET BRIDGE was immediately, and persistently, available throughout the transfer process. Additionally, the new option to pick up 5 cards at once from a tabletop deck was used extensively. This substantially reduced the need for one-by-one multi-card pick-ups, which, in turn, reduced participants' use of the pick-up counter on the SURFACE GHOST multi-card visualization.

Finally, the TABLET BRIDGE feedback also helped participants cope with hardware input errors, such as errors in touch or gesture detection on the tabletop and tablet devices or errors in user tracking on the tabletop. Participants found the additional visual feedback on the tablet helpful for detecting and managing these issues, as illustrated by the comments, "The slight finicky-ness [of the tabletop touch detection] was still a problem, but was helped by the display of cards being transferred at the top of the tablet ." (P23 EXPLICITSG+B questionnaire) and "[I] Felt the sensor wasn't working as well as the first game [a BRIDGE condition]. This could have been due to having less feedback when I picked up a card. Would have been nice to know how many cards were in transition." (P13 EXPLICITSG+NB questionnaire).

*Summary.* The study found that providing both TABLET BRIDGE feedback on the tablet and SURFACE GHOST feedback on the tabletop improved transfer awareness for both tablet-to-tabletop and tabletop-to-tablet transfers, thereby improving the overall utility of our T-MSE P&D transfer technique. The immediate and persistent feedback provided by the TABLET BRIDGE feedback helped compensate for several technical and usability

issues of the SURFACE GHOST mechanism.

## **Discussion**

Our three studies provided significant insights on supporting cross-device transfer in T-MSE settings. The studies also highlighted how point-to-point cross-device transfer techniques like P&D can be appropriated for within-surface transfers to help ameliorate usability issues related to dragging objects, especially across long-distances, on devices with imperfect touch input technologies (e.g. dropped objects due to lost or jittery input). We discuss these lessons learned below.

*Make Object State Apparent through Entire Transfer Process.* The results of Study 1 uncovered the need for visual feedback during P&D transfer. However, Studies 2 and 3 highlighted the specific need for feedback during the pick and drop actions of the three-phase P&D process (pick, transfer, drop). The limited visual feedback available on the tablet during pick operations in Study 2 hindered participants' perceived awareness for transfers originating on the tablet. Introducing the TABLET BRIDGE visualization (without associated BRIDGES portal functionality) in Study 3 provided persistent feedback during the entire P&D process: users could immediately see each picked card added to the row of miniature cards displayed on the TABLET BRIDGE, and see them disappear when cards were dropped on the target device. For tabletop-to-tablet transfers in Study 3, players could utilize either the SURFACE GHOST feedback on the tabletop or the TABLET BRIDGE feedback on the tablet to learn the state of cards involved in the transfer process, providing redundant feedback (when the SURFACE GHOST feedback was available on the tabletop). The BRIDGES transfer method from Study 1 provided similarly redundant feedback throughout the entire transfer process. Both the TABLE and TABLET BRIDGES displayed all cards being transferred (across a pair of devices), and at no time did cards disappear from view—they were either on the tabletop/tablet as full-size active cards, or they were visible on the TABLETOP/TABLET BRIDGES transfer portals. Not surprisingly then, Study 1 participants consistently reported high levels of transfer awareness in the BRIDGES condition.

*Consider Efficiency at All Stages of Transfer: Beginning, Middle, and End.* While the BRIDGES transfer method provided excellent awareness of transferred objects, it was also found to be extremely tedious to use in the DOMINION task context, which required frequent object transfers. The fact that each transfer operation required interaction to/from the intermediary BRIDGES containers added additional interaction steps to the overall transfer process. Participants found this to be especially effortful when performing multi-card transfers, of which there were many during the DOMINION games.

The point-to-point nature of P&D transfer allowed for more efficient transfer, especially as our implementation allowed for multiple cards to be

picked-up at the originating location and transferred at once. However, the frequent need in DOMINION to pick-up multiple (most often 5) cards each turn, introduced room for improved efficiency at the beginning of a multi-card transfer process. Indeed, the “pick up 5 cards” option added to the tabletop menu in Study 3 was highly appreciated, and utilized, by players. Allowing aggregated card transfer in the BRIDGES transfer method may be similarly useful for improving its efficiency, for instance, by allowing a deck of cards to be placed on the BRIDGES. This approach raises the design issue of whether the aggregated content (e.g. 5 cards) should be shown separately or in aggregated form on the BRIDGES containers. In Study 3, the TABLET BRIDGES visualization used the former approach: all transferred cards were displayed separately. Using this “show all” approach, users could then remove individual items “from the BRIDGE” on the target device, or could be given a mechanism (a gesture or button) to allow items to be removed together. Displaying an aggregated view would only allow for an all-at-once end-of-transfer action, and may also reduce some of the positive awareness benefits of the BRIDGES method.

P&D transfer outperformed BRIDGES for end-of-transfer efficiencies as multiple cards being transferred at once would all drop at the target location. The “tap to drop” convenience feature on the tablet (available when the tablet was empty) also improved the drop efficiency of P&D transfer over the “swipe-down to drop” interaction, as it was more forgiving due to the bigger interaction target of the whole tablet screen (vs. the top edge for the swipe-down action) and to the more robust touch detection in the central area of the tablets used in the studies. As mentioned above, end-of-transfer interaction, especially on the tablet, could be improved by providing a mechanism to allow all transferred items to be moved off the BRIDGE at once. This should be done in a task- and device-relevant way, for instance, in the DOMINION game, the TABLET BRIDGE could be augmented with a button located to one side that, when pressed, incorporated all content on the BRIDGE into the hand of cards on the tablet. This would be fairly simple, as there was only one possible destination for cards fully-transferred to the tablet. In contrast, automatically offloading the TABLETOP BRIDGE would be more complex on the tabletop, as the intended destination may be less clear. Here, a specific drag gesture (e.g. a 2-finger drag) that allows players to manually move the entire contents of the BRIDGE to the intended location may be more appropriate.

In Study 3, it was anecdotally observed that some participants misinterpreted the TABLET BRIDGE visualization to mean that cards picked up on the tabletop were automatically transferred to the tablet. This misperception was actually a commonly suggested improvement across the three studies, and one we have received from others during public demonstrations of our system. This approach would resolve many efficiency issues discussed above. However, the approach assumes that players always intend to move cards to their tablet. Yet, our studies revealed frequent use of tabletop-to-tabletop transfers, thereby introducing complexities for inferring when

cards should be transferred to one's tablet rather than be moved elsewhere on the tabletop. Nonetheless, the approach warrants further investigation as it has the potential to greatly improve the efficiency of tabletop-to-tablet transfers.

*Consider Post-Transfer State, Utilize Context if Available.* Another limitation of the BRIDGES method is its inability to infer the target location, and hence intended purpose, during tablet-to-tabletop transfers. Consequently, the same post-transfer state was applied to each transferred card: Cards were always transferred face-up onto the TABLETOP BRIDGE to facilitate the common "reveal a card" action. However, this design decision was not universally appreciated. The inability to control the post-transfer card state with BRIDGES prompted highly competitive players to adopt a "partial transfer" strategy, in which they left drawn cards sitting on the BRIDGES. This allowed them to keep cards face-down on the tabletop at the cost of not being able to fully view, or manipulate, cards on the tablet. These players strongly preferred the context-dependent manner of determining the post-transfer state used by the TA-P&D (and P&D) transfer method: Cards transferred to the tabletop took the face-up/down state of any deck/card they were dropped onto, or were placed face-up if dropped onto an empty area. This design decision was driven by the application task (i.e. the DOMINION game) and an early analysis of common game actions (and associated player intentions).

In the DOMINION game, the possible states of transferred objects were relatively limited: cards and card decks were the only application objects, card size and orientation were fixed on both the tabletop and tablet, and cards were either face-up or face-down. (In Study 1, orientation of cards (and decks) on the tabletop was automatically determined by whether they were located in a personal territory or the shared territory. In Studies 2 and 3, cards (and decks) were automatically (orthogonally) oriented toward the table side of the "owning" user after P&D transfers or drag actions.) However, in other task contexts, the possible object states that should be considered after transfer will vary, and may include, for instance, the scale (size) and orientation of content objects, or whether they are separate or aggregated, and for multi-dimensional objects, what side (or sides) is displayed. The size disparity between a large surface and smaller personal surface may play a factor. For instance, if a document that is currently being viewed on a smartphone display is transferred to a shared tabletop, it may be useful to display a larger portion of the document on the larger tabletop display than was visible on the smaller smartphone. Ultimately, if post-transfer state is determined automatically by the system, it should select a task- and device-appropriate state that best facilitates people's intended task activities. The selected state should optimize the overall efficiency of the transfer process by minimizing any necessary interactions to achieve a desirable post-transfer object state. Any contextual information available about the intended target location, transfer direction, task phase, etc. may be helpful in inferring a reasonable post-transfer state.



*Consider Within-Surface Transfer on Large Surfaces.* The studies revealed the common use of P&D transfer to move cards from one tabletop location to another. Almost all participants performed such tabletop-to-tabletop transfers. Analysis of the interaction logs for Study 2 showed no consistent pattern of participants' use of P&D transfer compared to drag actions related to the move distance: P&D transfers appeared to be as equally likely to use for short-distance tabletop moves as for long-distance moves. The video data revealed several possible motivations for choosing P&D over drag to move a card/deck on the tabletop. First, participants often appeared reluctant to drag cards/decks directly over other cards/decks, possibly due to uncertainty over the consequence of such actions (i.e. the deck/card may be disturbed). Thus, they sometimes dragged cards in a wide path around other tabletop content, or simply used P&D transfer to go above the tabletop content. Second, the imperfect touch detection on the tabletop sometimes caused the touch input to fail and cards to drop onto other content. One such instance in Study 3 prompted the user comment, "the deck just swallowed my cards". This type of input errors, unfortunately all too common in existing large-surface hardware, creates significant frustration for users. Long-distance drags are particularly vulnerable to lost-touch situations. The fact that the P&D transfer method required minimal touch interaction on the tabletop provided a reasonable coping strategy for moving content, especially across a long distance, giving the tabletop's imperfect touch detection.

## **Conclusions**

Our studies investigating cross-device transfer demonstrated how the existing cross-device transfer methods virtual portals (BRIDGES) and physical proxy (PICK-AND-DROP) can be applied to both "current" and "future" table-centric multi-surface environments. The studies revealed both methods, with our adaptations optimized for touch-based devices, effectively supported the significant amount of transfer required by the experimental task (the DOMINION card game). They also revealed several key interaction design requirements for cross-device transfer, including the need for persistent feedback throughout the entire transfer process, the need for efficient multi-object transfer, the need to preserve privacy and content secrecy throughout the transfer process when desired, and the need to consider post-transfer object state.

While our studies revealed many useful insights, further study is warranted in a number of directions. First, occasionally in our studies, players wished to transfer cards directly from one tablet to another when "giving a card" to another player. Moreover, one can imagine other task contexts, particularly during more cooperative group activities, where people might want to exchange task content directly from one tablet to another. Future design extensions should consider this functionality. Similarly, other design extensions might include the ability to "share" tabletop content on someone else's tablet to allow more cooperative transfer patterns between available surface devices.



## High-Performance Interfaces for Touch Surfaces

*Carl Gutwin, Andy Cockburn, Sylvain Malacria, Scott Olson, and Ben Lafreniere*

(Portions of this chapter first appeared in the following publications: Carl Gutwin, Andy Cockburn, Joey Scarr, Sylvain Malacria, and Scott C. Olson. 2014. Faster command selection on tablets with FastTap. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 2617-2626. DOI=<http://dx.doi.org/10.1145/2556288.2557136>. Carl Gutwin, Andy Cockburn, and Benjamin Lafreniere. 2015. Testing the rehearsal hypothesis with two FastTap interfaces. In Proceedings of the 41st Graphics Interface Conference (GI '15). Canadian Information Processing Society, Toronto, Ont., Canada, 223-231.)

### Introduction

Desktop interfaces often offer multiple ways to select the same command, and the different mechanisms can have very different performance characteristics. For example, selecting commands from menu hierarchies is slow when compared to keyboard shortcuts and toolbars, which provide access to commands with fewer actions. Having these types of shortcuts in an interface can substantially increase a user's efficiency over time, by allowing the user to learn quicker methods of invoking the commands they use most often.

On portable touch-based surfaces, however, interface shortcuts are seldom available. The lack of a physical keyboard means that there are no keyboard shortcuts for quick selection, and limitations on screen real estate leave little or no room for always-visible components such as toolbars. Touch-based command interfaces, therefore, often take the form of tedious menu hierarchies, with no way of making a transition to an expert method of interaction. This greatly decreases the utility of touch devices for productivity tasks.

One widely-studied interface that supports expertise on touchscreens is the marking menu, a type of radial menu that allows visual inspection of menu items for novices, and rapid gestural interaction for expert users (Kurtenbach and Buxton, 1991). However, the contact-move-lift actions for marking menu gestures may take longer to perform than a simple tap action. With multi-touch capabilities widely available in modern tablet computers, there

are opportunities for interfaces that support rapid command execution for experts, as well as smooth transitions from novice to expert use (e.g., Bailly et al., 2010; Roy et al., 2013).

In this paper, we present FastTap, a new rapid-access interaction technique that allows fast command selection on multi-touch devices for both novice and expert users. As shown in Figure 1, FastTap uses the entire screen to present a spatially-stable grid of commands – based on the recent CommandMap interface (Scarr et al., 2012). The command overlay is hidden by default, and is shown when the user holds their thumb on the grid activation button. The interface can then be inspected, and commands can be selected with a finger; when the user lifts their thumb, the grid disappears.

Novices use the interface by showing the grid and visually searching for the commands they need. As users become familiar with commands, however, they remember item locations in the grid, leading to expert behavior – experts can select a command with a single ‘chorded’ tap using the thumb and forefinger, without waiting for the grid to appear. Similar to marking menus, this design follows Kurtenbach’s principle that ‘guidance should be a physical rehearsal of the way an expert would issue a command’ (Kurtenbach, 1993) – in other words, since the novice and expert interaction methods require similar motor actions, users can develop spatial and muscle memory of the action required for each command through natural use.

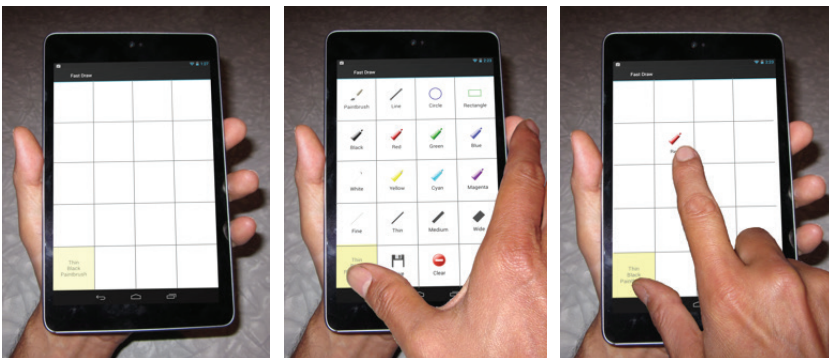


Figure 1. FastTap interface. Left: default state of the interface (gridlines enhanced). Center: FastTap grid overlay after touching the activation button. Right: FastTap selection by chording with the thumb and forefinger, without waiting for the overlay.

To assess the performance of FastTap, we carried out a controlled study comparing selection time and errors with FastTap and marking menus. Our study showed that selection time was significantly and substantially faster with FastTap (mean 1.6 seconds per selection) than with marking menus (mean 2.4 seconds). In addition, FastTap was significantly faster at all levels of expertise with the interface, and provided an additional speed benefit when multiple commands were carried out in sequence. We found no differences in terms of errors, effort, or preference.

Our work makes three main contributions. First, we introduce a new interaction technique for tablets that was significantly faster than marking menus in an initial study. Second, we further demonstrate the power of spatial memory as an organizing principle for visual interfaces, and demonstrate how spatial memory can be exploited together with multi-touch input to produce rapid command selection interfaces. Third, we provide empirical results about user performance with FastTap and marking menus.

### **Background: Supporting Transitions to Expertise**

Understanding skill acquisition has long been a basic objective in psychology, and in HCI, numerous techniques have been proposed to help users achieve higher performance. These techniques fall into four main groups: intra-modal improvement, which aims to boost performance within the current interaction mechanism; inter-modal improvement, which involves improvements through switching to a faster style of inter-action; vocabulary extension, which tries to increase users' knowledge of the commands that are available within an application; and task mapping, which involves improving the user's task comprehension or solution strategy (see survey by Cockburn et al., 2014).

Many different types of techniques have been suggested in these areas, including different training methods, shortcuts for experts, memory-based retrieval interfaces, adaptive interfaces, and task-based customization. Memory-based expert techniques – including keyboard hotkeys, gestural interfaces, command languages, and spatial-memory-based interfaces – have been shown to be particularly fast for experienced users. These techniques are rapid because they involve fewer and faster operations when a user is experienced – rather than navigating or searching for a command like novices do, the expert user can just remember and execute the command. Importantly, these techniques normally involve inter-modal changes (users must switch from one interaction method to another to increase performance) – for example, switching from mouse-and-menu operation to hotkeys. Rehearsal-based techniques, reviewed below, seek to minimize this inter-modal transition (Kurtenbach, 1993).

Learning and skill development are critical foundations to the idea of expert interfaces for surfaces, and several concepts are important. First, interface learning can be organized into three stages as proposed by Fitts and Posner (1967) – cognitive, associative, and autonomous. During the cognitive phase, users learn what the interface contains, and they rely on visual search to identify commands. During the associative phase, users know what the interface contains, and they begin to focus more on how the execution occurs. They begin to remember where in the UI each command is located, and can move there more and more quickly as they build experience. During the autonomous phase, people attain automaticity – they can execute commands quickly, without attention, and in parallel with other activities.

In addition, research has shown that although incidental learning is possible,

particularly with spatial locations, the depth of mental effort put into learning an interface can be correlated with their eventual memory of the interaction mechanisms (e.g., the gestures in a command set, or the location of items in the UI). Craik and Lockhart's (1972) "levels of processing" framework suggests that a deeper, more effortful, mental encoding in memory leads to faster retrieval and longer persistence. The relationship between effort and learning has been demonstrated in research on learning object locations and learning shape-writing. In the motor learning literature, deliberate practice has been identified as a key requirement for acquiring expert performance.

Despite the increased performance ceiling of expert interfaces, however, several studies of real-world use show a tendency for users to persist with slower, suboptimal methods. Researchers have suggested several reasons for this phenomenon, including:

- Satisficing. Users may opt for a strategy that they know is "good enough" for their current purposes, even if they know that a better solution exists (Simon 1987).
- Paradox of the active user. Carroll and Rosson (1987) suggest that users who are engaged in ongoing tasks will often continue using known methods rather than learning new ones, and will generally apply known methods to new problem situations.
- The value of feedback. Fu and Gray (2004) suggest that users can prefer well-practiced novice methods if these provide fast and incremental feedback (particularly in the associative phase).
- The "guidance hypothesis." Guidance provided to facilitate learning of an expert technique (e.g., feedback provided during an action) can become relied upon, degrading retention and performance when the guidance is no longer present (Schmidt, 1991).
- Local optimality. For any single action, using a known but slow mechanism is likely to be faster than learning a new one (Gray et al., 2006).
- Performance dips. Switching to a new interaction modality usually incurs a performance dip (as users must learn the new techniques); users may therefore be reluctant to switch because it means a (temporary) reduction in performance (Scarr et al., 2011).

Researchers have considered several methods for helping users over these obstacles – for example, by punishing the use of the novice method, by increasing awareness of the expert method, by providing feedforward to support expert command execution, or by showing the user how much their performance could increase if they switched to the expert method. An alternate approach, however, is to design techniques that do not require overt methods of encouraging or forcing the user to switch to the expert method, and rather provide a natural and gradual transition from novice to expert behavior.

Several interfaces have been proposed that attempt to avoid the “performance dip” between novice and expert use. These systems use Kurtenbach’s (1993) principle of rehearsal to enable knowledge transfer from novice to expert methods. The principle states that novices should carry out selection actions in the same way that experts do; therefore, incidental learning will happen through everyday use, and as users gain experience with the interface, they will gradually build up the memory that they need to use the expert method. Feedback and guidance appear for novices, but as users become more experienced, these supports can be removed.

Kurtenbach explored rehearsal in detail with the Marking Menu technique (Kurtenbach, 1993). Novices use this technique as a standard radial menu, in which the menu’s visual representation appears a short time after the user holds their stylus down on the screen. As users gain experience with the locations of items in the menu, they can start converting the navigation motions needed to reach the item into a gestural “mark” – which can be performed without needing to wait for the visual guidance of the menu. Once expert, users simply draw the marks that correspond to the items they want to select, which is much faster. Several other techniques have also used the principle of rehearsal. For example, the SHARK text input technique allows users to move from touching individual keys on a virtual keyboard to shapes for words, where the shapes are a fast version of the novice’s movement from key to key. Similarly, the ExposeHotKey system (Malacria et al., 2014) allowed people to select toolbar items using the same mechanism as they used for hotkeys; as users learned the key combinations, they used the visual guidance of the toolbar less and less.

### **Design Goals for Shortcuts on Touch Surfaces**

As shown in Figure 1, the FastTap interface works by displaying a CommandMap over the main workspace when the user places their thumb on an on-screen activation area. In this section, we discuss the design goals behind FastTap and frame them in the context of related work.

#### **Design Goal 1: Enable rapid command execution**

Modern touch-screen applications typically use hierarchical menus and dialogs, with a few commands accessible on the main display, and others requiring several pointing actions that slow their selection. Gesture-based systems, such as marking menus, are one alternative to hierarchical linear menus, with some implementations appearing in commercial products (e.g., Autodesk Sketchbook Pro). Marking menus allow practiced users to traverse a radial menu hierarchy in a single gesture, speeding up interaction. However, navigating the hierarchy can still be slow, and marking menus are inherently limited in terms of the number of items that can appear at each hierarchical level (see Design Goal 3 for extensions to marking menus that can reduce this problem).

Multi-touch technology has created new opportunities for designing efficient command selection interfaces that exploit the higher bandwidth available

with multiple concurrent contacts. For example, Wu and Balakrishnan (2003) describe multi-finger and whole-hand interaction techniques for tables, including a selection mechanism that posts a toolglass with the thumb, allowing selection with another finger. Similar techniques are used and studied in Wagner et al.'s (2012) BiTouch system, but with a focus on handheld tablets where the thumb of the non-dominant hand is used to post interface components.

Multitouch marking menus (Lepinski et al., 2010) and finger-count menus (Bailly et al., 2008) both allow users to specify a menu category by changing the number of fingers used to touch the screen (thus reducing the number of levels that must be traversed). Other techniques parallelize the hierarchy traversal: for example, Banovic et al.'s (2011) multi-finger pie menu allows users to post the menu with one finger and select an item with another; Kin et al.'s two-handed marking menus (2011) allow users to draw the marks for different menu levels simultaneously, by using both thumbs. However, these higher-bandwidth techniques do not always improve performance, since a more-complex control action may take more time to retrieve and execute.

One key characteristic that determines whether a command selection interface is efficient is the number of separate actions needed to navigate to an item. Reducing this number is a main design goal of Scarr et al.'s (2012) CommandMap technique, which uses a full-screen overlay to display as many commands as possible at once. These commands can be selected with a single action, which is faster than the multiple navigational steps needed for hierarchical menus and ribbons. Scarr et al. also showed that navigational errors (e.g., choosing the wrong menu) substantially increased the time needed for hierarchical organizations. In this work, we adapt the CommandMap's flat and spatially stable design to work with mobile devices.

### **Design Goal 2: Support a transition to expertise**

One of the primary advantages of marking menus is the way in which they support a smooth and rapid transition to expert use. After activating a marking menu, a novice user can wait for a short time to see a labeled radial menu appear, from which they make their selection with a touch gesture; an expert user can make the same gesture without waiting for the menu to appear. Since the motor actions for the novice and expert uses of the menu are identical, users learn the expert gestures through normal interaction.

This principle of rehearsal is extremely important to the development of user expertise. FastTap is therefore designed to support rehearsal during novice use. In a similar manner to marking menus, the command grid only appears on-screen after a delay; users with spatial or muscle memory of the interface can interact instantly without waiting for the visual display, using the exact physical action they used as novices; intermediate users suspecting the desired command location but unwilling to execute it without confirmation can also benefit from FastTap, by anticipating the location of the target, positioning their finger over it while waiting for the grid to appear, and

selecting the command after visual confirmation.

### **Design Goal 3: Support a large number of commands**

While marking menus are generally limited to eight or twelve commands per level (Kurtenbach, 1993), various extensions significantly increase this limit. Polygon menus and flower menus both allow more commands by increasing the types of gestures available. More recently, Roy et al. (2013) developed Augmented Letters, a system whereby users draw the first letter of a command on the screen, then select from a radial menu of resulting candidate commands. OctoPocus recognizes gestures by shape, and provides visual suggestions for the remaining gesture based on the initial movements (Bau and Mackay, 2008). While these systems increase the number of commands that marking menus can support, they still rely on gesture-based interaction.

Rapid execution is our priority for FastTap, but we also intend that it will support a wide command vocabulary. In FastTap, the number of items at each level is limited only by the size of the screen; our prototype uses a 5x4 grid, with one cell being used as the FastTap activation button. However, this number can be increased through the use of different activation buttons, or command tabs, which can be arranged along the bottom of the screen. We consider these design possibilities further in the Discussion.

### **Evaluating the Performance of FastTap**

We carried out a study to assess the performance of FastTap for command selection on tablets. We compared FastTap to marking menus, which allow fast command selection for experts and support a smooth transition to expertise. We compared the two interfaces in a controlled experiment where participants selected a set of commands over several repeated blocks, allowing us to examine both novice and more expert selection behavior.

Both FastTap and marking menus were implemented in a functional multi-touch drawing application (see Figure 2). As described above, FastTap provides modal access to a grid of command buttons. Selections are made by pressing a command button, either after invoking the grid display, or simultaneously with the invocation button (i.e., by chording). There is no difference in the selection mechanism for novice or expert use – experts who know the item locations simply tap the command before the interface is shown. After a chorded selection, feedback on the selected command is given by displaying the command icon for 500ms (Figure 1, right). The interface used in the experiment contained sixteen command buttons in a 4x4 grid, of which eight were used as study targets. The sixteen commands were organized into four rows that grouped similar commands together (see Figure 2). Marking menus were implemented as a 16-item marking menu with a two-level hierarchy, adapted from Kurtenbach's previous work. Once again, only eight of the 16 items were used as study targets. Upon invocation of the menu (described below), users move a finger towards one of four categories shown on screen (Shapes, Colors, Line Style, Line Width),



and then to one of the items in that category (see Figure 2, left). The items in each category were the same as the row groups used in FastTap (Figure 2).

Participants completed a demographics questionnaire, and then performed a sequence of selections in a custom study system with both marking menus and FastTap. For each trial, a command stimulus (consisting of one, two, or three command names) was displayed at the top of the screen; the participant then selected the command(s) using the interface provided. Trials involved selecting a combination of one, two, or three individual tools and properties that could be used within the drawing application (e.g., 'Red', 'Red Line', 'Red Thin Line'). Trials were timed from the appearance of the stimulus until all targets were successfully selected. In the case of multiple-command targets, command names in the stimulus were crossed out as they were selected, and participants could select commands in any order. Participants were instructed to complete tasks as quickly as possible, and were told that errors could be corrected simply by selecting the correct item. Completion times included the time for correcting errors.

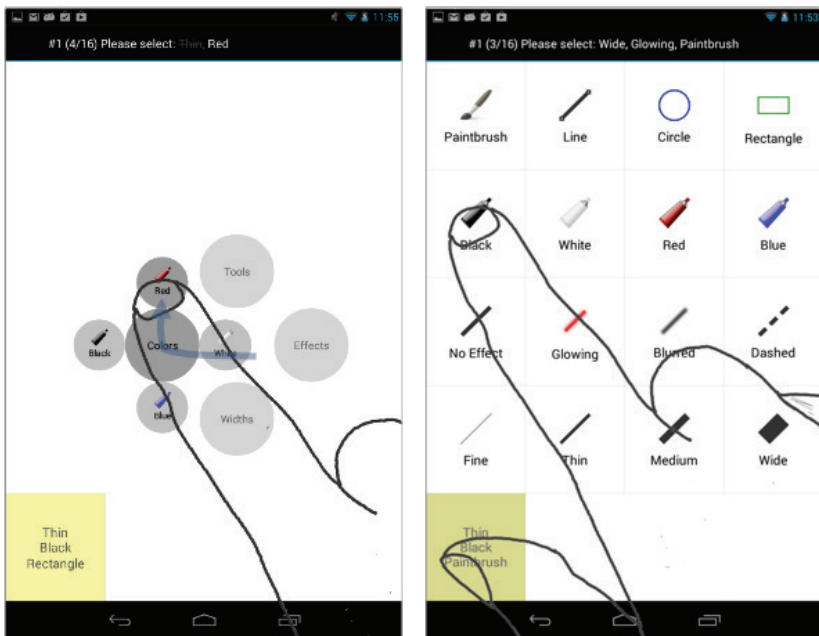


Figure 2. Study UIs. Left: marking menu (arrow shows gesture path). Right: FastTap. Cue appears at top of screen.

Of the sixteen commands in each interface, only eight were used as stimuli, in order to allow faster development of spatial memory and expertise for both interfaces. Two commands were used from each interface category (Shape, Color, Line Style, Line Width). Multiple-command targets were also composed from these eight commands.

For each interface, selection trials were organized into blocks of sixteen selections (eight one-command targets, four two-command targets, and four three-command targets). Participants first performed one practice block of sixteen trials (data discarded) to ensure that they could use the interfaces successfully. They then carried out 10 blocks of sixteen selections. Targets were presented in random order (sampling without replacement) for each block. Short rest breaks were given between blocks. After each interface, participants filled out a NASA-TLX questionnaire; at the end of the study, they also answered summary questions about their preferences.

Sixteen participants were recruited from a local university (8 female; mean age 26.2 years). The study was conducted on a Nexus 7 Android tablet (7-inch screen, 1280x800 resolution). The software was written in Java, and recorded all experimental data including selection times and errors. The study used a  $2 \times 3 \times 10$  within-participants RM-ANOVA with factors Interface (FastTap, MarkingMenu), NumberOfCommands (1, 2, or 3 commands per trial), and Block (1-10). Dependent measures were command selection time, and errors per command selection. Interface was counterbalanced between participants. Hypotheses were:

- Mean selection times for FastTap will be faster than for MarkingMenu.
- FastTap will be faster than MarkingMenu both for novices and for experts.
- FastTap will show an added speed benefit for two- and three-command targets.
- There will be no evidence of a difference in error rates between the two interfaces.
- There will be no evidence of a difference in perception of effort for the two interfaces.
- Users will prefer FastTap over MarkingMenu.

### **Results: Selection Time**

We calculated the selection time for each command by dividing the total trial time by the number of commands in that trial. As shown in Figure 3 (rightmost bars), mean selection times were about 0.8 seconds faster per command with FastTap (1.60s, s.d. 0.52s) than with MarkingMenu (2.39s, s.d. 0.65s), giving a significant main effect of Interface ( $F_{1,15}=84.37$ ,  $p<.0001$ ,  $n_2=0.85$ ). We therefore accept H1 – FastTap selections were 33% faster overall than the marking menu.

Figure 3 also shows selection time by the number of commands per trial. ANOVA showed a significant main effect of Number of commands ( $F_{2,30}=3.7$ ,  $p<.05$ ,  $n_2=0.20$ ), and also an interaction with Interface ( $F_{2,30}=6.69$ ,  $p<0.005$ ,  $n_2=0.31$ ). As suggested by Figure 3, both effects can be attributed to FastTap permitting faster selections when commands are joined into groups of two or three. Post-hoc t-tests (Bonferroni corrected)

show a significant difference between the two interfaces for each number of commands (all  $p < .0001$ ). We therefore accept H3.

As shown in Figure 4, selection times decreased across trial blocks; ANOVA showed a significant main effect of Block ( $F_{9,135}=17.49$ ,  $p < .0001$ ,  $n_2=0.54$ ). There was no interaction with Interface ( $F_{9,135}=1.43$ ,  $p=.183$ ) or with NumberOfCommands ( $F_{18,270}=1.29$ ,  $p=.194$ ). We also analysed the performance difference between interfaces at each block. Bonferroni-corrected t-tests showed that FastTap was faster than MarkingMenu at all levels of expertise (all  $p < .001$ ). Since the advantage of FastTap over marking menus is consistent across trial blocks, we accept H2.

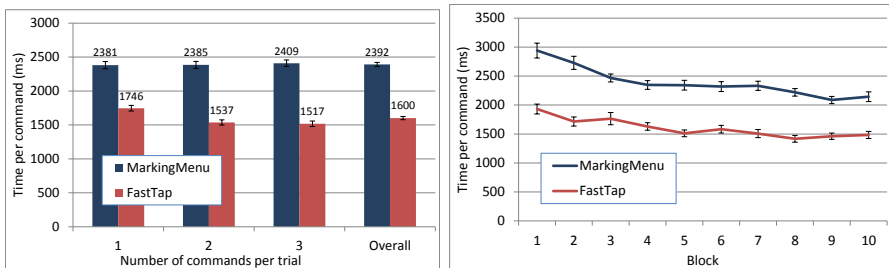


Figure 3. Mean selection times by Interface and Number (left); trial times by block (right).

### Results: Errors

As with selection time, we analysed errors per command, dividing the number of errors in a trial by the number of commands in that trial. Errors were counted as any incorrect selection (note that multiple-command trials could be carried out in any order). ANOVA showed no effect of Interface on errors, with FastTap at 0.10 errors/command, s.d. 0.13, and MarkingMenu at 0.08 errors/command, s.d. 0.11 ( $F_{1,15}=2.96$ ,  $p=.11$ ). We therefore accept H4 (errors are considered further in the discussion section). There was also no effect of Block ( $F_{9,135}=0.41$ ,  $p=.93$ ) or NumberOfCommands ( $F_{2,30}=0.51$ ,  $p=.61$ ) on errors per command; there were no interactions between any factors.

### Results: Subjective Responses

User response was positive to both interfaces, but with no strong differences in NASA-TLX scores (compared using the Friedman test; see Table 1). No significant differences were found on any effort question, and the mean scores were close in all cases; therefore, we accept H5. We also asked participants which interface they preferred in terms of several qualities (see Table 2). As with the effort scores, counts were very close with no quality showing a significant difference. When asked about overall preference, seven participants preferred FastTap, eight preferred marking menus, and one had no preference. We therefore cannot accept H6.

Like the preference results, participant comments were about evenly divided

between the two selection techniques, and participants often mentioned the characteristic features of the designs when explaining their preferences. For example, participants made several comments about how spatial stability and quick activation helped the speed of FastTap: one person commented on “the simple and stable location of each icon”; another said “the grid is faster because it only requires a two finger tap, with one finger always in the same place. The menu, however, demands specific movements (swipes) which expend a few milliseconds more time.”

It was also clear that some participants liked the semantic categories of the Marking Menu, and others liked FastTap’s access to all commands at once. In favour of marking menus, participant comments included “the menu compartmentalized the options much better”; “the menu required less memory”; and “visually a grid with all available options is not as easily navigable as a menu with divisible submenus.” In favour of FastTap, comments included: “the options are all available at one time which makes it easier to pick, and memorization also helps a lot in the grid”; “it is easier to see the items that you want to select”; and “there is only one level to be memorized, where [the marking menu] has two levels.”

Some of the participants were initially concerned with the number of commands in FastTap, but ultimately preferred it: “after a while, the grid became easier to remember”; “I was able to catch on much quicker [with FastTap] and be able to visualize where everything was.” Finally, one participant also mentioned that the sliding motions needed for the marking menu could present problems on a touch surface, and that FastTap’s tapping action did not have this problem: “sliding over a surface can be impeded by many factors like moisture [on the] hand and friction over the surface. [...] I feel the tapping action in the grid is more comfortable.”

## **Discussion**

*Why did FastTap work well for both novices and experts?* Our experiment showed that FastTap improved (by 33%) on a well-known technique (marking menus) for selecting commands on tablets; in addition, FastTap was faster at all stages of learning. The performance advantage of FastTap can be explained through an analysis of the steps required for command selection, in both novice and expert cases. For novices, there are three steps required: activating the command mode, searching for the desired command, and executing a selection action. Activating the command mode was the same in both interfaces, but the interfaces differed in the second and third steps. Searching for a command in FastTap involves only visual search over all of the concurrently displayed items, whereas search in marking menus involves first choosing and selecting a semantic category (Colours, Tools, etc.) and then searching in a much smaller set of items. Although some participants found the semantic categories helpful, previous research has shown that the decision and selection time costs of traversing the hierarchy can exceed that of a broader visual search (Scarr et al., 2012). The visual search needed for FastTap could take longer initially, since there are more items to inspect,

but the rapid development of spatial memory means that novice users will quickly make a transition from full visual search to remembering where items are located. The third step – execution of the selection action – is faster in FastTap, since a tapping action can be carried out more quickly than the sliding motion of a marking menu.

For experts, selection requires only two steps: retrieval of the command action (either a location or a gesture) from memory, and execution of that action. The performance advantage for expert use of FastTap most likely arises from the speed of execution, since the memory-retrieval step is similar for both techniques. Execution of a thumb-and-finger tapping is a faster action than a drawn gesture, leading to a performance advantage for FastTap.

*Why were multiple selections faster than single selections?* Multiple commands selected with FastTap were each about 200ms faster than single commands. We believe that this speed-up is due to people's ability to visualize multiple command locations within FastTap's grid interface, and optimize their movements for multiple selections. For example, we observed participants re-ordering the selections to reduce finger movement (e.g., ordering the selections by grid row allowed people to move their finger in a straighter line). With the marking menu, there is less opportunity for optimization (e.g., re-ordering the commands does not reduce overall movement).

*Does FastTap lead to more errors?* Error rates were high overall in both techniques (10% and 8% for FastTap and marking menus). This high error rate is probably an artifact of our experimental protocol, which explicitly instructed participants to select items as quickly as possible, while noting that errors could be corrected afterwards. Although FastTap had a slightly higher mean error rate (10%) than marking menus (8%), the difference was not statistically significant ( $p=.11$ ). However, if we assume that effect of higher errors with FastTap is actually a reliable one, we see three candidate explanations. First, the quick execution of a selection action in FastTap may have encouraged participants to view errors as amenable to rapid correction, thereby encouraging users towards a 'guess and correct' mode of operation. Second, participants may have found the post-selection visual feedback in FastTap more clearly communicative of the selected item than the marking menu, again encouraging faster but more error prone selections. Third, it is possible that people's memory of an item's spatial location was imperfect, and so participants may have experienced 'near misses' more often than with marking menus. Further work is needed to determine whether the difference in error rates is reliable, and to properly explain its cause if it is.

*FastTap and Marking Menus with Larger Item Sets.* Our study tested FastTap and marking menus with a small command set (sixteen items), and it is worth considering how the two approaches would compare in the case of

a larger interface. The main performance differences between FastTap and marking menus are in initial visual search (novice behavior) and the number of actions needed for a learned item (expert behavior). These differences arise due to the properties of FastTap's single-level presentation, compared with marking menu's hierarchical organization.

With a larger command set, it seems likely that FastTap will require more visual search than marking menus – provided the hierarchical organization of the marking menu items is clear (which can be a non-trivial design problem). However, several prior studies suggest that the novice period of use is relatively short and users will soon be interacting with commands that they are familiar with. In this case, FastTap selection time will be similar to that observed in our study – since only a single invocation and selection action is needed, regardless of the size of the command set. Marking menus, however, must create hierarchies because they use less space, and so experts must continue to execute multiple navigational steps (even if these are executed as marks).

Recent work on multi-touch marking menus suggests ways that these navigation interactions can be speeded up – e.g., by encoding top-level menu categories with different finger postures, or by parallelizing the execution of navigational marks. Although we believe that FastTap will continue to compare well against other selection techniques as command sets increase in size, further work is needed to explore the potential differences between the spatial-memory approach of FastTap and the gesture-memory approach of marking menus.

### **Design issues for FastTap**

*How many commands can the interface accommodate?* Our drawing interface contained 19 commands in a 5×4 grid, but this is not the limit for the FastTap approach. In general, the number of commands in the interface is limited by the size of the device, the minimum desired size of the targets, and the size of the user's hand (to facilitate chording). Using the average width of an adult index finger (16-20mm) as a guideline, it would be possible to have a grid of up to 40 buttons in an 8×5 layout on a 7-inch tablet. If we assume that touch targets should be no smaller than 9.6mm (Parhi et al., 2003), a maximum density of 150 items in a 15×10 arrangement is possible. However, this size guideline is debatable – larger button sizes have been shown to provide higher success and satisfaction rates, yet much smaller targets can also be used, as demonstrated by smartphone virtual keypads (which can be operated substantially eyes-free with sufficient practice). There are interesting research questions around users' ability to form spatial and motor memories for varying numbers of targets at different target densities.

Importantly, however, the possibilities for adapting FastTap to high functionality applications, or to small displays, is not necessarily limited to the 'all commands at once' designs examined in this paper. The capacity of

the interface to display candidate targets can be multiplied through the use of multiple trigger buttons, or 'tabs', which organize the commands into multiple categories – each trigger button would then show a separate set of commands. As discussed under Design Goal #3, this technique requires the allocation of additional trigger-button regions in the bottom row of the grid. Further research is required to determine how well people can remember these two-finger combinations.

Finally, it is also possible that if the grid interface cannot accommodate all of the application's commands, it could still function as a shortcut list for frequently-used items.

*Issues regarding device orientation.* Mobile devices such as tablet computers can be used in different orientations, which changes the aspect ratio of the screen. There are three possibilities to accommodate orientation changes. First, the grid could maintain its overall aspect ratio and scale to fit the smaller dimension of the new orientation, requiring that users adapt to a different-scale interface. Second, the grid could change its aspect ratio to fill the new orientation, requiring that users adapt to a stretched version of the grid. As shown by Scarr et al. (2012), these two transformations would cause only a minor disruption to spatial selections. Finally, the grid could maintain its size regardless of the orientation, fitting the most constraining orientation.

*Multiple simultaneous selection.* Normal expert selection in FastTap involves two digits (normally the thumb and one finger; but sometimes two different fingers). This combination selects one command; however, there is no reason why additional commands cannot be selected simultaneously in the same chorded tap. These kinds of selections already work in our prototype application. Added-finger selections work well only for certain combinations (because of constraints on positioning the fingers); therefore, if these are to be used in interface design, further work must determine which fingers combinations are possible, and which command combinations are desirable.

## **Conclusions**

Although multi-touch tablets are now common, and are starting to be used for productivity work, there are few techniques for these devices to support rapid command selection. In this paper, we presented a new selection technique for multi-touch tablets called FastTap that uses thumb-and-finger touches to show and choose from a grid-based overlay interface. FastTap allows novices to view and inspect the full interface, but once item locations are known, FastTap also allows people to select commands with a single quick thumb-and-finger tap. The interface helps users move toward expert use, since the motor actions carried out in novice mode rehearse the expert behavior. A controlled study with 16 participants showed that FastTap was significantly faster (0.8 sec/selection, 33%) than marking menus, both for novices and experts, and without reduction in accuracy or subjective

preference.

Our research thus far with FastTap gives us strong initial results, and future work will continue in three directions. First, we will continue development of the drawing application and release a fully-functional version of the app, in order to gather real-world usage and performance data from a wide audience. Second, we will develop new prototypes that explore some of the design issues described above, including tabbed command sets, complex interface widgets, and interface scaling for larger devices. Third, we will develop FastTap prototypes for other applications that could benefit from fast access to commands and shortcuts (e.g., contacts and response options in a mail client, or favourites and page actions in a web browser). Our experience with FastTap suggests that the underlying ideas of spatial-memory-based expertise and quick tap-based access can be successfully and broadly applied across several application domains.





## **Transmogrification: Casual Manipulation of Visual Information**

*John Brosz, Miguel A. Nacenta, Richard Pusch, Christophe Hurter, and Sheelagh Carpendale*

### **Introduction**

It is through manipulation and exploration that we develop understanding of the world around us. Our goal in this work is to enhance our understanding of visual information by providing a method to enable flexible exploration and manipulation of image data.

One common feature of almost all information visualization techniques is that they require a great deal of work to achieve. They require data, often in particular formats, as well as a significant amount of time spent configuring the visualization itself. While these prepared visualizations can be effective, they cannot be used scenarios where time and data are not available, such as in discussions and meetings.

Transmogrification, the technique described in this chapter, was created with the goal of enabling visual arguments to be made when time is limited and data is not available. It particularly focuses on the scenario where visuals exist, but not in the configuration that would make the necessary point. Rather than requiring data, transmogrification makes use of digital imagery. In this fashion existing visualizations, digital images, pdfs, videos or any other materials capable of being shown on a digital display can be used as source material. Consequently, transmogrification is a technique for manipulation of visual information.

A historic example of where such visual manipulation was useful lies in the medieval route maps of Matthew Paris. In these maps (e.g., Figure 1), rather than showing a spatial layout of the journey that we would expect, the maps are made up of the way-points of the trip laid out almost linearly one after another. Transmogrification is designed to allow people to do just this, take a typical spatial map of a route and then straighten it into a linear presentation.

## Transforming in a Magical Way

So with this goal of enabling fast and easy manipulations of imagery we set out to, not just create software, but to design an interface that would empower these types of interactions.

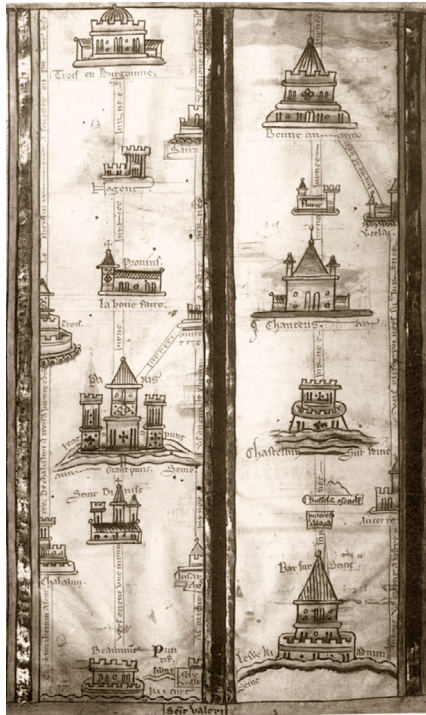


Figure 1. Matthew Paris, *Itinerary from Beaumont to Beaune*, British Library, ms Royal 14 C VII, fol. 2v.

There were three key elements that were necessary to support our goal:

- A multi-touch interface: specifying shapes with mouse input has often devolved into manipulating controls points and other tedious interactions. Multi-touch provides a great deal of control and freedom, an extremely fast mechanism for interaction, and a feeling of directly working upon the visual information.
- Data is not required: by aiming for casual environments we cannot assume that the people we are designing for have the data with them, or even access to the underlying data at all. With the prevalence of digital phones, it is easy to create imagery of whatever visual information one wishes to manipulate – whether that is a sketch on a napkin, or a precisely rendered scatterplot of millions of points.
- Understandable transformations: the last aspect is that for transmutations to be useful, it has to be clear to anyone using them how they work. This led us to two important elements of our interface; animated transitions & “live” results. Animated transitions are incorporated by, anytime a transmutation is specified, animating

the transformation from one shape to another so that it is clear why the resulting image appears as it does. The second element is that the result of transmogrication is real-time and updated on the fly. One can shift and move the source visual and the output will change with every movement, making the result easier to modify and understand, promoting further exploration and understanding.

### How to Transmoglify

Transmogrication operates by transforming an image from one shape to another. We call the starting shape the source and the final shape the destination.

So how do we control this transformation in an understandable way? To do this we describe shapes by specifying three curves: two curves that mark the edges of the shape, the boundary curves, and the spine curve. The spine is often midway between the boundary curves but, depending on the desired, transformation, may be placed differently. This technique is loosely based on Hsu et al.'s (1993) skeletal distortions. To provide intuition for how the transformation occurs, examine Figure 2 and consider that it is relatively easy to change any of the shapes on the left to the center rectangle by straightening the shape curves (no matter the form these curves may take). Conversely we can change the rectangular shape to any other by adjusting its curves as necessary. Consequently we can easily transform any shape defined by three such curves into any other.

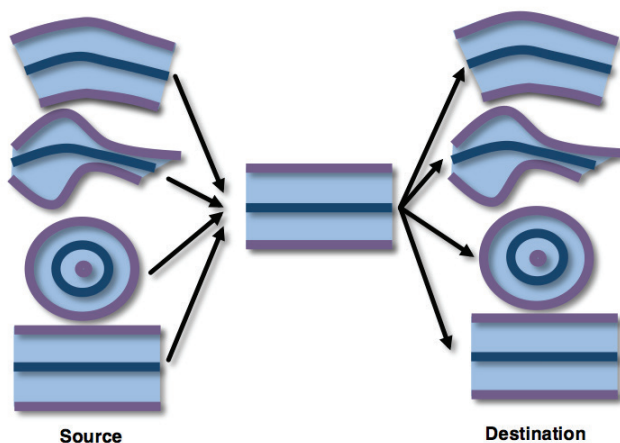


Figure 2. Transmogrication occurs through transforming a source shape into a destination shape. Boundary curves are purple, spine curves are blue. Any source shape can be transformed into any destination shape.

Once we know how to change shapes into one another, to transmoglify we need to transform images underneath such shapes into the new shape's configuration. To do so we make use of standard computer graphics texture mapping. Before going on, let us discuss how these shapes get rendered

(drawn). For each shape, in the terms of OpenGL, we would render by creating two sets of triangle strips, one between each boundary and the spine. Now to transmogrify we take the imagery under the source shape and then for each point on the source shape's three curves we calculate the corresponding texture coordinate of that image. That is, we would calculate the same texture coordinates as if we wanted to render the source shape texture mapped with the underlying image. However, instead of using the geometry of the source shape, we use the geometric positions of the destination shape. Texture mapping then performs all the hard work of distorting the imagery. For a more complete technical description of how this is achieved please consult Brosz et al. (2013).

### Transmogrification Examples

Now that the basics of transmogrification have been explained we will explore a variety of cases where transmogrification can be put to use. The first is in creating route maps. These are the aforementioned maps used by Matthew Paris and others to describe journeys. By aligning a transmogrification source with the path taken on a normal map (Figure 3) the transmogrified result is an easily identified route map. In its default behavior the transmogrification preserves the distance along the spine curve (the middle of the three green lines in Figure 3). This can be useful in comparing several route maps at once. In the example shown in Figure 4, six flight approaches into the Lyons airport are straightened so that the distance and changes in elevation of each flight's approach can be directly compared.

Another example of rectifying an image is the scenario of straightening rivers. This straightening was used in old atlases to provide a comparison for the length of rivers. An example of this can be seen in J. H. Colton's visualization that was recently analyzed by Tufte (1990, p. 77). We can easily recreate this visualization through transmogrification by merely rectifying the paths of the different rivers as shown in Figure 5. The straightened rivers retain differentiating characteristics while being straightened sufficiently for easy, visual comparison.

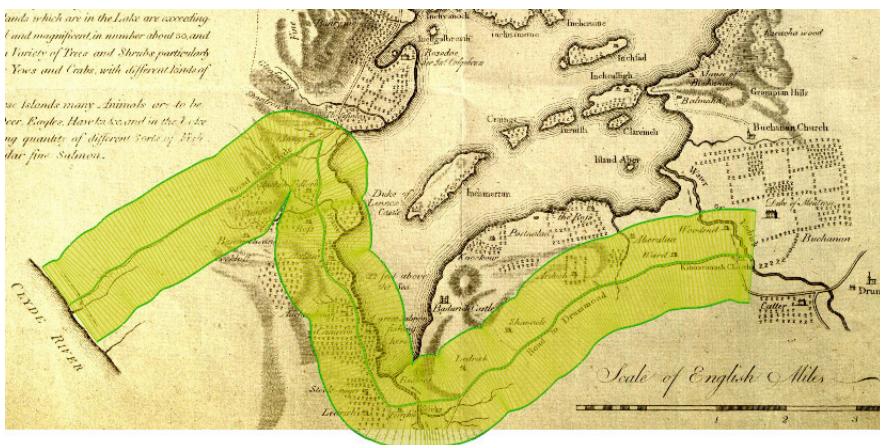




Figure 3. Creating a route map (bottom) by tracing a path (top).

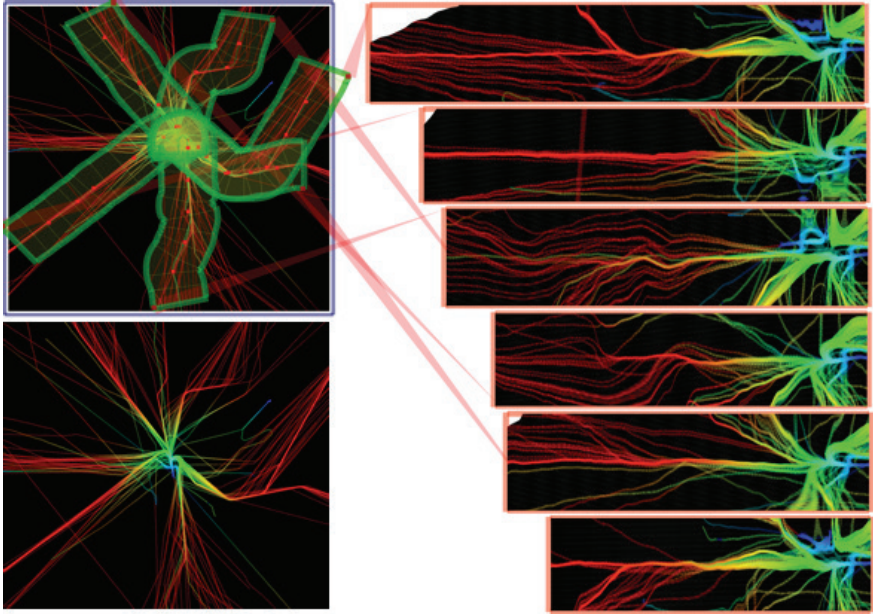


Figure 4. Transmogrified aircraft flightpaths. Original imagery is bottom-left, top-right image portrays the seven source shapes, while the transmogrified results on right.

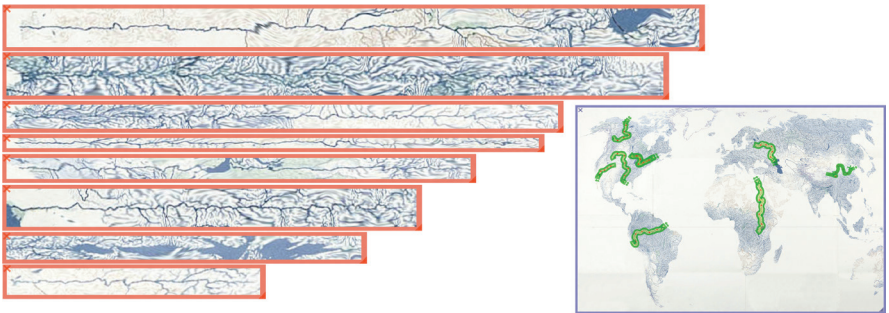


Figure 5. Straightening rivers to show side-by-side comparisons of their lengths (left). The green lines on the original map show the shape of the un-straightened rivers (right).

A more advanced scenario is shown in Figure 6 where a cycling route is straightened for comparison to distance-based elevation and heart rate data (Figure 6B). This allows viewers to easily determine which parts of the route required the most effort. However, it may be the case that viewers still have difficulty determining where high heart rates occurred along the route.

To address this, the heart-rate chart can then be transmogrified to wrap itself along the cycling route (Figure 6C). This makes it clear geographically where high heart rates occurred. Figure 6D takes this a step further showing both the heart rate overlaying the elevation data being wrapped along the cycling route.

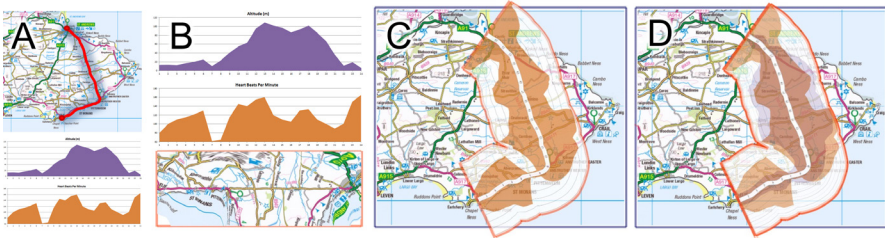


Figure 6. A cycling route (A) is rectified for direct comparison to elevation and heart rate data (B). The heart rate charts is then transmogrified around the route within the map to show effort within the spatial context. (D) is same as C, but with elevation data included. Map contains Ordnance Survey data © Crown copyright and database right 2013.

Another frequently occurring scenario exists when a chart is not within one's preferred format such as a pie chart that may be more easily read as a bar chart. In Figure 7 we change an icicle tree representation into a sunburst. This is done by using a rectangle as our source shape and a circle as the destination shape.

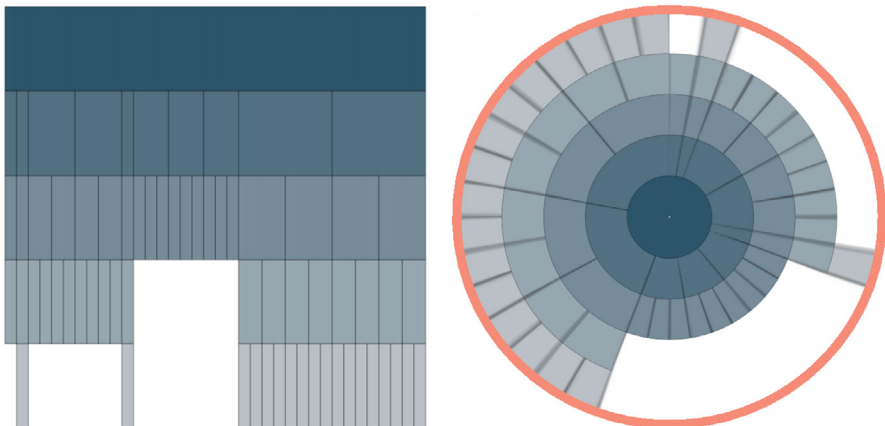


Figure 7. Icicle tree representation (left), transmogrified to a sunburst representation (right).

Transmogrifications can also be daisy-chained, that is we can further transmogrify imagery created through transmogrification. An example of this is shown in Figure 8 where two transmogrifications are used to remove the tied games out of a Nightingale chart outlining wins, losses, and draws of a football team. To do this a circle source shape is placed over Nightingale chart to transmogrify it to a rectangular destination creating a bar chart. Then another rectangle source shape is placed over the non-draw

bars (i.e., the blue and red bars related to wins and losses), which is then transmogrified back into a circle shape, creating a Nightingale chart of only the wins and losses.

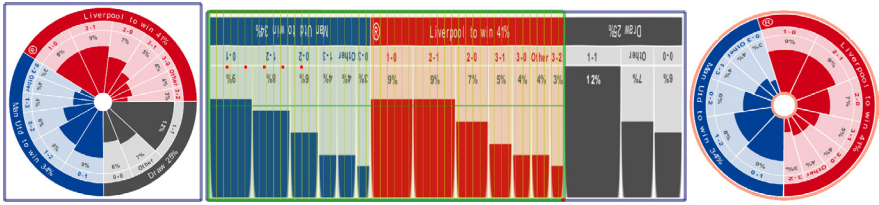


Figure 8. A chart (left) is changed to bar chart (middle), a subset of which is changed back into a Nightingale chart (right).

The fast nature of transmogrification makes it ideal to quickly prototype any sort of distortion-based visualization or interaction technique. To demonstrate this possibility we have re-created Mackinlay et al.’s Perspective Wall demonstration (1991) for a calendar using three transmogrifiers placed side-by-side (Figure 9). Note that while the figure is static, within the software the distortions occur in real-time. That is, different days can be magnified by moving the calendar image; this allows interactions with the distortion to be experienced.

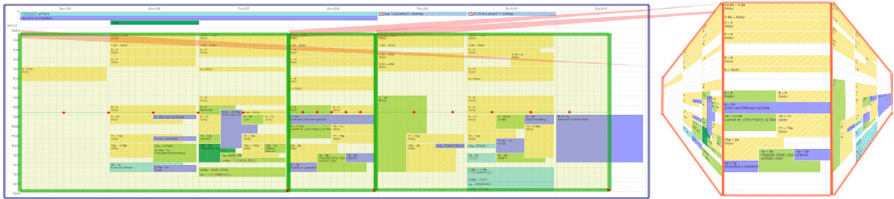


Figure 9. Creating a Perspective Wall distortion using three transmogrifiers. The three green rectangles are the source shapes, the three orange quadrilaterals are the destination shapes.

In all the examples shown thus far the transmogrification has preserved the distance along the center of the source shape to match the distance along the destination shape. However this mapping of space can be interactively or programmatically changed. One example is shown in Figure 10 where we transmogrify a driving route between Los Angeles and Los Vegas. The destination shape was created by scaling the path by the amount of time spent at each point on the route, stretching the parts of the route where one would travel slowly and compressing the parts travelled quickly. This is reminiscent of the hand-drawn route maps discussed by Agrawala and Stolte (2003).

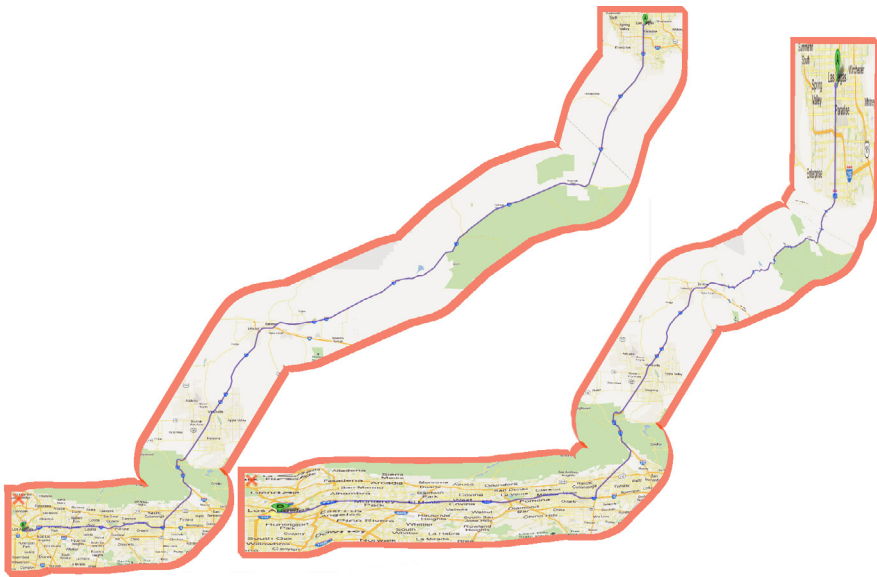


Figure 10. Transmogrification of a route (left) between Los Angeles and Los Vegas where slowly travelled regions are expanded and quickly travel areas are compressed.

Another example of disrupting the preservation of distance in transmogrifiers lies in the possible of correcting perspective distortion; that is, showing the wall as if it was orthographically projected. In the center image of Figure 11 we transmogrify a wall shown in perspective projection to a rectangle. While this adjusts the wall to a nice rectangular shape the spacing between windows and other details is not uniform as we would expect of an orthographic project. The right image of Figure 11 has used interactive scaling to shift points along the spine curve to create the uniform spacing of the wall's features.

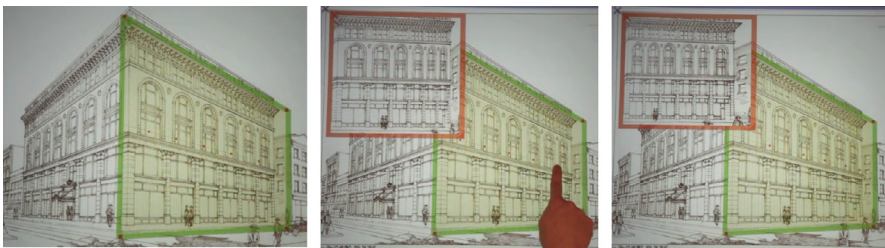


Figure 11: Transforming the side of a building from perspective to orthographic projection. The center image is the first attempt where windows and other features are expanded on the left and compressed on the right side of the wall. The right image is the result of adjusting the source shape's interactive spacing points (red dots) to create a uniform distribution across the wall.



## **Summary & Future Directions**

Transmogrification is a technique that provides a controllable and fast way to manipulate 2D images and visual data. It provides people with the opportunity for manipulation and exploration to better understand visual information.

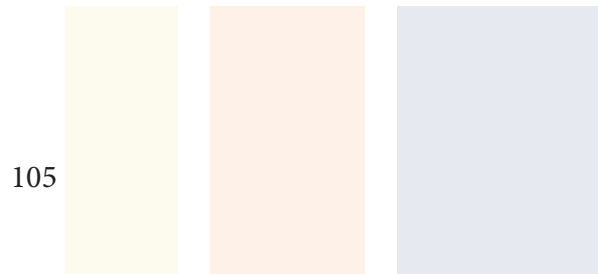
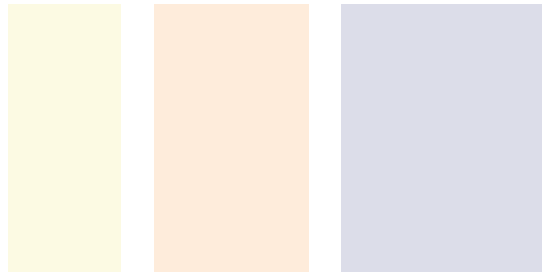
Through the included case studies we have shown that transmogrification is useful in a variety of scenarios: rectifying content, wrapping content around or within other content, combining different sources of content, providing distortions based on the features of the image, transform chart layout, and many other possibilities.

At this point, transmogrification has a limitation in that it can heavily distort text to the point that it becomes unreadable. To address this problem, text (or other symbols that are required to be legible in their original form) needs to be identified in the source imagery and then redrawn at their corresponding new positions in the transmogrified result. Automatically identifying such text and calculating this text's new position are challenging tasks.



Three vertical bars of equal height, colored yellow, orange, and blue from left to right, partially obscured by the text below.

# IMPROVING SOFTWARE TIME TO MARKET







## IMPROVING SOFTWARE TIME TO MARKET

*Robert Biddle, Carleton University*  
*Kevin Schneider, University of Saskatchewan*

Focus Areas:

- 2.1 Agile Development and Human-Centered Design
- 2.2 Application Specific Development Processes
- 2.3 Requirements Analysis and Testing
- 2.4 Development Tools
- 2.5 Collaborative Management of Software Development

### **I**ntroduction

Theme 2 of SurfNet concerns the process of building software in the context of surface computing. There are several aspects, but in particular this includes surface applications to support the development process, and also how that process might be adapted to support the development of surface applications. In the sections below, we outline highlights of work on theme 2 in the latter years of SurfNet.

#### **2.1 Agile Development and Human-Centered Design**

Sub-theme 2.1 involved exploring the links between software engineering and user interaction design. We were especially interested in the “agile” approach to software engineering, because both that and UI design are iterative and emphasize collaboration. There are were many projects contributed on this sub-theme. Exploring best practices was the most common approach, featuring in several projects led by Carpendale and Maurer in Calgary on geo-exploration (Rodrigues, 2014) and by Biddle

at Carleton on card walls (Gossage et al., 2015) addressed specialized domains, and the Carleton work on collaborative analysis surveyed a broader field (Brown et al., 2013). Projects led by Kienzle at McGill as part of his TouchRAM project (touch applications for reusable aspect modelling) featured in this area, and used UI design methods for model-based software engineering (Kienzle, 2013). Work led by Carpendale and Sillito at Calgary on lifecycles of diagrams and sketches also contributed (Walney et al., 2015). All these projects involved consideration of collaboration between the design and engineering efforts, in an effort to understand and explore how best to organize project work; issues relating to analysis and testing were especially of interest. Creating lightweight tool support for collaborative development was also common. This was involved in Maurer's project on discussion tools for shared understanding of data (Paredes et al., 2014), and in Biddle's projects on collaborative tools for large touch screens (Wilson et al., 2013; Wilson et al., 2014). The main ideas here involve interactive visualizations, and other ways of showing analysis results and design decisions to help software teams better understand and reflect. A third approach was to focus on usability evaluation of surface applications. The projects led by Maurer on the "TableNOC" system (Sharma et al., 2012) and led by Mandrake (Dergousoff et al., 2015) and Gutwin (Cechanowicz, 2013) on Gamification, tackled this issue. These projects especially explored on usability evaluation for specific kinds of software, monitoring systems and gamified work-support software. Addressing a more general approach, Issa, Sillito and Garousi investigated "visual testing" (Issa, 2012).

## **2.2 Application Specific Development Processes**

Sub-theme 2.2 had a focus on processes for specific kinds of software. The idea was that some kinds of software are created with processes not typical across software engineering. Games are a common example, because of the need for collaboration across an especially wide range of disciplines, and also because of the delicate balance necessary to make games engaging to play. Specific areas addressed were game development teamwork and game prototyping. Graham's group at Queens with their project on "Game Orchestration", involves innovative work on "live" gameplay creation, using ideas from the "game master" in typical in role-playing games (Graham et al., 2012 and 2014). Also, Hancock's work at Waterloo concerns prototyping for a new kind of game to foster social innovation, and is done to help collaborative solutions to complex cultural, economic or policy problems (Watson et al., 2013) Of course, not only games require special processes. One general idea, software "product lines", involves creating a range of software applications all at once to improve reuse. This topic was the focus of much earlier work by Maurer's group (Ghanam and Maurer, 2010), and also a motivation for work at Saskatchewan on elements of complex image processing software (Asaduzzaman et al., 2013; Haque et al., 2013).

## **2.3 Requirements Analysis and Testing**

Sub-Theme 2.3 had a specific focus at what are the traditionally the two ends of the software process, requirements analysis and testing. The reason

is that these are the places where the engineering process must especially connect with the greater context the software must serve. Usability testing was mentioned in sub-theme 2.1, above, but here we addressed functionality testing. One approach involved “Test-Driven Development” for surface interaction, and had two motivations. One is that in agile processes testing work starts early, and the tests are specified early in order to “drive” development rather than merely check it afterward. The other motivation is that for surfaces, the touch-interface typically features novel interaction, and we are interested in seeing how that affects the design. Maurer’s group did much work in this area earlier in SurfNet (Hellmann et al., 2010 and 2011). Maurer’s “TableNOC” project, a new approach to a monitoring and control environment, contributes to this by positioning the requirements as “tests” to be met (Sharma et al., 2012). A more flexible approach to testing is involved is “Exploratory Testing”. This supports a more epistemic process where the findings emerge and reframe our findings and expectations about the software. The TableNOC project also contributes here, because the idea is explore how monitoring software should work. Ehud’s work on “Two-sided Transparent Displays” also has contributions because of the novelty of the platform, where interaction happens on both sides of a surface. A prototype two-sided transparent display has been developed using projection from both sides, and it supports touch and tracking of hands on either side of the display (Li et al., 2014) Biddle’s group at Carleton also took a new approach to requirements emphasizing the how surfaces seem to encourage more exploratory interaction (Gossage et al., 2015; Simonyi, 2015).

## **2.4 Development Tools**

Sub-theme 2.4 had a focus on development tools, and we mean this to include a range of work. In particular, we intended to emphasize tools that help create surface applications, but we also wanted to include tools that were themselves surface applications. When we began the network, one articulated interest was conversion, that being tools for migrating desktop applications to surfaces”. In later years, this was addressed Schneider’s work with his colleague Roy and students exploring tools to support software maintenance, where their interest was leveraging tools on detection of cloned elements of software, converting them to use surface interaction (Zibran et al., 2013; Saha et al., 2013). Several projects in the network addressed tools to support surface application debugging and evaluation. These included the earlier work of Gutwin’s group on visualization of version-based collaborative processes, again addressing the challenges of testing complex software applications involving concurrency (Xue et al., 2011). Even more projects, however, involved surface computing tools to support software development. The projects contributing include Schneider’s and Maurer’s projects mentioned above, but also a number of projects across the theme. Maurer’s “discussion tools” project (Paredes et al., 2014) is involved, and Gutwin’s project on version-based collaborative processes also involves tool support, as does Kienzle’s “TouchRAM” project (Kienzle, 2013), Sharlin’s two-sided transparent display (Li et al., 2014), and Biddle’s group work on add-ons to improve collaborative exploration (Simonyi et al.,

2015, Mirza et al., 2015). This is a wide range of projects with varying aims, but the common thread is this: as we build all kinds of surface applications, we also build supporting tools that are also surface applications.

## **2.5 Collaborative Management of Software Development**

Sub-theme 2.5 was about collaborative management of the development process. In particular, this is about the use of software technology in the development process, and so has a special relationship with the software teamroom application area. One area was on management of distributed software development. Another area involved issue management, and so bug tracking. There was also work on reviewing source code, which includes Roy and Schneider's work on tools to detect code clones (Zibran et al., 2013; Saha et al., 2013). This area also includes the ongoing work of Biddle's group on "Understanding Artefacts for SE-UI Collaboration" where fieldwork identifies the kinds of artefacts (diagrams, conceptual devices, etc.) used by teams to coordinate specific activities; for example the new idea for understanding multi-tasking in operations centres (Samaroo et al., 2013), and the findings on digital agile card walls (Gossage et al., 2015). The area also includes several contributions from Anslow's work with Biddle on Software Visualization for collaborative review (Anslow et al., 2013 and 2015), and also the work led by Maurer and Anslow on visualization in the development process (Paredes et al., 2014, Bhaskar et al., 2014).

## **Conclusions**

Looking back on the SurfNet work in Theme 2, several trends emerge. The main force at work is the compelling nature of the interaction enabled by touch surfaces. Of course, this is the topic of Theme 1 (Humanizing the Digital Interface) and the technology to support that is the topic of Theme 3 (Building Infrastructure for Digital Surfaces). But in the work on Theme 2, we see that the new interaction also affects the software process. The interaction newly supported in surface computing includes touch and gesture. These seem more "direct" than earlier interaction styles, in much the same way the Graphical User Interface (GUI) mouse-cursor interaction seemed more direct than commands and menus. This point was made well in the paper by Lee et al. (2012). The trends we can identify roughly correspond to the five sub-themes discussed above, and we outline them as follows:

*Alignment with Interaction Design.* Both interaction design and agile software engineering are iterative human-centred processes, but combining the two processes can be difficult, even if promising (Fox et al., 2008). Our work suggests that it may be promising to document best practices that connect surface interaction with the underlying software. Moreover, there seem to be some general patterns about where and how these connections occur.

*Some Domains Match the Strengths of New Interaction.* The support for easy exploration afforded by surface computing seems to suit some domains especially well, namely those where some kind of exploration



is critical. This is important in game design, for example, because of the delicate balance that allows playability; it is also important in kind of image analysis, and also in other kinds of analysis work. However, GUIs were not always superior to the command line, which for experts is fast and allows scripting. Identification of the equivalent distinction between surfaces and traditional GUIs may lead to more insight.

*Liminal Development Processes Suit New Interaction.* Software development involves great care and precision, but not all the steps cannot be determined precisely. In particular, both requirements analysis and testing involve a kind of exploration of possibilities. They are liminal in that they are on the edge of the development process. The need for exploration in these processes means they have a special connection with surface computing, where the interaction can be strongly supportive.

*New Interaction Means New Development Tools.* Surface applications can support many domains, as the work in Theme 1 shows. However, software development is itself a domain, and surface tools can therefore support the domain. We showed this in building several novel surface computing tools that support the development process itself. And not only do these help development work, but our experience suggests another advantage. When developers themselves use surface computing to do their work, they will better appreciate when where and how it works well. This understanding will bring designers and developers closer together.

*Collaborative Development is Exploratory.* Collaboration is a core principle identified in the "Agile Manifesto". Another idea in many agile processes is self-managing teams. It turns out that collaboration in analysis domains is typically exploratory. This is, when working together we try out ideas on one-another, give feedback, and refine. This is the main idea of "intersubjectivity" in the study of human communication. What this means is that surface computing may be especially relevant to self-managing agile teams, where surface computing tools will support the process well.



## Understanding Sketching Practices for Surface Interface Design

*Jagoda Walny, Samuel Huron, and Sheelagh Carpendale*

### Introduction

Interactive surfaces are a natural fit for applications that support and integrate with everyday visual thinking processes, including brainstorming, conceptualizing, and collaborative work. The high degree of freedom of input, including pen- and multitouch input, sparks the possibility of more flexible interfaces that approach the richness of the interactions we already have with non-digital objects. In particular, it is now possible to create interfaces that support freeform visual thinking similar to the kind of visual thinking people do physically on whiteboards or in notebooks. However, with this richness come an overwhelming number of options for interface design. Gaining a better understanding of existing, analog versions of visual thinking practices can guide interaction design for such creative, knowledge work contexts.

Sketching is a common way of externalizing internal thoughts for a variety of purposes, which, Kirsh (2010) explains, variously optimize thinking tasks, for example through offloading memory, enabling the solution of problems that the mind is unable to simulate, completing tasks more efficiently, and working with others. Tversky (2008) states that sketches can reveal what a person is thinking. Arnheim (1980) argues that visual perception and cognition are intrinsically intertwined. Visual thinking frequently occurs on analog whiteboards, as noted by Mynatt (1999): “All manner of incomplete and seemingly vague content was written as participants used their whiteboard as a scratch surface while pondering concepts much larger than their surface representations”.

Interest in computational support for sketching dates back to Sutherland’s Sketchpad (Sutherland & Sketchpad, 1963) and remains an active area both in research (including the work of Haller et al. on Anoto pen interfaces, e.g. (Brandl, Haller, Oberngruber, & Schafleitner, 2008), and work into sketch-based interfaces such as QuickDraw (Cheema, Gulwani, & LaViola, 2012)

and SketchStory (Lee, Kazi, & Smith, 2013)) and commercially, with recent products that augment predominantly touch interfaces with styluses meant for sketching, as in the case of the Microsoft Surface or the Apple iPad Pro. There have also been several research projects for creating full electronic whiteboard systems, including Flatland (Mynatt, Igarashi, Edwards, & LaMarca, 1999), Tivoli (Moran & Van Melle, 2000), ReBoard (Branham, Golovchinsky, Carter, & Biehl, 2010), Range (Ju, Lee, & Klemmer, 2008), and whiteboard systems for designers (Mangano, LaToza, Petre, & van der Hoek, 2014). These are powerfully enhanced electronic versions of existing analog media. In contrast, our aim is to create software environments — particularly information visualizations — that do not necessarily emulate existing analog environments, but that support visual thinking needs for thinking about digital artifacts such as data.

To better understand these visual thinking needs, working primarily with the goal of supporting everyday thinking processes in information visualizations, we have observed analog sketching-based thinking practices from several angles, including: examining the lifecycles of important sketches within software development projects; analysing the visual constructs left behind as residue on office whiteboards; and moving towards integration of sketching practices with data by studying people’s sketched representations of a small dataset. This has led to an expanded understanding of some of the facets of visual thinking practices that could be integral as we move to design surface interfaces that support visual thinking with data.

### **Understanding the Lifecycles of Sketches and Diagrams**

To better understand the role of informal sketches and diagrams in everyday workflows, we (Walny, Haber, Dörk, Sillito, & Carpendale, 2011b) ran a qualitative, interview-based study of reiterated sketches; that is, sketches that were created, re-created, and used at various stages of a project. Cherubini et al. (Cherubini, Venolia, DeLine, & Ko, 2007) observed that software developers use reiterated sketches in their workflows. We asked software developers to describe in detail sketches that were important to a recent project, and from these we deduced a number of rich, varied sketch lifecycles that we viewed through the lens of transitions and social contexts. In this section, we summarize the results of this study; full results can be found in (Walny, Haber, Dörk, Sillito, & Carpendale, 2011b).

We interviewed eight academics who are actively involved in developing software. They were a rich source of information about unconstrained visual thinking because of the freedom they had in choosing when and how to sketch. Unlike architects, engineers or, designers, who are generally constrained to sketching physical artifacts, software developers tend to work — and think — with non-physical, abstract concepts such as data structures, algorithms, and user interfaces, all of which can be represented in a multitude of ways. As academics, our participants were not part of strict software development teams and therefore had relatively high freedom to design their own preferred workflows.

## Sketch Lifecycle Diagrams

Our interviews were semi-structured and began with the question, “tell us about a sketch or diagram that was important to you in a software development project.” Further questions focused on the context in which sketches were drawn, their characteristics, the roles of the sketches, the tools and techniques used to create the sketches, the reasons for each stage of creating or re-creating the sketch, and personal experiences during the creation of the sketches. We analyzed all of our transcribed video data using an open-coding approach (Strauss & Corbin, 1998) and visualized the results as individual sketch lifecycles, shown in Figure 1.

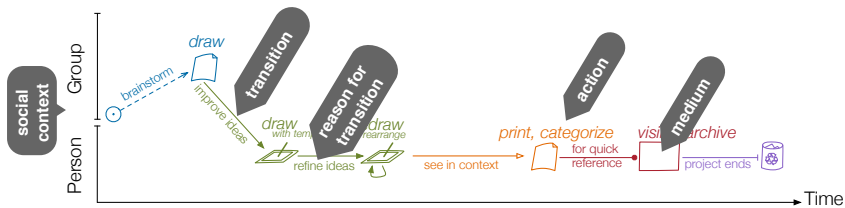


Figure 1. Lifecycle diagrams demonstrate the lifecycle of a single reiterated sketch. Each node represents an instance of the sketch; the icon represents the medium on which that sketch was drawn. Node labels describe how a sketch instance was created or used. Links between nodes represent a transition of each sketch — a re-creation of or edit to the previous sketch. Link labels explain why each transition occurred. The horizontal position of a node indicates the approximate time at which it was created. The vertical position represents the social context of the sketch instance: lower nodes were created in or for an individual setting, while higher nodes were created to share with a group. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011b).

Our key insight was in the use of transitions as a lens to look at the lifecycles. We identified the following transitions in our gathered sketch lifecycles:

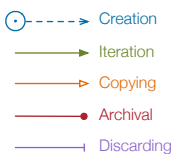


Figure 2. We identified five main transitions that a sketch would undergo: . ©IEEE. Reprinted, with permission, from (Walny, 2011b).

- Creation is the transition from idea to an expression of that idea in sketch form.
- Iteration refers to a change in the form of the sketch, whether through redrawing, annotating, or summarizing previous sketches.
- Copying transitions are direct reproductions of an existing sketch, e.g. scanning or photographing.
- Archival transitions denote a change in status of the sketch from active to inactive. Such a sketch is stored but not actively accessed.
- Discarding transitions denote a (usually) deliberate discarding of a sketch.

We also used social context as another lens to look at the lifecycles:

*Personal:* Sketches were often created and re-created for personal use.

*Group sharing (informal):* In some cases, sketches were created, augmented, or re-created while sharing ideas informally with a small group. This kind of sharing often involved annotations or small augmentations to existing sketches, some of which had been created in a personal setting.

*Group sharing (formal):* In other cases, sketches were used for presentation to larger groups (some of which had a stake in the decision making for some part of the project). These sketch instances were usually quite polished. Such sketches were rarely changed or polished after this formal sharing stage.

Lastly, we used symbols to encode the kinds of media on which each sketch instance was created, as shown in :

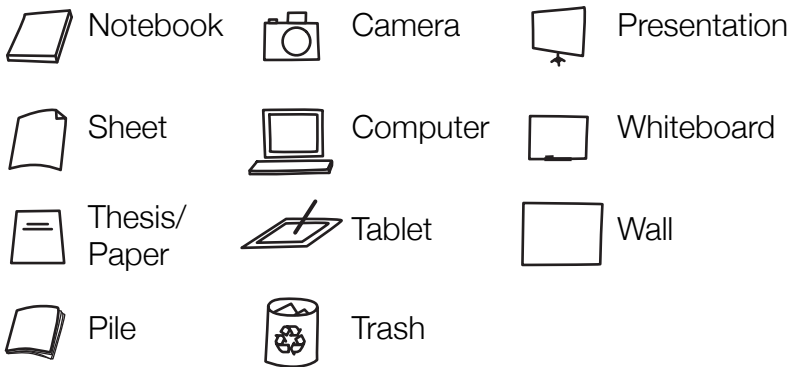


Figure 3. Types of media on which sketches were created. ©IEEE. Reprinted, with permission, from (Walny, 2011b).

### Lifecycle Variations

Our participants can be grouped into three general categories according to their level of investment in their sketching process. Participants with thoroughly considered workflows deliberately sketched as a part of their work process and put considerable effort into choosing the medium of their sketches and the way in which they shared and archived them. Participants with freeform workflows were heavy sketchers, but were more opportunistic in terms of how and when they sketched how they organized their sketches afterward. Participants with low sketching activity generally preferred other alternatives to sketching and therefore did not put much effort into managing their sketching workflow.

We describe here a selection of lifecycles from each type of participant; full lifecycles are described in (Walny, Haber, Dörk, Sillito, & Carpendale, 2011b).

## A Thoroughly Considered Lifecycle (P4)

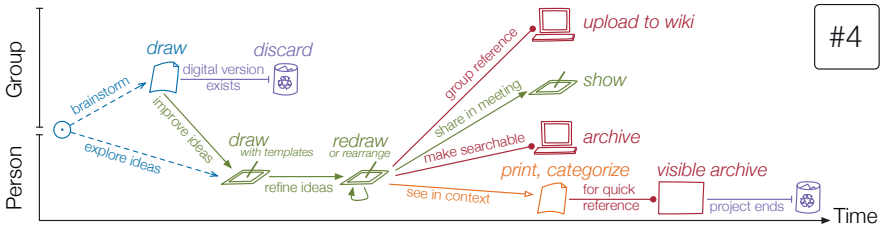


Figure 4. A sketch lifecycle from P4, who thoroughly considered his sketching workflow. ©IEEE. Reprinted, with permission, from (Walny, 2011b).

Let us consider one participant with a thoroughly considered workflow, whom we will call P4. This individual thought deeply about his workflow and was very partial to paper, but was actively trying to digitize his workflow so that he could take advantage of more streamlined transitions between iterations of sketches and the potential of template reuse. Let us walk through the lifecycle of a typical sketch that P4 would make, shown in Figure 4.

The first half of P4's sketch lifecycle reveals that he would draw sketches in meetings or on public transit, either on paper or on a pen-based tablet. Paper sketches were particularly used for brainstorming, but would be quickly redrawn on a tablet to improve the ideas, then discarded once a digital version existed. The pen-based tablet was used to create sketches that explored particular ideas; it contained pre-existing templates for rapid idea iteration. These sketches would be redrawn or rearranged several times in an iterative process of idea refinement.

The second part of P4's sketch lifecycle shows the result of the carefully planned workflow. The digital version of the sketch would be uploaded to a wiki for group reference; shown on the tablet to share in a meeting setting; and archived to make it searchable. In some cases, P4 felt he needed to see the sketches in context of his work, so he might print and categorize them, then pin them up in his workspace in a visible archive, to be explicitly discarded once the relevant project would end.

## A Freeform Lifecycle (P5)

Let us examine the lifecycle of a sketch from a participant, whom we call P5, with a freeform workflow (Figure 5). This individual was a graphics programmer and this particular sketch was used for debugging purposes.

P5 described to us a situation where she was debugging some graphics code using a debugging environment, but it was not helping. She grabbed the most readily available drawing surface — a piece of paper from a stack of loose scrap paper on her desk — and began to sketch out her problem. When this did not help, she decided she needed a change of environment and walked to a communal whiteboard in her workspace to redraw the same sketch. This helped her to discover a missing special case she had not

previously accounted for.

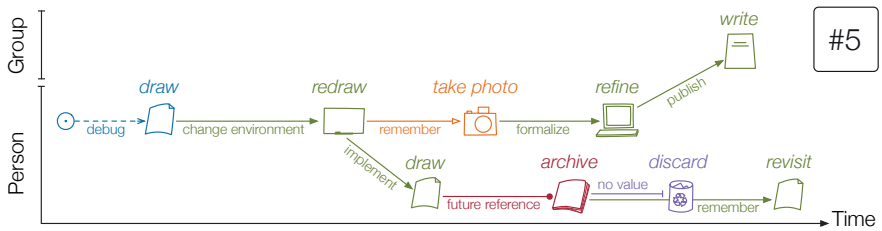


Figure 5. A sketch lifecycle from P5, who had a freeform sketching workflow. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011b).

In the bottom branch of the lifecycle, we see that P5 returned to her desk and created a sketch to plan out her implementation of the missing case. She kept this sketch on her desk to revisit it as a memory aid and told us she would discard it once it no longer held value as a memory aid.

Meanwhile, as can be seen in the top branch of the lifecycle, she also took a photo of the original whiteboard drawing so that she could remember it, re-create a more formal version of it digitally, and eventually publish it in a paper.

P5’s lifecycle illustrates that, for some people, working in different environments and on different media can be beneficial, and even crucial, to a thinking process. The sketches that led to solving of problems or working out implementations were clearly valuable to P5, as she saved them as both a visible reminder and a basis for later formal documentation.

### Low-Sketching Activity Lifecycle (P7)

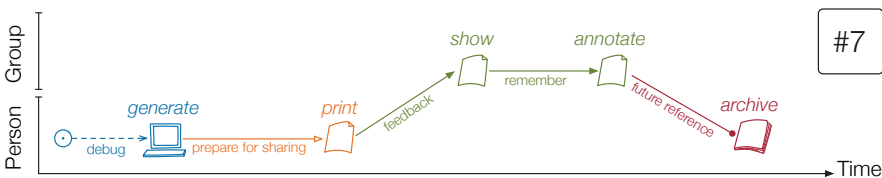


Figure 6. Sketch lifecycle from a participant with low sketching activity (P7). ©IEEE. Reprinted and modified, with permission, from (Walny, 2011b).

An example of a participant with low sketching activity was a researcher, P7, who was working on optimizing network protocol parameters (Figure 6). Part of his work included automatically generating performance graphs for debugging. In contrast to the other lifecycles we gathered, the creation step here was done digitally, and the generated graphs were printed so that they could be shared in meetings.

At these meetings, P7 would show the graphs to a supervisor, annotate them throughout the discussion, then store each group of graphs, stapled,

in a pile on his desks. Although P7 had the ability to re-generate the graphs at any point in time, he found himself referring back to the annotated graphs because the annotations from the meetings were so valuable.

P7's lifecycle illustrates the value of handwritten annotations even for some with primarily digital workflows.

### **Overview of Transitions**

We now summarize the characteristics of each transition among our participants.

The creation transition generally happened when a participant had an idea or problem and needed an external representation. This often happened in a personal context, for brainstorming, ideation, planning, or debugging. In several cases, creation was a rapid transition that used the most readily available medium. Participants with thoroughly considered workflows tended to ensure that their preferred medium was readily available.

The iteration transition was the most significant part of the lifecycles. Sketches were iterated upon for idea generation and refinement, to receive feedback, to preserve a record of meetings, or to beautify them for formal presentation.

The copying transition was used primarily to support sharing a sketch with others or to place the sketch in a place where it could act as a memory aid, whether a consistently visible one in the workspace or an archived one to be kept in case of future need.

The archiving transition was implicit for most of our participants; they tended to avoid deliberately discarding sketches unless they were leading to clutter, so sketches were archived by default. The clear exception here was when participants deliberately made searchable, accessible archives of their sketches for sharing with groups. Note that none of our participants talked much about retrieving sketches from an archive; this seemed to be a rare activity, but participants liked the security of having access to these sketches.

The discarding transition was usually deliberate and tended to happen most with analog sketches for reducing clutter. Sketches were discarded when they lost their value to a participant, either because they represented an idea that had since changed or because they were no longer needed as a memory aid.

### **Other Highlights**

*Beautification.* Our participants tended to talk about beautification only in the context of sharing, and particularly formal sharing with wider groups. Some participants were aware of the effect that "sketchiness" can have on the amount of feedback people are willing to give —more formal sketches



tend to look more finished, and therefore inspire less feedback, as noted by (Landay, 1996).

*Sketches as contracts.* One of our participants used sketches as contracts, that is, records of agreement. He and a colleague would agree on a task based on a sketch and upload it to a wiki. Later, if there was any dispute over what the task had been, it was easy to check it on the wiki.

*Summary sketches.* Two of our participants created what we call summary sketches (see Figure 7 for an example). These sketches marked an inflection point in a project. A series of previous sketches were used to explore an idea. Once the researcher had settled on a particular idea, they would draw a clean summary of the chosen idea that, in their minds, marked a decision point and replaced many of the previous sketches. In the case of our participants, these were the sketches that they chose as having been important to a recent project. Some of the preceding sketches were now deemed less valuable, and therefore more likely to be deliberately discarded.



Figure 7. Example of a summary sketch.

*Strong opinions about paper.* Some of our participants had strong opinions about paper. Participants with thoroughly considered workflows tended to be very fond of drawing on paper due to its easy accessibility and its particular qualities, but some of these participants found the advantages of digital media to be strong enough that they pursued fully digital or mixed digital and paper workflows. Participants with freeform workflows tended to be fond of paper. Participants with low sketching activity were vocal about their dislike of paper because of the clutter it created and its lack of searchability.

## Discussion

Among our participants, it was clear that sketches were highly used for ideation, communication, and distillation. While some ideation sketches were only transient, those that had rich lifecycles were clearly valued as part of participants' workflows, going through multiple transitions across media and contexts, serving as a record of thought, a trigger for new thoughts, and a memory aid, and there was a clear reluctance to discard them. In

addition, analog media were clearly dominant amongst our participants when it came to creating ideation sketches.

Sketch lifecycles were also shaped by participants' communication needs. Sketches were used to share ideas; redrawn to communicate and clarify work; and were used to record agreements about decisions made during work processes. For formal communication, sketches were transformed to different media and beautified, but original hand-drawn sketches were often saved for personal use.

Sketches were also used for distillation. Most transitions in the lifecycles that did not result in exact copies were for refining ideas, clarifying solutions to problems, and for discussing ideas with others.

### Summary

The study of sketch lifecycles contributes a fuller understanding of the richness and variation in the process of creating sketches for everyday thinking tasks and of the significance of transitions within this process. It is clear that for some people, these visuals are important cognitive tools throughout their workflows. It is also clear that it currently takes a great deal of setup effort to use purely digital media for this type of activity, particularly for heavy sketchers; and that this is not without its limitations, particularly where sketches are used as a memory aid within the workspace. Despite this, there are advantages to integrating visual thinking into digital workflows, judging by some of our participants' willingness to do so. This highlights the importance of considering the overall work context and people's various visual thinking and communication needs when designing digital tools.

### Understanding Sketched Visual Constructs on Whiteboards

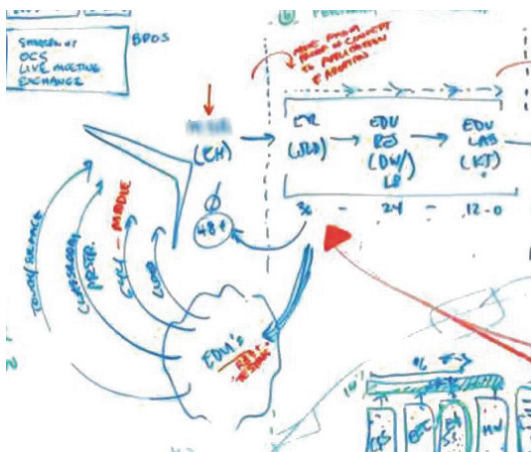


Figure 8. A whiteboard diagram for thinking; collected in our study. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011b).

A key facet of visual thinking encompasses the visual constructs that

are created and left as residue of visual thinking. There is a whole class of these visualizations created everyday all around us — spontaneous visualizations, or diagrams sketched to illustrate abstract concepts or data while thinking or collaborating. These are usually drawn on freeform media such as paper or whiteboards. Their residue can be studied to gain a better understanding of the kinds of ways in which information visualizations could support visual thinking. To this end, we gathered photographs of unaltered office whiteboards and studied the visual constructs on these whiteboards qualitatively, using information visualization as a lens through which to understand these spontaneous visualizations. The full results of this study, which was done in collaboration with Microsoft Research, are available in (Walny, Carpendale, Riche, Venolia, & Fawcett, 2011a).

### **Study Design & Analysis**

The focus of the study was to better understand the kinds of visual constructs people create and use in their daily routines. This includes better understanding of:

- The types of visuals that people create.
- Whether people develop their own visual representations.
- The techniques that people use to arrange their visuals.
- Whether any of these techniques parallel established information visualization techniques.

Our goal was to observe a breadth of constructs. We took photos of 82 analog whiteboards from 69 participants over the course of a few days (Please note: We are unable to provide high-quality images in this section. However, the key aspect of these images is their structure, not their textual content). We used a “moment-in-time” approach, wherein we visited offices unannounced and took photos of unaltered whiteboards.

Our participants were mostly researchers in a range of disciplines including computer graphics, mathematics, linguistics, and theoretical computer science. There were also a small number of software engineers, designers, managers, administrators, and technical support staff.

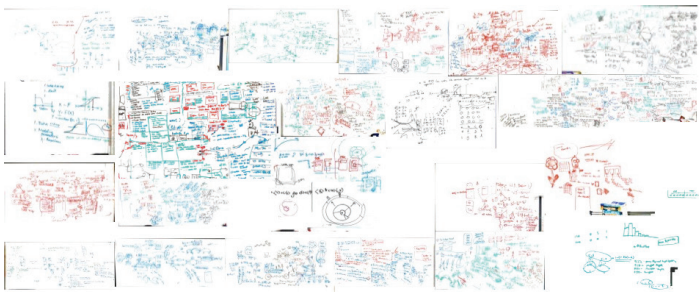


Figure 9. We observed a breadth of visual constructs.

We used an open coding approach (Strauss & Corbin, 1998) to analyze the

photos for information visualization constructs. All five of the paper authors (see (Walny, Carpendale, Riche, Venolia, & Fawcett, 2011a)) were involved in the coding process, and together we made two full passes on the data, counting the number of whiteboards that contained various constructs or factors. We then conducted 10 follow-up interviews with some of our participants to validate the accuracy of our coding and to learn about their whiteboard usage in more depth. We transcribed these interviews and used affinity diagramming to extract their major themes.

## Findings

We found a large variety of different diagrams drawn on these whiteboards, used as visual representations of problems, ideas, thoughts, and work processes. They were hugely inventive and clearly meaningful to participants. Our findings fall under three major themes: the relationship between words and diagrammatic constructs, use of recognizable information visualizations, and the use of whiteboards as a medium.

### Findings I: The Relationship between Words and Diagrammatic Constructs

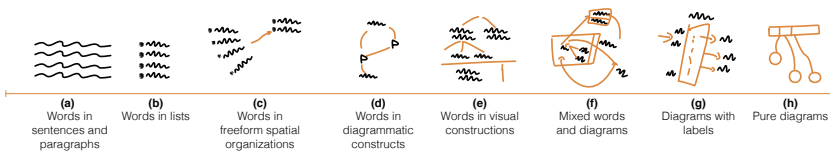


Figure 10. The words-to-diagrams spectrum. ©IEEE. Reproduced, with permission, from (Walny, 2011a).

Our open coding process led us to the observation that visual constructs contain both words and diagrammatic elements in a variety of relationships to each other. We place these on a spectrum that we call the words-to-diagrams spectrum, shown in Figure 10. At one end of the spectrum lie constructs with pure text; at the other, constructs with purely diagrammatic constructs. In between these two extremes lie various configurations of words and diagrammatic constructs, suggesting that words do not always play the role of labels or text. Rather, they can be essential parts of the structure of diagrams, and their own spatial configuration can have meaning.

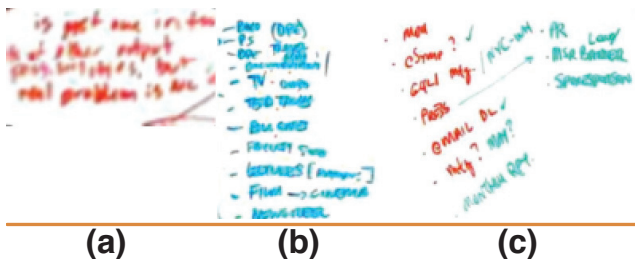


Figure 11. Examples from the word-dominant side of the words-to-diagrams spectrum. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011a).

Let us examine the words-to-diagrams spectrum in more detail. On the left of the spectrum, shown in Figure 11, are primarily word-based constructs. Pure text in sentences or paragraphs is shown in (a), followed by a structured spatial organization of words into lists (b). Next, words become organized in more freeform spatial organizations, often using techniques such as orientation or layering.

In the middle of the spectrum, shown in Figure 12, lie different varieties of mixed words and diagrams. In (d), words create a spatial structure and are linked by diagrammatic elements. Next, in (e), words are arranged in more familiar diagrammatic constructs, such as trees. And in (f), diagrammatic constructs are more prominent, but words are still structurally integral.

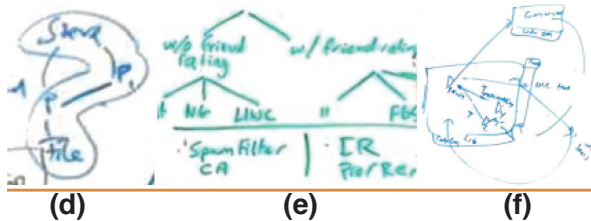


Figure 12. Examples from the middle section of the words-to-diagrams spectrum. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011a).

Lastly, on the far right of the spectrum, shown in Figure 13, are diagrams where words play a labeling role rather than a structural role, or are not present at all. Most of these diagrams that we collected were sketches of data charts or geometric sketches.

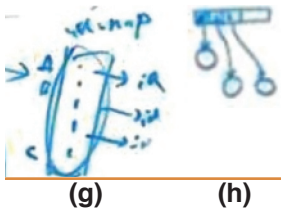


Figure 13. Examples from the diagram-dominant side of the words-to-diagrams spectrum. ©IEEE. Reprinted and modified, with permission, from (Walny, 2011a).

Along the majority of this spectrum, words are being used as primary objects, not as simple labels. It is clear that words play an important role in visual thinking. Some of our interviewees indicated that they saw a clear link between words and maturity of thought, and suggested that if a thought could be represented in words, it was more developed. Some claimed that drawing a diagram with words was easier than writing because it is easier to use spatial organization and diagrammatic constructs than to search for the correct wording. In addition, words can be used to represent abstract entities that don't have obvious visual representations. For instance, one interviewee stated:

*"I wanted to come up with my own theoretical framework about cognitive resources, so I was just like basically trying to relate them all. Because I couldn't do that [in a word processor]. I could create*

*the links, but I couldn't have kind of like a mental picture in my mind of how they relate."*

### Selected Implications

The observation of the words-to-diagrams spectrum suggests several areas of exploration that may be of interest to the information visualization community:

- Explore visualizations in which words are treated as primary to the representation. This could be useful for visualizing abstract concepts.
- Explore the idea of transitioning from visual data representations to words once an analyst has made a connection between data and concept. For example, a person might use such a tool to move from a data representation, to summarization in words, to manipulating the relationships between these words.

### Findings II: Use of Recognizable Information Visualization

Using information visualization as a lens onto our collected data, we also focused on finding recognizable information visualization factors and constructs in the collected visual thinking residue.

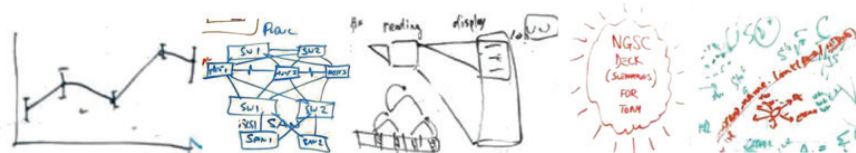


Figure 14. Collected examples of a data chart, a node-link drawing, a focus-plus-context technique, emphasis, and layering. ©IEEE. Reprinted, with permission, from (Walny, 2011a).

We found many examples of these, including data charts, node-link drawings, focus and context techniques, emphasis, and layering information (see Figure 14 for examples). For this chapter, discuss two in depth: data charts and layering. Further details about the remaining types can be found in (Walny, Carpendale, Riche, Venolia, & Fawcett, 2011a).

### Data Charts

Data charts such as line graphs, scatterplots, and bar charts appeared in 30% of our collected whiteboards.



Figure 15. Example of a line graph, a scatterplot, and bar charts collected during our study. ©IEEE. Reprinted, with permission, from (Walny, 2011a).

These charts rarely contained data with actual numbers, though there was evidence of underlying data, such as trend lines or data bars. Our interviewees suggested several possible reasons for this:

- They could not remember the data while drawing it.
- It would take too long to draw the data.
- The gist or trend was sufficient for the purpose of the conversation or brainstorming. As one participant put it: *"We didn't re-plot anything to draw this, we're just kind of saying, here's the pattern"*

This suggests that, in this case, the whiteboard was used either at the early stage of data analysis, or at a later, communicative stage.

Notably, one of our participants was somewhat disturbed by this lack of data, saying that, for his work, *"any small amount of data is going to be suggestive and possibly be completely misleading"*.

### Layering

There was ample evidence of an interplay between current and historical usage in the form of palimpsests (43%) — traces of old diagrams visible under new ones — and erasing (44%) (see Figure 16). This suggests that people are comfortable reading through layers of information, particularly if they have personally created these layers. It also suggests that people find value to preserving this historical information. During our interviews, participants confirmed that they use this layering as a way to manage temporal information. For example, one participant used different colours each time he started a new session of writing at the whiteboard. He said:

*"I choose different colors deliberately. If I use all the same colors, I don't know what my latest thinking was."*

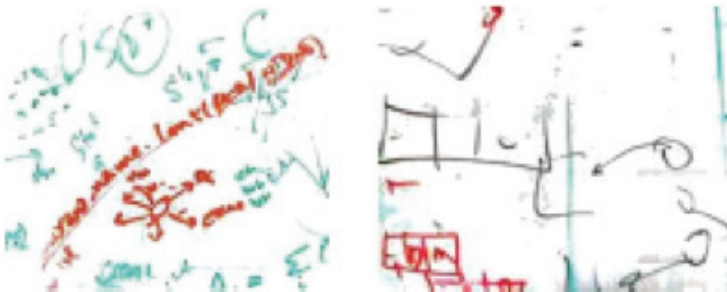


Figure 16. Examples of palimpsests (left) and erasing (right), demonstrating an interplay between current and historical whiteboard usage. ©IEEE. Reprinted, with permission, from (Walny, 2011a).

### Selected Implications

Whiteboards have some characteristics that would be powerful for data exploration: they make it easy to collaborate and to make rapid modifications to the display. However, they are tedious for performing real

data exploration because, of course, they are not connected to real data. Given that data charts do show up as a part of visual thinking on analog whiteboards, one research direction that emerges from these observations is to explore an augmented whiteboard design that supports access to data while still maintaining the thought-supporting freedoms of the whiteboard. One stream of research has already followed up on this by integrating access to data with a sketch-based interface: (Browne, Lee, Carpendale, Riche, & Sherwood, 2011) et al's pen-based interface, followed by (Walny, Lee, Johns, & Riche, 2012)'s wizard of oz study of a freeform pen-and-touch interface for data exploration, which culminated in Lee et al's SketchInsight (Lee, Smith, Riche, & Karlson, 2015) and SketchStory (Lee et al., 2013).

Another potential stream of research is to consider the usefulness of layering as a presentation technique. The concept of layering has appeared in other systems, e.g. Magic Lenses (Bier, Stone, Pier, Buxton, & DeRose, 1993), but in these cases, new layers hide existing layers to reduce clutter. However, since layers seem to serve as history and collaboration awareness tools, and since people seem to have a certain level of tolerance for the clutter of layers they have created on their own, it would be interesting to investigate how to enable our tools to help people build these multiple layers of information.

### Findings III: Whiteboards as a Medium

During the course of our analysis, several observations stood out as being characteristic to the medium of whiteboards. There is a considerable body of work that has expanded our understanding of the medium of whiteboards, including the work of (Mynatt, 1999), (Branham et al., 2010) , and (Tang, Lanir, Greenberg, & Fels, 2009). Our findings confirm and add to this body of work from an information visualization vantage point.

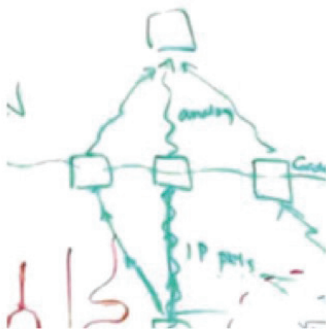


Figure 17. An example of deliberate sketchiness. ©IEEE. Reprinted, with permission, from (Walny, 2011a).

### Immediacy

Our interviewees made it clear that one of the defining characteristics of whiteboards was their immediacy — they could be used instantly, without interrupting a thought or a conversation. This immediacy was important enough that participants were willing to give up color choice, tidiness, real data, and precision in order to rapidly externalize their thoughts.



One interviewee remarked that explaining a concept to someone doesn't require precision. Such precision can be had with a mathematical graph drawing program, but it takes time (about 20 minutes, in his case) that he prefers not to spend in a meeting setting. This results in him avoiding that level of precision in typical meetings entirely. This indicates a potential missed opportunity for discussing precise details in collaborative settings.

### Sketchiness and Forgiveness

Two other key characteristics of whiteboards were sketchiness and forgiveness. While most marks made on the whiteboards were sketchy in nature, some were clearly deliberately so (see Figure 17). For example, several of our interviewees claimed that as their thoughts clarified, they tended to redraw their drawings in a neater, less sketchy fashion (see Figure 19).

Whiteboards also have a very forgiving nature. Our participants did not feel pressured to "get it right" the first time — they could easily be imprecise or even mistaken while using a whiteboard, whereas they felt much more pressured to be perfect while using software (see Figure 18).



Figure 18. Whiteboards allow mistakes.

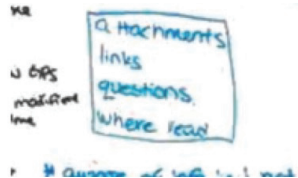


Figure 19. Sketchiness could indicate the maturity of a line of thought. ©IEEE. Reprinted, with permission, from (Walny, 2011a).

### Messiness

Nearly half of the whiteboards we photographed were densely packed with information. Some of these were arranged relatively neatly, using various separations and groupings to keep diagrams in order. However, many were very messy, particularly those used for communication rather than organizing one's own thoughts.

One interviewee commented that his whiteboard drawing may look "messy" to others, but he actually found that a digital copy of that same whiteboard was much messier to him:

*"[Diagramming software] for me is a lot messier than this board. For me to replicate this, software doesn't give me enough*

*constructs. Kind of messes up my thought process. You start using artificial shapes and places just to fit it in with the software. Over here [whiteboard], I can do a lot more thinking. And this is a lot less stressful."*

## **Selected Implications**

The whiteboard medium has several characteristics that are very unlike those found in software, characteristics that were clearly highly valued by some of our participants. It is important to consider the tradeoffs of introducing software-based solutions intended to replace whiteboards. For instance, while augmented whiteboards may be powerful tools, our data indicates that people may not be eager to use them in place of an analog whiteboard unless there is enough immediacy. Additionally, while much work on information presentation is focused on clutter reduction, our study indicates that some people are able to work with cluttered representations, particularly if they have constructed them by themselves over time. In fact, there is some indication that reducing clutter can inadvertently remove the ability to imbue a representation with implied meaning, for example about contextual and temporal information.

## **Summary**

Our qualitative study of spontaneous visualizations on office whiteboards has made it clear to us that people have the capability to create inventive, spur-of-the-moment visual representations of their problems to help themselves think. Moreover, examining these representations reveals that the ways in which people think visually do not always align with the ways in which visual constructs are made available in software. By decomposing the relationship of words to diagrammatic constructs, we have seen that words are often used as primary, rather than supporting, objects in these visuals. Viewing the visual constructs from the perspective of information visualization revealed that discussions of data are taking place at whiteboards, with immediacy being valued over data accuracy, even where more accurate tools were available. We have also seen that there are cases where there is value in clutter and sketchiness, in using a medium that allows for freedom to make mistakes, and in a medium through which a rich variety of visual constructs can be created to support thought. This understanding leads to a variety of open research questions regarding the ways in which software environments could support this kind of thinking freedom. This is particularly relevant for software on interactive surfaces, which are the natural digital equivalent to the ubiquitous whiteboards.

## **Sketching Representations of Data**

Having explored when and why visual thinking sketches are created as well as the characteristics of visual thinking constructs, we initiated explorations into how data representations would look when sketched by novices. We performed an exploratory observational study in which we asked people with varying amounts of experience in visualization to visually represent a small dataset using just paper and coloured pencils, and to tell us what they

learned about the data. This study broadened our perspective of the scope of possible data representations. Full results were presented at the EuroVis 2015 conference (Walny, Huron, & Carpendale, 2015).

Sketching data is a normal practice in our lab and in the classes we teach, as a way of both exploring new representations and of understanding the data we are working with. Our goal was to better understand the impact of sketching on data representation and understanding. Sketching has several benefits over digital tools for this kind of scenario: it has an extremely low barrier to entry, meaning that complete novices can do it, even if they do not think highly of their drawing abilities; it is quite rapid, meaning that it can be done within the timespan of a normal study; and it is completely freeform, meaning there are no restrictions on the representation beyond the dimensions of the page. This last characteristic is very important, because the lack of constraints meant we could minimize — though not eliminate — imposing pre-existing notions of how a representation “should” look.

### **Study Description**

Our study setup was quite straightforward. We ran three one-hour-long sessions with 7, 8, and 7 participants each, for a total of 22 unique participants. Participants were given coloured pencils, a printout of the dataset, and as many pieces of blank, letter-size paper as they wanted. We then asked them to “represent the data on the blank sheets of paper” in “any way [they] wished”. To minimize biasing our participants or suggesting representation types, we carefully scripted the introduction to each session and pre-planned answers to anticipated questions. After participants were done sketching, or after 45 minutes, we administered a simple demographic questionnaire that also included the following question with approximately half a page of room to answer: *“Please describe what you learned or found interesting about this data during the session. (There are no wrong answers)”*.

The dataset that participants sketched is a set of behavior-situation appropriateness scores, shown in Figure 20. We were very careful in choosing this dataset, as we wanted it to be very accessible but still interesting to a wide range of people. This is an engaging dataset, with combinations such as sleeping at a job interview, which was rated as quite inappropriate at 0.75 or eating on a date, which rated quite highly overall at 7.79. The key advantage of this dataset is that we can consider all of our participants to be “experts” in this data — not from a social psychology perspective, but from a human perspective.

We encourage readers to try sketching a portion of this dataset themselves. All experimental materials for the study, including the entire dataset, are available on our supplementary material website at <http://innovis.cpsc.ualgary.ca/supplemental/Data-Sketching/>.

Mean Appropriateness Ratings for 225 Behavior-Situation Combinations

Situation	Behavior														
	Run	Talk	Kiss	Write	Eat	Sleep	Mumble	Read	Fight	Belch	Argue	Jump	Cry	Laugh	Shout
Class	2.52	6.21	2.10	8.17	4.23	3.60	3.62	7.27	1.21	1.77	5.33	1.79	2.21	6.23	1.94
Date	5.00	8.56	8.73	3.62	7.79	3.77	3.12	2.88	3.58	2.23	4.50	4.42	3.04	8.00	3.79
Bus	1.44	8.08	4.27	4.87	5.48	7.04	5.17	7.17	1.52	2.15	4.17	3.12	3.08	7.10	3.00
Family dinner	2.56	8.52	4.92	2.58	8.44	2.29	2.54	3.96	1.67	2.50	3.25	2.29	3.21	7.13	1.96
Park	7.94	8.42	7.71	7.00	8.13	5.63	5.40	7.77	3.06	5.00	5.06	7.42	5.21	8.10	6.92
Church	1.38	3.29	2.38	2.85	1.38	1.77	3.52	3.58	0.62	1.42	1.92	1.71	3.13	2.60	1.33
Job interview	1.94	8.46	1.08	4.85	1.73	0.75	1.31	2.48	1.04	1.21	1.83	1.48	1.37	5.88	1.65
Sidewalk	5.58	8.19	4.75	3.38	4.83	1.46	4.96	4.81	1.46	2.81	4.08	3.54	3.71	7.40	4.88
Movies	2.46	4.98	6.21	2.73	7.48	4.08	4.13	1.73	1.37	2.58	1.71	2.31	7.15	7.94	2.42
Bar	1.96	8.25	5.17	5.38	7.67	2.90	6.21	4.71	1.90	5.04	4.31	3.75	3.44	8.23	4.13
Elevator	1.63	7.40	4.79	3.04	5.10	1.31	5.12	4.48	1.58	2.54	2.58	2.12	3.48	6.77	1.73
Restroom	2.83	7.25	2.81	3.46	2.35	2.83	5.04	4.75	1.77	5.12	3.48	3.65	4.79	5.90	3.52
Own room	6.15	8.58	8.52	8.29	7.94	8.85	7.67	8.58	4.25	6.81	7.52	6.73	8.00	8.17	6.44
Dorm lounge	4.40	7.88	6.54	7.73	7.19	6.08	5.50	8.56	2.40	4.00	4.88	4.58	3.88	7.75	3.60
Football game	4.12	8.08	5.08	4.56	8.04	2.98	5.23	3.69	2.04	3.85	4.98	7.12	4.31	7.90	7.94

Note:  
 0 = "The behavior is extremely inappropriate in this situation."  
 9 = "The behavior is extremely appropriate in this situation."

Figure 20. We used on a dataset from a 1974 social psychology study in which researchers asked people to rate the appropriateness of particular behaviours - such as running, talking, or eating — in various situations — such as at a job interview, in your own room, in the park, or in church (Price and Bouffard, 1974). The dataset shows mean results from the study.

### Results: Numeracy to Abstractness

Regardless of sketching ability or visualization expertise, all of our participants were able to produce at least one sketch. We collected 35 representations in total from the 22 participants, as well as their datasets and their questionnaires. Our aim was to broaden our perspective on the space of visual representations of data, so our analysis approach was designed to help us view the representations we received in a new way. To analyze the sketched representations, we used a combination of careful examination, affinity diagramming, several coding passes, and finally working together until agreement was reached.

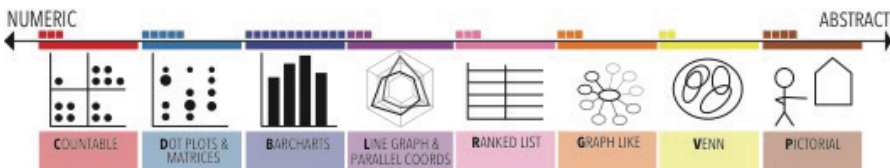


Figure 21. Types of sketched representations that we collected, arranged on a continuum from numeracy to abstractness. Square tokens indicate number of collected representations of each type. © 2015 The Author(s), Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Reprinted, with permission, from (Walny, 2015).

After this process, the strongest characteristic that stood out for us in looking at the representations was the variations in how numeric they were. We called a representation numeric if it represented directly the raw data values in the dataset. We called a representation abstract if it represented some level of abstraction of the original raw data. Representations are binned roughly by type as shown in Figure 21; tokens above each bin represent

number of representations we collected of each type. See Figures 22 – 27 for examples of selected representations.

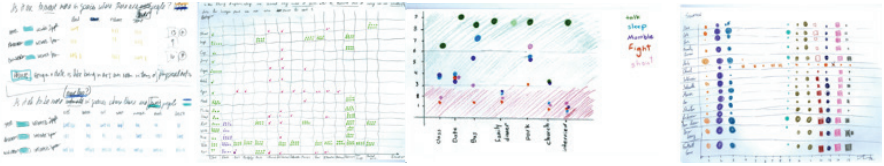


Figure 22. Countable representations: a tally and countable dots; A dot plot, in which both the position and size of the points encodes the numerical value and the colour of the background bins the values into high, medium and low appropriateness; A matrix that is a fairly straightforward mapping of the original raw data to size of the shape. (3rd image from left © 2015 The Author(s), Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Reprinted, with permission, from (Walny, 2015)).

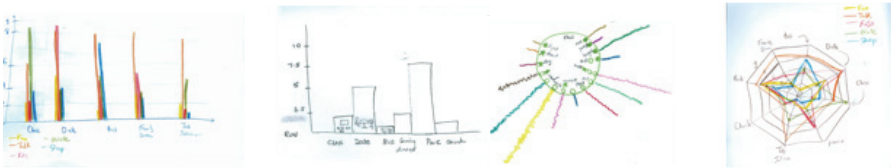


Figure 23. Examples of bar charts and line charts, including radial charts. These are still quite numeric, but can have various groupings or aggregations applied to the raw values.

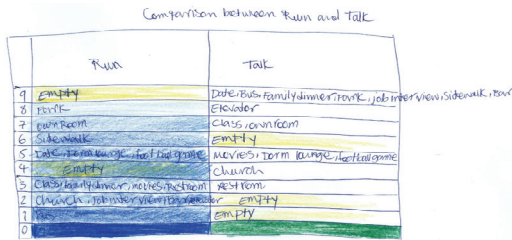


Figure 24. Ranked lists abstract numeric data into an ordering.

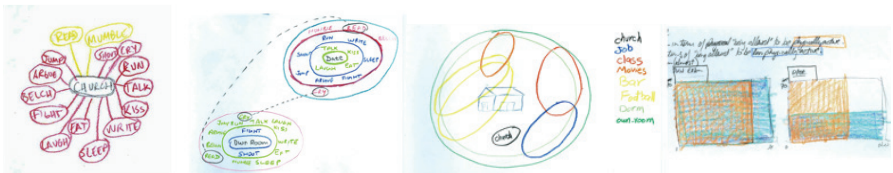


Figure 25. These graphs, which bin and link values in various ways, demonstrate representations on the abstract side of the continuum; A Venn diagram relates different situations to each other; In the hybrid Venn diagram / bar chart, external information is included. The orange area represents physically active behaviours and the blue represents less physically active behaviours. (Top images © 2015 The Author(s), Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Reprinted, with permission, from (Walny, 2015)).

At the most abstract end of the spectrum, we have pictorial representations, shown in Figures 26 and 27. It is easy to disregard these at first glance because they prominently feature cartoons and very little actual data. However, these proved to be more interesting than they seemed, as demonstrated in the later integration of results.

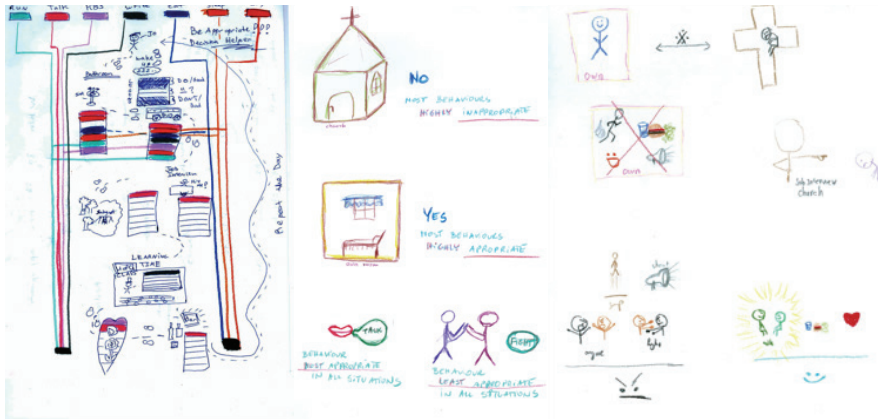


Figure 26. Examples of pictorial representations. A pictorial decision support tool depicts a “day in the life” of a person and uses the data to guide decision making on appropriate behaviours in particular situations; A set of cartoons summarizes the dataset with “YES” and “NO” statements; Another set of cartoons summarizes several situations and behaviours. (Left image © 2015 The Author(s), Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Reprinted, with permission, from (Walny, 2015)).

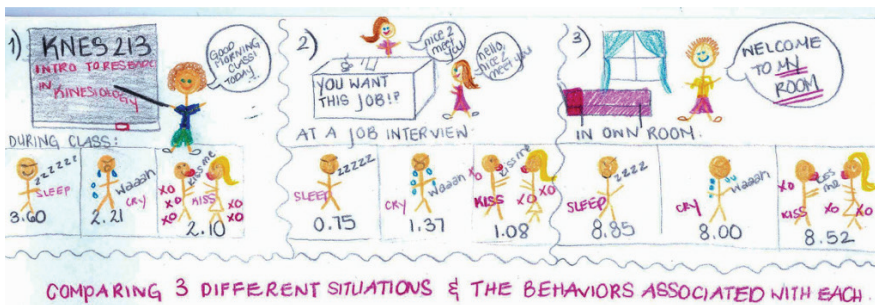


Figure 27. A set of pictorial vignettes showing an overview of selected situations and behaviours.

## Results: Spectrum of Data Reports

Independently of the data sketches, we analyzed the questionnaire responses to the open question: “Please describe what you learned or found interesting about this data during the session (there are no wrong answers)”. We divided each response into individual statements, then used open coding to analyse them. This led us to a classification we call the “spectrum of data reports”, which we label from A – F and describe as follows:

### **A – C: Statements communicating information intrinsic to the dataset.**

A. On the left side of the spectrum we have statements about individual data values, such as, “fighting in church is inappropriate”.

B. Next, there are statements that summarize entire rows or columns, such as, “there aren’t many behaviours appropriate in church”.

C. These statements contain comparisons between two rows or columns, such as “Date and own room have the similar rating for ‘kiss’. Other ratings in these two situations are close to each other.”

### **D: Statements containing dataset-level trends and comparisons.**

Statements in this category:

- Compare three or more rows or columns,
- Group items by value, or
- Make global comparisons.

For instance: “Several situations which have a similar rate for one specific behavior tend to be similar for other behaviors”; and, referring to values as: “completely appropriate”, “somewhat appropriate”, “highly inappropriate”.

### **E: Including Extrinsic Information**

Some statements included information extrinsic to the dataset. For example, some people would classify values using external concepts such as “comfortable”, “safe”, or “aggressive”. Others compared the values to their own expectations, for example, “mumbling + talking diverged more than expected.” And others would explain the values within the domain context, for instance, “people care a lot in job interviews.”

### **F: Analytic Potential**

Lastly, there were some statements that indicated hypotheses or conjectures about the reasons behind the values in the dataset. Because this was a half-page open question, we were not expecting a full analysis. However, we still received a few statements that indicated analytic potential. For instance, one participant thought that the park and your own room might have similar values due to their relative anonymity: *“it appears the park might be the same as one’s own room... anonymity?”*.

Another participant hypothesized that there were more women than men in the original study because the acceptability of talking in bathrooms was, in his opinion, rated quite highly: *“I found out that there seem to be more women in the dataset than men because most inappropriate behaviours to men (i.e. Talking in the restroom) is still above 5.”*

### **Integration of Results**

We integrate the two independent analyses together to see a more holistic perspective on each participant. We summarized this in a graphic shown in Figure 29. In this graphic, every row represents all of the artifacts

returned by a single participant. Consider the single row shown in Figure 28, which shows the participant's reported visualization experience level, the types of statements the participant reported, and the classification of the two sketches this participant created (both are classified G for graph; classifications match the highlighted letters in Figure 21).

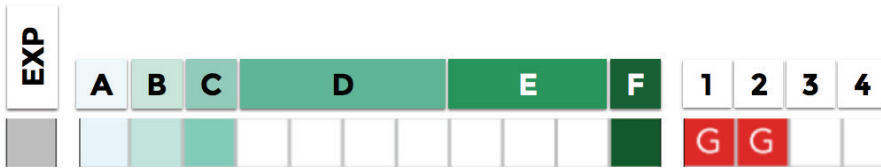


Figure 28. Integration of results for one participant. The first column shows their self-reported visualization experience rating, which is medium. (Darker colours indicate more experience.) The green columns show the kinds of statements this participant reported in their questionnaire, corresponding to the data report spectrum. This participant had statements in the A, B, and C categories, which were statements about raw data values and comparisons between values — and also statements in the F category, with analytic potential. The red columns show the kinds of sketches this participant returned. This participant returned two sketches, both graphs (as denoted by 'G'; see highlighted letters in Figure 27 for a legend). The saturation of the red colour indicates that these were positioned closer to the abstract side of the spectrum.

The full integrated results for all participants are shown in Figure 29, which orders participants based on type of representation returned, from most abstract to most numeric (where multiple types of representations were submitted, the ordering uses the most abstract representation). The first column shows that there was a range of visualization experience among participants. In the green data report columns, it can be seen that most participants returned some statements in the left-most columns — A, B, and C — and much fewer returned statements referencing extrinsic information (E) or statements with analytic potential (F). The rightmost red columns show that most people returned only one sketch, although a few created several sketches.

In Figure 29, we condense the sketch columns to get a single column with the most abstract sketch from each person. We have ordered the whole graphic top to bottom from most abstract to most numeric.

If you look at the top part of this matrix, you can see that where we have the highest concentration of abstract sketches — particularly the pictorial ones — we also have a concentration of the statements about extrinsic information and statements with analytic potential. This means that the participants who created the cartoon storytelling images seen in Figures 26 - 27 — images least similar to standard visualizations — actually demonstrated deeper levels of thought about the data and how it fits within the context of the world.



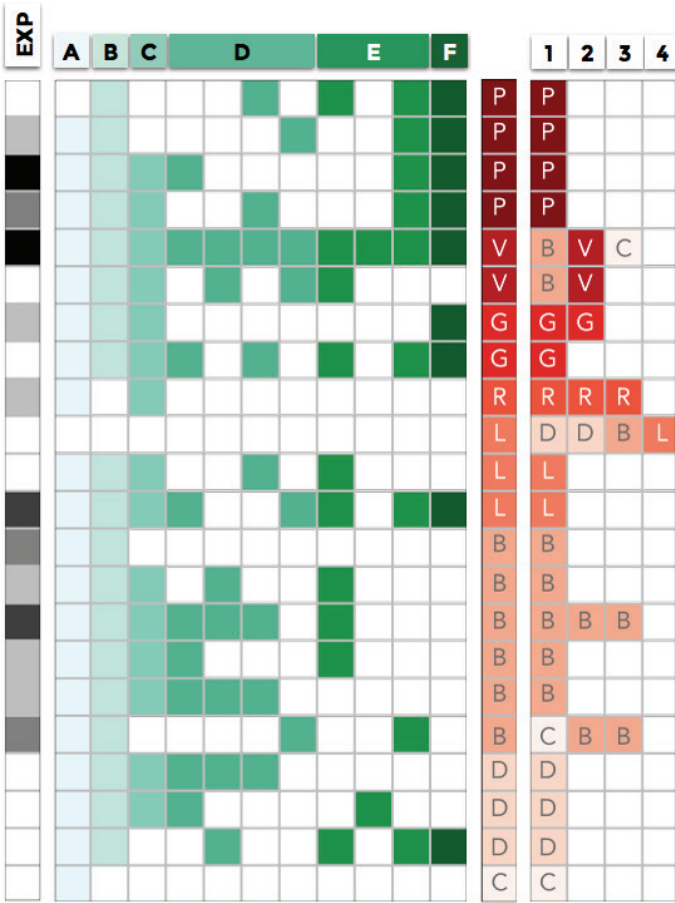


Figure 29. Full integrated results for all participants, ordered by representation type, from most abstract to most numeric. See Figure 28 for explanation of how to read each row.

In Figure 29, we condense the sketch columns to get a single column with the most abstract sketch from each person. We have ordered the whole graphic top to bottom from most abstract to most numeric.

If you look at the top part of this matrix, you can see that where we have the highest concentration of abstract sketches — particularly the pictorial ones — we also have a concentration of the statements about extrinsic information and statements with analytic potential. This means that the participants who created the cartoon storytelling images seen in Figures 26 - 27 — images least similar to standard visualizations — actually demonstrated deeper levels of thought about the data and how it fits within the context of the world.

Of course, we cannot make any conclusive claims from this observation, but we include it here because it inspired us to see the abstract representations

in a completely new light. We cannot know if some participants were more creative than others, affecting both their sketches and their thinking about the dataset; if they took steps to analyze the data before they represented it; or if the act of sketching happened to have an influence on their thinking. However, because the end goal of the information visualization community is to understand data and not just represent it, and because the representations associated with deeper thought about the data were so different from standard visualizations, it is worth studying this association further.

### Levels of Data Description

The association found in the integration of results leads us to a more formal way of describing the information content of a representation. We call these the levels of data description, and they can describe representations of varying numeracy or abstractness. This was inspired by Bertin’s elementary, intermediate, and overall levels of information (Bertin, 1981), but is based on our more complex dataset and the results of our data report spectrum.

What follows is an intuitive explanation of the levels of data description. A formal definition can be found in the full paper (Walny et al., 2015).

We begin with a dataset (represented in Figure 30 by the empty table), the world external to the dataset (represented by the globe icon), and the data report spectrum (represented by the green bar with letters A – F).

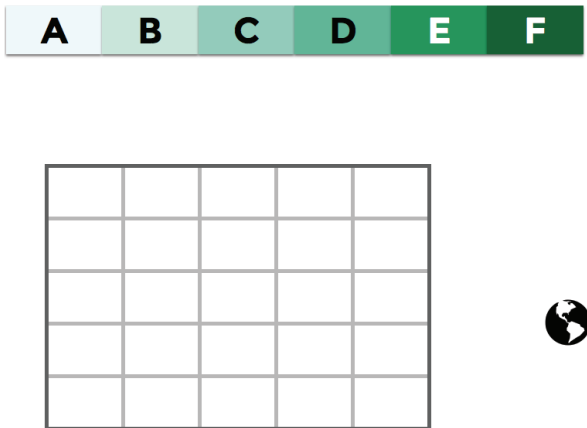


Figure 30. Basic components underlying the levels of data description: the data report spectrum (top), a dataset, and the external world.

### Value Level of Data Description

Representations that describe the value level of a dataset communicate the individual raw values, pairwise comparisons, and trends of 3 or more individual values from the original dataset. Generally, such representations would fall at the numeric end of the numeracy-to-abstractness continuum. For example, the countable matrix shown in Figure 31 shows only the raw values.

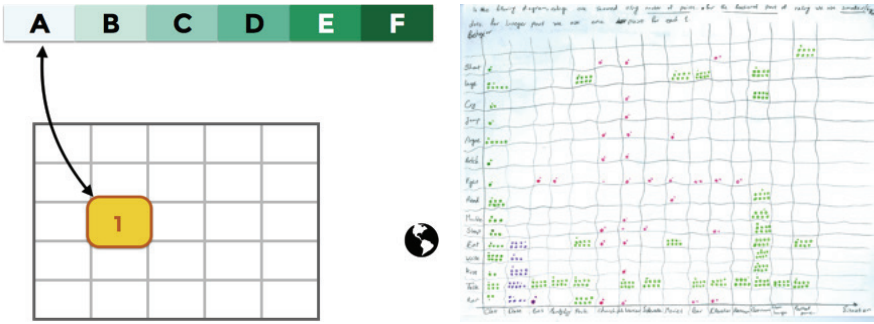


Figure 31. Value level of data description (left) with an example from our collected sketches: a countable matrix.

**Dimension Level of Data Description**

Representations that show the dimension level of a dataset communicate summary descriptions of individual dimensions or groups of dimensions, pairs of individual dimensions or groups of dimensions, and dimensional trends. The example shown in Figure 32 summarizes the average appropriateness values for each situation. On the left, "own room" has, overall, pretty high appropriateness ratings overall and, on the right, "church" has pretty low appropriateness ratings overall.

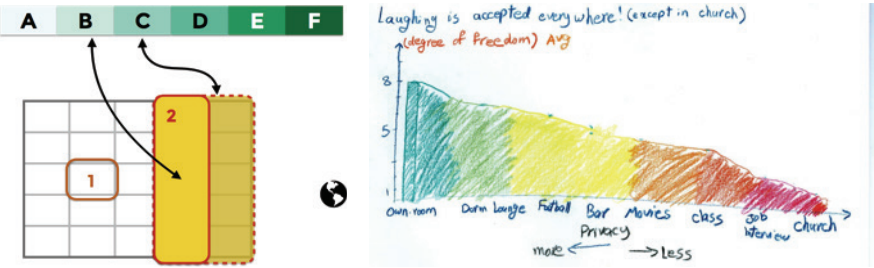


Figure 32. Dimension level of data description (left) with example from our collected sketches, which shows average values for each dimension.

Representations that show the dimension level of a dataset communicate summary descriptions of individual dimensions or groups of dimensions, pairs of individual dimensions or groups of dimensions, and dimensional trends. The example shown in Figure 32 summarizes the average appropriateness values for each situation. On the left, "own room" has, overall, pretty high appropriateness ratings overall and, on the right, "church" has pretty low appropriateness ratings overall.

**Global Level of Data Description**

Representations that describe the global level of a dataset meaning they give an overview of the shape of the entire dataset. It is difficult to provide such a global overview in 45 minutes of sketching, so we did not receive many examples of this kind of sketch. However, the example shown in Figure 33 comes quite close: it provides an approximate overview of the value distribution of each row and column in the dataset, all at once. Another

example might be a matrix representation of the data that is clustered by similarity, giving a feel for the overall shape of the data.

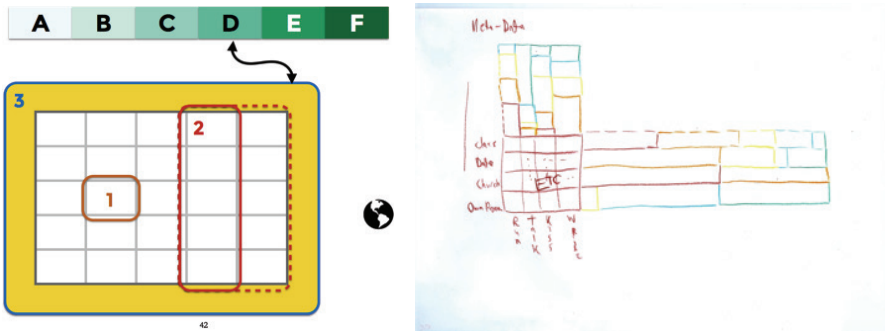


Figure 33. Global level of data description (left) with example providing an overview of the distribution of values of each row and column in the dataset.

### External Level of Data Description

Lastly, representations that convey the external information level of data description relate any of the previous levels to external concepts. For example, the representation in Figure 34 orders situations — on the left — by level of privacy, and it groups the bars —the behaviours — by type, i.e. active, emotional, usually frowned upon, or negative.

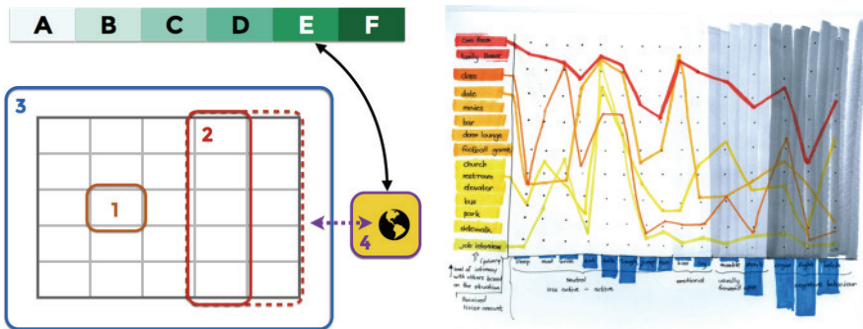


Figure 34. External level of data description (left) with example representation that classifies and orders data based on information not contained in the dataset. (Left image © 2015 The Author(s), Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Reprinted, with permission, from (Walny, 2015)).

### Discussion & Conclusion

From this exploration, some of the most interesting contributions are open questions for further study. Some highlights include:

1. It is worth investigating the potential usefulness of viewing representations in terms of their levels of data description. While a representation might not include a particular level, it might have

validity — and utility — at another level.

2. We also think it is worth investigating the sketching process itself in terms of data understanding. This was highlighted in particular by those participants who made abstract and pictorial sketches and also provided insightful data reports. Given the correspondence here with data understanding, this relationship merits further study, regardless of whether it has to do with process, pre-existing skill, or some other factors.

We have presented a continuum of representations, from numeric to abstract, a spectrum of data reports ranging from statements about individual data values to statements with analytic potential, and four levels of data description, from value-level through to external-level. What we have seen is that there is more to representation than direct transcription of data. Data has meaning within a context, and that meaning can be represented together with the data. This may even have some relationship to insight generation. Therefore, it is worth investigating the benefits of representing data at various levels of data description in both numeric and abstract ways.

## **Conclusion**

We have studied the lifecycles of sketches, the visual constructs in visual thinking, and how data is represented given the freedom of sketching. These studies have expanded our understanding of the valued characteristics of sketching for visual thinking, including: the large variety of workflows that sketches fit into; the structural importance of words; the layers of meaning that can be present within a freeform sketch, and the correspondingly restrictive nature of rigid software-based visual constructs; and the levels of information that a data sketch can contain beyond the precise raw values of a dataset. This expanded understanding will provide guidance for designing new surface interfaces for thinking visually with and about digital information.

## **Acknowledgements**

The co-authors of the full papers summarized here include Jonathan Haber, Marian Dörk, and Jonathan Sillito from the University of Calgary, and Nathalie Henry Riche, Gina Venolia, and Philip Fawcett from Microsoft Research. In addition to the generous support of SurfNet, this research was partially funded by NSERC, AITF, GRAND, and SMART Technologies.



## Designing Tabletop and Surface Applications Using Interactive Prototypes

*Tulio de Souza Alcantara and Frank Maurer*

### Introduction

*"We must design our technologies for the way people actually behave, not the way we would like them to behave" (Norman, 2007).*

Designing Windows, Icons, Menus and Pointers (WIMP) based applications is a well-known challenge. This challenge becomes even bigger for touch-based devices and gesture-based applications (Hesselmann, Boll, & Heuten, 2011), (Hinrichs & Carpendale, 2011), (Wobbrock, Morris, & Wilson, 2009), (Lao, Heng, Zhang, Ling, & Wang, 2009), (North et al., 2009) and (Morris, Wobbrock, & Wilson, 2010). The increasing popularity of multi-touch tabletops and surface computing opens up new possibilities for interaction paradigms, allowing designers to create applications that can be interacted with in new and different ways, through gesture-based and touch-based interactions that can improve or hamper the user experience (Norman, 2007), (Norman & Nielsen, 2010) and (Hesselmann et al., 2011). Interactive Tabletops and Surfaces (ITS) are highly visual systems, which are usually controlled by touches and touch gestures performed on the device, enabling users to directly interact with information using their hands or tangible objects.

For ITS applications, a preferable user interface integrates gesture-based interactions into the applications (Wobbrock et al., 2009). Frameworks such as Windows Presentation Foundation (WPF) ("Windows Presentation Foundation,"), provide a set of pre-defined gestures that application developers can use easily ("Microsoft Surface User Experience Guidelines,") and ("GestureWorks, a multitouch application framework for Adobe Flash and Flex.,"). However, the literature shows many examples of gestures that are not available 'out of the box' (Wobbrock et al., 2009) and (Khandkar & Maurer, 2010). When creating gestures for interacting with ITS applications, interaction designers have to determine if users consider them natural, understandable and easy to use (Wobbrock et al., 2009). Interactive prototypes can help answer this question.

## **Challenges in designing ITS applications**

In the context of ITS applications, designers can explore innovative ways for users to interact with their applications. Innovative interactions might drastically hamper the user experience if Human Computer Interaction (HCI) principles are not taken in consideration (Norman & Nielsen, 2010). What is necessary is a way to help designers follow HCI principles not only on the design of the interface of ITS applications, but also the interactions themselves.

Previous research on gesture-based interaction has shown problems with the design of gestures, the meaning of touch and gestures, and how context influences them (Hesselmann et al., 2011), (Hinrichs & Carpendale, 2011), (Wobbrock et al., 2009), (Lao et al., 2009), (Long Jr, Landay, & Rowe, 1999), (North et al., 2009) and (Morris et al., 2010). In the gesture design scenario, there are two main challenges:

- The effort, time and technical expertise required to create gestures (Lyons, Brashear, Westeyn, Kim, & Starner, 2007), (Kin, Hartmann, DeRose, & Agrawala, 2012) and (Plimmer, Blagojevic, Chang, Schmieder, & Zhen, 2012);
- The design of gestures that is suitable for specific tasks, context and users (Hinrichs & Carpendale, 2011) and (Long Jr et al., 1999).

Research on multi-touch applications shows a lack of processes and tools to support the design of these systems (Hesselmann et al., 2011), (Hinrichs & Carpendale, 2011) and (Wiethoff, Schneider, Rohs, Butz, & Greenberg, 2012). The authors of these studies bring up the need to allow designers to follow up on methods to improve the design of multi-touch applications, such as user-centered design (Morris et al., 2010).

Hesselmann and Boll propose Surface Computing for Interactive Visual Applications (SCIVA), a user-centered and iterative design approach addressing some challenges in designing ITS applications (Hesselmann et al., 2011). Their design process gives a general overview of the most important aspects in design of ITS applications. Studying ways to interact with tabletops, Hinrichs and Carpendale found that the choice and use of multi-touch gestures are influenced by the action and social context in which these gestures are performed, meaning that previous gestures and context influence the formation of subsequent gestures (Hinrichs & Carpendale, 2011). Also supporting the contextualization of interaction is Krippendorff (Krippendorff, 2006), highlighting that design is not only about making things but also about allowing users to make sense of things. Both studies suggest that to evaluate gestures, it is necessary to contextualize them in the scenario that they will be used.

North et al. (North et al., 2009) studies how users interact with objects in multi-touch surfaces and how designers can create intuitive and natural gestures. They start from the assumption that interacting with objects on a

multi-touch surface is an experience closer to manipulating physical objects on a table than using a desktop computer with keyboard and mouse. They study whether familiarity with other environments influences how users approach interaction with a multi-touch surface computer, as well as how efficiently users complete a simple task. They show that users who started with the physical model had a better performance when accomplishing the task on the surface, which supports their initial assumption, but they also suggest that more complex gestures, (e.g.: using two hands for a selection) might not work well on a surface tabletop. This means that there should be a balance between physical metaphors and supporting gestures to invoke commands.

Trying to understand users' preferences for surface gestures, Morris et al (Morris et al., 2010) compare two gesture sets for interactive surfaces: one created by end-user elicitation and one authored by three HCI researchers. The study used the feedback of 21 participants on 81 gestures. Their results showed three main findings:

- Their participants had similar gesture preference patterns;
- These preferences were towards physically and conceptually simple gestures;
- These simple gestures had been designed by larger sets of people, even though participants did not know how many authors created the gesture.

Their findings suggest that participatory design methodologies should be applied to gesture design, such as a user-centered gesture elicitation methodology.

Studying the inconveniences that can be generated by touch based interactions, Gerken et al. (Gerken, Jetter, Schmidt, & Reiterer, 2010) focuses on how users compensate for conflicts between non-interactivity and interactivity created by unintended touch interaction when using a multi-touch enabled tabletop. They conclude that touch-enabled devices can lead to "touch-phobia", reducing pointing and leading to less efficient and fluent communication. They suggested solution is to make touch smarter and more context-aware.

Norman and Nielsen (Norman & Nielsen, 2010) published a usability study that highlights concerns that should be addressed by designers when creating new touch-based interfaces and ways of interacting with them. The authors propose a balance between creative means of interacting while preserving basic HCI principles. However, guidelines for processes that can help designers follow a user centered design approach in the development of ITS applications are limited (Hesselmann et al., 2011). Hence, there needs to be an objective way to evaluate the usability of gesture-based applications in early stages of the design and having users involved in early stages of the design, helping designers follow a user-centered approach.



## **Prototyping for ITS applications**

Having users involved early in the process through iterative prototypes has been widely researched and the advantages of sketching and prototyping to improve the design of applications has been proven successful (Moggridge & Atkinson, 2007), (Rudd, Stern, & Isensee, 1996), (Sefelin, Tscheligi, & Giller, 2003), (Virzi, Sokolov, & Karis, 1996), (McCurdy, Connors, Pyrzak, Kanefsky, & Vera, 2006), (Lim, Stolterman, & Tenenberg, 2008) and (Derboven, De Roeck, Verstraete, Geerts, & De Grooff, 2010). Especially in the scenario of gesture and tangible based applications, where Norman and Nielsen (Norman & Nielsen, 2010) argued that gestures might be harmful for usability designers, there is a need to evaluate if the gestures are improving usability.

As shown by Moggridge (Moggridge & Atkinson, 2007), Krippendorff (Krippendorff, 2006) and Buxton (Buxton, 2010) sketching has shown to be a valuable aid to designers in order to validate ideas with users in early stages of the design. In any activity of design, sketching has been proven to be a crucial part of it and many contributions (Moggridge & Atkinson, 2007), (Wiethoff et al., 2012), (Rudd et al., 1996), (Sefelin et al., 2003), (Virzi et al., 1996), (McCurdy et al., 2006), (Lim et al., 2008), (Derboven et al., 2010), (Krippendorff, 2006), (Buxton, 2010), (Mommel, Gundelsweiler, & Reiterer, 2007), (Van den Bergh, Sahni, Haesen, Luyten, & Coninx, 2011), (Holzmann & Vogler, 2012), (Unger & Chandler, 2012) and (Obrenovic & Martens, 2011) defend the importance and the benefits of sketching and prototyping to improve design ideas by failing early, often and then, learning from mistakes. While these authors defend the use of paper as a medium to transmit ideas, sketching the dynamics of applications is not possible without tool support (Buxton, 2010).

Mommel et al. (Mommel et al., 2007) studied how prototyping can elicit requirements. They recommend the use of abstract prototypes, as filling in details too early might lead to premature decisions, leading to wasted effort and time spent on these details. Abstract prototypes help designers understand important aspects of the content and organization of the user interface while deferring details about how the final application will look like and operate (Constantine, 2004). These studies motivated our research by showing the importance of prototyping in a UCD process, which ultimately can help designers fit the workflow in agile iterations.

Paper prototypes allow designers to evaluate the output of a system, while the input is assumed obvious; they allow designers to evaluate what users want to do, while how users want to do certain tasks is not trivial (Derboven et al., 2010). If the interaction input is more complex, paper prototypes are not sufficient (Rudd et al., 1996).

Based on the limitation of designing ITS applications, our motivation is to make developing of usable gesture-based applications easier and better fitting user's needs. For this issue, a desired solution has to:

- Make it easy to design gestures, respecting the time and cost constraints of prototyping;
- Make it easy to evaluate if these gestures are usable.

By using prototypes, not only design ideas but also requirements of software can be brought to attention and properly addressed in early stages of the development. The advantage of prototyping is that it allows designers to experiment and invent (Robertson & Robertson, 2012). Interactive prototypes then help “interaction designers to define user interfaces, and evaluate usability issues in early stages of design” (Van den Bergh et al., 2011). While designing ITS applications, interactions play an important part in the design of interfaces and can drastically improve or hamper interfaces by having intuitive or non-intuitive gestures for interacting with them.

Having “interactive sketches” allows designers to take advantages of having users involved and providing feedback about the interactions in ITS applications. Memmel et al. (Mommel et al., 2007), propose the iterative use of low-fidelity prototypes in order to validate steps of design and development, resulting in a more iterative and agile process. Further in the design process, sketches can become more sophisticated and goal oriented, thus the time spent onto them changes, also changing the expectations regarding them. This distinction defines the sketches as prototypes (Buxton, 2010).

Rudd et al. (Rudd et al., 1996) suggest advantages and disadvantages of low-fidelity and high-fidelity prototyping, as shown in Table 1.

Type	Advantages	Disadvantages
Low-Fidelity	<ul style="list-style-type: none"> <li>• Less time &amp; lower cost;</li> <li>• Allows multiple design concepts;</li> <li>• Communication device;</li> <li>• Address screen layout issues;</li> </ul>	<ul style="list-style-type: none"> <li>• Limited usefulness for usability tests;</li> <li>• Navigational and flow limitations;</li> <li>• Facilitator-driven poor specification;</li> </ul>
High-Fidelity	<ul style="list-style-type: none"> <li>• Partial/complete functionality;</li> <li>• Interactive;</li> <li>• User-driven;</li> <li>• Clearer navigational scheme;</li> <li>• Use for exploration and test;</li> <li>• Marketing &amp; sales tool;</li> </ul>	<ul style="list-style-type: none"> <li>• Time-consuming creation;</li> <li>• Inefficient for proof-of-concept designs;</li> <li>• Blinds users to major representational flaws;</li> <li>• Management may think it is real;</li> </ul>

Table 1. Low- & High-Fidelity Prototyping, based on (Rudd et al., 1996), extracted from (Mommel et al., 2007).

The disadvantages regarding interactivity that motivated this research are drawbacks in low-fidelity prototyping that could be addressed and some of the features from high-fidelity prototyping that could be incorporated. For this, the tool .ve (de Souza Alcantara, Ferreira, & Maurer, 2013) was created,

a prototyping tool that incorporates interactivity on the level of high-fidelity prototypes allowing usability tests based on interaction but having the low effort cost of low-fidelity prototypes allowing the evaluation of multiple design and interaction concepts.

ProtoActive was designed to help designers of ITS applications, a prototyping tool that:

- Elicits user feedback through sketch-based prototypes that take into consideration the size constraints of an ITS application;
- Allows the evaluation of how users interact with the application by having prototypes that can be used through a pre-built set of gestures
- Supports the development of custom gestures through a tool that allows the creation of new gestures without requiring any programming effort.
- 

ProtoActive is a storyboard sketching based prototyping tool for multi-touch devices that integrates with a gesture-learning tool (IGT)(Alcantara, Denzinger, Ferreira, & Maurer, 2012) to evaluate custom gestures in prototypes. This chapter will explain the process of designing ProtoActive involving requirements gathered from related work and from a qualitative study with participants from industry. The following sections will also explain ProtoActive features and the workflow of a designer using ProtoActive to create interactive prototypes.

To gather requirements for a sketch-based prototyping tool, it was used:

- Existing research about computer-based prototyping tools and problems found in existing tool support for prototyping (Obrenovic & Martens, 2011), (Segura, Barbosa, & Simões, 2012), (Lin, 1999), (Bailey, Konstan, & Carlis, 2001), ("Balsamiq,"), ("Pencil: Add-on for Mozilla Firefox,"), ("iPlotz: Wireframes, mockups and prototyping for websites.,"), ("Axure RP: Interactive wireframe software and mockup tool.,"), ("Mockingbird: Wireframes on the fly.,"), ("Microsoft Sketchflow.,"), ("ForeUI: Easy to use UI prototyping tool.,"), ("Proto.io : Silly-fast mobile prototyping.,") and (Smith & Graham, 2010);
- A qualitative study that consisted of semi-structured interviews with five User Experience (UX) designers from different companies.

ProtoActive was created to address the drawbacks of existing prototyping tools and from a qualitative user study with five UX designers from industry. The following sections explain how ProtoActive addresses these drawbacks and the requirements gathered from the UX designers.

### **Improving the paper experience**

Sefelin et al. (Sefelin et al., 2003) compare paper prototyping with prototyping using software tools. Their study suggests three scenarios

where paper prototyping would be a preferable medium. ProtoActive addresses these scenarios as follows.

1. It allows expression of ideas and customization from the designers: ProtoActive allows designers to create free-hand sketches on a drawing canvas.
2. Require minimum expertise to use: In order to simulate the paper experience, ProtoActive has an intuitive and easy-to-learn interface that allows designers to create prototypes without requiring much time to learn the application. In order to do so, ProtoActive is a sketching tool with basic commands: free-hand sketching, sketch eraser, color picker, and strokes selection. Two features were added: adding and removing a background (which allows the designer to import existing images into their prototypes). Finally, to have a flow between the pages, designers can specify areas in the prototype page that when interacted with, will trigger a page movement set by the designer.
3. During evaluation, ProtoActive allows participants to easily sketch over the interface as a medium of feedback.
4. A designer is able to save multiple copies of a prototype, so during evaluations users can suggest modifications on the prototype itself. The tool is simple enough that making modifications to a page is as simple as sketching on a paper.

### **Help designers follow design guidelines for ITS applications**

SCIVA 3 is an iterative process for designing gesture-based interfaces for interactive surfaces. In order to help designers have a more systematic approach in the design of ITS applications, ProtoActive helps designers follow three steps of the SCIVA design process: defining the right visualization, conducting user studies to create gestures and evaluating the system with the user to detect flaws from previous steps. The following sections explain how ProtoActive addresses these steps.

*Defining the right visualization.* In ITS applications, there is a tight coupling between input (gestures and touch) and output (visualized objects on the screen). ProtoActive allows designers to create prototypes without constraining creative ideas by supporting the creation of free-hand sketch prototypes that will allow for any type of object on the screen. If designers decide to have a more accurate visualization, ProtoActive allows designers to import high-quality pictures into their prototypes. ProtoActive helps brainstorming and gathering feedback from users. It also helps in optimizing visualizations according to characteristics of ITS. The visualization optimization can be achieved by letting designers create and evaluate the prototypes in the ITS devices themselves. This allows for a realistic evaluation of distance, position and orientation of objects in the screen.

*Conducting user studies to create gestures.* For ITS applications, there

are sets of defined gestures offered ("GestureWorks, a multitouch application framework for Adobe Flash and Flex."), ("Microsoft Surface User Experience Guidelines,") that help designers define the input of ITS applications. However, these sets can be insufficient due to particularities of devices, location, context and orientation. In order to improve the user experience, it is necessary to evaluate the interaction of ITS applications with users. ProtoActive gives designer a pre-built set of gestures that can be extended by using a gesture recorder application (IGT). These gestures can be evaluated to interact in the context of the ITS application by being the input of prototypes in ProtoActive.

*Evaluate the system to detect flaws resulting from previous steps.* ProtoActive helps designers involve users in early stages of the design process by taking advantage of the low-fidelity prototype' features of the tool, by being easy and fast to use, and by having prototypes with a dirty look, thus eliciting more user feedback as affirmed by Buxton (Buxton, 2010).

A key drawback among the studied prototyping tools was the lack of gesture customization for users to interact with the prototypes during usability studies. Allowing designers to create custom gestures allows the creation of new ways to interact that might better suit for a certain task or group of users. ProtoActive provides a set of pre-built gestures that can be expanded using IGT (Alcantara et al., 2012), a tool that allows designers to provide samples of a gesture to create new gesture definitions that can be used to interact with the prototypes. This feature was gathered from drawbacks of the following tools: CrossWeaver (Sinha & Landay, 2003), Balsamiq Mockups ("Balsamiq,"), Pencil ("Pencil: Add-on for Mozilla Firefox,"), Fore UI ("ForeUI: Easy to use UI prototyping tool. ,") and Proto.io ("Proto.io : Silly-fast mobile prototyping. ,").

The prototyping tools that allow custom interactions also come with the cost of requiring a programming step for customization. This was seen as a drawback as it adds to the cost of prototyping (more time or even the involvement of software developers to create the customization). ProtoActive allows designers to fully create a prototype without requiring programming skills. Creating prototype pages, linking them through gestures, creating custom gestures and evaluating them can be accomplished in ProtoActive through its GUI. This feature was gathered from drawbacks of the following tools: Raptor (Smith & Graham, 2010), Sketchify (Obrenovic & Martens, 2011) and Microsoft Sketch Flow ("Microsoft Sketchflow,").

By allowing designers to sketch in a similar fashion as sketching on paper, ProtoActive allows designers to create interfaces that are not constrained by a pre-built set of controls. Among the tools studied, a constant problem was the lack of a feature that allows designers to free-hand sketch pages. Having pre-built UI widgets might increase the productivity and the speed of creating prototypes, but this comes at the cost of constraining creativity, especially for the design of ITS applications as that field is still evolving (and

so are the UI widgets used in these applications). ProtoActive is a sketch-based prototyping tool based on Buxton's principles (Buxton, 2010) about low-fidelity prototypes looking quick and dirty to encourage users to provide more feedback. Having a sketch-based prototyping tool was derived from drawbacks of UISKEI (Segura et al., 2012), SILK (Lin, 1999), DEMAIS (Bailey et al., 2001), Balsamiq Mockups ("Balsamiq,"), Pencil ("Pencil: Add-on for Mozilla Firefox,"), iPlotz ("iPlotz: Wireframes, mockups and prototyping for websites.,"), AxureRp ("Axure RP: Interactive wireframe software and mockup tool.,"), MockingBird ("Mockingbird: Wireframes on the fly.,"), Microsoft Sketch Flow ("Microsoft Sketchflow.,"), Fore UI ("ForeUI: Easy to use UI prototyping tool.,") and Proto.io ("Proto.io : Silly-fast mobile prototyping.,").

ProtoActive allows designers to create prototypes that focus on usability as well as interaction. In order to improve the sketching experience in ITS devices, ProtoActive drawing features consist of:

- A canvas area that can be drawn using fingers or stylus pen;
- An eraser functionality;
- A selection button to select strokes on the canvas to move, resize or remove;
- A color button to change the color of the stroke;
- An undo button;
- A gesture area button that allows designers to draw an area on the canvas to define a gesture area.

A gesture area is an area defined in a prototype's page can contain a list of gesture and prototype's page associations. When a designer defines a gesture area in a prototype, he can associate a gesture with this area by choosing from a list of pre-defined gestures or define a custom gesture. After selecting the gesture, the designer is asked to choose which page she wants the prototype to navigate to when the gesture is recognized during a user study. To remove selected strokes or gesture areas from the canvas, a designer drags the selected strokes or gesture area to the right side of the canvas (to the trashcan area), in a similar fashion as a designer would move or remove an object placed on top of a sheet of paper.

The following sections will explain ProtoActive in two aspects: as a tool and its features to design prototypes, and as a tool to evaluate prototypes.

## **ProtoActive**

ProtoActive keeps the sketching area to a maximum (which can be seen in Figure 1 as the empty white space). Sketching in ProtoActive can be conducted with free hand or with a stylus pen. No drag-and-drop or sketch recognition was implemented in ProtoActive and the only filter added to the sketching canvas is a "FitToCurve" feature that smooths out the stroke.

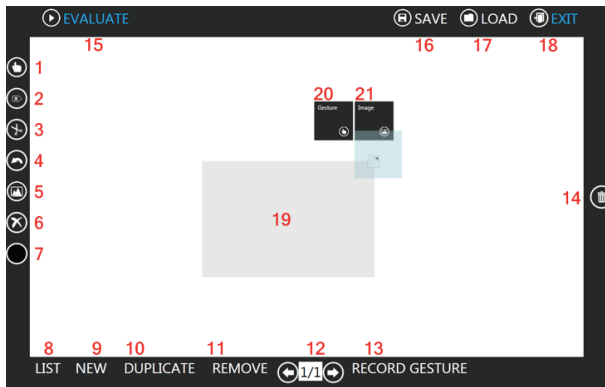


Figure 1.

During usability studies, prototypes in ProtoActive can be interacted with via gestures. The gesture area button (Figure 1, item 1) allows designers to define gesture areas by performing a lasso on the screen, the bounding area of the lasso will become a gesture area in the prototype.

To help designers simulate movement and zoom features in their applications, ProtoActive allows strokes to be selected, resized and moved on the canvas. By clicking on the scissors button (Figure 1, item 3) a designer can perform a lasso on the canvas to select all the strokes inside the lasso. With the selected strokes, a designer can move them on the canvas, remove them (by dragging them to the trashcan area) or resize them (by using zoom or pinch gesture).

ProtoActive allows designers to set the background of a page, by the two buttons: add background (Figure 1, item 5) and remove background (Figure 1, item 6). This was required in order to create prototypes that work with a static image as a background. Unger and Chandler show an example of the value of this feature in their book *UX Design* (Unger & Chandler, 2012). In the prototype chapter of the book, the authors explain how a designer can create prototypes in a What You See Is What You Get (WYSIWYG) tool such as Dreamweaver CS4 ("Dreamweaver CS4,") and export the prototype pages as separate images and set them as a clickable background in an HTML page. A similar solution can be achieved with ProtoActive by using the set background (Figure 1, item 5) feature to set the background of a prototype page with images from other applications. In ProtoActive, after setting the background of a page, a designer can still draw on the top of the image and add gesture areas, which allow setting specific parts of the background to respond to gestures and trigger page transition.

The new button (Figure 1, item 9) creates a new empty page, while the duplicate button (Figure 1, item 10) creates a duplicate of the current page in the prototype with the same drawings, and gesture areas of the original. The remove button (Figure 1, item 11) will remove the current page of the prototype. To facilitate the removal of gesture areas and strokes, the right

side of ProtoActive has a trashcan area (Figure 1, item 14) where gesture areas and selected strokes can be dropped and removed from the canvas. Finally, the exit button (Figure 1, item 18) closes the application.

The evaluate button (Figure 1, item 15) switches ProtoActive into evaluation mode. In evaluation mode, the canvas turns full screen, non-editable and the pages can be navigated using gestures.

ProtoActive has drawing tools to help users create prototypes: the eraser button (Figure 1, item 2) allows the designer to erase strokes using his finger, the undo button (Figure 1, item 4) allows designers to undo stroke mistakes. In order to allow the design of colored prototypes (allowing designers to highlight some areas with a specific color) the color button (Figure 1, item 7) allows the designer to select the color of the stroke on the canvas.

A navigation control allows users to navigate through the prototype's page. "List" (Figure 1 item 8) shows a list of thumbnails of all the pages in the prototype. The navigation buttons (Figure 1, item 12) changes the page to the previous (if there is any) or to the next page (if there is any), and the save (Figure 1, item 16) and load (Figure 1, item 17) buttons allow designers to export their designs to different devices running ProtoActive.

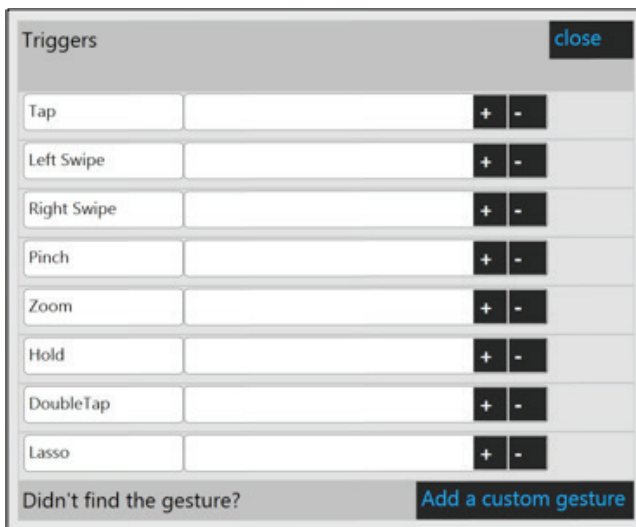


Figure 2. Gesture area trigger selection.

Finally, gesture areas (Figure 1, item 19) are movable and resizable areas on the page of a prototype that can be bound to one or multiple pairs of gestures and pages. From a gesture area, the designer can select the gesture menu (Figure 1, item 20), which will bring up the gestures triggers dialog (Figure 2) where a designer can use a pre-built set of common gestures and bind its detection to showing a specific page on the prototype chosen by the designer. The list of predefined gestures includes:



- Tap, a single tap with the finger on the surface;
- Double Tap, subsequent taps with the finger on the surface;
- Pinch, gesture using two fingers moving towards each other;
- Swipe left, single finger moving left;
- Swipe right, single finger moving right;
- Lasso, single finger gesture of an arbitrary shape establishing a closed loop;
- Zoom, gesture using two fingers moving in opposite directions.

If a designer wants to use a gesture that is not listed, he can create custom gestures using an embed gesture creation tool: Intelligent Gesture Toolkit (IGT) by clicking on Add custom gesture button (Figure 2) or Record Gesture (Figure 1, item 13). ProtoActive is integrated with IGT (Figure 3), allowing the designer to create custom gestures and evaluate them with the prototypes. Any custom gesture created in ProtoActive through IGT will be automatically available for all the projects on the gesture triggers list (Figure 2).

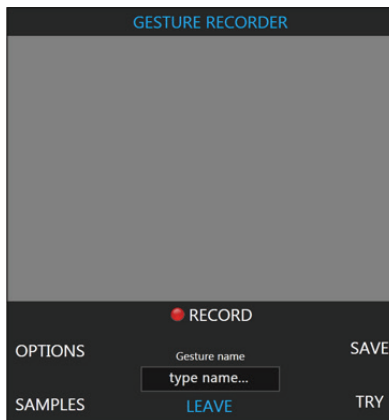


Figure 3. IGT screenshot.

### Recording gestures in IGT

The IGT gesture definition workflow can be seen in Figure 4. A gesture definition needs to be as broad as necessary to surpass the nuances of different users performing the gesture in different moments. A gesture also needs to be as precise as possible to avoid conflict with other gestures and to be detected only when this gesture is really intended by the user. In order to gather the terms and the different nuances to define a gesture, IGT asks the designer to train the tool by performing samples of the gesture they want to create. It is up to the designer to provide samples that cover all the nuances they desire the gesture definition to cover. It is also up to the designer to create the gestures providing the samples or by asking users to provide samples to generate gesture definitions.

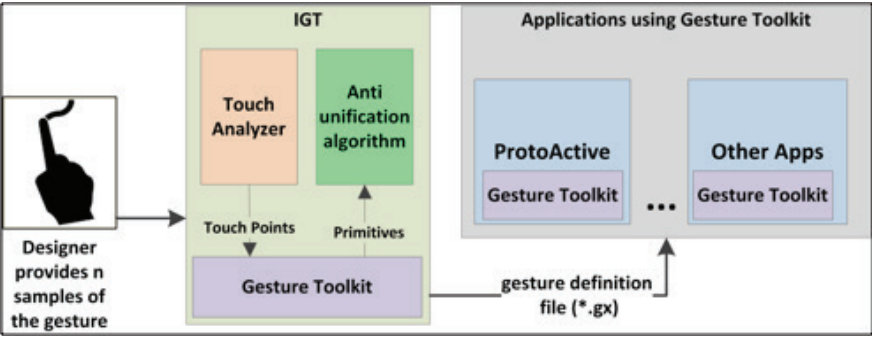


Figure 4. IGT Architecture.

Based on experimentation and custom heuristics, a sample is considered outside the standard when less than 20% of the gesture matches any of the previous samples. The designer can choose to keep the non-standard sample, which creates a more general gesture, or the designer can remove the sample and add another one.

Using the Gesture Definition Language (GDL) to define gestures allows the designer to read the definition of the sample as seen in Figure 5. If the designer does not agree with the gesture definition, she can remove the gesture definition and submit a new sample.

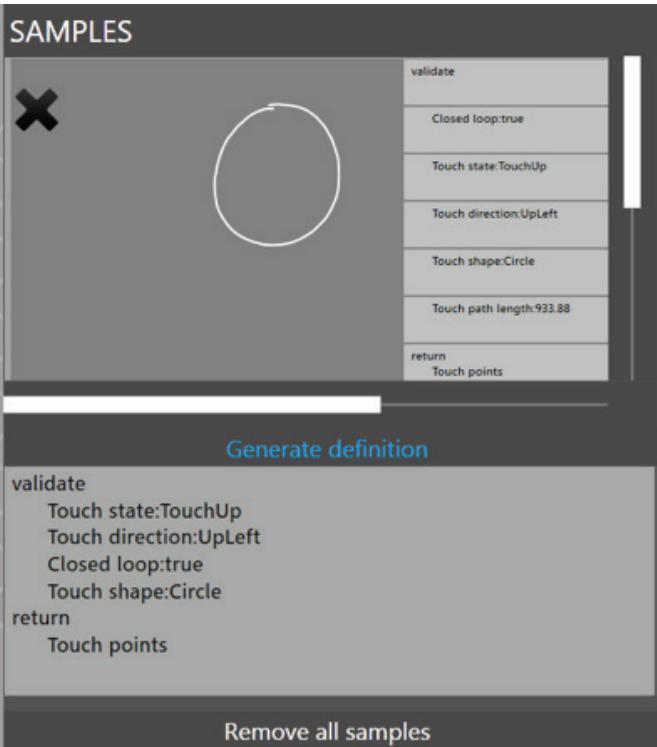


Figure 5. IGT Sample repository.

If a gesture definition is too specific, the designer has two options:

- Provide more samples to generate more variances for the anti-unification algorithm
- Change the "MATCHING ACCURACY".

When the designer agrees with the gesture definition provided, he can try the gesture in IGT by clicking "TRY" button (Figure 3). If the designer is satisfied with the gesture recognition, he can save the gesture thus making it available to any application using Gesture Toolkit with the IGT extension to GDL.

### **Pilot Evaluation**

Prototyping is one of the steps of user-centered design that has been proven to be an effective way to include users early in the design process, producing products that better fit user's need by getting early feedback. ProtoActive allows designers to evaluate two aspects of ITS applications: layout ideas through sketches of the prototype and interactions through pre-built or custom gestures. To evaluate the approach, we conducted a sequence of pilot evaluations.

The first pilot gathered different gestures that participants created to perform similar tasks. The variety of gestures created for the same task suggests that designers could benefit from such a tool as ProtoActive to evaluate different and innovative interactions.

Additional pilot studies had participants use ProtoActive and discuss its gesture creation and evaluation feature. The second study gave the same task to participants and asked them to create a gesture to accomplish a given task. The variety of gestures and the feedback from the participants suggest that such a create-custom-gesture feature might allow designers to innovate and try new design ideas with users. Due to the low cost and easiness to create and evaluate different ideas, more alternative designs can be explored.

The third pilot study evaluated ProtoActive's ability of evaluating interactions through tangibles, using fiduciary markers. The received feedback anecdotally suggests that using ProtoActive to appraise interactions that would normally be time consuming to create allows designers to experiment with ideas in an early stage. Additional feedback coming from experienced designers highlights ProtoActive's potential to reduce development effort for ITS applications.

### **Conclusion**

Our approach offers a pragmatic prototyping solution for ITS application development that is supported by two integrated tools. The first is ProtoActive, a sketch based prototyping tool for ITS applications. The main contribution of ProtoActive is to allow designers to evaluate not only the output of sketch-based prototypes, namely: can a user accomplish a task,

but also the gesture interactions for accomplishing the task. ProtoActive provides a pre-built set of gestures and supports customized gestures created with an embed gesture learner tool, IGT.

IGT uses samples of a gesture performed on the device to create a gesture definition that can recognize all the samples provided for a specific gesture. The novelty of IGT relies on the unique anti-unification approach used to identify all the common aspects between the samples, thus creating a gesture definition that is the most specific template covering all samples.

Enabling designers to create custom gestures allows the evaluation of different interaction ideas within similar costs and time constraints of low-fidelity prototyping. Providing designers with ways to evaluate custom gestures in the final application context (through using the custom created gestures in interactive prototypes) allows these innovative interactions to be developed following a user-centered approach as recommended by Norman and Nielsen (Norman & Nielsen, 2010).



## Pairing for Designing Visualizations

*Shahbano Farooq, Sheelagh Carpendale and Frank Maurer*

### Introduction

"A graphic is no longer 'drawn' once and for all: it is 'constructed' and reconstructed manipulated until all the relationships which lie within it have been perceived...a graphic is never an end in itself: it is a moment in the process of decision making." (Bertin, 1981). As Bertin said, while creating data representations one learns more about the data and the relationships within the data. With the help of this increased understanding of the data, one can improve the design of visualizations. Domain experts have stronger and deeper knowledge about the data and its relationships than the visualization designers. On the other hand, visualization designers have a better grasp of design concepts. Therefore, we suggest that the best outcomes can be reached by a close collaboration between the two.

Visualizations are becoming a widespread method for understanding data due to the availability of generalized business intelligence tools that support simple and interactive visual design, such as Tableau (Mackinlay, Hanrahan, & Stolte, 2007) and Spotfire (T.I.B.C.O, 2014). These tools have enabled domain experts to quickly transform data into simple as well as complex diagrams and charts. However, when the domain and the tasks are too complex, design expertise is needed to create visualizations that are easy to understand and supporting analysis as well as exploration tasks. In these situations, designers and domain experts need to work together to create custom visualizations for the complex domain and requirements.

According to the existing process of visualization design in the field (Sedlmair, Meyer, & Munzner, 2012), domain experts do not actively participate in the design of visualizations. They are limited to providing requirements and feedback on visualization designs. We wanted to overcome this limitation by creating a process that allows designers and domain experts to synchronously collaborate on creating complex visualizations. The process is inspired by pair programming (Williams, Kessler 2002) and allows both participants to take on active roles in the design process.

## Background – Visualization Design Process

To understand collaboration between the domain expert and visualization designer, we first need to understand how visualizations are designed in the real world. (Sedlmair, et al., 2012) have outlined the design study process, suggesting nine activities are carried out by a visualization designer, as illustrated in Figure 1.

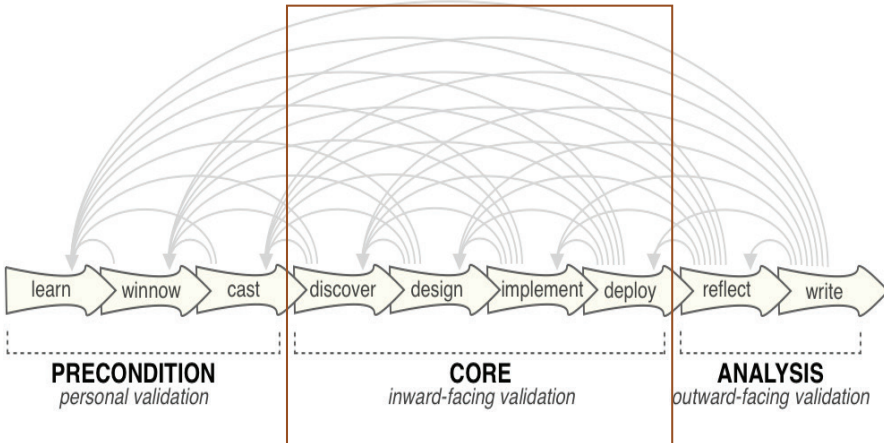


Figure 1. The nine stage Design Study Methodology Framework. Modified from (Sedlmair, et al., 2012). Illustrates activities carried out by the visualization designer while conducting a design study.

In this nine stage process, the precondition stage defines the tasks necessary to allow the designer to gain a general understanding of the data. These tasks include Learn, Winnow, and Cast.

The Core phase consists of the visualization design activities – Discover, Design, Implement, and Deploy. The Discover stage allows the visualization designer to develop an understanding of the domain, the users, and the problem, using user-centered design approaches, such as, observational studies, contextual enquiries, and interviews (Dix, 1998). The Design stage requires the Visualization designer to create low-fidelity paper or programmatic prototypes. The steps performed by the visualization designer include data collection & abstraction, mapping data to visual encodings and visual representation & interaction (Card, et al., 1999). The role of the domain expert(s) at this stage is to review the prototype designs and select the most useful.

In the Implement stage, a visualization designer implements the selected prototype, tests it using usability evaluation techniques and modifies the tool to overcome usability issues. The Deploy stage allows the domain expert to test the visualization in their day to day work activities so that she can provide usability feedback to the visualization designer.

According to this process, visualization designers are responsible for understanding the domain, the requirements, and designing a useful

visualization tool whereas the domain expert's role in the process is to provide input in the form of requirements, review, and feedback. Moreover, the domain experts and the visualization designers work asynchronously, with communication points for sharing information and feedback. We propose a modification to the Core Phase of the Visualization Design study Methodology, illustrated in Figure 1. Our research interest is to involve the domain expert in designing visualizations in close collaboration with the visualization designer (Pretorius & Wijk, 2009). We looked into existing literature to determine the current state of research.

### **Related Work**

Collaborative Information Visualization is a relatively new sub area of information visualization. (Mark, Kobsa, & Gonzalez, 2002) state that "that given the right visualization system, groups do better than individuals in finding more accurate results.". Recent interest towards "big data" analysis is also considering collaborative analysis of data on large displays (Isenberg, et al., 2010), (Forlines, et al., 2005). Current synopses in collaborative information (Heer, et al., 2008), (Stusak, 2009), and (Isenberg, et al., 2011) provide an overview on research on how to support a team of experts during visualization analysis. However, there is limited research on how to support a team of experts during visualization design.

### **Two Expert Challenge**

Visualization researchers have noticed discomfort between the visualization designer and the domain expert during data collection and requirement analysis activities. (Van Wijk, 2006) believes that a knowledge gap exists between a domain expert and a visualization designer. By knowledge gap the researcher is referring to their diverse areas of expertise and use of different terms and terminology to express themselves, which can result in confusion and frustration. Wijk suggests that this gap can be filled by educating domain experts to define visualizations. Some researchers have tried to bridge this gap while performing a long-term case study (Lloyd & Dykes, 2011). They educated the domain experts on a comprehensive set of possible visualization designs and interactions by giving them a lecture. Then they asked the domain experts to sketch possible designs for the data and tasks. Visualization designers were able to identify important design and interaction requirements from the sketches. It is evident from this research that teaching information visualization to domain experts and taking design requirements from them is useful. However, in this case study, the visualization designers did not assist the domain experts in creating the sketched paper prototypes. Recent research indicates that novice users, like domain experts newly trained on visualization techniques, face difficulty in creating and analyzing visualizations accurately on their own (Grammel, et al., 2010). We propose that domain experts should always be involved in visualization design, however in collaboration with visualization designers.

### **Multiple Prototypes**

(Sedlmair, et al., 2012) have explained that when the data and tasks are

complex or the scope of the domain is huge, the metadata information is partially in the head of the domain experts. (Pretorius & Wijk, 2009) with evidence from their experiences in design studies, have highlighted that information about the data and the tasks evolves through prototyping in close collaboration with domain experts. "Rather than trying to fine-tune a single technique", the researchers suggest "an exploratory approach where a number of prototypes are developed in close collaboration with users" and "when a promising idea is uncovered, it is then possible to nurture it to a mature solution." (Pretorius & Wijk, 2009). Visualization designers make use of paper or programmatic prototypes to get feedback from domain experts. We decided to support the scenario of a domain expert and a visualization designer creating prototypes together quickly and interactively using a visualization tool.

### **Paired Analytics**

Pair programming is a well-know collaboration approach in software development, coming from agile methodologies. Pair programming means that two programmers work together on the same machine. One programmer, the driver writes code, while the other, the navigator, reviews and helps the driver. The two programmers exchange roles frequently. According to a survey on pair programming (Cockburn & Williams, 2000), pair programming improves design quality, reduces defects, and improves team communication. (Arias-Hernandez, et al., 2011) used this concept to study visual analysis. They paired a domain expert and a visualization designer to study visual data analysis activities and referred to it as "Pair Analytics". The researchers found that Pair Analytics provided them with a more natural means of capturing analytic reasoning rather than "think aloud protocol". Research evidence suggests that tightly coupled work environments lead to a natural means of discourse between team members (Tang, et al., 2006). We propose that pairing domain experts and visualization designers during visualization design activities can lead to a natural discussions on data, task requirements, and visualization designs.

### **Iterative Design**

(Gammel, et al., 2010) conducted a study to learn how novice users with limited knowledge of visualization design create visualizations on their own. An important finding of their research is that participants repeated visualization design activities with different representations till a useful visualization was found. The research informs us that these iterative visualization design activities support learning in three ways: understanding the data with different representations, finding the accurate representation, and gaining experience in visualization design. We propose that iterative construction and discussion can support knowledge sharing between the two experts and can contribute to better design decisions. This idea is also used in design education and is known as learning by design or problem-based learning (Kolodner, et al., 1998). The approach a very effective practice in supporting collaborative designs in a classroom setting and helps students create better designs based on their own learning through



problem-solving and critique from their peers.

### **Reviewing Existing Visualization Tools**

There are different types of Information visualization users and many commercial tools and open source toolkits have emerged to support their differences. (Heer, et al., 2008) have categorized these differences based on the following:

*User Skill and Knowledge of Visualization Design and Analysis.* Domain experts can create standard visualizations using commercial tools, such as Tableau (Mackinlay, et al., 2007) and Spotfire (T.I.B.C.O, 2014). When these tools do not provide adequate results, visualization experts can programmatically create novel visualizations and interactions using toolkits, such as Processing (Reas & Fry, 2014) and D3 (Bostock, et al., 2011) to satisfy unique and complex requirements. None of the existing tools support the scenario of interactive design involving novice users and programmatic enhancement by experts (Heer, et al., 2008).

*Visualization Design Requirement: Exploratory vs. Explanatory Design.* Some visualizations are static in nature and provide the outliers or trends in a single glance of the view – they are explanatory visualizations. In our case, we need a tool to explore the data and find the appropriate view that can highlight the trends and outliers in the data. Existing commercial visualization tools support quick and interactive means of exploring data through different visualizations. We want to support a designer and a domain expert in collaborative visualization prototyping.

*Number of Users: Single or Collaborative Visualization Design and Analysis.* Most commercial visualization tools support single or asynchronous design and analysis of visualizations. In recent years some research tools have emerged that support collaborative analysis, but none have looked into supporting collaborative design of visualizations (Isenberg & Caprendale, 2007), (Isenberg & Fisher, 2009), (Tobiasz, et al., 2009), (Forlines, et al., 2005), and (Forlines & lilien, 2008).

*User's Role in Visualization Design and Analysis.* Tools also need to support the visualization designers and the domain experts during the visualization design process. Visualization tools for requirements gathering, collaborative designing, and visualization testing are yet to be designed.

On reviewing existing tools and our requirements, we came to the conclusion that existing tools do not support the scenario of collaborative visualization prototyping between a domain expert and a visualization designer.

### **Requirements**

As discussed in the Related Work section, we were faced with the challenge of creating a visualization prototyping tool to support collaboration between the domain expert and the visualization designer on the data, tasks,

and the visualization designs. The following section describes the major requirements we followed to create a tool named PairedVis. The tool can support the two experts in collaborative prototyping. PairedVis is designed based on the following functional requirements:

*R1. Two Expert Challenge.* The tool should provide an interface for discussing the data and the underlying relationships in the data that are in the mind of the domain expert to aid in the selection of useful visual representations (Sedlmair, et al., 2012), (Pretorius & Wijk, 2009). Similarly, we need to support a visualization designer in explaining existing visualization templates and how to perceive them accurately to aid the domain experts in understanding, designing, and interpreting visualizations accurately (Grammel, et al., 2010).

*R2. Multiple Prototypes.* The tool should provide quick and interactive means of creating visualizations to support prototyping. The quick and simple interactions can enable a domain expert to actively participate during prototyping. On the other hand, we also need to support a visualization designer in programmatically enhancing the existing templates into functional prototypes (Heer, et al., 2008).

*R3. Paired Analytics.* We want the two experts to take turns in discussing their expertise during visualization prototyping. As a result, we want to support tightly coupled work on a tabletop in a collocated environment (Tang, et al., 2006).

*R4. Iterative Design.* The tool should support quick and interactive means of switching between visualization templates to support iterative design.

## Interface Design

We designed a tool, PairedVis to support collaborative design activities between a visualization designer and a domain expert using visualization templates. The interface of PairedVis is designed based on the data state reference model (Card, et al., 1999). As a result, the interface has four panels; the data panel, the data transformation panel, the view transformation panel, and the code panel. The main interface design is shown as an abstract representation in Figure 2. The screen can show two panels at a time. Arrows can be used to flow back and forth between the panels at any time.

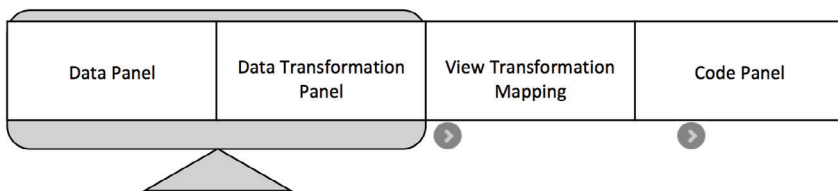


Figure 2. PairedVis interface with four panels.  
The screen showing two panels at a time.

The Data panel holds the dataset in a table format. The second panel, the Data Transformation panel, can be used to create relationships between the data columns. The third panel enables mapping of data to visual representations, therefore it is called the View Transformation panel. The Code panel provides the code behind the visualization for sharing or customization.

### Discussing Data

Before creating visualizations, the domain expert and a visualization designer need to have a shared understanding of the data and the requirements. Therefore, we have designed the data transformation panel to support domain experts in discussing the data and the relationships between the data variables. Our approach to providing discussion on the data is inspired by concept mapping diagrams (Novak & Canas, 2008) such as Class Diagrams from software engineering and entity-relationship diagrams in database modeling. In PairedVis, data variables of interest can be selected from the tabular data in the Data Panel. The selected attributes (or: data variables) are represented as bubbles (circles) on the data transformation panel for concept mapping, as shown in Figure 3.

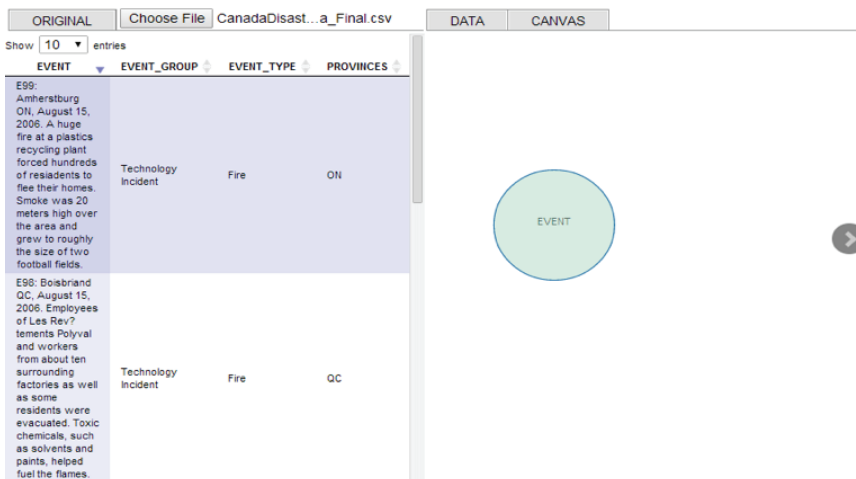


Figure 3. Left panel for uploading data and selecting attributes. Right panel for data transformation. On tapping an attribute (column) on the left, a bubble appears on the right representing the variable. A user can interact with the selected data variables.

As shown in Figure 3, we represent the selected data variable as a bubble (circle) on the data transformation panel. The domain expert can use the Data Transformation panel on the right to explain the relationships between the data. We were inspired by relationships based on entity-relationship diagrams in database modelling. In entity-relationship diagrams there are two major types of relationships, hierarchical/parent-child relationships and associative relationships.

*Hierarchical.* Assume that the domain expert is interested in exploring disastrous events data of Canada. He/she can explain that events can be categorized based on event types; such as, wild fires, landslides, thunderstorms, and so on. Event types can be further categorized under event groups, such as Natural Disasters, Industrial Accidents, War, and so on. This hierarchical relationship can be represented using a bubble inside a bubble, as shown in Figure 5. “Events” are placed inside “Event\_Type” and “Event\_Type” is placed inside “Event\_Group”. The hierarchical representation of a bubble inside a bubble can also be used for grouping. For both hierarchy and grouping we used the same interaction, because the nesting operation is required to facilitate both grouping and hierarchy of data.

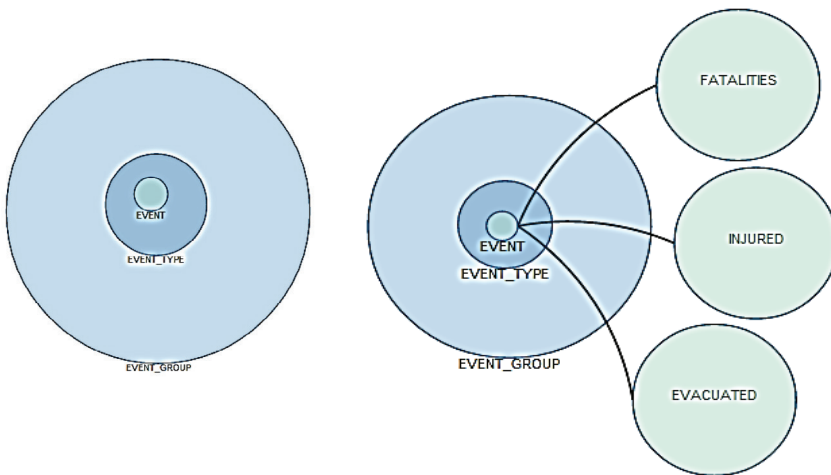


Figure 4 (left). Using bubbles inside a bubble to show hierarchical relationship; Figure 5 (right). Links between the bubbles showing an associative relationship.

*Causal or Associative Relationship.* This relationship is used when one data variable is associated or dependent on the other but cannot be categorized as inheritance. For example, a domain expert might want to explain that for each event he has information about the number of injuries, evacuees, and fatalities that occurred. This relationship can be represented with the use of links between the bubbles, as shown in Figure 5.

The use of these two relationships results in a graph structure. Prefuse (Heer, et al., 2005), made use of a graph structure between the data and the visualization, to facilitate data transformation operations. The difference in our tool is that we have provided a visual form for representing data and have provided interactions to show relationships between the data variables in visual form.

### **Discussing Visualization Design**

After understanding the data, the visualization designer can suggest

appropriate visual representations for the data. We have presented sample visualization representations in a sliding thumbnail bar at the top of the view transformation panel, shown in Figure 6.

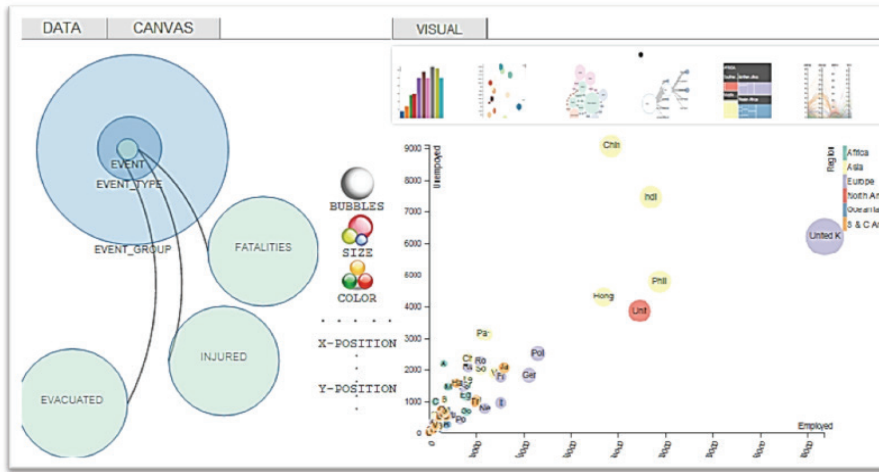


Figure 6. View Transformation Panel on the right. The thumbnail bar on the top shows the six templates mapped to sample datasets. The scatterplot has been selected and is in view. The data variables available with the scatterplot are in the middle of the data transformation panel and the view panel.

The view transformation panel currently supports six representations: bar chart, scatterplot, bubble chart, Reingold tree layout, treemap, and parallel coordinates. Each of these representations are mapped to sample datasets. Figure 6, shows the Scatterplot mapped to the sample data about immigrants to Canada based on country of birth. We also provided a breakdown of visualizations into their basic components to help explain visual mappings to novice users as suggested by (Grammel, et al., 2010) (Kwon, et al., 2011). To create a visualization, the two experts need to map a data variable to one of the visual variables shown in the center of Figure 6. By selecting a visual variable in the template, the visualization changes and only shows the selected visual variable in the view. This serves as a simple beginning for domain experts to understand some of the complex visualizations developed by the community.

### Addressing Interactive and Iterative Design

PairedVis enables the two experts to move between the panels with a simple swipe to support fluidity during iterative selection of data and mapping to visualizations. Selecting and switching between templates and visual variables is supported by simple drag and drop interactions. The major goal of PairedVis is to facilitate data exploration and discussion through a few visualization templates. However, PairedVis also allows visualization designers to enhance the existing templates programmatically. As a result, there are no extra panels to perform data transformation operations, selecting colors from a palette and, so on. These operations are handled

automatically. The data transformation operations, such as nesting occurs based on the relationships created between the data variables in the data transformation panel. The bubble inside a bubble results in a nesting operation. Similarly, view transformation operations such as color encoding is also automated. Personalized choices are left to the programmatic customization of the visualization through the code panel.

To support platform independence, PairedVis is designed in HTML5 and JavaScript. The visualizations are created using the JavaScript based toolkit D3 (Bostock , et al., 2011). D3 is an open source JavaScript library and there are many open source templates designed by the community that do not just provide data representations but also interactions to support data exploration. For example, we used the treemap template that facilitates zooming in on a parent to view only the children that belong to it. We however, made small modifications to the interactions provided by default with these templates to facilitate touch interactions. The following section describes the initial laboratory study we conducted as a first step towards evaluating PairedVis.

## **Evaluation**

PairedVis is designed to support collaborative prototyping involving a domain expert and a visualization designer. As a result, it would have been natural to study this collaboration in a real world setting. However, PairedVis is in the early stages of development and is currently a functional prototype. Therefore, we decided to get initial feedback in a laboratory setting.

## **Study Goals**

PairedVis is designed to support both the domain expert and the visualization designer in sharing their knowledge. The interface of PairedVis enables a domain expert to share his knowledge of the data and the visualization designer to share his knowledge of visualizations. Moreover, PairedVis interface was made simple to ensure that domain experts can understand how to map data and analyze representations. Therefore, our study goal was to investigate whether knowledge sharing activities occur during visualization design with PairedVis. We also wanted to investigate whether both experts critique the selected visualizations and discuss their limitations in satisfying data and task requirements.

## **Study Methodology**

To evaluate our research goals, we conducted the study in a laboratory environment. We decided to use a fresh pair of participants, a domain expert and a visualization designer in each experiment. We took a qualitative approach to investigating the impact of PairedVis on two questions:

- How does PairedVis support the participants in sharing their knowledge and experience?
- How does their collaboration critique existing representations and their limitations with respect to data and user requirements?

## Participants

Twelve university students were recruited for this study through mailing lists and word of mouth. Two participants worked together as a pair resulting in six experiments that took place in one week. Participants with two or more years of experience in visualization design were given the role of a visualization designer and were paired with a participant with no experience in visualization design, who took up the role of a domain expert. We could not recruit professional domain experts and visualization designers. However, in the first 20 minutes of the study, we motivated them to take up the role of a domain expert or the visualization designer. The domain expert was provided 20 minutes to get familiar with the data, while the visualization designer was given the same time to learn how to create visualizations with PairedVis.

## Setup

The study environment consisted of two labs in close proximity, Lab A and Lab B. Lab A was setup with a touch-enabled tabletop connected to a keyboard and a mouse. PairedVis was running in the browser on the tabletop before the experiment. A camera was positioned on top of the table to capture participants' activities on the tabletop and record the conversation between the participants. Lab B was setup with data and tasks on paper, as well as on an electronic tablet, to facilitate data and tasks on both mediums.

## Procedure

The study required two researchers, one to assist each participant in the two labs. The visualization designer was invited to Lab A, whereas the domain expert was invited to Lab B. The study consisted of three parts. Part 1 took 20 minutes of the study and during this time the participants were given the information necessary to take up their respective roles. Part 2 took 30 minutes during which the domain expert and the visualization designer created visualizations together using PairedVis. During Part 3, the participants shared their experiences in a follow-up interview separately.

## Tasks

We used a simple dataset that provides details about disastrous events that occurred in Canada (Statistics, 2013). The events were described in ten columns, consisting of the Event\_Group, Event\_Type, Provinces, Fatalities, Injured, Evacuated, Days, Cost, Year, and Month. The domain experts were provided with a task sheet on paper, consisting of six tasks, as shown in Table 1.

Data Analysis Tasks	
T1	List two most significant disastrous events that occurred in Canada with respect to fatalities?
T2	What types of events have effects for larger number of days?
T3	What types of events cause more fatalities and injuries?
T4	List two types of events that are most disastrous?
T5	Which provinces are most affected by the major types of events identified in Q4?
T6	Are there any event types reoccurring in the same season?

Table 1. Tasks and the expected results of these tasks.

## **Data Collection**

We observed and videotaped the participants during the second part of the study, while they were creating visualizations in collaboration. We did not consider how much time was taken to complete the tasks or how many tasks were completed in each study. After the study, we interviewed them to gain more insight into the experience of the participants. We had determined a few questions to guide us through these open ended interviews.

## **Data Analysis Methods**

We used a qualitative approach to analyzing the video recordings of the collaborative visualization activities. This helped us in investigating the communication between the two experts during the study. We transcribed the video based on the major activities carried out during visualization design. These activities were repeated in a cycle for each task and can be described based on the Data State Reference Model (Card, et al., 1999); data abstraction, visual representation selection, visual mappings, and visual analysis. This is a similar approach to (Isenberg, et al., 2010). As a result, a visualization design cycle starts when a task is read and ends when the task result is written in the task sheet. During the second parse of the video recordings we closely observed discussions while these tasks were performed and found other important activities, such as task and data clarifications, representation explanations, and critique. This study is different from other studies (Grammel, et al., 2010), (Isenberg, et al., 2010), (Kwon, et al., 2011), because we looked at the discussion between a domain expert and a visualization designer while creating and analyzing visualizations together.

*Data Abstraction.* In all the studies the domain experts would start with dictating the task and the data. In certain cases, the visualization designer would ask clarification questions to understand the task or ensure that the selected data was correct. For example, in experiment1, the visualization designer asked, "Do we need to select the country as well?" and the domain expert replied, "The dataset is only from Canada". As a result, there was a natural flow of communication between the two during this phase to understand the task requirement.

*Visual Representation Selection.* During this phase if the visualization designer had selected a representation for the first time, they would explain it to the domain expert. In experiment 3 and experiment 4, the visualization designers made use of the animations to break down the representation into its' individual components. In some cases, the visualization designer would explain why they selected the particular representation. For example, in experiment 1 the visualization designer explained "as the dimensions go more than two it is better to use these new charts" and starts pointing to the scatterplot and moved the finger towards the parallel coordinates.

*Visual Mappings.* After selecting a representation, the visualization designers mapped the data to the available visual variables and in most



cases explained what was being mapped. The domain experts liked the links between the visual variables and the data variables. One of them commented in the interview, "The easy thing to understand was, oh you make a connection from color to a certain column. That is very explicit." Another domain expert liked how the visualization changed when each visual variable was mapped. "I also like the feature that your visualization changed dynamically, you see the visual variable you mapped to." The domain experts suggested mappings to visualization experts in all the experiments.

*Visual Analysis.* Only the parallel coordinate's representation was explained during analysis. It could be due to the fact that the other representations were common. In experiment 3, after the visualization designer had mapped the data to the treemap and was analyzing the data, the domain expert asked what each rectangle meant, and the visualization designer explained.

## **Discussion**

In general, using paired participants results in a natural continuous conversation between the participants. Overall the participants liked the interface of PairedVis and two visualization designers described the experience as enjoyable. The domain experts easily understood how to create representations. One domain experts said, "it was pretty straightforward".

The visualization designers in all the experiments made use of the sample datasets to explain the visualization templates to the domain experts, and in two cases also used the animations that show one visual variable at a time. The domain experts in all the experiments requested for a different representation, when the task requirement was not met and explained their requirements in greater detail. For example, in experiment 4, the domain expert said that I would like to group fatalities and injuries together. As a result of such discussions, the visualization designer would look over the thumbnail bar for representations and think before choosing a more useful one. Then the data was mapped again and the analysis was performed with the new representation. These steps would iterate until both of them were satisfied with the visual representation and the results of the analysis. Especially for Task3, in all the experiments the visualization designers switched to at least 3 different representations.

The Analysis of our study support that PairedVis enables both the experts, a domain expert and a visualization designer in sharing their knowledge. PairedVis had facilitated the two experts with quick and simple interactions in order to map data to different representations. A domain expert had noticed this and said, "...you can instantly try out different charts, usually for excel if you pick one chart, trying to change it to other things for same data takes time but this one switching between charts, its design to actually for people to use different charts." Visual mapping was described as "quite explicit" by a visualization designer.

## **Conclusion**

The main aim of our research was to better support collaboration between a visualization designer and a domain expert during visualization design activities. Our overview on current literature in information visualization processes and tools led us to think that domain specific visualization designs are either created on paper or programmatically by visualization designers. In this case, domain experts are limited to reviewing and providing feedback on these designs. However, when the domain is simple, commercial business intelligence tools help domain experts in creating visualizations on their own. When the domain is complex, we proposed that they can create visualization designs in collaboration with visualization experts. Pretorius and Van Wijk [10] have suggested an exploratory approach to creating prototypes in close collaboration with domain experts. However, we propose the use of adjustable templates in order to explore and discuss representations.

We elicited requirements for a tool that can facilitate both the domain expert and the visualization designer in creating and discussing visualizations. We found that existing tools are not designed to support collaboration between a domain expert and a visualization designer. As result, we designed a tool, PairedVis to support collaboration between the two experts. We conducted an experiment in a laboratory study to investigate whether this tool can support discussion between the two experts. Our evaluation supports that collaboration between the two experts results in sharing knowledge and expertise. Moreover, collaborative prototyping results in critique more templates.



## **Constructive Visualization: A New Paradigm to Empower People to Author Visualization**

*Samuel Huron, Alice Thudt, Bon Adriel Aseniero, Tony Tang, and Sheelagh Carpendale*

### **Introduction**

During the past two decades, information visualization (InfoVis) research has created new techniques and methods to support data-intensive analyses in science, industry and government. These have enabled a wide range of analysis tasks to be executed, which vary in terms of the type, and volume of data involved. However, the majority of this research has focused on static datasets, and the analysis and visualization tasks tend to be carried out by experts.

In more recent years, social changes and technological advances have meant that data have become more dynamic, and are consumed by a wider audience. These social and technological changes give rise to multiple challenges as most existing visualization models and techniques are intended for experts, and assume static datasets. In spite of this, only a few studies have been conducted to explore these challenges.

In this chapter, with my collaborators, I provide a pictorial overview of two papers that address these challenges (Huron, Jansen, and Carpendale 2014; Huron, Carpendale, et al., 2014). In these paper we define construction as a design paradigm for non-experts to author simple and dynamic visualizations. This paradigm is inspired by well-established theories in developmental psychological as well as past and existing practices of authoring visualization with tangible elements. We describe the simple conceptual components and processes underlying this paradigm and a preliminary study we employed to assess it. The results of this study confirm that non-experts in InfoVis can create, update, and annotate a visualization in a short period of time. Moreover, this study allowed us to articulate a primary model of how people perform the authoring of visual mappings using this paradigm.

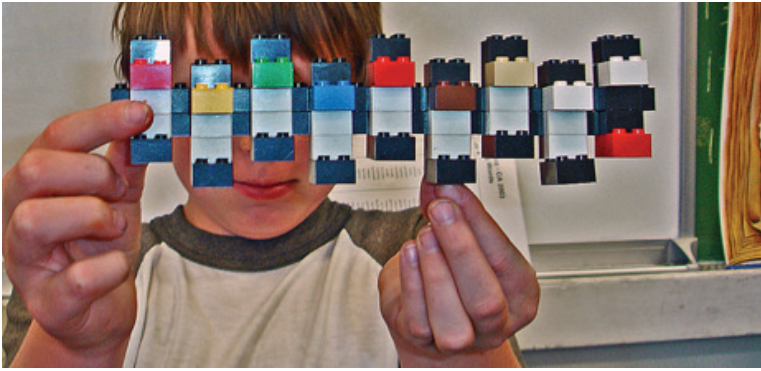


Figure 1. Photo of a kid showing a construction made with Lego bricks.  
Credit: Michael McCauslin.

## Part 1. Constructive Visualization Paradigm

### Design Challenges

Democratizing visualization authoring is challenging. Below we describe our three main challenges we are considering:

*Keeping it simple.* It can be said that actions are simple and accessible if they are similar to the actions we have been comfortable with since early childhood. A good example of this is sketching, for which one of the best advantages is, that we all can do it.



Figure 2. A sketch of a car drawn by a 4 year old child. Copyright Emran Kassim.

*Enabling expressivity.* We are looking for a creation process that provides sufficient freedom to enable people's ability to express their own ideas. Our ideal is to support the expressivity of sketching and the flexibility of digital tools by incorporating the concept of plasticity, or the ability to remodel during the creation process.

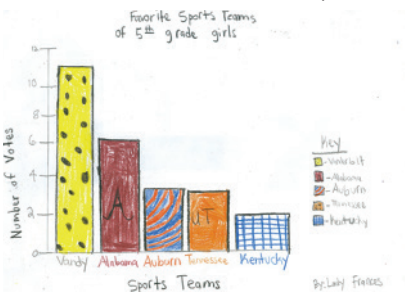


Figure 3. Drawing of a bar chart with different textures.

*Incorporating dynamics.* One of the biggest challenges in making the creation of visualizations more generally accessible is that, thus far, visualizations can only be made adaptable to data dynamics through coding. By this we mean that the visualization can change in response to a change in the data stream. However, coding remains, and is likely to remain, a skill of comparatively few people.



Figure 4. An icon symbolizing “update.”

### Three Design Paradigm to Create a Visualization

Previously in his talk “Drawing dynamic visualization” Bret Victor’s (Victor 2013) introduced three visualization design paradigms: Use, Draw, and Code. Below, we will summarize these three approaches, and introduce a new additional paradigm.

*Using.* The first paradigm, using, refers to the possibility of pushing a magic button in a software (Figure 5) which directly transforms a dataset into a traditional, pre-coded visualization. This is a simple way to produce visualization, if you know the location of the button and how it functions. Moreover, when the data changes, the visualization gets dynamically updated. However, this is not an expressive tool—you cannot personalize the visualization.

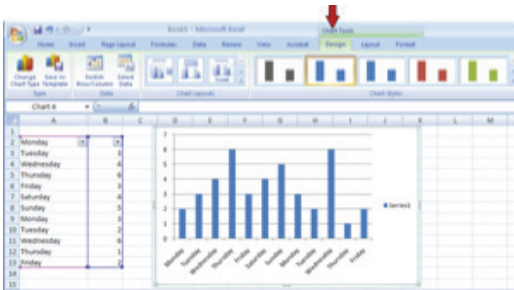


Figure 5. Microsoft Excel interface, illustrating the using design paradigm.

*Drawing.* The second paradigm of producing visualizations is to draw, either by hand or by using a drawing software such as illustrator (Figure 6) (Walny et al. 2012; Walny, Huron, and Carpendale 2015). In this case, the process could be quite simple and very expressive, but it is not dynamic. If you want to update the data visualized in the drawing, you will have to redraw most, if not all of it.

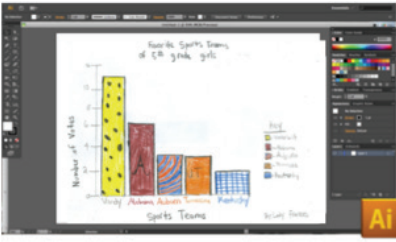


Figure 6. Adobe Illustrator interface, illustrating the drawing design paradigm.

*Programming.* The last paradigm is coding (Reas and Fry 2007). Through the use of a programming language, programmers can encode abstract symbols (Figure 7) then compile and run their code to show the visual result. This process is not simple, but it can handle dynamic data changes, and can be very expressive.

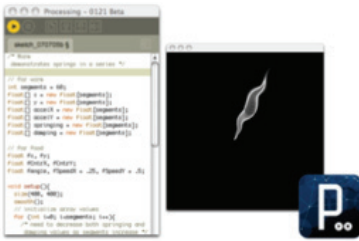


Figure 7. Processing interface, illustrating the programming design paradigm.

## Summary

We can summarize these three approaches to get an overview (Figure 8):

- Using is simple, handles dynamic data, but is not very expressive,
- Drawing is simple and expressive, but does not handle dynamic data,
- And coding is not simple, but allows data dynamics, and is very expressive.

All of these approaches are well studied. Grammel et al. (Grammel et al. 2013) recently surveyed the different ways to create information visualization. If we classify all these papers among these three approaches, we get the distribution shown in Figure 8. In this chapter, we present and describe a new design paradigm, which addresses the three challenges: simple, dynamic, and expressive, and has not been previously study.

	Simple	Dynamic	Expressive	Papers
Using	✓	✓	✗	36
Drawing	✓	✗	✓	13
Coding	✗	✓	✓	17
Constructing	✓	✓	✓	0

Figure 8. Summary of the three main design paradigms according to our design challenges.

## A New Design Paradigm for Visualization: Constructing Visualization

We define constructing visualization (Huron, Carpendale, et al. 2014) as creating visualization by assembling components that represent data. To define this paradigm we presented tree main aspects:

- The historical inspiration
- The components and process
- The some real life example

### Historical Inspiration

#### 1. Frederich Froebel, The Invention of Kindergarten

Our first source of inspiration is Frederich Froebel (Figure 9), the German pedagogue who invented the Kindergarten in 1837 (Brosterman, Togashi, and Himmel 1997; Manning 2005). Froebel's challenge was to teach mathematics to children who do not know to read and write. To solve this problem he designed some building block toys called Gift (Figure 10). A part of his pedagogical approach was to teach children that each block corresponds to an abstract unit (Figure 11), and by manipulating the blocks, they could process mathematical operations such as addition, subtraction, and multiplication among others.



Figure 9.



Figure 10.

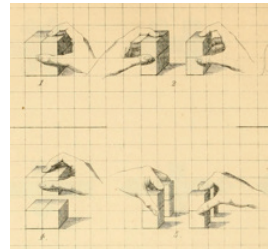


Figure 11.

Figure 9. Portrait of Friedrich Wilhelm August Fröbel. Picture in the public domain;

Figure 10. Picture of the Third Froebel Gift. Credit: Samuel Huron – Creative

Commons; Figure 11: Image from page 53 of  
"A practical guide to the English kinder-garten".

#### 2. Jean Piaget, The Constructivism Theory

Our second source of inspiration is Jean Piaget (Figure 12) a Swiss psychologist, who is known for research about children's development (Chapman 1988; Piaget 1989). Piaget used building blocks similar to the ones designed by Froebel in his experiments (Figure 13). According to Piaget, children construct most of their knowledge by manipulating, and experimentation with physical objects. Piaget provided a solid theory that helps us understand the learning stages during children's cognitive development.



Figure 12.



Figure 13.

Figure 12: Photograph of Jean Piaget—Picture in PD; Figure 13: Screenshot of a Youtube video of a Jean Piaget experiment. Accessible online at: <https://youtu.be/0XwjruMI94?t=27m51s>.

### 3. Seymour Papert, from Constructivism to Constructionism

The third source of inspiration is Seymour Papert (Figure 14) and his colleague Alan Kay. Papert was a MIT mathematician, computer scientist, and educator. He built on top of the constructivist theories and extended the idea of pedagogical manipulative materials to computer programming (Papert and Harel 1991). Papert founded the “Lifelong Kindergarten Group,” a research group at the MIT MediaLab (Figure 15). One of the major works this group is known for, is the programming environment Scratch (Figure 16). Scratch was inspired by Froebel’s methods, transforming the building block idea into a visual representation of the “command block”. This approach was so successful, the Scratch logic is now integrated in programming interfaces of commercial products such as Lego Mindstorm.



Figure 14.



Figure 15.



Figure 16.

Figure 14. Portrait of Seymour Papert. Credit Rodrigo Mesquita; Figure 15. MIT Medialab LifeLong Kindergarten logo; Figure 16. Command block from Scratch. Some Right reserved by Andrés Monroy-Hernández.

### Summary

From this three sources of inspirations, Frederich Froebel, Jean Piaget and Seymour Papert we learned:

- That the understanding of abstract and mathematical concepts can be developed through the manipulation of simple elements such as wooden blocks. We also learned that this approach is proved to be continually accessible and effective, as it has spread across the world, and is still in use today. This meet our Frist design challenge: Simplicity
- That this approach also allows people to modify and understand their constructions over time. This meet our third design challenge:



Dynamicity

- That this approach is highly creative and generative. This meet our second design challenge: Expressivity

## Components and Processes

In the following section, we describe the components and process of constructive visualization.

The first component is a token, which is mapped to a unit (Figure 17, Co1). For instance, the blue square on Figure 17 could be considered as a graphical token which maps to a single “yes”.

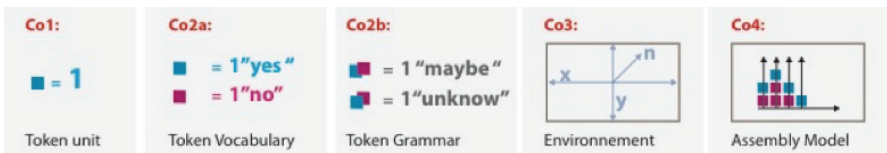


Figure 17. The components of constructive visualization.

Credit: Samuel Huron - Some right reserved.

The second element that needs to be defined is the token grammar and vocabulary (Co2a, Co2b). The token vocabulary is the relation between the different token properties and data properties. For instance, if I want to self-monitor my consumption of apples against soda, the pink square can stand for one soda and the blue square for one apple. The token grammar contains the rules one can define about the relationship between two or more tokens.

To organize these tokens, we need to have an environment (Figure 17, Co3). The environment is the space that provides constraints on how tokens can be assembled together using the token grammar. The properties of this space can include many different types of constraints such as the number of dimensions (e.g., 2D or 3D), space limitations, grids, gravity, and others.

The last component is the assembly model (Figure 17, Co4). The assembly model is the rules of the construction process. These rules are defined by creating the visualization, and concern the spatial organization over time.

## Process

The process for providing a constructive information visualization environment is based on four steps:

- P1: Environment initialization.
- P2: Mapping data to “tokens”, and data properties to token properties.
- P3: Assembling the tokens.
- P4: Evolution over time.

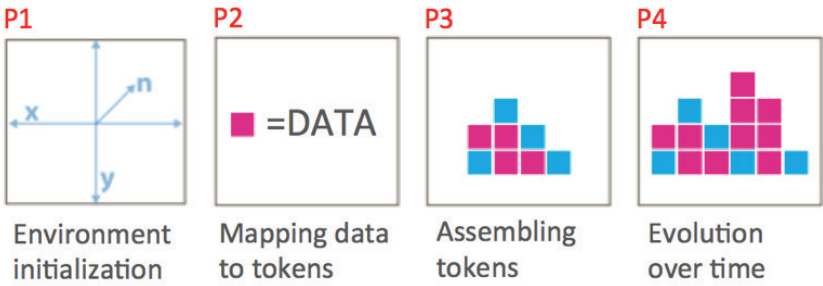


Figure 18. The process of constructive visualization.  
 Credit Samuel Huron - some right reserved.

### Real Life Examples

In this subsection we will present two real life scenarios, other examples can be found in a previous paper (Huron, Carpendale, et al. 2014):



Otto Neurath (1882–1945)  
 Philosopher  
 Otto creates the Isotype principles.



Michael Hunger, contemporary  
 Programmer  
 Michael creates personal Infovis.

### Example 1: Otto Neurath, Isotype Principle

Otto Neurath wanted to democratize statistics of socio-economics datasets. For that, he created a specific type of visualization called Isotype (Jansen 2009; Neurath and Vienna 2009; Neurath 2009). The following are the steps he used to create an Isotype visualization:



*Drawing Tokens.* First Gerd Arntz, Otto Neurath's graphic designer, draws pictograms to represent a specific semantic type of data. These pictograms will be used as tokens, and the symbol of the pictogram defines its meaning.



*Duplicate Tokens: Molding and Printing.* To quickly duplicate the pictograms, they used a mold. Using this mold, they were able to produce as many pictograms as they needed.



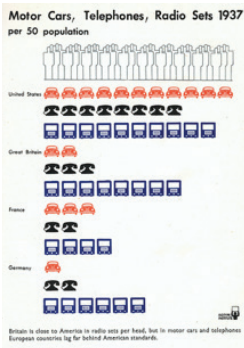
*Duplicate tokens: Clipping.* The pictograms are then clipped to be discreet elements that can be manipulated.



*Composing.* After clipping, the pictograms are assembled on top of a white canvas. On the picture to the left, you can see Marie Neurath, Otto Neurath's wife, positioning each pictogram into an assembly model. You can also see in this picture, the frame that plays the role of a two-dimensional assembly environment. During this phase the assembly could be updated and changed as necessary.






*Photographing.* Later, when they are satisfied by the resulting composition, they take a picture of it as seen on the picture to the left.



*Printing and distributing.* With this picture, you can see a poster resulting from this process.

Let us analyse the different components of this visual representation. The tokens are the following symbols:



-  = 1 car by 50 persons
-  = 1 car by 50 persons
-  = 1 car by 50 persons

The assembly model follows a horizontal bar chart principle in which each group of three lines represent a country, and each line represents the distribution of a good within the country. The environment is a two-

dimensional canvas.

### Example 2 : Michael Hunger, Personal Visualization

Michael Hunger is a computer engineer, he works on many different projects and he is having troubles managing his time. As he explained on his blog, (<http://goo.gl/Qz554q>) he has already tried software techniques such as Outlook, Spreadsheets (Figure 19), as well as more tangible techniques such as tally sheets, pen and paper to-do lists, sticky notes, or using a notebook.

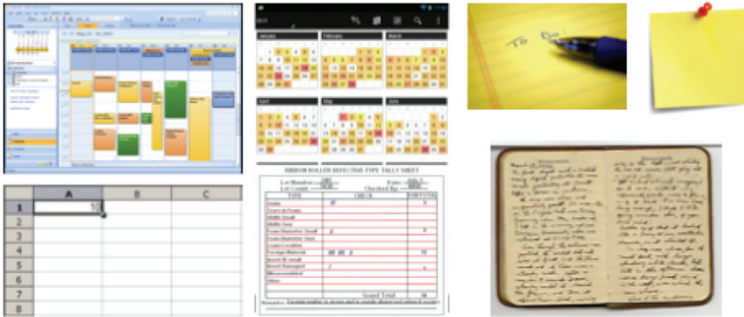
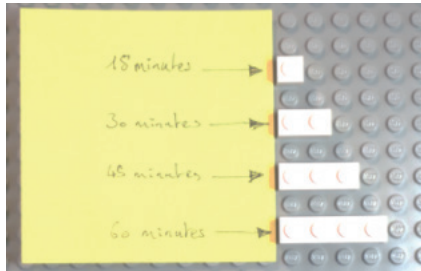


Figure 19. Collage of different tools for personal time management, from left to right, top to bottom: Outlook express interface, schedule overview with d3.js, pen and paper, post-it, spreadsheet, tally sheet, and a personal notebook.

Finally, he decided to design his own solution out of Lego blocks. The following are the steps he performed to create his solution out of legos:

*Token Mapping: Time to size*  
First, he decided to map time frames to the size of Lego™ bricks:

- 15 minute to 1 pin brick,
- 30 minutes to 2 pin brick
- 45 minutes to 3 pin brick
- an hour to 4 pin brick.



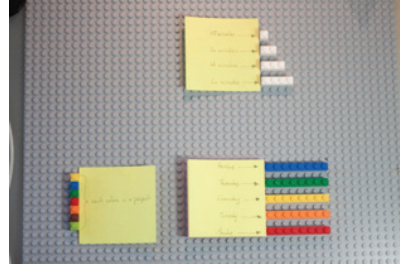
*Token Mapping: Color to Project.* Then, he mapped the different project types into the different colours of the bricks.



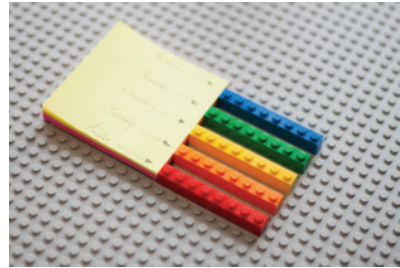
*Token Mapping: Long Brick to Days.* Finally, he used long coloured bricks to represent the days of the week. Monday is red, Tuesday orange, Wednesday yellow, and so on.



*Token Grammar.* Michael self-defined his own token grammar to construct a visualization. He mapped different dimensions of the data to different attributes of the tokens. Here, we can also see that the environment is a Lego board.



*Environment of Assembly.* Let us simulate how Michael constructs and updates this visual and tangible representation. It is 10 in the morning, the Lego board is still only contains the long coloured bricks symbolizing the days of the week.



*Assembling the Tokens Over Time.* Michael arrives late and spends an hour to read and reply to his emails. The email processing task is symbolized by blue bricks. Hence, he adds a blue brick on the brick representing Monday.

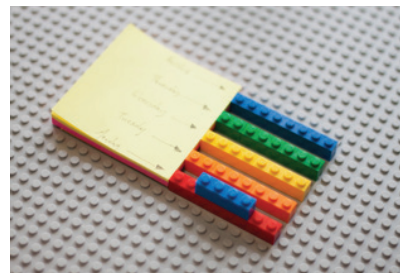


Figure 20. (Left) A day of work in the simulation; Figure 21. (Middle) Two days of work; Figure 22. (Right) A week of work.

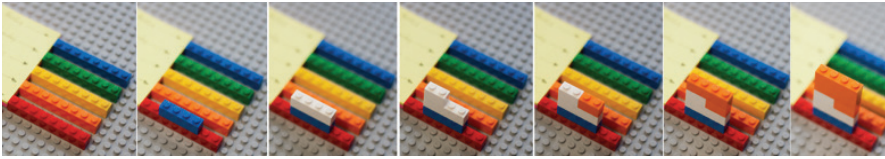


Figure 23. The simulation of a day of work on Michael's Lego time management system.

This is how Michael tracks his time for Monday. Figure 23 reveals the actual version of Michael time management tool. You can get more information on his blog at the following URL: <http://goo.gl/Qz554q>.

## Summary

We have introduced constructive visualization as a new paradigm, which can help realize the democratization of information visualization. We disclosed our historical and theoretical inspirations for its conceptualization, and presented the components and process of constructive visualization. Lastly, we presented two case studies of real-life examples, emphasizing their constructive processes and components.

Most paradigms of visualization creation focus first on creating data representation, and then developing interaction to suit data needs and tasks. The basic approach for constructive visualization is different. The focus is on creating an interactive environment where people can assemble, from modular data-linked units, visualizations that directly fit their needs.

This paradigm reveals new perspectives on the visualization design process, calling for:

1. New sets of possible design and experimental studies,
2. The development of guidelines for designing constructive visualization environments,
3. And lastly, the creation of new tools for supporting constructive visualization.

## Part 2. Constructive Visualization: A Study

### Introduction

The authoring of information visualizations by a wide audience has been identified as a major challenge by several researchers. For instance, in 2006 Johnson et al. declared in a NSF visualization research report that "the goal is to make visualization a ubiquitous tool that enables ordinary folks to think visually in everyday activities" (Chris Johnson, Robert Moorhead, Tamara Munzner, Hanspeter Pfister, Penny Rheingans 2006). In 2012, 6 years later, Heer and Shneiderman wrote that "novel interfaces for visualization specification are still needed. [...] New tools requiring little to no programming might place custom visualization design in the hands of

a broader audience” (Heer and Shneiderman 2012). Similarly, during his keynote at the conference IEEE VIS 2014, Alberto Cairo emphasized the importance of building tools for non-experts to create visualization (Cairo, 2014). These challenges were also raised by Brett Victor in his talk about drawing dynamic visualization (Victor, 2013). In this talk, Victor summarized visualization authoring in three approaches: use, draw and code. As a response to these challenges, we proposed a new design paradigm called constructive visualization in the previous part of this chapter.

However, all of these four approaches—using, drawing, coding and constructing visualization—are specific ways to process visual mapping. This process is an important element of the information visualization reference model (Jansen and Dragicevic, 2013; Stuart K. Card, Jock D. Mackinlay, 1999). The visual mapping defines the mapping of a dataset to a visual representation. While research has focused on finding perceptually efficient visual representations, the way humans perform visual mappings themselves, is still a black box that needs to be opened and explored.

Our goals are to explore:

- Whether novice people in InfoVis can construct their own visualizations using tokens.
- How these people are constructing their visualizations using certain materials.
- The types of visualizations they creating.

To investigate these questions we ran an exploratory study where we asked information visualization novices to create a visualization of a simple dataset using tangible tokens.

### **Study Design**

First, we recruited 12 participants from a variety of disciplines and educational backgrounds. We made a specific effort to not select InfoVis experts, avoiding people with backgrounds in visualization, human-computer interaction (HCI), and computer science domains in general. Thus, we did not study people in an HCI lab. Figure 24 summarizes the demographic information of participants using a Bertifier visualization (Perin, Dragicevic, and Fekete, 2014).

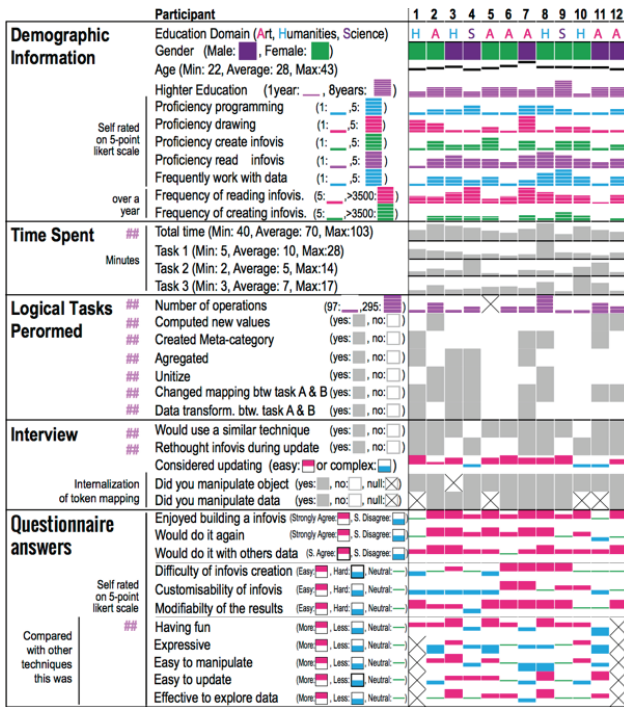


Figure 24. Visualization of the demographic distribution of our participant. This visualization was created with Bertifier (<http://www.bertifier.com>) and freely adapted to our needs.

## Setup

We invited the participants to sit in front of a desktop like the one described in Figure 25. The top of the desktop contained:

- #1 A printed dataset.
- #2 A box of tokens (token box).
- #3 A note suggesting participants to map a single token to 25 units.
- #4 A white canvas as the assembly environment for constructing a visualization.

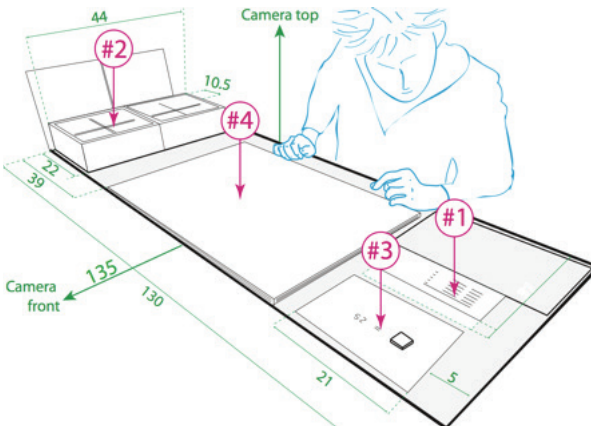


Figure 25. Setup of study.



## Dataset

We used an aggregated version of a bank account statement as our dataset. The participant saw three months of expenses on a single sheet of paper. All expenses were grouped into categories such as “amusement,” “bar and restaurants,” “groceries,” etc. To simplify the participants’ data processing, all values were rounded to the nearest 25. An update of the dataset containing 1 month of expenses (November) was provided during the experiment.

Month	Category	Amount
08 August	Amusement	125
08 August	Bars & Restaurants	100
08 August	Epiceries, courses	150
08 August	Transports	125
08 August	Voyage	150
09 September	Amusement	125
09 September	Bars & Restaurants	175
09 September	Epiceries, courses	150
09 September	Transports	200
09 September	Voyage	50
10 October	Amusement	50
10 October	Bars & Restaurants	150
10 October	Epiceries, courses	175
10 October	Transports	150
10 October	Voyage	125

Figure 26: Screenshot of the first three months in the dataset.

## Token Box

The tokens were 25mm wooden square tiles. There were six colours, with 36 tokens per colour. The tokens were contained inside two boxes with four compartments taped together on to the table. As seen on Figure 27, only six compartments contained tokens of different colours.



Figure 27. A photo of the token box, viewed from above.

## Tasks

We first asked participants, to create a visualization based on the given dataset (Figure 28). We then interviewed them after finishing the task. We then gave them a new dataset and asked them to update their visualization (Figure 29). Afterwards, we conducted a second interview. Lastly, we asked the participant to annotate their visualization such that another person would be able to understand it later (Figure 30).



Figure 28. Mosaic extract from the top camera of participant 1, during the task CREATE.



Figure 29. Mosaic extract from the top camera of participant 1, during the task UPDATE.



Figure 30. Mosaic extract from the top camera of participant 1, during the task ANNOTATE.

## 2. Results and Analysis

All participants were able to complete the three tasks in a short amount of time. They spent, on average, only 11 minutes to create, 6 minutes to update, and 7.5 minutes to annotate the visualization. As seen on Figure 31, while some participants simply recreated well-known visualizations such as bar charts, others developed unexpected diverse visual mappings. Most of the participants (10 out of 12) said that they would use a similar technique in the future. We also got surprised as one of them said that she had already used a similar approach with real coins to plan her future budget.

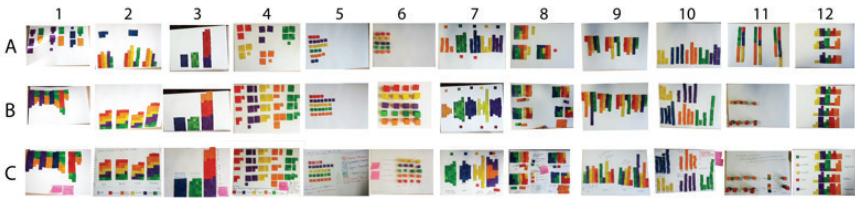


Figure 31. Mosaic of all the visualization produced by the participants (top numbers corresponding to their ID) during task A (create), B (update) and C (annotate).

### How Did They Do That?

To answer this question, we analysed the videos taken during the study using a qualitative data analysis approach. The coding of the video was performed through several passes in an iterative process. We identified 11 different subtasks, named after the logical task (WHAT), and grouped by their underlying goals (WHY). In Figure 32, we classified these actions into three categories: construction, computation, and storytelling. Each of these 11 tasks require several actions in different combinations and in different orders of execution. During the coding of the videos we observed a high diversity of actions committed by the participants. This diversity indicates that while people used the same actions, they did not adhere to the same sequence.

	<b>Why</b> <i>What (logical task)</i>	<b>How</b> <i>(mental and physical actions)</i>
<b>Construction</b>	1. Load data	READ, COMPUTE, SELECT COLOR, GRASP, CREATE
	2. Build constructs	ORGANIZE, MOVE
	3. Combine constructs	ARRANGE, ALIGN
	4. Extend	READ, COMPUTE, SELECT COLOR, GRASP, CREATE, ORGANIZE, MOVE, ARRANGE, ALIGN
	5. Correct	INCREASE, DECREASE, REMOVE
<b>Computation</b>	6. Categorize	SELECT COLOR, ARRANGE, MERGE, SPLIT
	7. Aggregate	MOVE, MERGE
	8. Compute New Value	SPLIT, COMPUTE + LOAD
	9. Unitize	ORGANIZE, ARRANGE, SPLIT, MERGE
<b>Storytelling</b>	10. Highlighting	SPLIT (temporarily)
	11. Marking	CREATE, SELECT COLOR

Figure 32. Summary of the logical, mental and physical tasks.

### Analysis

As seen in our results, the participants' process of constructing visualizations is pretty chaotic. However, we summarize the most common relationships between the subtasks in this flow diagram (Figure 33). The mental tasks are shown as purple circles and the physical tasks as blue circles. The grey oblongs linking two circles represent possible co-occurring actions. Tasks that impact the assembly model are marked with red circles. The grey background rectangles illustrate the logical tasks.

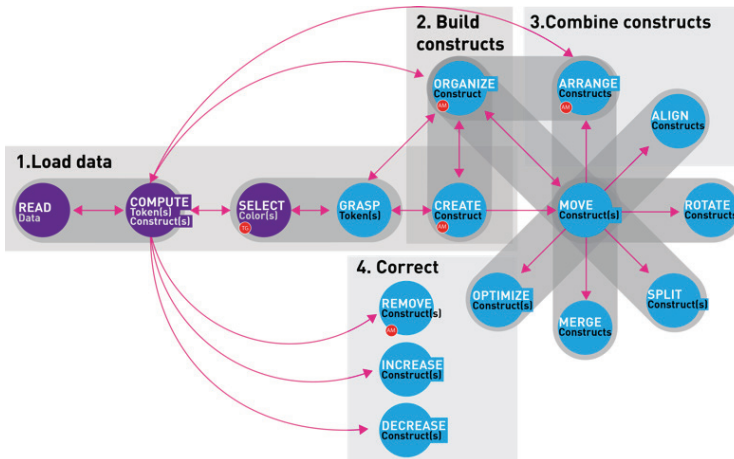


Figure 33. Flow diagram representing the different microtasks performed by participants. The arrows represent the most common paths taken between microtasks.

### In Detail...



Figure 34. Participant 4, (1) first loads the data on the canvas into tokens, (2) he then organizes the red tokens into squared constructs, and (3) he extends the organization he defined with the red tokens to all the others tokens.

*Load Data.* In Figure 34, the participant first reads the dataset, then computes the right number of tokens to grasp. These two actions are concurrent. He then selects the tokens colour and grasps the tokens. Then, he creates a construct, in this case, a heap of tokens, just after he repeats these subtasks for the next two months of the same category. These operations correspond to the logical task called loading data. By processing this subtask, the participant defines rules of assemblies that can be reused.



Figure 35. Part of the flow diagram concerning the logical task loading data.

*Extend (Load Data).* Extend refers to the task of applying existing rules of an assembly to other data cases. This logical task is illustrated by Figure 34. Between vignettes 2 and 3, the participant applies the assembly model he defined for the red tokens to all of the other tokens and the rest of the dataset.

*Build Constructs.* In Figure 34, we can see how the participant organizes

each heap into a different type of constructs. The participant is building squares to represent subunits for better readability. Two subtasks compose this building operation, create and organize, and are a part of the building construct task. Most of the time, the organize subtask is co-occurrent with the move subtask, and the create subtask with the organize subtask.

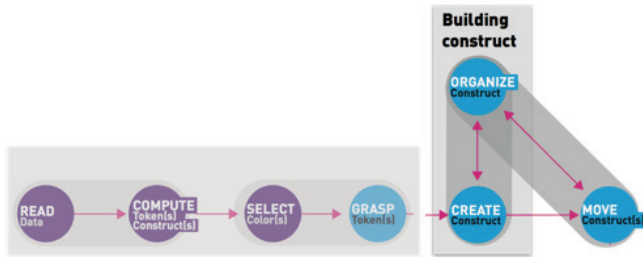


Figure 36. Part of the flow diagram concerning the logical task “Building a construct”.

*Organize.* Sometimes the organization of tokens into a specific construct happens in the hands of the participant, between the subtasks of grasping and creating. We can observe this with participant 7 who grasped some tokens with her two hands to organize the tokens into a 3d pile, and then placed it on the canvas.

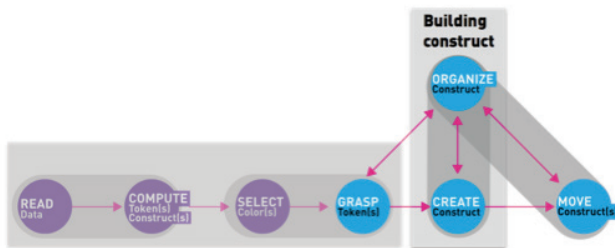


Figure 37. Part of the flow diagram concerning the subtask “Organize a construct”.

*Combine Construct.* In Figure 38, we can observe a participant that first merges a red sub-construct with another one. She then arranges the two columns on her right to be closer to the rest of the tokens after aligning the top of the two columns with the other construct. These three actions allow combining constructs.

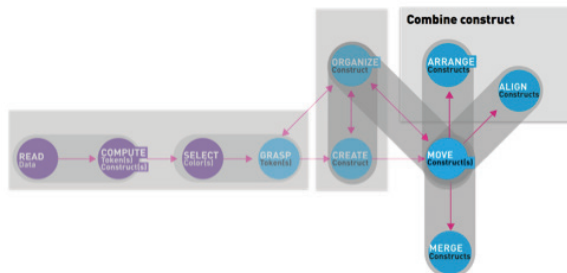


Figure 38. Part of the flow diagram concerning the action relative to the logical task “Combine a construct”.

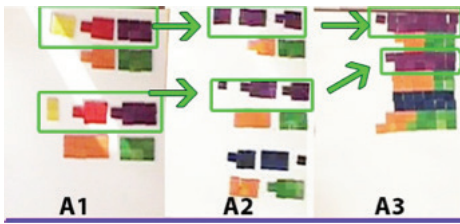


Figure 39. Participant # combined a construct over time. First (transformation A1 to A2), he changed the colour of three group of tokens (red and yellow) into purple, resulting in the aggregation of these three categories into one. Second (transformation A2 to A3), the participant combined all the token construct into one.

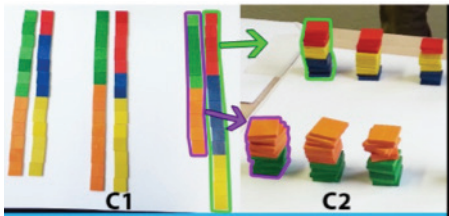


Figure 40. The participant changed the token construct between task A (C1) and task B (C2), while keeping the same colour coding over time. These two constructs represent the same data but with different spatial configurations: 2D and linear for (C1), and 3D and stacked for (C2).

### 3. Discussion

#### Bottom-Up vs Top-Down Procedure

All the participants had their personal going back and forth between different types of actions throughout the authoring process. However, we observe two distinct classes of procedures. The most common one (10 out of 12), which we call bottom-up procedure (Figure 41), consisted of participants starting from a simple data case, to progressively build higher level structures for an axis or a category. The second one is called top-down procedure (2 out of 12) (Figure 42). In this case, participants started by positioning higher-level structures such as the dimensions and axis and then populated them with data. Only two participants used it.

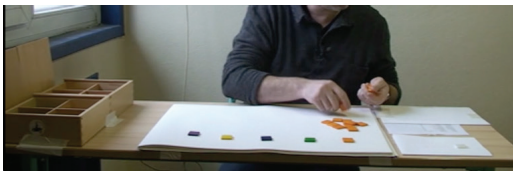


Figure 41. Participant 7 already defined the axis and colour coding before loading the data. He used a top-down approach, first defining the visualization model then rendering it. A full video of this participant is accessible online on <http://constructive.gforge.inria.fr/#!videosmd>.



Figure 42. Participant 9 first played with the tokens and then progressively constructed the visualization. She used a bottom-up approach, she defined the visualization model while she is constructing the visualization. A full video of this

participant is accessible online on  
<http://constructive.gforge.inria.fr/#!/videos.md>.

## How Information Visualization Novices Construct Visualizations

Although our study has inherent limitations, it is generalizable; our results suggest that creating constructive visualization environments in which people can assemble their own visualizations from tokens may be beneficial and merits further research. In 2010, Grammel, Tory and Storey conducted an inspiring study (Grammel, Tory, and Storey 2010) with similar goals but with a different setup, authoring tools, data complexity, and protocols. For this reason, the results of these two studies are different and complementing each other. In the previous study, Grammel identified different barriers relative to novices authoring of InfoVis, proper to their setup. In our setup, the major barrier we observed—and on which participants commented—was the initial transformation from the number printed on paper, to a number of tangible tokens.

## Internalization of Data to Token Mappings

We were interested in how far participants internalized the token mapping. A good example of token mapping internalization in our day-to-day life is the money. Do you think of a dollar bill as a piece of paper or as its value? Do you think a coin as a piece of metal or as the value it stands for? To investigate this further, we systematically asked our participant two questions:

(Q1) “What did you manipulate during your construction process?”  
and depending on their answer: (Q2) “What was the value (or meaning) of [the declared object in Q1]?”



Figure 43. (Left) A Canadian one dollar coin – Credit: Kevin Dooley Some rights reserved; Figure 44. (Middle) One and two Euro coins isolated on a white background. – Credit: Image of Percy Some right reserved; Figure 45. (Right) An American one dollar bill Credit: Thierry Ehrmann – Some right reserved.

They replied:

- A. Half of them (6 out of 12) replied to Q1 by referencing the object first, then the data.
- B. Four other participants spoke only about the object
- C. Two replied with only the data or the data first and then the object.

This result suggests that our participants have a clear awareness of the coupling between the data and their tangible proxy.



Figure 46. Pie chart of the replies. Blue for reply A, Red for reply b, and green for reply C.

## 4. Implications

### Exploiting Processing FLUENCY

Processing FLUENCY have been previously defined as “the subjective experience of ease with which people process information” (Alter and Oppenheimer 2009). The method we provided to construct visualization was originally designed 200 years ago to teach mathematics to non-literate kindergarten children. The result of this study showed that people without specific skills in InfoVis can construct useful visualization, when they use a method with for which they already possess fluency. This implication opens some questions such as:

- Will a constructive authoring tool implemented in a digital environment provide the same benefits as the tangible version?
- How can we transform more complex InfoVis techniques into more fluent ones?



Figure 47. A young kid constructing a bar chart with building blocks is doing math operations.

### Tangible Constructive Design

Participants criticized several aspect of the wooden tiles, for instance they accidentally destroyed parts of their construction during moving actions. This could be addressed by using other materials. For instance, Lego bricks, or materials with programmable properties (Figure 48). This raises some question such as:

- Which material properties are better for supporting constructive strategies?
- Which material properties are most effective?
- How does the complexity of programmable properties affect people’s proficiency with such environment?





Figure 48. From left to right - First pictures: A visualization made with Lego bricks, Credit General Motors; an extract of the video "Claytronics - Physical Dynamic Rendering" <https://goo.gl/tgxtrB>.

## Summary

In summary, this study has several contributions. First, we demonstrated that visualization novices are capable of creating meaningful visualizations in a short period of time in a tangible, constructive environment. Second, we opened the "black box" of "visual mapping" to present a first preliminary model. Third, we revealed many processes internal to this step, and presented in a model. This study is also an empirical proof that supports the new design paradigm we presented in the first part of this chapter. We expect that such research will help researchers and designers to create tools, which support visualization non-experts in their future activities.

## Conclusions

In this chapter, we presented a new design paradigm and an empirical study of this paradigm. This paradigm is particularly suitable for information visualization novices as it addresses the following design challenges: simplicity, dynamicity, and expressivity. We first defined this paradigm by presenting its underlying components and processes, as well as our historical inspirations for its conception.

To empirically explore this paradigm, we designed a study in which we asked information visualization non-experts to construct a visualization using this approach. The results of the study confirmed our hypothesis: in a constructive environment, information visualization non-experts can create, update and annotate visualizations within a short period of time. Moreover, these results allowed us to investigate how people perform visual mapping—a phenomenon that has not been studied before. We presented a preliminary model of constructive visualization, making it easier for the research community to investigate and support this process for a wide range of visualizations. We finished by presenting the implications that can lead to future research and design.



# An Approach to Automated GUI Testing

*Theodore D. Hellmann and Frank Maurer*

## Introduction

Automated software tests are a foundation of effective software development. By providing evidence for the correctness of development work that has been done on a project, tests allow teams to confidently, quickly, and repeatedly release software to end-users. However – inconveniently – the type of software testing that most directly reflects how users will interact with a system, graphical user interface (GUI) testing, is also the least stable for use in automated testing environments. In this book chapter, we discuss our approach to making automated GUI tests (AGTs) more compatible with use in automated software testing.

GUIs are part of most modern software for the simple reason that they make it easier for people to interact with a system – assuming that both the system and its GUI are functioning correctly. It makes intuitive sense, then, to simply test a system through its GUI to verify the correctness of both simultaneously, but testing a system through its GUI is much more difficult than testing a system’s code directly. AGTs work by interacting with the system in a way similar to how users would interact with the system. An AGT needs to locate a specific part of the application on-screen, interact with it by triggering actions that a user could perform – like mouse clicks and keyboard input – and then evaluate correctness based on what the GUI displays onscreen as a result of these interactions. These actions are very easy for a human to perform, but very difficult for an automated algorithm.

Creating and maintaining AGTs is very time consuming and expensive. Every time the GUI is revised, AGTs that touch on these revisions may need to be updated. It’s very likely that the GUI will need to change over time, since this is the portion of the software that users interact with directly, and their feedback will necessitate changes. Each change comes with a risk of causing test failures – either due to changing the functionality of the system or by changing the way in which tests need to interact with the GUI – and the later causes overhead over and above normal test maintenance. AGTs essentially navigate through a system in order to reach the functionality they need to test, and any changes to the GUI that change this navigation will

cause test failures (see: Figure 1).

Even given the added cost, AGTs are still useful for evaluating whether a system is correct from the point of view of its users because the test interacts with the system the way a human user would. If an application is not tested in the same way in which it will be used, errors that users are able to trigger may slip through the testing process unnoticed and be released to end users. Further, performing GUI testing on an application manually would be unfeasibly slow and labor intensive. Based on this, we chose to investigate how AGTs are used in practice and propose a methodology for integrating AGTs into a software development process.

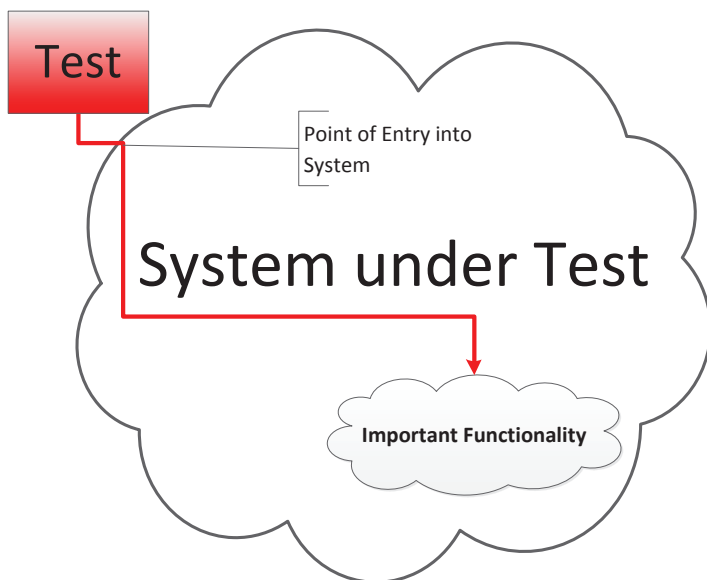


Figure 1. Visualization of an AGT navigating through the GUI of a system to reach important functionality. Imagine the red line as a sequence of interactions with the GUI leading to the important functionality. Any changes in this navigation will cause the test to fail.

In this chapter, we discuss motivations behind the use of automated GUI tests in practice. Based on the results of this, we describe our methodology for integrating automated GUI testing into an Agile development process. Related work is discussed at the end.

### **Motivation for Automated GUI Testing**

Researchers investigating GUI testing have been looking into solutions for the various issues with GUI testing for many years and have repeatedly described potential solutions to common issues with GUI testing (Hellmann, 2015). Despite this, these solutions have not seem to catch on in software development organizations. In order to make our solution more likely to actually benefit practitioners, we began our study by actually interviewing practitioners in order to understand how to tailor a solution to meet their

needs. We specifically focused on investigating, which uses of AGTs were most important to practitioners and which issues prevented them from actually benefiting from GUI testing. The results of these interviews are presented here as motivation for what a successful approach to AGTs must provide and can be used to evaluate how successful our approach is.

Semi-structured interviews were conducted with 18 participants who had some amount of experience in the use of tools for the creation of AGTs. However, we quickly noticed that participants with more limited experience (less than one year) tended to focus on issues with specific tools rather than with the process of GUI testing generally. Of course, this means that one of the findings of our study was that the usability of many AGT frameworks is a major barrier to adoption, but we wished to come up with a more general perspective to AGTs not bound to specific tools. In order to focus on an approach that is tool-agnostic, we focused our analysis on the 8 participants with more than one year of experience. Full details on the study can be found in the full report (Hellmann, et al., 2014). For the present discussion, we provide a brief overview of results.

Our interviews showed that practitioners use AGTs primarily for two purposes: acceptance testing and regression testing. Automated acceptance tests are traditionally created in collaboration with the customer as an encapsulation of expectations about how a feature should work. Automated acceptance tests take the form of tests that operate at the system level to demonstrate that a feature is working from the point of view of the system's target users – the use case described in the introduction of this chapter. Further, in Agile development settings, acceptance tests tend to be written before development work on a feature has begun. This is because acceptance tests serve as a contract between technical team members and their customers: when the acceptance tests for a feature pass, the feature should be working. For our present investigation, then, we will make this a requirement for AGTs: *AGTs must be creatable before the system they are testing exists so that they can be used as a specification of correctness.*

Regression testing is the process of running the same test suite repeatedly against a system over time to determine if it is at least as correct as the last time tests were run. If at any point in time a test that was previously passing begins to fail, this indicates that a regression error has been introduced into the system. One of the crucial points is the rate of false positive results – cases where the tests identify a failure in the system when none exists. When this sort of failure occurs frequently, developers begin to wonder first if the test itself is to blame rather than what regression error the failure represents. From this, we draw our second requirement for a new approach to AGTs: *AGTs must have a low rate of false positive test failures so that developers trust the results of test runs.*

In order to make both of these goals possible, we propose an integration of the creation and evaluation of AGTs into another process that takes place

before development and which is geared towards reducing the amount of change required during the development process: user interface design. GUI testing and UI design go together nicely since both processes are aimed at verifying that the GUI is correct from the perspective of end-users. Additionally, one can expect a collaboration between testers and designers early in the process of creating a system to come up with GUIs that are more verifiable, usable, testable, and stable than when those groups work independently.

### **Driving GUI Development with Tests and Design**

In short, we propose a process for systems development in which:

1. Interactive low-fidelity prototypes are created by designers
2. Usability evaluation is carried out with end-users using the prototypes
3. AGTs are created using the prototypes (potentially in collaboration with end-users in the spirit of acceptance testing)
4. AGTs can be run against the prototypes to demonstrate functionality
5. AGTs, with slight modification, can be run against the actual GUI as it is developed

Based on the requirements outlined in the previous section, we conceived and evaluated a way to integrate the creation and use of AGTs into a design and development process in order to promote their use by practitioners as both acceptance tests and regression tests. By encouraging collaboration between testers and designers, we hope to increase the testability of GUIs early on by including testers as stakeholders in the design process. By encouraging usability testing early in the design process, we hope to increase the stability of the resulting GUIs after development, which would also decrease the amount of maintenance effort that would need to be expended on AGTs over time. Overall, this is expected to increase the use of AGTs both as acceptance tests and as reliable regression tests – thus addressing both issues raised in our interview study.

This process could be referred to as “graphical user interface design-driven testing and test-driven development” but for simplicity we will use the acronym UITDD.

### **Prerequisites for UITDD**

Before UITDD can begin the project vision must be sufficiently clear. Because UITDD will involve some up-front design – normally a practice that is avoided in Agile software development – we will need to ensure that this work is aimed towards the actual problem the project is intended to solve. We recommend that the team carries out project visioning exercises to ensure that the whole team understands what the goal of the project should be and understands what a successful project would be able to provide end-users with.

Building upon a solid project vision, requirements elicitation should be performed in order to understand what features will be necessary to enable end-users to accomplish the goals laid out in the project vision. This doesn't need to be done in exacting detail – as would be common in traditional waterfall-style projects – but the main features of the system do need to be clear enough to understand the prioritization of different features.

Based on the prioritization of the major features of the project, two things can happen. First, a project roadmap, release plan, user story map, or similar product planning tool can be developed in order to understand what the tentative timeline for the project as a whole will look like. It's important to have this understanding in place because, without it, there's too much risk of building the wrong system. However, this should be understood as a living document that can be easily changed throughout the project, as the process of UITDD is designed to encourage learning about the project and to understand changes users will need throughout the process.

Once this high-level understanding of the project as a whole exists, UITDD proper can begin. The integration of design, testing, and development helps to ensure we are building a high quality system.

### **Creation of Low-Fidelity Prototypes**

Requirements elicitation focus in on the details of the features that will be developed in the next few iterations. To assist with this process, storyboards of prototypes should be used. Storyboards are sequences of sketches of the user interface of a system that demonstrate how it would react to different input from the user. Prototypes can be as simple as pen-and-paper sketches (low-fidelity prototypes) or as complicated as sequences of linked web pages (high-fidelity prototypes), but the key is that they should not look like a finished system or take a lot of effort to create or change.

The purpose of low-fidelity prototypes is to evaluate whether the interactions they describe are good ways of helping users accomplish functionality in the system, and if users are presented with prototypes that actually look like a finished system, this can encourage feedback about how the system looks rather than about how the system functions.

Designers should cast a wide net with initial prototypes and create a wide variety of interaction methods before narrowing in on a small number to move forward with. Once the customers – in collaboration with the designers – have decided on a prototype to move forward with, the prototype could be translated from a paper-based low-fidelity prototype into an interactive prototype, such as Moqups ([moqups.com](http://moqups.com)), Pencil Project ([pencil.evolus.vn](http://pencil.evolus.vn)), SketchFlow ([microsoft.com/silverlight/sketchflow](http://microsoft.com/silverlight/sketchflow)), or any other of the number of similar existing tools. An interactive prototype is for our purposes better than a paper prototype for three main reasons. First, it will allow users to interact with the prototype on a computer, in the same way in which they will interact with the actual product, and thus encourage them

to take usability evaluations more seriously. Second, it will force designers to understand the complexity of the user interface they will be asking developers to create and thus understand how difficult their designs will be to realize. Third, it will result in the creation of a digital representation of the system – such as a webpage linking together different pages of the prototype – to be developed which can be used for the creation of AGTs.

### **Usability Evaluation with End-Users**

Once the prototype exists, it can be used for usability evaluation (Buxton, 2007) (Barnum, 2002). Usability evaluation can either be done with experts – heuristic evaluation where experts look over the system for well-understood, general usability issues – or with end-users. Methods like heuristic evaluation tend to identify minor problems (at least when skilled designers created the prototypes), so we recommend performing evaluations with end-users (Jeffries, et al., 1992). Working with actual using techniques like Wizard of Oz evaluation tends to be more valuable in terms of discovering issues which would have necessitated changes to the system down the line (Jeffries, et al., 1992). We recommend creating high-fidelity prototypes for these evaluations as without an interactive prototype the results can be inconsistent due to fatigue on the person running the evaluation, making results difficult to compare between participants (Bernsen, et al., 1994).

However, with a digital, interactive version of the prototype, issues with the reliability of the “wizard” are not an issue, since the wizard is replaced by a simple system linking different pages of the prototype to one another, as for example with ActiveStory: Enhanced (Hosseini-Khayat, et al., 2010). Additionally, this sort of usability testing can be performed with a much larger number of users, since the prototypes can be posted online to collect statistical information about the way users interact with the system in order to allow designers to focus in on the more important issues with the system (Hosseini-Khayat, 2010). And, as mentioned earlier, the digital representation of the prototype as a series of linked web pages will make it very easy to interact with the prototype for the purpose of creating tests.

At this point in time, in parallel with the evaluation of the system with real end-users, designers should also consult with testers and developers. Testers will need to give input on the general testability of the design while developers will need to do the same with implementation details. Again, it's easy to design a system that can end up being technically difficult to test or implement.

### **Create AGTs From Prototype**

One of the most important side-effects of creating a digital, interactive prototype is that capture-replay tools (CRTs) will be able to pick up interactions with this type of interface. CRTs work by monitoring a user's interactions with a GUI and recording them in sequence so that they can later be replayed automatically against the system. Normally, this is performed against an existing system for the purpose of regression testing:

if the system is working when a test is recorded, any changes to the system that break the replay of the recording will indicate a problem with the system. Examples of common CRTs include Visual Studio's Coded UI Tests ([msdn.microsoft.com/en-us/library/dd286726.aspx](https://msdn.microsoft.com/en-us/library/dd286726.aspx)), Selenium ([seleniumhq.org](https://seleniumhq.org)), and Sikuli ([sikuli.org](https://sikuli.org)).

A CRT doesn't require any sort of specialist knowledge to use. This is extremely important because it means that non-technical customers can actually participate in the process of creating acceptance tests. Strictly speaking, customers should be the ones creating acceptance tests or acceptance criteria for functionality because they are the ones to actually approve/accept the software in the end, but in practice testers will usually create acceptance tests based on their understanding of the system functionality. With a CRT, customers themselves can create acceptance tests by simply interacting with a prototype and can verify that the test is behaving as they expect by watching the execution of a replay of the test.

There are other benefits to the use of a CRT in this scenario.

### **Demonstrating Functionality to Developers**

A recording of how a user wants to interact with functionality can be replayed against the prototype to demonstrate how interactions should work. This can make it a lot easier for developers to understand what they're being asked to implement because they can see a visual representation of functionality. In the experiment described in the following section, this was a very common use of these recorded tests.

### **Perform Test-Driven Development of the Actual System**

By modifying the tests in very simple ways, the way CRTs locate the elements they use in order to accomplish a test can be altered in order to target the same elements in a different interface. Most CRTs work by creating a dictionary of elements of the user interface so that they can be looked up simply during testing. This may be done either using a keyword – a unique name assigned to each element – or using a heuristic search – in short, looking for an element in a GUI that matches a certain set of criteria. Two things can be done to make use of this property of CRTs: either the dictionary can be altered so that the test will be able to find elements in the actual system's GUI that match the corresponding element in the prototype, or the actual GUI as it is developed can be created to match the information in the prototype. In reality, a combination of these two methods will probably be needed.

The benefit of synchronizing the lookup information in the prototype and the actual system is huge because it enables test-driven development (TDD): tests can be recorded from the prototype and then run against the actual system as it is developed in order to ensure that it meets all of the expectations expressed in the prototypes. This allows the developers to develop functionality, to be sure that the system they are developing is



functioning correctly at all points in time, and to gain other benefits of TDD (see, for example: (Jeffries, et al., 2007)).

### Iterate and Increment

Of course, as with any Agile process, this approach to design, testing, and development is intended to be both iterative and incremental. The extra design and testing work are not intended to lead to a completely finished feature in order to prevent changes, but rather to lead directly to the implementation of releasable functionality early in production and to encourage changes to the system early in the process, when they will be less expensive to undertake. Design should not be done for the system as a whole – with the exception of perhaps overall interaction themes – but rather for each feature. As design work on a feature finishes and testing and development begin, the design team would move on to working on the next feature – but always focus on single features at a time. In this way, the system is built up incrementally, one feature at a time.

An example of this general process follows in the next section and a controlled experiment of UITDD is presented in the section afterwards.

### An Example of Testing Portion of UITDD in Action

In order to demonstrate the process for UITDD, assume we intend to develop a calculator application. Full results of this example were first reported in (Hellmann, et al., 2010). For now, we are focusing on basic addition functionality, as this is the functionality that our hypothetical users are most interested in. In order to demonstrate how addition would work, we come up with a paper prototype which might look something like Figure 2.

			⊗
←	C	.	*
7	8	9	÷
4	5	6	-
1	2	3	+
⊗			=

Figure 2. Prototype for calculator interface.

After performing Wizard of Oz style evaluation with potential end-users, the designers proceed to make a digital version of the prototype. This digital version is based around a use case scenario for the functionality, titled “adding five and nine should result in fourteen” and with a corresponding set of steps for users to follow in evaluations. This digital version was made, as a proof-of-concept, using ActiveStory: Enhanced. Figure 3 shows several

pages of the prototype after interactions with a user. The yellow areas show elements of the prototype that are selected by the user, while the arrows show transitions between these pages as a result of these interactions. So, by clicking on various parts of the prototype, the user will be presented with different pages so that it looks as though s/he is interacting with a functioning system.

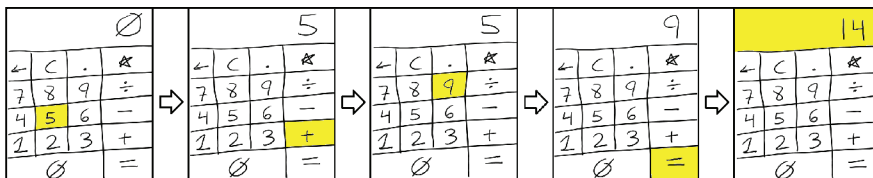


Figure 3. Storyboard of a test sequence. Highlighted areas represent mouse clicks in the first four states and the field to be verified in the last state.

Now that the prototype exists and we have a scenario as a basis for testing, testers can record a test demonstrating our “adding five and nine should result in fourteen” scenario. This test can be replayed against the prototype to demonstrate to developers how the actual calculator should behave, and it can also be run against the actual calculator as the developers implement this addition functionality to see if they are done with development. In this example, LEET (leet.codeplex.com) was used as the CRT for testing. The test ended up looking like Figure 4.

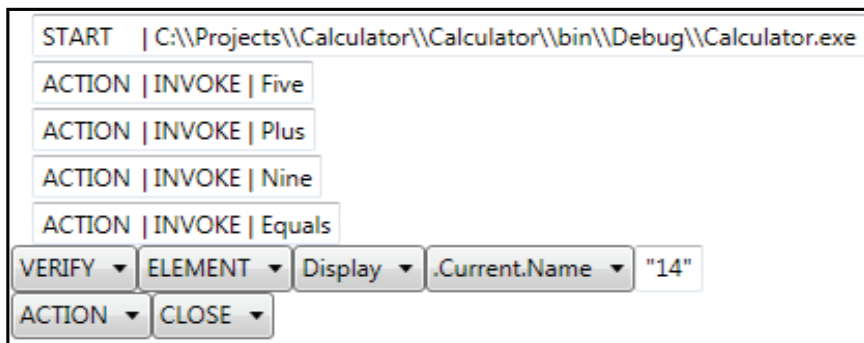


Figure 4. Test for the calculator’s simple addition use case scenario. Each line represents a single interaction with the GUI.

Because LEET works with keyword-based UI element lookup, during development all that was necessary to make these tests work against the actual system was to make sure that their AutomationID – their unique keyword – matched those of the corresponding element in the prototype. Due to the way LEET works, now all that was needed was to change the START command to start the actual application instead of the interactive prototype. Development can now take place on the actual GUI, with the developer able to run the test repeatedly throughout development to see if the application is meeting the specifications expressed in the test.



Figure 5. A complete interface. The original test still passes.

In order to see if this approach was usable by people who actually do GUI development, a controlled pilot experiment of this methodology was conducted with three developers.

View Totals		New Report		Modify Report	
Entity:	John Smith/1				
Name:	John Smith				
ID:	1				
Transport:	750				
Lodging:	500				
Meals:	150				
Conference Fees:	100				
Subtotal:	1500				
Paid:	<input type="text" value="0"/>				
Owing:	1500				
<input type="button" value="Update Report"/>					

Figure 6. One page from the SketchFlow prototype of ExpenseManager.

### Controlled Pilot Experiment with GUI Developers

In this pilot study, three developers were given an interactive prototype, tests that ran against the prototype, a GUI with no logic implemented, and tests that ran against the GUI and asked to implement logic for the GUI. The GUI was for an application, ExpenseManager, that, when completed, would include functionality for entering expense reports, clearing expense

reports, saving expense reports, modifying expense reports, and viewing totals for all saved expense reports. Full results of this pilot experiment are reported in (Hellmann, et al., 2011).

In this pilot evaluation, the tests were still created in LEET. However, the interactive prototypes were created using SketchFlow.

Participants were given one hour in which to implement four features: clearing entries, modifying saved entries, saving entries, and ensuring that multiple entries could be saved. Completeness of implementation was based on whether the tests for those features passed. All participants were able to implement functionality for clearing and saving reports. No participants were able to implement functionality for modifying saved reports, and only one participant was able to implement functionality for saving multiple reports. Based on these results, it would seem that it was possible for UITDD to be used to develop some features, but, as there was no comparative sample group, we cannot conclude that it is better than other development methods.

Participants also completed a post-experiment interview to evaluate how useful they felt UITDD would be in practice. All participants indicated that they found UITDD to be at least “useful” (3 on a scale of 1 to 5) and indicated that they were interested in seeing if they could apply the technique to their own work. Again, full results of this pilot evaluation are reported in (Hellmann, et al., 2011).

### **Concluding Remarks**

This chapter presented an overview of an approach to integrating design, testing, and development of GUIs in order to promote better customer communication, better collaboration between testers and developers, and meet the needs of real-world practitioners in terms of providing a way to conduct acceptance testing and regression testing using AGTs. Additionally, we provided a method that makes it possible to make use of CRTs in this process, which would make it easier to integrate customers and end-users into the testing process. The method also makes it possible to perform test-driven development of GUIs.

Interestingly, if a team is already creating prototypes, conducting usability evaluations early in the design process, and using AGTs as part of their test process, this methodology won't require many changes to their design, test, and development processes. Because of this, we believe that it has potential for adoption in practice.



## **Agile Product Line Engineering Case Study: Vertical & Horizontal Displays**

*Yaser Ghanam and Frank Maurer*

### **Introduction**

Variability management plays an important role in defining and handling the parts of the system that may vary. This is often needed when a number of similar – yet not identical – systems are to be derived from a common platform to satisfy different needs. This software paradigm is called Software Product Line (SPL) engineering (Clements, P., and Northrop, L., 2001). Companies consistently report that SPLs yield significant improvements. Some reported reductions in the number of defects in their products and cuts in costs and time-to-market by a factor of 10 or more (Schmid, K., and Verlage, M., 2002). Commonality between systems is what makes SPLs economically effective; whereas variability is what makes mass customization possible. SPLs deal with similar systems as a family of products sharing a library of core assets. But since customer requirements are rarely exactly the same, shared assets have to accommodate a certain degree of variability. For instance, the customer of an intelligent home system should be able to choose a subset of components that fulfills her wants. It should also be possible for customers to tailor certain aspects of these components to meet their specific needs. A security module, for example, offers different techniques to secure access control such as PIN protected locks, access by magnet cards and finger print authentication. When choosing to have a security system component, customers may select one or more of these options.

### **Problem Statement**

Traditionally in SPL engineering, variability analysis is conducted upfront during a phase called domain engineering. A comprehensive analysis is conducted to specify the commonalities and variations in the prospective SPL. Commonality and variability analysis is concerned with determining the requirements of the members of the software family, and defining how these requirements may vary. This includes determining all sources of variation (i.e. variation points) as well as the allowed values (i.e. variants). After the domain engineering phase comes the application engineering phase. As a starting point, application engineers use the reference architecture, the

reusable artifacts, and the variability profile – that were all defined in the domain engineering phase. Based on the specific requirements of a certain product, application engineers make decisions on what variants should be selected for each variation point. The outcome of this phase is an instance of the system that represents a specific product. Ideally, application engineers should provide feedback to domain engineers pertaining to problems and limitations of the current architecture or variability definition.

For agile organizations, the focus has been to develop software systems that satisfy their current customer base, without worrying about best practices to handle variations of requirements in the future. Recently, the agile community has been investigating ways to scale agile up to the enterprise level rather than the team level like in (Leffingwell, D., 2007) and (Shalloway, A., Beaver, G., and Trott, J., 2009). This will eventually require that agile organizations find a way to adopt SPL practices to manage variability in customer requirements in a more effective way. However, adopting SPL practices in their traditional form is challenging. For one, agile organizations foster a culture of minimalism in upfront investment and process overhead including documentation. This is in direct conflict with traditional approaches to SPL engineering where a whole phase, namely domain engineering, is dedicated for domain and requirement analysis upfront. Moreover, especially during domain engineering, documentation is deemed essential to communicate knowledge to application engineers. Secondly, agile organizations depend heavily on fast delivery as a mechanism for quick customer satisfaction and feedback, which is too difficult to achieve when a domain engineering phase is to occur before delivering any products. Thirdly, the flexibility to accommodate changes in requirements and new customer requests is an important characteristic of agile teams. This characteristic will be compromised if two separate processes – namely domain engineering and application engineering – are introduced, because it may slow down the feedback loops between teams.

## **Goal**

Our goal is to reconcile conflicts between traditional SPL engineering and agile software development. We argue that for agile organizations to adopt a SPL approach, a reactive – as opposed to proactive – framework is more befitting. This chapter presents a framework that shall allow agile organizations to incrementally and reactively construct variability profiles for existing and new systems. The framework leverages common agile practices such as iterative software development, refactoring, continuous integration and testing to introduce variability into systems only when it is needed.

The rest of this chapter will be structured as follows. First we review related literature. Then, we describe the proposed approach. After that, we evaluate our approach using a case study of a real experience. Finally, we discuss the advantages and limitations of our approach.

## **Incremental and Reactive Variability Management**

In our research we stress that for an approach to fit well with agile principles and practices, being incremental and reactive is key. By “incremental”, we exclude big-bang transitional approaches. And by “reactive”, we exclude proactive approaches in which a great amount of upfront speculation is required. The quest for an incremental and reactive approach to establishing and managing product lines is a relatively new phenomenon. For one, organizations did not want to throw away their investments in legacy systems and start all over again. Also, for many organizations the transition to systematically managed variability in their systems was too big a change if they were to follow the strict domain-then-application engineering model.

Kruger (Kruger, C., 2002) contributed ideas and commercialized a tool to ease the transition to software mass customization. The main idea is that domain engineering and application engineering should not be separate. Their tool utilizes the concept of separation of concerns to realize variability in software systems. The tool is closed source and not available for academic evaluation. Reactive approaches, with the support of tools like the one in (Kruger, C., 2002) has been reported to require orders of magnitude less effort compared to proactive approaches (Buhrdorf, R., Churchett, D., and Krueger, C., 2003). Clegg et al. (Clegg, K., Kelly, T., McDermid, J., 2002) proposed a method to incrementally build a SPL architecture in an object-orientated environment. The method provides useful insight into realizing variability in an incremental manner, but does not discuss how to communicate variability from the requirement engineering phase to the realization phase. The aim of our work is somewhat similar to the abovementioned efforts. However, we differ in that we are not only concerned with realizing variability in a system. Rather, we are interested in the process of managing variability as it evolves in an agile context, as will be detailed later.

## **Agile Product Line Engineering**

Agility in product lines is a fairly new area of research. In 2006, the 1st workshop on agile product line engineering was held as part of the 10th international SPL conference (Cooper, K., and Franch, X., 2006). The workshop aimed at bringing researchers from the agile community and the SPL community to discuss commonalities and points of variation between the two practices. The theme of the discussions in that workshop was around how feasible it is to integrate the two approaches. One of the presented efforts was the iterative approach proposed by Carbon et al. (Carbon, R., Lindvall, M., Muthig, D., and Costa, P., 2006). This approach is based on PuLSE-I (Bayer, J., Gacek, C., Muthig, D., and Widen, T., 2000) which is a reuse-centric application engineering process. The proposed approach gives agile methods the role of tailoring a product for a specific customer during the application engineering process. The approach does not discuss the role of agile methods in the domain engineering phase. In a different venue, Hanssen et al. (Hanssen, G., and Fægri, T., 2008) described how SPL techniques can be used at the strategic level of the organization, while agile

software development can be used at the medium-term project level. Also, Paige et al. (Paige, R., Xiaochen, W., Stephenson, Z., and Phillip J., 2006) proposed building SPLs using Feature Driven Development. They assert the method worked well when giving special considerations for the product line architectural and component design. While these efforts are interesting attempts to combine concepts from agile software development and SPL engineering, their goal is different from that of our research. While their goal is to find ways to introduce or enhance agility in existing SPLs, our goal is to enable agile organizations to incrementally and reactively build and manage SPLs by adopting frameworks that align well with agile principles and practices. Our goal goes hand in hand with the recommendations of McGregor (McGregor, J., 2008) who presented an interesting theoretical attempt to reconstruct a hybrid method. He concluded that competing philosophies of the two software paradigms make their integration difficult. But he asserts that the two can be tailored under the condition that both should retain their basic characteristics. In our research, we try to tailor variability management to fit within an agile context such that the advantageous characteristics of SPL practices are attained and the agility of software development is not deteriorated.

## **THE PROPOSED APPROACH**

This section will present the proposed approach to manage variability in a reactive manner using agile practices. The recommended process involves a number of steps, namely: eliciting new requirements, conducting a variability analysis, updating the variability profile, refactoring the architecture, running the tests, realizing the new requirements, and finally running the tests once again. This is an iterative process that repeats whenever new requirements are available. Each one of the steps is discussed in detail in the following subsections.

### **A. Eliciting New Requirements**

This is the first natural step in any software development process. Traditionally – and especially in the case of SPL engineering – this is a fairly heavyweight process, because it involves domain analyses to predict what requirements may be needed in the future. In agile software development, it is sufficient to get only the available set of requirements and divide them into work items that can be achieved in 2- to 4-week iterations. Speculation is to be avoided as much as possible. In our approach, we adopt the agile way of requirement elicitation. We also use a customer-driven elicitation process. This means that unless something is actually requested (or needed) by a known customer, we do not invest into incorporating it in the product line/application system.

### **B. Variability Analysis**

Variability analysis is traditionally conducted upfront in the domain engineering phase. Elicited requirements are analyzed in terms of what they share in common, and in what aspects they may vary. Sources of variations are determined, and they are called variation points. The allowed values



for these variation points are also determined, and they are called variants. In our approach, we avoid a one-shot upfront variability analysis, simply because it does not fit within the iterative nature of requirement elicitation in agile methods. Rather, we conduct a variability analysis every iteration between the current requirements in the system and the newly elicited requirements.

During variability analysis, we use lightweight techniques to determine the commonalities and variations between the new requirements and the existing ones. Although we do not specify a certain technique to conduct this analysis, we recommend the use of a simple issue-implication table that lists all the issues that may cause variability in the system, and their implications in terms of variability. In each iteration, the expected outcome of this step is a list of changes to the variability profile. This includes new variation points, new variants for existing variation points, and new abstraction of common aspects. In Section IV, we use a case study to illustrate in detail how this is done in a real setting.

### **C. Updating the Variability Profile**

By variability profile we refer to the list of all variation points in the system and their variants. They are usually expressed in a formal representation or using a feature model (Kang, K., Cohen, S., Hess, J., Novak, W., and Peterson, A., 1990). In this chapter, we use this simple notation to illustrate the idea:

*Variation Point X = {Variant A, Variant B}*  
*Variant A = [feature1, feature2, feature3]*  
*Variant B = [feature1', feature2', feature3']*  
*Where: {} implies OR grouping; [] implies AND grouping.*

After the variability analysis step in each iteration, we update the variability profile with any new variation points or variants arising due to the new requirements (in cases where there are no changes to the variability in the system, we may not need to do that). It is important to keep a variability profile for the system to ensure that all aspects of variability are traceable to code artifacts and that they are communicated well to all stakeholders through and after the development process. Variability profiles are also used to explicate any dependencies and constraints between variation points and variants. In (Ghanam, Y., and Maurer, F., 2010), we explain in great detail how to maintain variability profiles using feature models and executable acceptance tests.

### **D. Refactoring the Architecture**

Using the refactoring techniques, the architecture has to be refactored in order to accommodate the new variability. For example, new architecture layers can be introduced to abstract common aspects, and other layers can be specialized to handle variable aspects. It is important to note that the goal of this step is to refactor the architecture to be ready to accommodate

the new version of the variability profile, and not to realize this variability. The actual realization of that variability happens at a later step. For example, suppose a feature *x* existed in the system before the current iteration. If feature *y* in the new requirements is just another variation of feature *x*, then a new variation point is defined. Although we have two different variants *x* and *y*, at this point we only consider the existing, not the new, variant. Thus, the architecture is refactored to accommodate a variation point with the variant *x*. This is important because we would like to separate the side effects of refactoring from those of adding new functionality.

### **E. Running the Tests**

To make sure the refactoring process in the previous step did not have any side effects, we run all the tests in the system. This includes executing automated unit tests and acceptance tests as well as running all manual regression tests (usually used to test user interfaces and hardware related functionality). If a test fails, this means the refactoring process needs to be fixed, undone or redone to make this test pass again. We should not proceed to the next step until all tests are in a passing state.

### **F. Realizing the New Requirements**

Having refactored the architecture to be able to realize the new variability (if any), in this step developers implement the new functionality. The developers should produce test artifacts either before (using test-driven development) or after writing the production code.

### **G. Running the Tests (again)**

This step is similar to step E. All tests for the new functionalities as well as the older ones have to be run in order to make sure the new changes are actually verified and validated, and that the old functionality is not impacted by these changes. When all tests pass, a new iteration of the process can take place when needed.

## **CASE STUDY**

### **A. Experience Context**

The application we will discuss throughout this chapter is called eHome. It is a software system to monitor and control smart homes. Generally, the interface of the application consists of a floor plan representing the smart environment to be controlled, a number of items that can be dragged and dropped on the floor plan, and a set of graphical user interface (GUI) controls. A screenshot is shown in Figure 1.

Interacting with eHome occurs in two modes, namely:

- (a) user mode which allows the dwellers to obtain information about climate variables in the home such as temperature, humidity, CO<sub>2</sub> levels and other sensory information, check the current status of certain devices in the home such as lights being on or off, change the status of devices



*New Technologies.* As we went along, we wanted to deploy eHome on a large-scale SMART DVIT Table with an older version of the SMART SDK. A later request from our partner was to deploy eHome on a digital tabletop they had recently purchased. Specifically, it was the New SMART Table which supported multi-touch input and had a newer version of the SMART SDK. Later on, we obtained a Microsoft Surface and we decided to include it within the hardware platforms that we should support. As more platforms were supported, more decisions were revisited and the software design underwent drastic yet incremental changes. These changes were mainly driven by the two factors we mentioned in Section 1: technical issues and usability issues. Examples of such issues include:

- Three different SDKs that dealt with touch point input, one for each hardware platform.
- Conventional GUI elements like menus and tabs assumed a single orientation (vertical).

*Sources of Variability in eHome.* The technical and usability issues were not the only sources of variability in eHome. In fact, the first source of variability was business-driven. Smart homes vary widely with regards to what smart devices exist in the home, and what kind of monitoring and controlling is requested by a given customer. This variation in requirements often results in delivering a different application for each smart home. However, in spite of the differences between these applications, they share a lot of underlying functionality and business logic. Therefore, it is better to think of these applications as a family of systems that are somewhat similar yet not identical – which is the general understanding of what a Software Product Line (SPL) is. In this chapter, we will not discuss SPLs in terms of business-driven variability – but we will focus on technical- and usability-driven variability due to the utilization of vertical and horizontal displays.

## **B. Using the Approach**

When dealing with a new and fast-changing technology like digital tabletops, uncertainty about the future can be high. This in turn might render useless any efforts to speculate these needs. In the development of eHome, we avoided huge investments in upfront work. Instead, we followed a bottom-up, evolutionary approach to develop and maintain the SPL. We incrementally embraced new variations as needed, and allowed our common platform to evolve gradually. The following sections will discuss this matter in more detail.

In the discussion to follow, each section talks about one variability aspect. For each aspect, we analyze the issues we encountered and their implications on our system, and then we describe our approach to contain them. Although the examples we provide are specific to our system, this does not deteriorate the generality of the analysis or the proposed approach – because we believe that researchers and practitioners in this field will encounter similar issues and implications that can generally be resolved

using the same approach.

*Variability within Vertical Displays.* By vertical displays, we refer to the normal PCs that were used by developers to develop eHome as well as the HP TouchSmart PC on which eHome was initially deployed. The differences between these two groups were issues related to the mouse-versus-touch input. Table 1 describes these issues and their implications.

Issue	Implication
Right-click events do not make sense on a touch screen.	An alternate way (provided by the HP machine) to capture the right-click event on the touch screen was 'press-&-hold'.
The tip of the mouse cursor is tiny and accurate compared to the tip of a finger.	All GUI objects have to be larger to accommodate the finger touch more precisely.
When applying a touch on the vertical surface, the body of the finger covers some content on the screen (Figure 2a).	A vertical slider that was used to control the intensity of a light was changed into a horizontal slider (Figure 2b).

Table 1. Variability between a normal PC and an HP Touchsmart PC

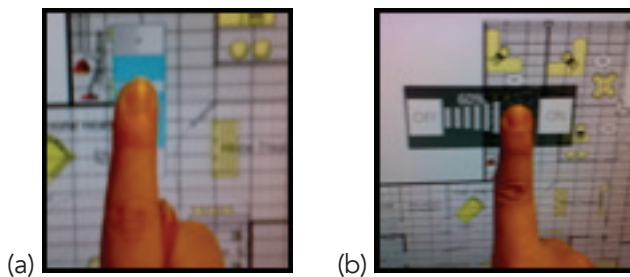


Figure 2. (a) part of the vertical slider is blocked by the body of the finger.  
(b) the horizontal slider solves this issue.

As mentioned earlier, the development for normal PCs and HP TouchSmart PCs was the initial stage in the evolution of eHome. At that stage, the architecture of eHome looked like the one in Figure 3a. The Presentation layer included all the view-related elements, whereas the UI Controller managed the communication between the Presentation layer and the Data Object Model. The Hardware Controller was responsible for communication between the actual hardware devices with the Model or the UI Controller. External Resources included the hardware devices, XML configuration files, and web services.

At first when we only considered the first issue (right-click vs. press-&-hold) as a source of variability, a conceptual layer was added to reflect this variability as shown in Figure 3b (previously, input was managed within the Presentation layer). The common platform included everything but the Input Manager where variability occurred. One variation point was defined as "input mechanism" with the two variants "mouse" and "touch." Later, when the other two issues were to be managed, variability penetrated down

to the Presentation layer as shown in Figure 4. That is, the variability profile we had so far could be described as:

*Input Mechanism = {mouse, touch}*  
*Mouse = [scale factor x, vertical slider, right-click]*  
*Touch = [scale factor y, horizontal slider, press-&-hold]*

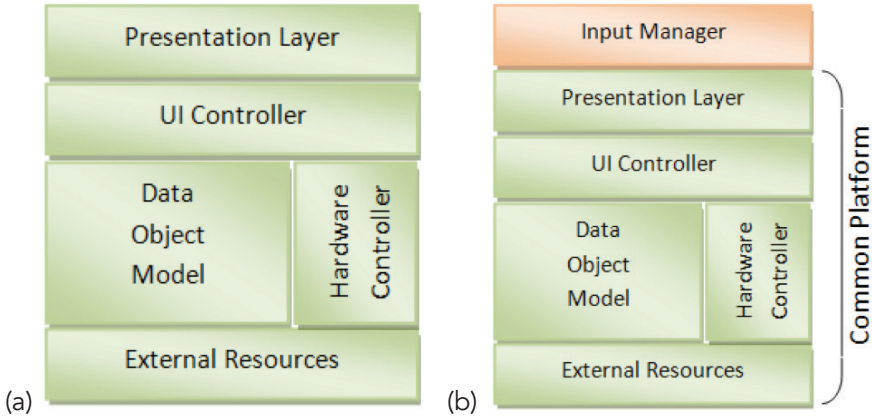


Figure 3. eHome architecture (a) before and, (b) after considering variability at the Input Manager layer.

*Variability between Vertical & Horizontal Displays.* To migrate eHome from a vertical surface to a horizontal one, we initially deployed eHome on a horizontal display without any modification to understand the differences. After a number of usability observations and going back and forth between the vertical and horizontal settings, we realized a raft of issues. Table 2 lists these issues and their implications on the migration process. In this chapter, we do not argue that these implications improved usability as this is yet to be appraised. The point, however, is that usability issues introduced new sources of variability. At this stage, we realized new variability occurring at the same two layers of the architecture. Not only did we have to go back and modify the variability we had previously defined in the Input Manager, but we also needed to explicate more variability in the Presentation layer. All the other layers were left intact. The updated variability profile included the following:

*Input mechanism = {mouse, single touch, multi-touch}*  
*Mouse = [right-click], Single-touch = [press-&-hold],*  
*Multi-touch = [press-&-hold, two-touch-zooming, gesture support]*  
*Layout = {normal PC, TouchSmart PC, digital tabletop}*  
*Normal PC = [scale factor x, vertical slider, conventional GUI controls, textual feedback]*  
*TouchSmart PC = [scale factor y, horizontal slider, conventional GUI controls, textual feedback]*  
*Digital tabletop = [scale factor z, circular slider, redundant GUI controls, text-less feedback]*

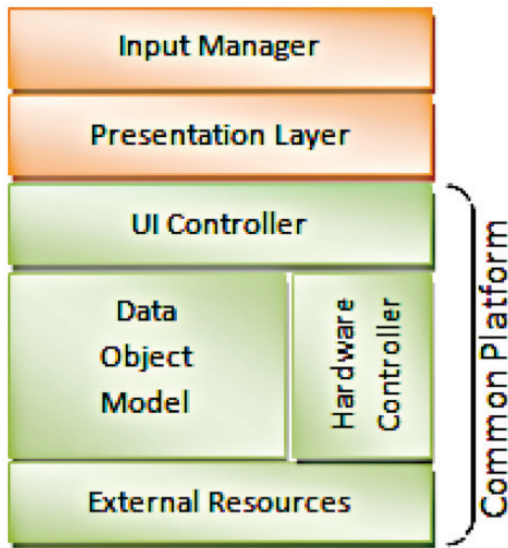


Figure 4. Architecture after considering variability at the Presentation layer.

*Variability within Horizontal Displays:* In the previous sections, we discussed variability due to differences between vertical displays. We then discussed variability due to the migration of eHome from a vertical display into a horizontal one. This section will discuss variability that was due to differences between horizontal displays. By horizontal displays, we namely refer to three hardware platforms: SMART DViT Table, New SMART Table, and Microsoft Surface. As illustrated in Table 3, we dealt with three different SDKs, two of which were different versions from the same vendor. The first tabletop on which eHome was deployed was the SMART DViT Table. We utilized the dual-touch capability of this table by adding a feature that allowed the user to place two touch points on the floor plan to zoom in and out. This kind of interaction required the hardware platform to support at least two simultaneous touches, which made the interaction irrelevant to the previous hardware platforms. For this reason, we chose not to include this interaction with the rest of the interactions in eHome that were common to all platforms.

Rather, a specialized controller was introduced in the UI Controller layer to manage all communication between eHome and the touch handlers in the SMART SDK, as shown in Figure 7 – A. By this separation, it was easier to plug this feature in and out. The new controller was responsible for managing three events, namely: TouchDown, TouchUp and TouchMove. In case the touch events were part of a zooming interaction, the specialized controller will handle the zooming. Otherwise, the touch events were rerouted to mouse events we had previously defined in the UI Controller for the previous platforms in order to maximize code reuse and avoid code redundancy.



Figure 5. eHome on a horizontal display has redundant GUI elements to support multiple orientations.

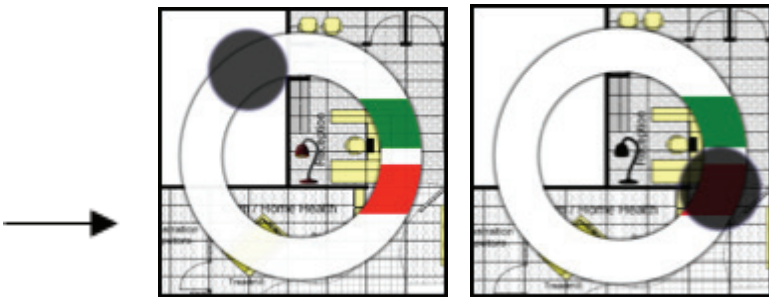


Figure 6. Circular slider to control light intensity.

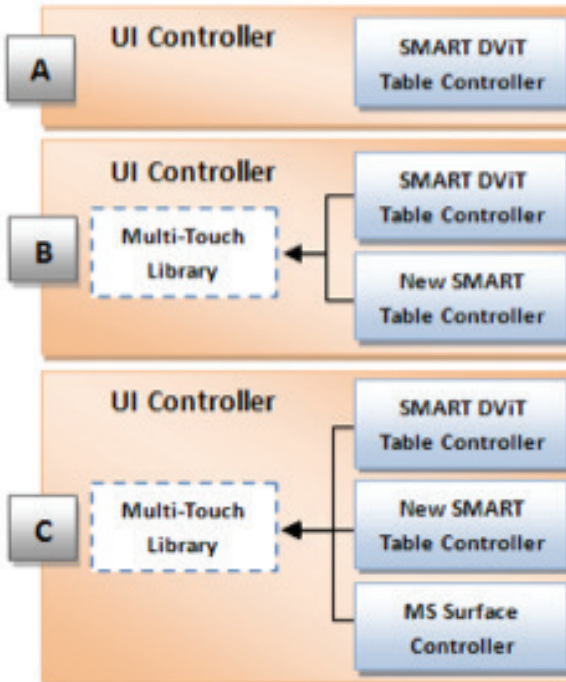


Figure 7. The evolution of variability due to differences in the SDKs.



The second step was deploying eHome on the New SMART Table. The New SMART Table came with its own SDK, and the technology was different from the older table. Therefore, a new specialized hardware controller was also created to manage communication between eHome and the touch handlers in the new SMART SDK. At this stage, we had two different controllers one for each table. These controllers, however, shared common aspects such as the main triggering events and the zooming interaction. These common aspects were abstracted in a new layer we called “Multi-Touch Library” as shown in Figure 7 – B. The new layer was abstracted in a way so that it was completely agnostic to the target hardware platform – all specificities were kept in the specialized controllers.

Later on, this abstraction served well in accommodating the new digital tabletop – MS Surface. It only took one day’s worth of work to deploy eHome on MS Surface, because all we needed to do was create a new specialized controller to communicate with the Surface SDK, while all other aspects were managed by the Multi-Touch Library. Figure 7 – C shows the final organization. As was done before, variability was evolved to include a new layer, namely the UI Controller layer. The following variation point was added to the variability profile:

*Multi-Touch SDK = {SMART DVIT Table, New SMART Table, MS Surface}*  
*SMART DVIT Table = [old SMART SDK], New SMART Table = [new SMART SDK], MS Surface = [Surface SDK]*

## **DISCUSSION**

In the previous sections, we discussed an approach to reactively manage variability in systems using agile practices. We also reported a case study where we used the approach to manage variability in an application that was to be deployed on a number of different hardware platforms. In this section, we discuss the advantages and limitations of our approach as learned from the case study.

### **A. Opportunistic Reuse of Code and Test Artifacts**

In the case of eHome, about 60% of the code (production and testing) is reused amongst all platforms. This figure could even be higher for systems that have a thinner presentation layer than the one in eHome. Maximizing reuse is desirable because it lessens the time and effort to produce new products and maintain existing ones. For instance, if the underlying technology for a certain feature (e.g. item tracking) changes, we need to make the proper modification in the common platform only once. Then we re-instantiate different products for the five different platforms we support. Also, say a vendor produced a new digital tabletop technology. All the work we need to do is at the UI Controller layer. The common platform can be used as is without any changes. However, this flexibility to change, adapt and reuse is achieved through a good understanding of the variability profile of the product line – which makes explicating and managing variability

essential.

## **B. Explicating and Managing Variability**

Adopting a SPL practice provides a systematic approach to think about and handle variations in the family. That is, before deciding to support a new digital tabletop platform, we need to know what is different about the new platform that cannot be supported by the existing product line. If there is any difference, then decisions need to be taken on where in the architecture this variation should be accommodated and what impact it will have on other platforms in the family. Without having an explicit variability profile of the SPL, taking such decisions becomes more difficult and is accompanied with higher risks. More importantly, with the variability profile the instantiation process of different products can be formalized by looking at each product in the product line as a function of the variation points. That is, any product P in the family is formalized as:

$P = f(vpa, vpb, \dots) = f(\{v1a, v2a, \dots\}, \{v1b, v2b, \dots\}, \dots)$  Where *vp*: variation point, *v*: variant, *{}*: OR operator

For instance, let's consider the variability profile of eHome. To produce a product that is specific to the HP TouchSmart PC, we need to specify the variants as:

*Input mechanism > Single-touch = [press-&-hold]*

*Layout > TouchSmart PC = [scale factor y, horizontal slider, conventional GUI controls, textual feedback]*

Or:  $P_{TouchSmart PC} = f(\text{input mechanism, layout})$   
 $= f(\text{single-touch, TouchSmart PC})$

This formal representation is then fed to the SPL through a configuration file or any other mechanism in order to start the instantiation of a specific product.

Issue	Implications
Horizontal displays are, typically, physically larger than vertical ones.	A new scaling adjustment factor is defined for UI objects to make them bigger, and hence easier to interact with, on larger displays.
Horizontal displays deal with multiple touch points not only single touch points or mouse clicks.	This new input mechanism needs to be incorporated into the Input Manager layer as a new variant.
Conventional GUI elements like buttons, menus and tabs were oriented in a top-down fashion, which for a horizontal surface did not seem natural because people sit on different sides of the table.	The conventional GUI elements were replaced by panels available on each of the four sides of the tabletop, as shown in Figure 5. Instead of one Exit button on the top left corner of the screen, an Exit button was added on each corner of the tabletop. The "change mode" button (user/designer) was removed. Instead, the change of mode on the digital tabletop happens automatically.
Feedback to the user was provided using a status bar at the bottom of the screen, which was not suitable for a multi-oriented surface (i.e. horizontal display).	Alternative ways to provide feedback were used. For example, when a certain operation executes successfully, the corresponding icon on the surface glows.
When using a slider control, vertical and horizontal sliders seemed counterintuitive if there were people sitting around the table (e.g. if you go up in a vertical slider, it seems as if you are going down for a person sitting opposite to you).	A circular slider was used with clearly flagged ON/OFF positions, as shown in Figure 6. Regardless of where you sit around the table, if the handle of the slider is moving towards the ON button, then the intensity is increasing and vice versa.
Some features were not easy to use for everybody around the table because the UI controls were closer to a certain part of the screen.	For deleting an object, instead of a single trash can on the bottom right corner of the screen, if the user touches an object while in the designer mode, the user has the option to drag it to any of the trash cans distributed on the corners of the screen.
Readability of text on the horizontal display was limited because of the presumed top-down orientation.	The horizontal interface includes far less text than the vertical one. Descriptive icons and UI controls, animations, as well as visual cues like pulsation or glowing are used to replace text.
With multi-touch capabilities, horizontal displays provided new interactions that were not possible on vertical displays (This was specific to our case – new versions of the HP TouchSmart PCs support dual-touch interactions).	On horizontal displays, it was made possible to zoom in and out of the floor plan using two finger touches.
On a big scale tabletop, drag-and-drop became difficult due to the physical limitations on the reach of an arm.	Gestures were made available as additional (not substitutive) ways of executing certain features. For example, to delete an object, one can use a scratch gesture.

Table 2. Issues leading to variability between vertical and horizontal displays.




	Picture	Issues			Implications
		Dimensions L x W cm	Touch Points	SDK	
<b>SMART DVIT Table</b>		240 x 100	2	SMART SDK old version	<ul style="list-style-type: none"> <li>- The aspect ratio of the SMART DVIT Table is 2.4 (compared to 1.33 for the New SMART Table and 1.56 for MS Surface). This introduced challenges in treating all four sides of the table equally. For example, instead of four panels, we only put two panels, one on each long side of the SMART DVIT Table, because the floor plan could not rotate with its full size except for a full 180 degrees.</li> <li>- An abstraction layer was introduced to embrace the different ways the SDKs deal with touch points.</li> <li>- Although we did not encounter this problem, we anticipated that when we add features that should support collaborative work, the limitation of 2 touch points might be a source of variability.</li> </ul>
<b>New SMART Table</b>		55.9 x 41.9	40	SMART SDK new version	
<b>MS Surface</b>		108 x 69.0	Large number - exact number unknown	Surface SDK	

Table 3. Differences between the SMART DVIT Table, New SMART Table, and MS Surface.

### C. The Ability to Form Combinations

One more advantage of the systematic treatment of variability is the ability to combine different variants to come up with diverse products. For example, suppose we want to support the new HP TouchSmart PC that enables two simultaneous touches. We can come up with a new combination of variants to add the zooming behavior:

*Input mechanism > Multi-touch = [press-&-hold, two-touch-zooming, gesture support]*

*Layout > TouchSmart PC = [scale factor y, horizontal slider, conventional GUI controls, textual feedback]*

Or:

$P_{\text{New TouchSmart PC}} = f(\text{multi-touch, TouchSmart PC})$

That is, by choosing a different variant for a given variation point, we ended up with a different product for the new platform. Constraints are usually defined to filter out invalid combinations.

We understand that some of these advantages are inherited from the SPL practice itself. However, it is imperative to point out that using our iterative approach allows organizations to realize the same advantages in a way that is more cost effective (because it is lightweight) and less risky (because it minimizes speculation), and with a faster return on investment (because systems are continuously delivered as opposed to waiting until the application engineering phase).

### Limitations

The main limitation of our approach is that there is no clear definition of the roles needed in the different steps. For example, who in a typical

agile organization should conduct the variability analysis? Can developers assume the responsibility of updating the variability profile? This is vital because variability analysis and profiling require a wide knowledge of existing requirements in the system. Therefore, a developer who only worked on a certain aspect of the system may not be qualified for this role. A second concern we had about the proposed approach is the amount of discipline needed to implement the approach successfully. For example, the approach relies on the premise that tests are written for all features in the system and that sufficient test coverage is available. In our case, eHome had an automated testing coverage as high as 90% of the model code. We also defined a suite of regression tests to be conducted manually to test UI and hardware related issues. We are not sure what the consequences are if good testing practices are not present in the organization. A more systematic evaluation is needed in order to draw more reliable conclusions on the advantages of our approach as well as its limitations.

### **Conclusion**

The general goal of our research was to reconcile conflicts between traditional SPL engineering and agile software development. This chapter presented a framework that allows agile organizations to reactively construct variability profiles for existing and new systems. The framework leverages common agile practices such as iterative software development, refactoring, continuous integration and testing to introduce variability into systems only when it is needed. We showed, by example, how to use the proposed approach, and we discussed the advantages that can be realized, and the limitations that may hinder successful adoption of the approach. Future work includes evaluating the approach in an agile organization to form a better understanding of the practicality and feasibility of the approach.





# **BUILDING INFRASTRUCTURE FOR DIGITAL SURFACES**







## Building Infrastructure for Digital Surfaces

*Nicholas Graham, Queen's University*

*Carl Gutwin, University of Saskatchewan*

### **I**ntrouction

SurfNet's Theme 3 focused on creating tools, frameworks and system infrastructure for software engineers to rapidly develop applications for surfaces. The motivation for carrying out this research was the substantial gap between the needs of surface applications and development tools: for example, operating systems provide little support for key interaction techniques such as multi-touch input and gesturing, and many software architectures do not support basic surface issues such as combining multiple devices and displays in a single application. As a result, researchers and developers were often forced to re-implement basic interaction techniques for every application.

In this theme, we carried out explorations of how to simplify the development of surface-based applications, particularly those involving multiple surfaces, multiple users and multiple surface types. This theme's organization uses a reference architecture for surface-based systems which has five layers, of which the middle three represent the core work areas of this theme.

More than forty SurfNet projects investigated infrastructure for surface-based applications. Several projects were primarily resident in other themes, but list the development of infrastructural requirements as part of their goals – this approach was taken to ensure that the infrastructure developed in the project matches the needs of application developers. Research has been carried out on each of the three main service layers. Concrete tools have been developed at the basic services layer (e.g., the EquisFTIR

toolkit or the Haptic Puck Toolkit). At the middleware layer, we explored the design space of tools and APIs enabling multi-surface development. Our approach relies on creation of both special-purpose infrastructure (e.g., the iOSRemoteConnector), as well as more generic networking techniques (e.g., the .Networking GT toolkit). At the user-interface service layer, we have explored the next generation of user interface widgets applicable to multi-surface applications. A diverse set of projects was carried out at this level, from the use of synthesized audio to improve workspace awareness, a proximity toolkit for detecting peoples' positions around surfaces, and the libjtouch toolkit for bringing touch-based input to the Java language.

We now survey specific types of infrastructure developed in the theme.

<b>Application</b>	<ul style="list-style-type: none"> <li>• Basic Services: Includes basic abstractions for low-level surface hardware, such as projector tiling and input processing.</li> <li>• Middleware: Network support for multi-surface applications, which involves issues such as abstracting distributed systems, allowing programmers to treat MSEs as single surface, and dynamic recruitment and use of available surfaces.</li> <li>• User Interface Services: User interface toolkits at the surface level, which is built on the results of Theme 1 (Humanizing the Digital Interface: Departure From Desktop Computing). For example, this layer involves multi-user widget sets, abstractions for rotation and orientation, and generalized pointing and selection techniques for surfaces.</li> </ul>
<b>User Interface Services</b>	
<b>Middleware</b>	
<b>Basic Services</b>	
<b>OS/Hardware</b>	

### Multi-Surface Environments

Multi-surface environments (MSEs) are composed of a number of devices, such as digital tabletops, large shared displays and personal tablets. MSEs are designed to allow people to work together effectively, allowing sharing, collaboration and private work. The goal of an MSE is to allow people to fluidly move between devices depending on the style of work or collaboration that they are performing.

The difficulties of developing MSEs can be seen in the numerous examples developed in SurfNet. The C4i Emergency Operations Centre (Calgary) and the MACCH Coordination Hub (Waterloo) both provide support for emergency operations, providing digital tables and large shared displays for collaboration as well as tablets and personal computers for private work. In both cases, a major technical challenge was the real time processing and display of data from disparate sources. Researchers at Calgary developed the C4 programming language and API to address these problems.

In Queen's OrMiS environment for simulation-based training and in Calgary's SkyHunter system for visualizing geological information, the key technical problem was ensuring the consistency of different views provided by different surfaces. Several technologies were developed in SurfNet to aid the connection and synchronization of views in MSEs. .Networking GT (Saskatchewan) provides basic networking features specialized for collaborative systems. Janus' Timelines model (Queen's) eases the sharing

of time-sensitive data between surfaces. Web-technologies, as described below, played a major role in connecting surfaces, through Saskatchewan's web-application framework for multi-surface environments (WAMS), Calgary's REST and JSON-based IntAirAct, and Carleton's PanUI for web-based deployment of multi-surface applications.

An important aspect of multi-surface interaction is the localization of surfaces in physical space, such as in TerraGuide's use of a tablet as a magic lens over a shared tabletop for terrain exploration (Queen's). Researchers at Calgary developed the SkyHunter toolkit that allows a Kinect camera to track the position of handheld surfaces in real-time.

### **Web Technologies**

One of the main technology changes over the course of the SurfNet project was the establishment of the World-Wide Web as a platform for full-scale application delivery. As a result of this advance, SurfNet researchers explored ways in which surface-based applications can be organized and provided using web technologies. Although different native platforms have support for multitouch surfaces in large and small devices, there are numerous practical advantages to targeting web browsers as platforms for application development. Using this approach, applications need no installation process, and can be developed once for a wide range of devices. For example, researchers at Carleton University have developed several surface-based applications for decision support that run entirely on the web, and have captured their knowledge in a new web architecture for distributed surface applications (the PanUI toolkit). The toolkit brings together existing JavaScript libraries and provides an extension framework to integrate diverse devices into our distributed web application for surface applications.

Other SurfNet research has explored the performance of web-based applications. One of the main requirements for multi-surface and multi-user applications delivered over the Web is for reliable and efficient network communication, but little was known about the new networking approaches that appeared in web browsers during the time of the SurfNet project (e.g., AJAX Comet and HTML5 WebSockets). Researchers at the University of Saskatchewan showed that web-based networking performs well and can support the communication requirements of many types of real-time multi-user applications. In another project between Saskatchewan and Queen's University, researchers explored the problem of what happens when the various devices and users of a multi-surface application become temporarily disconnected, and developed the DiscoTech toolkit that enables application developers to handle disconnection in ways that preserve the user's experience in the application.

### **Group Awareness**

Awareness of the presence, locations, and activities of the other members of a group is recognized as an important part of real-world collaboration,

and has been extensively studied in the area of Computer-Supported Cooperative Work. SurfNet researchers carried out several projects to extend our understanding of group awareness to situations where people collaborate on tables, multi-surface environments, and distributed surfaces. At the architectural level, researchers at Saskatchewan and Cornell Universities developed the OpenMessenger framework which structures multi-surface applications around the assumption that individual behaviors occur in anticipation of and in response to the behavior of others.

At the interface level, several projects have been carried out. Researchers at the University of Waterloo built the Event Timelines system to support situation awareness in tabletop environments that involve automation, allowing groups to perceive changes in the system, comprehend them, predict future events, and, ultimately, make optimal decisions. Other systems have focused on capturing the information available in gestures above a table surface and showing it at remote tables. The KinectArms toolkit (Universities of Saskatchewan and Calgary) captures and isolates arm images using a depth camera, and shows the arms on remote surfaces – allowing richer gesturing and pointing over tables. Finally, SurfNet researchers also explored the viability of sound effects as an awareness cue on tabletop systems, and produced a toolkit for generating dynamic sound using granular synthesis.

### **Proximity Toolkit**

An important insight to arise from SurfNet is that interaction with surfaces is not limited to touch. For example, as seen in the SkyHunter and TerraGuide projects discussed earlier, the relative positions of surface devices can be an important part of the interaction. Researchers at Calgary have developed the foundational underpinnings of how proximity to surfaces can moderate interaction, allowing for example a television to configure its interface based on who is near to it. The proximity toolkit has been developed and publicly released to enable development of surface-based applications which consider proximity of users and other surfaces. The toolkit has been widely adopted internationally. Examples of its use at Calgary include the proximity-based universal remote controllers, devices that can control literally anything that they are close to. At Saskatchewan, researchers have shown how body-based input can be used to create a variety of application styles controlled by physical motions performed in the proximity of surfaces.

### **Game Development Frameworks**

Digital tabletops naturally enable co-located collaboration, and so provide an opportunity to bring traditional board games to the digital realm. Example tabletop board games investigated in SurfNet include Pandemic (Queen's and Waterloo) and Dominion and Pax Romana (Waterloo). There has until now been little support for creating board games for digital tabletops, however. Researchers at Waterloo have created the Tabletop Board Game Framework that extends the Unity game development engine to better support board games. The framework includes touch-based

widgets for common board game features such as scorekeeping and shuffling and dealing cards. Beyond traditional board games, researchers at Saskatchewan have explored in their GAMS framework how to support the development of multi-surface games where the location of the surfaces is part of the game. GAMS is accessible to developers through its use of web technologies. Finally, large touch displays (such as tabletops) provide a natural environment for the monitoring of complex distributed multiplayer games, such as massively multiplayer online roleplaying games. McGill's tabletop monitoring tools demonstrate how surfaces can support the back-end operation of such games.

### **Enriched Interaction Toolkits**

Several projects in SurfNet involved the development of infrastructure to support techniques for enriching interactions on surfaces. These projects range from programming APIs to novel techniques. For example, researchers at the University of Calgary developed C4, an API to support creative coding and enhanced expression on multi-touch devices; the system provides a prototyping language suited for the rapid development of expressive mobile applications. A similar approach, but designed for multi-touch control of music, was seen in the JunctionBox system. In the more focused domain of idea generation, researchers at Queen's University built MACS On Top, a surface-based tool that allows small groups of designers to collaboratively draw and edit diagrams, rapidly supporting creation and comparison of many design alternatives. Several other projects have developed infrastructure that supports techniques from SurfNet's Theme 1. For example, researchers have built reusable components that support interactions at different proximities (Calgary), visualizations of hands above a surface (Saskatchewan), novel scrolling techniques for radiology (UBC), and pointing-based command selection (Saskatchewan).

### **Conclusion**

This theme has made two broad contributions to the state of practice in creating surface-based applications and multi-surface environments. First, we have identified the types of infrastructural problems faced by developers of this kind of application, and second, we have developed concrete software architectures, APIs and toolkits that help resolve these problems. This is captured by the reference architecture shown above. Examples of basic services included detection of proximity of users and surfaces and tools capturing the positions of users' arms over a table. Considerable research has been performed in SurfNet on middleware, driven largely by the need to support MSE development. As we have seen, a particular focus has been on the development of tools based on web standards to simplify the development of distributed systems dynamically composed from a set of surfaces. Finally, user interface services include interaction techniques for visualization of earlier application states on tabletop-based applications.

While the reference architecture has helped to conceptually unify the work performed in this theme, we found it best to provide tools focused on

specific application domains and development contexts rather than try to provide a single unified tool for surface development. This was appropriate since, for example, tabletop games are appropriately developed using the Unity game engine, whereas tools created in collaboration with companies such as SkyHunter and C4i more appropriately adopted the companies' own development environments. We see great promise going forward in the use of web technologies as a means of providing surface development tools in a platform-agnostic manner, and this theme's work has laid the foundations for such further work.



## **Society of Devices Toolkit and Projected Pixels**

*Sydney Pratte, Teddy Seyed, Alaa Azazi, Edwin Chan, Yuxi Wang, and Frank Maurer*

### **Introduction**

Multi-surface environments (MSE) are becoming increasingly popular in research today due to their ability to enhance application interactivity, group collaboration and an inherent “coolness” factor. There is a broad variety of devices and sensors available that can form the basis for MSEs. Devices range from small wearables up to multi-touch wall screen displays. MSEs are examples of the ubiquitous computing concept. Ubiquitous environments allow people to access and share information continuously through the environment across different devices. Spatial awareness of a ubiquitous environment is a system’s ability to understand the location and orientation of people and devices in a space. Spatial-awareness is derived from the sensors, which are either embedded in the environment or in device devices inside the environment. Challenges arise with how information and tasks can be performed effectively across different devices, which have different degrees of spatial-awareness (Seyed et al., 2012).

Past research into ubiquitous environments has generally focused on the types of interactions performed by people and the devices (Weiser, 2001) using proxemics to define the interaction spaces (Ballendat et al., 2010; Greenberg et al., 2011). However, these approaches are limited in real world setups due to the difficulties in creating a multi-surface environment. Many development kits use complex software and hardware setups and do not support multi-sensor or cross-platform devices (Houben and Marquardt, 2015).

The Society of Devices (SoD) Toolkit (or SoD Toolkit, <http://sodtoolkit.com>) was developed to help mitigate the software and hardware limitations of previous spatially-aware ubiquitous environments (Seyed et al., 2015). The primary research goal was “to allow for novel explorations of different types of multi-device, spatially-aware (through multi-sensor fusion) ubiquitous environments that can be augmented with a multitude of newer sensors and device platforms” (Seyed et al., 2015). To achieve this goal, the toolkit uses a modular architecture allowing developers to use multiple sensors with off-the-shelf technologies to create a spatially aware ubiquitous

environment. With this style of architecture, developers can easily integrate additional modules with future technologies. The SoD Toolkit includes APIs for several different platforms including iOS, Android and Windows and web-based systems including HTML5, Node.js and Javascript. Allowing for a wide variety of devices to communicate in an everyday ecology. The toolkit also offers methods of prototyping without needing any hardware setup for quick testing even if hardware is unavailable.

In this chapter, we present the SoD Toolkit including its key features and architecture. Next we describe a real world application in Emergency Response that utilizes the key features of the Toolkit. We then demonstrate the flexibility of the Toolkit by describing a project that integrated projection feedback into the ubiquitous environment.

### **Related Work**

The SoD Toolkit is built off of prior work on proxemic interactions, multi-device interactions and application programming interfaces (APIs) and toolkit design (Seyed et al., 2015). Greenberg et al have richly explored proxemics used in ubiquitous environments, looking at spatial relationships between objects and users, specifically applying five proxemic dimensions: orientation, distance, motion, identity and location (Greenberg et al., 2011). Greenberg et al also looked at the use of sensors to track users and devices in a space to understand the differences in explicit and implicit interactions (Ballendat et al., 2010). This group of researchers also discovered many challenges faced in proxemics including privacy and security, connecting different devices and providing meaningful feedback for interactions (Marquardt et al., 2012). Research into proxemics interactions in ubiquitous environments was a motivation for creating SoD Toolkit as a platform for integrating technologies within a space.

The different devices in today's technical ecology (e.g. smartphones, tablets, smart-watches) leads researchers to explore how information or content can be moved across these devices (Seyed et al., 2012). Moving content within a multi-device space has been investigated in a number of ways. Rekimoto designed a 'pick-and-drop' technique where a pen is used to transfer information from one device to another synchronized across multiple computers (Rekimoto, 1997). Another example of a synchronized multi-device interaction is Hinckley's method of bumping or stitching tablets together (Hinckley, 2014; Brumitt et al., 2000) and Lucero's pinching gesture to share content (Lucero et al., 2009).

Multi-device interactions are directly influenced by proxemic and spatially aware technologies (Chen et al., 2014). For example Marquardt et al explored a gradual engagement pattern that mapped inter-device proximity for different types of interactions (Marquardt et al., 2012). Wilson and Benko used spatial awareness for different interactions between and on physical surfaces in the LightSpace project (Wilson and Benko, 2010). These are only a few examples of different interactions found in research, the SoD Toolkit



easily allows developers to explore new and yet to be looked at forms of interactions (e.g. a smart watch and a tabletop).

A related area of research is in the development of multi-device toolkits. Many toolkits have focused on web-based interfaces for multi-device interactions, such as Conductor (Hamilton and Wigdor, 2014) and Panelrama (Yang and Wigdor, 2014) and others (Chi and Li, 2015; Konig et al., 2009; Schreiner et al., 2015). XDStudio on the other hand uses a GUI builder for multi-device applications (Nebeling et al., 2014). One of the most well known toolkits in this area of research is the Proximity Toolkit, which gathers spatial data from various tracking sensors for larger ubiquitous environments (Marquardt et al., 2011). With its high-end tracking system, this toolkit is ideal for research prototyping however it is not appropriate for real-world applications (Nebeling, 2014). The SoD Toolkit is similar to the Proximity Toolkit and the XDKinect (Nebeling, 2014). XDKinect is a toolkit that uses a single Microsoft Kinect sensor to enable device interactions and proxemics. SoD Toolkit offers support for prototyping and building multi-device and multi-sensor applications for creating ubiquitous environments with a focus on off-the-shelf sensors and everyday devices. In addition, our source code is also freely available to download as open source (<http://sodtoolkit.com>).

### SoD Toolkit

In this section, we provide an overview of the SoD Toolkit. We discuss some of the architectural components such as the locator and the communication, then we cover the client SDKs available in SoD Toolkit and finally the visualization tool used for creating ubiquitous environments. A diagram of the system architecture is provided in Figure 1.

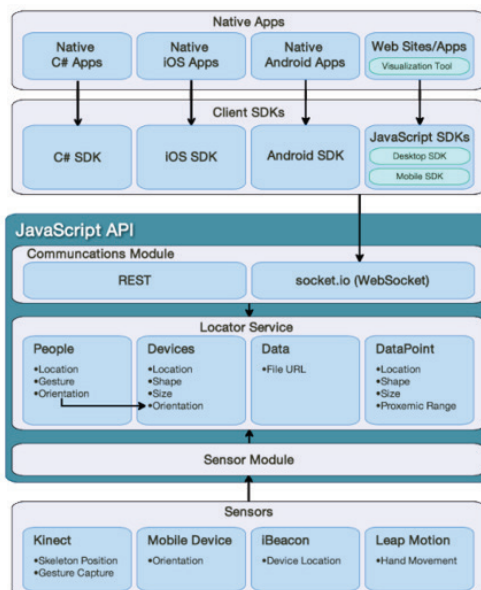


Figure 1. Overview of architecture.

## **Locator Services**

The locator service is the central hub that fuses all spatial data from the various sensors and devices in the environment (Seyed et al., 2015). The Locator Service receives raw positional data from connected sensors and devices and converts the device-based coordinates into a common coordinate system of the room, creating a common picture of the environment and allowing all devices in the area to be aware of the entities around them and their relationships in the space. The entities that are tracked by the Locator Service include devices (with orientation), sensors and the users in the space. All entities are mapped to a 3D common coordinate space.

The Locator Service is implemented in an event-driven design and the clients choose which events from the locator that they want to subscribe to. For example, for proxemic interaction a wall display application can subscribe to an event that watches for when a user is within a specific distance from it or if other devices are pointing at it (Seyed et al., 2015). The Locator Service also tracks data point entities. Data points are physical locations in the environment where virtual data can be “attached”. Data points can be created dynamically in both code and in the Visualizer tool (see below). Similar to the other entities in the space, a data point can also have proxemics ranges set where different interactions can take place. For example, if a user is in range of a data point they could perform gestural or device-based interactions with the data.

## **Communications**

Communication for SoD Toolkit is built into the Locator Services as the central hub. All devices communicate with the up, providing sensor information to it and receiving callbacks for subscribed events from it. The module is implemented in Node.js and follows the client-server style architecture. Node.js was chosen due to its scalability and efficiency (Seyed et al., 2015). All messages from the server to the client libraries are in standard JSON format and use the WebSocket protocol, allowing for bidirectional communications with less overhead than traditional HTTP methods. The modular design of SoD Toolkit allows for easy extension of functionality with limited development effort.

## **Client SDKs**

There are two main functions for the client SDKs, they provide the Locator Services with the spatial-awareness data from the sensors on the devices and they provide the native application platform for development. The sensors that are currently being used include the Microsoft Kinect, the Leap Motion and Apple’s iBeacon. In addition if a device has built in sensor data (e.g. accelerometers, gyroscopes) then this information is also sent to the Locator Services.

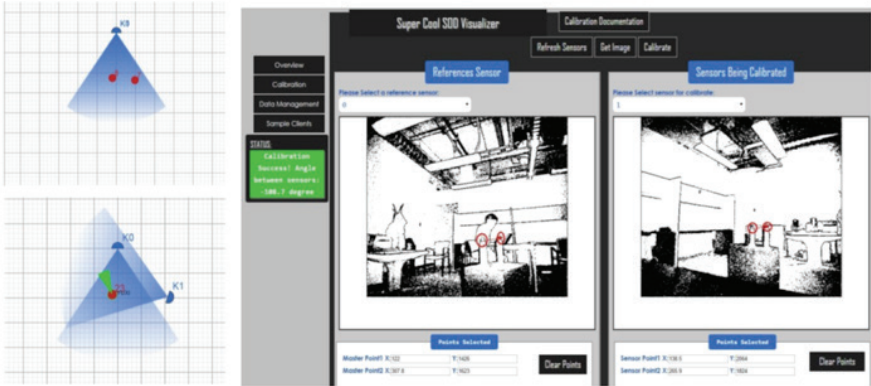


Figure 2. Sensor Fusion using multiple Kinect sensors. (a) 2 uncalibrated Kinect sensors (overlapping) showing skeletons tracked in the environment. (b) Interface for calibration. (c) Properly calibrated environment with overlapping Kinect areas (Seyed, 2015).

The Microsoft Kinect (version 1 and 2) sends skeletal data, position, identity and gestures from the users in the space at a rate of 30 skeleton frames per second (Seyed et al., 2015). Since the Kinect sensor has a limited tracking range (1.2m to 4.5m), multiple Kinects track users through sensor fusion, improving tracking quality as well as reducing occlusion issues. To integrate data from multiple Kinects, we need to map their device specific coordinate system into a common coordinate system. This fusion is accomplished through three calibration steps with two Kinects at a time:

1. an object is placed in the common view of the sensors to pair,
2. a point on this common object is chosen in two views (one per Kinect) and
3. one Kinect's vector is translated to the other Kinect's vector (allowing us to determine the transformation matrix between the two coordinate systems).

Figure 2 shows the sensor calibration process seen through the Visualizer tool. This process is then repeated for any additional Kinect sensors in relation to the reference sensor. Once all the Kinect sensors have been calibrated, the Location Service ensures that a user that is visible by more than one Kinect is only being tracked once.

To track a mobile device, we pair in with a user in the room and track its location by tracking the user. In addition, we fuse sensor information from the device with its location. This allows for interactions as "flicking" or "pouring" (Seyed et al., 2012). The use of multiple low-cost sensors working cohesively to spatially track users and devices provides a less costly approach, as opposed to more expensive and harder to setup tracking systems (Marquardt et al., 2012). Multiple sensors types also allow for more degrees of detail when tracking in the ubiquitous environment: to reduce

hardware costs, different areas in a space can be tracked at higher or lower accuracy based on application needs.

For example, on a finer level of detail the Leap Motion sensor provides finger tracking for a user in the space. Typically, the Leap Motion is fused with a sensor that has a larger tracking range in order to provide more “meaningful interactions”. By adding a Leap Motion to the environment, developers can, for example, provide touch capabilities to non-touch displays while using SoD for multi-surface integration.

As an intermediate sensor level, Apple’s iBeacon provides position data such as close, near and far. The iBeacon sensor can get the position information from both Android and iOS devices or from a person as long as they are wearing a iBeacon tag. The iBeacon sensor is not as accurate as the Kinect. However, it is most useful in a large ubiquitous environment and supports tracking people between higher-accuracy spaces.

One of the goals in the development of the SoD Toolkit was to reduce the learning effort for developers. The clients for SoD are all implemented in the various sensor/devices native development platform, including C# for Windows, Objective-C and Swift for iOS, Java for Android, and HTML5/JS. Support for the corresponding IDEs is also provided, such as Visual Studio, Android Studio and Apple’s Xcode. For other programming languages and environments developers can write wrappers for the existing libraries. Example applications for each library are also provided for developers to help with easy startup.

### **Visualization Tool**

The Visualization tool (Figure 3) for SoD Toolkit provides the general overview of the environment for testing and debugging purposes. It shows the current location of data locations, people and devices in the locator coordinate system. It also provides a list of all the connected clients and devices in the environment, the sensors available, and the pairing state. For instance, in Figure 3 there is a sensor available, one web client is running and one person (unpaired to a device) is being tracked. Also there is a data point available. If a mobile device was added to the system, the user could quickly be paired to the device. As stated earlier, the Visualization tool is also used for the calibration of multiple sensors.

The Visualization tool aids developers throughout the development process by supporting fast and mixed prototyping. Sample clients and sensors can be added to the environment through the Visualizer and will function like the real entity in the system. This allows researchers to conceptualize applications, environments and inter-device interactions regardless of the stage of development and without having to make significant changes once physical hardware is introduced (Seyed et al., 2015).

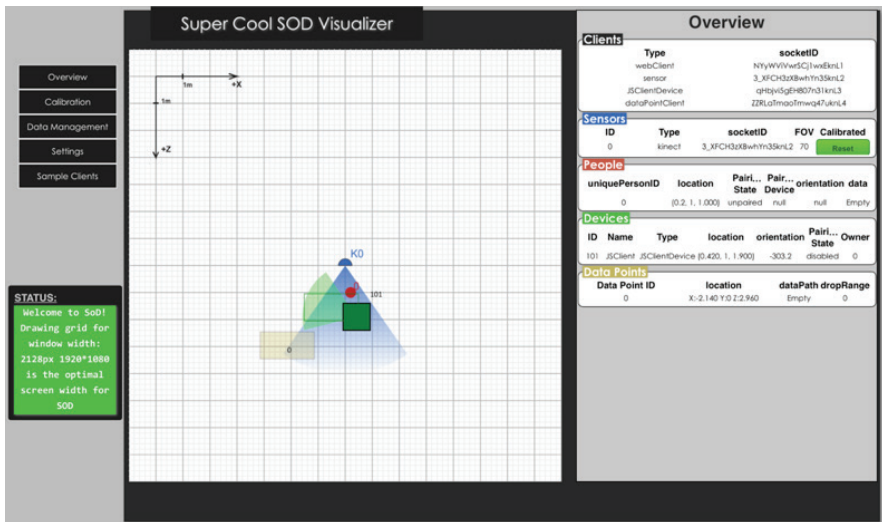


Figure 3. The Visualization Tool for SoD Toolkit.

### Real-World Application

The University of Calgary is collaborating with C4i Consultants, a software development company that focuses on military and emergency response training programs, to design the Emergency Operations Center of the future (EOC-F). In order to carry out a coordinated and synchronized response during an emergency, EOCs collect information from all parties involved to create a Common Operating Picture. An EOC is an extremely collaborative environment, bringing together a representatives from a number of organizations in a physical space to handle emergencies. Each interacts with their own systems and devices but to coordinate the response inter-agency collaboration is essential. This makes an EOC an ideal candidate for a ubiquitous environment. In the EOC-F project, we are designing an environment that tracks multiple stakeholders and allows them to communicate and share information and plans using the spatial awareness created by SoD.

An EOC setup may include a wall display, touch table, digital whiteboard (Kapp Board), and a number of tablet devices (Figure 4). One or more Kinects enable proxemics interactions within the EOC.

The wall display is intended to provide high-level information about the emergency, increasing awareness of the situation. It is instrumental in forming the Common Operating Picture (COP). A common reference point creates a shared understanding of basic and vital information among EOC operators. The information is aggregated from various information sources and streams, including social media.

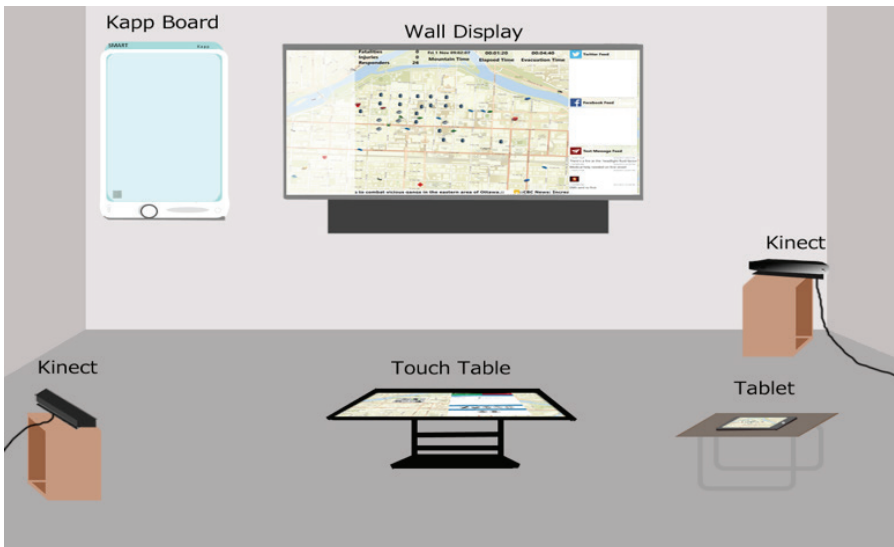


Figure 4. Typical EOC-F configuration.

The touch-enabled table (Figure 5) is a central collaborative space for decision makers, and provides an interactive map of the emergency site. Operators can draw up response plans with the annotation tools, and directly interact with field units by controlling them on the tabletop map. For example, hazard sites could be marked up, before an evacuation zone is created. Field units inside the evacuation zone are automatically routed to outside the area, while moving units outside the zone will be routed around it.

The digital whiteboard provides a familiar planning tool present in existing EOCs, but uses automatic capture of handwritten notes to quickly distribute information to other EOC operators and field responders. Information on the whiteboard can be viewed on the tabletop, or sent to field operators to communicate key objectives and planning details.

The tablet devices (Figure 6) represent the mobile aspect of EOC-F, and can be used both in the EOC and in the field. Similar to the table, users are presented with an interactive map and planning tools. However, tablets are role-dependent, and provide different tools suited to the user's role. A police officer using the tablet could place roadblocks, while a HazMat specialist could create regions around chemical spills and annotate it with relevant information. Plans drawn on tablets remain private and role-specific, until they are explicitly shared. The tablet also supports communication between field responders and the EOC, through video calls and SMS texting capabilities.



Figure 5. Touch-enabled tabletop, with various planning tools displayed.

EOC-F uses SoD for proxemic interactions, for example for flicking or pouring information from a tablet to a table or for pulling information from a table to a mobile device.

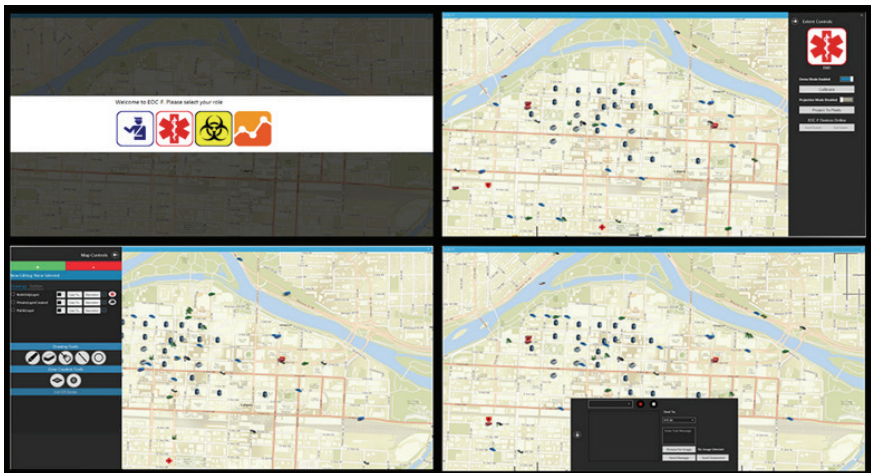


Figure 6. Various states of the tablet device: 1) Role selection, 2) general settings, 3) planning tools, and 4) video and SMS communications.

## Flexibility of SoD Toolkit

In order to demonstrate the flexibility of the SoD Toolkit for integrating new technologies, we describe the Projected Pixels project. The Projected Pixels project was a Surfnet collaboration to bring a low resolution full-coverage room display into the ubiquitous environment created with SoD Toolkit.

*Projected Pixels.* A challenge that can arise in a multi-device ubiquitous environment is providing meaningful feedback that conveys information such as an interaction has occurred and the target of the interaction. If no feedback is given on the devices or in the environment, then users can be left confused as to whether an action was successfully performed and what actually has happened. One possible solution to mitigate these feedback challenges is to add visual feedback to spatial and cross-device interactions to the environment itself. The Projected Pixels project utilizes a method of creating a low-resolution computer output on floors, walls, ceilings, furniture, etc. – in order to assist in providing visual feedback for communication and interactions in the projection-enhanced ubiquitous environment.

We explored how projection feedback could be used to show information transfer between devices and the location of virtual information (data points). We developed a Windows application using the SoD Toolkit for device communication and spatial tracking, and used the ASPECTA toolkit (<http://aspecta.cs.st-andrews.ac.uk>) for the projected pixels. The ASPECTA toolkit only requires a standard projector, a hemispherical mirror and a PC to run its API. The Windows applications for the Projected Pixels project (the monitor display, tabletop, wall-display and tablet device) were each developed using the C# client SDK provided by SoD Toolkit. A C# client wrapper was created to directly communicate with the APSECTA toolkit's server running on the computer connected to the projector.

To evaluate the approach, we implemented three tasks:

1. sending information from one device to another,
2. receiving information from another device, and
3. finding a data point within the environment.

For each of the three tasks two variations of feedback were designed. We looked at non-animated verses animated feedback. Animated feedback appears when the action is initiated and is removed at the conclusion. Non-animated feedback is static and visible throughout the duration of a specified time. The sending task involved sending an image from the tablet device to one of the three displays. The display that the information was sent too, displayed the image once it was received. The tablet device sends a message to the display, which listens for the event. The Projected Pixels wrapper also listened for the event and projected the appropriate feedback between the send and the receiving display. The receiving task works the same but in reverse. Figure 7 shows the non-animated and



animated feedback respectively for the sending task and Figure 8 shows the same for the receiving task.

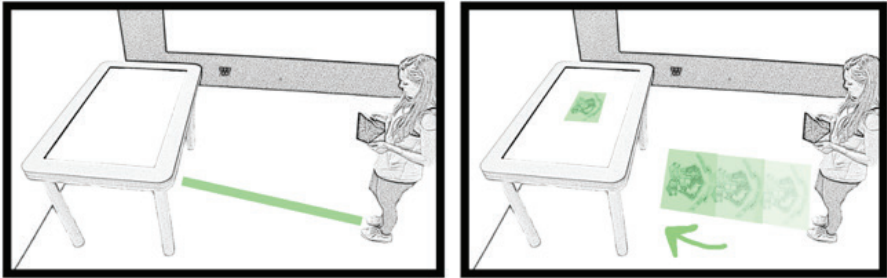


Figure 7. (a) Left image, non-animated feedback and (b) Right image, animated feedback for sending task (Pratte, 2015).

For the finding task when the person walked into the defined location the information was received on the tablet device. For the animated feedback shown in Figure 9b the image was projected on the floor as soon as a person enters the data point and disappearing again when they leave. The non-animated feedback shown in Figure 9a, a projected image at the data point location appeared for the duration of time. For the finding task once the person enters the data point location a message is sent to the tablet device paired to the person, which is listening for the event.

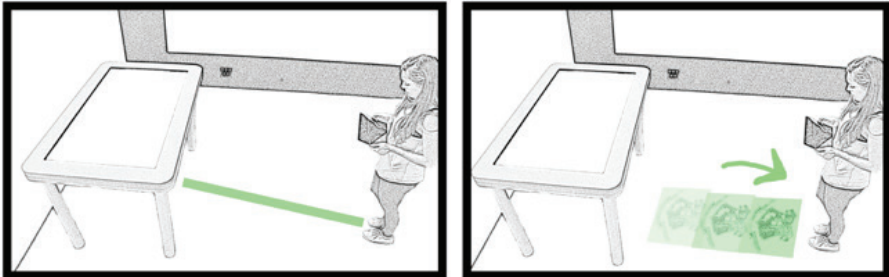


Figure 8. (a) Left image, non-animated feedback and (b) Right image, animated feedback for receiving task (Pratte, 2015).

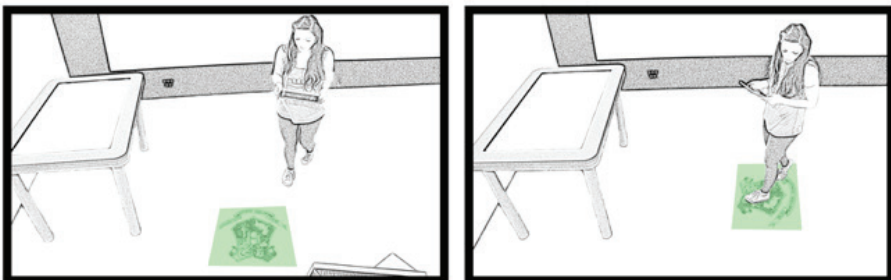


Figure 9. (a) Left image, non-animated feedback and (b) Right image, animated feedback for finding task (Pratte, 2015).

*Integrating into SoD Toolkit.* The integration into SoD Toolkit was fairly simple due to the modular setup of the Toolkit. A projector module was added to create a projector object and to pass the events to the ASPECTA toolkit's server. Events from the devices are added to the Locator Service and redirected to the projector module.

## **Conclusions**

Creating a large ubiquitous environment with multi-surface and multi-sensor interaction techniques is a complex task. In this chapter, we introduced the SoD Toolkit that reduces the implementation complexity of such environments. The SoD Toolkit builds on previous exploration into ubiquitous environments, such as XDStudio (Nebeling et al, 2014), Conductor (Hamilton and Wigdor, 2014) and Panelrama (Yang and Wigdor, 2014), and extends their approaches to incorporate a more diverse set of devices and sensors, supporting sensor fusion and the ability to implement novel multi-surface interaction techniques. Researchers and developers can build and explore their own designs for ubiquitous environments. The toolkit allows novice researchers and developers to explore new and existing sensors and devices for creating real-world ubiquitous environments while allowing more expert developers to tailor the toolkit to fit their needs (e.g. the sensor fusion). As a result the SoD Toolkit reduces development complexity and overhead.

One of the limitations of this toolkit is the usage of the Kinect hardware, which does not offer the same accuracy in tracking as larger scale, more expensive systems like the Vicon, even with multiple Kinects. However, we believe this is a necessary compromise when developing for a real-world ubiquitous environment. Future work on the SoD Toolkit includes integrating technologies that have device-centric sensors like the Google Tango and the Microsoft HoloLens, for different interaction in a ubiquitous environment.



## **Filling the Space Between: Augmenting Multi-Surface Environments with Low-Resolution Full-Coverage Displays**

*Carl Gutwin, Miguel Nacenta, and Julian Petford*

(Portions of this chapter previously appeared in the following published papers: Miguel A. Nacenta, Regan L. Mandryk, and Carl Gutwin. 2008. Targeting across displayless space. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM, New York, NY, USA, 777-786. DOI=<http://dx.doi.org/10.1145/1357054.1357178>. Robert Xiao, Miguel A. Nacenta, Regan L. Mandryk, Andy Cockburn, and Carl Gutwin. 2011. Ubiquitous cursor: a comparison of direct and indirect pointing feedback in multi-display environments. In Proceedings of Graphics Interface 2011 (GI '11). Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 135-142.)

### **Introduction**

Multi-surface environments (MSEs) are systems in which several display surfaces create a single digital workspace, even though the physical displays themselves are not contiguous. There are many different types of MSE: dual-monitor computers are a simple (and now ubiquitous) example, but more complex environments are also now becoming feasible such as control rooms with multiple monitors in multiple locations, meeting rooms with wall and table displays, or ad-hoc workspaces with laptops and mobile devices.

MSEs are now becoming more common as large displays and mobile devices become increasingly available. The archetype of these environments is the Smart Office, where it is common to see interconnected tablets, wall-mounted displays, laptops, and projected surfaces all being used concurrently and cooperatively (e.g., Benko and Feiner, 2005; Nacenta et al., 2006) (Figure 1).



Figure 1. A Multi-Surface Environment with table, wall display, tablets, and laptop.

One main issue that has arisen as people become more experienced with MSEs is what to do about the space between displays. Many current multi-display interfaces are direct adaptations of single-display designs, and therefore tend to ignore the gaps between surfaces – this is called “warping”. Warping means transporting the cursor directly from one display to another, without moving through the physical space between monitors. Several techniques for warping have been developed, such as stitching (which warps the cursor as it moves across specific edges of different displays) (Hinckley et al., 2004), wormholes (which warp the cursor when it moves into a specific screen region), warp buttons (in which pressing a software or hardware button moves the cursor to each display) (Benko and Feiner, 2007), or named displays (in which the user selects the destination display from a list) (Biehl and Bailey, 2004).

Although warping can be effective, this approach has a number of problems: for example, users must remember an additional mapping which might take time to learn; and with some techniques (such as traditional display stitching) the mappings may become incorrect when the user moves to a new location in the environment. The main problem, however, is that warping techniques disrupt the relationship between motor and visual spaces (i.e., between how the mouse is moved and how the cursor changes position in the physical world) – they ignore the existing (and obvious) arrangement of surfaces in the real world, which force users to learn different ways of moving within and between displays. As a result, warping techniques are distinctly less natural than regular mouse movement: they introduce an extra step into standard targeting actions, make it more difficult for the user to plan and predict the result of ballistic movements, and can cause tracking and interpretation problems for other people in the MSE who are trying to follow the action.

An alternative to warping is to include the space between displays – in fact, the entire interior of the room – in the model of the computational workspace. This idea was introduced by Baudisch and colleagues (2004) as “mouse ether” to deal with the space between dual monitors, but is here extended so that the entire environment is considered to be part of the workspace. The visible parts of the workspace, corresponding to the physical displays, are then arranged based on what the user can see from their current location and perspective. Combining ether and perspective in MSEs provides a workspace in which cursor movement behaves as the user expects, and in which the arrangement of displays corresponds exactly to what the user sees in front of them.

This approach, however, comes at the cost of having to include the ‘ether’ in the digital workspace. This implies that in order to get from one display surface to another, users must move through a displayless region where there is no direct feedback about the location of a cursor. This is not a major problem with ray-casting solutions (e.g., ‘laser pointing’), but does affect indirect pointing devices such as mice or trackpads. Although the displays in the room can provide indirect feedback about cursor location (e.g., using arrows or halos), the large empty spaces in a room-based MSE can make it very difficult for users to navigate between surfaces, because they have to perform (sometimes complex) estimation and inference to determine the cursor’s actual location.

Here we propose a different approach that makes it possible to realize a full perspective-ether solution, that fully recognizes the arrangement of displays in space, and that provides full feedback about the space between displays as well. Our approach uses a full-coverage display to cover the entire inside surface of a room with pixels (e.g., see Figure 2). These kinds of displays have been proposed and discussed many times in HCI – predictions often state that displays will become a commodity that will be purchased by area, and so inexpensive (e.g., “a dollar per square foot”) that we will be able to use them like wallpaper (Welch et al., 2000). These full-coverage displays will provide many benefits and opportunities compared to current monitors and screens, which cover only a small proportion of the environment.

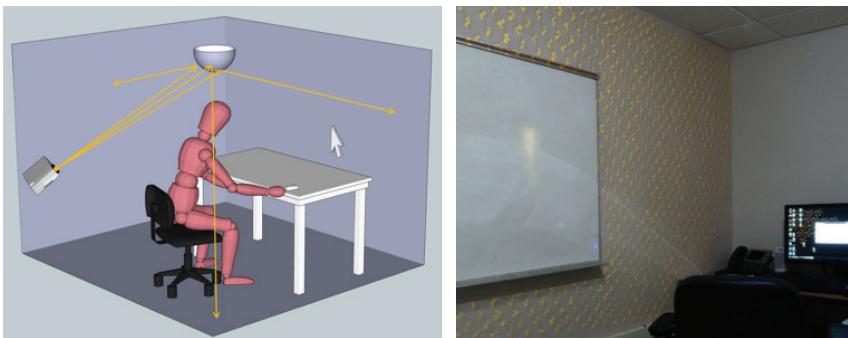


Figure 2. Full-coverage display schematic, and implementation projecting pattern on wall.

Previous work has introduced the idea that one way of covering a room with pixels is to combine a projection system with a dome lens. However, these systems are primarily used for output (e.g., planetariums); in this paper, we look at the use of full-coverage displays as a way to fill the space between displays in a multi-surface environment. Using a simple display system with a standard data projector, a dome mirror costing less than \$100, and middleware to handle display geometry and integration with existing desktop applications, we can provide basic feedback to users of MSEs.

In this chapter we introduce the general problem of space between surfaces, show that this space is important in interactive systems, describe a prototype full-coverage display that can fill the space between, and report on an experiment used to test the value of filling the space between. We conclude by exploring other uses for this novel display technology.

## **Background**

An MDE is a combination of several displays where some kind of interaction can take place across displays. MDEs enable a dramatic increase in the available pixels of an interactive system and have therefore been commonly adapted for commercial desktop systems; they have also been studied by the HCI community for several decades (e.g., Bolt, 1980; Welch et al., 2000). MDEs join together several displays of various types (e.g., monitors, tablets, or tables) into a single logical workspace. Although multi-monitor computers are the most common type of MDE, more complex environments are possible that integrate display surfaces located around a room. Some MDEs take advantage of the different characteristics of the various displays in the environment: e.g., Baudisch and colleagues' (2001) 'focus plus context' display paired a large but low-resolution projector with an inset high-resolution LCD monitor. The system provided large-scale peripheral context as well as detail in the area of focus. Other systems take similar advantage of large displays around the user's workspace to provide context and awareness (e.g., Birnholtz et al., 2010).

One of the obvious operations that needs to take place in an MDE is the movement of visual elements from one display to another. Previous research has introduced a number of techniques to achieve cross-display object transfer, including direct touch (Rekimoto, 1997), world-in-miniature (WIM) representations of the display space (Biehl and Bailey, 2004), laser-pointer based interaction (Myers et al., 2002), head-pose and gaze tracking-based interaction (Ashdown et al., 2005), and mouse-based techniques (e.g., Waldner and Schmalstieg, 2010; see Nacenta et al., 2009 for a survey). Although all technique types have advantages and disadvantages, we decided to focus on techniques exclusively based on mouse operation since the mouse is a common, accessible, and inexpensive device with proven performance, and has several advantages over other technique types; for example, it does not cause the same fatigue or inaccuracy seen in ray-pointing techniques (Jota et al., 2010; Nacenta et al., 2006), it allows access from a distance (unlike direct-contact techniques (Hinckley et al.,

2004)), and it does not require a change in visual context, such as WIM techniques (Biehl and Bailey, 2004).

One of the recognized challenges of interacting with MDEs through indirect input devices such as the mouse is displayless space, the real-world space between displays that cannot represent any information. In previous work, Nacenta and colleagues (2008) showed that in a flat dual-monitor environment, performance diminishes proportionally to the amount of displayless space (following Fitts's Law). This study also showed that when the displayless space is modeled as part of the workspace, performance can be improved with indirect feedback such as Halos (Baudisch and Rosenholtz, 2003); but the best performance was seen when displayless space is ignored (i.e., a warping approach that resembles the standard way that current operating systems connect multiple monitors).

The approach we explore in this chapter is to use a projection-based system that covers the entire interior of the MSE's room with usable pixels. Visions of ubiquitous large-scale display surfaces have appeared for many years in popular media and in HCI research (e.g., Bolt, 1980). For example, Bill Buxton has stated that 100-DPI displays will be as cheap per unit of area as standard whiteboards within a few. Although these kinds of full-coverage display technologies are not yet available, several areas of research have investigated different aspects of this idea.

Several researchers have worked on visualization and inter-action issues for large planar displays (sometimes called 'video walls'). These surfaces are typically constructed of tiled monitors, which can provide very high resolution out-put, but which require novel software for connecting multiple computers and graphics subsystems (e.g., Ebert et al., 2010). A variety of interaction techniques for working with large wall displays have been proposed: for example, techniques such as laser pointers and tracked gestures have been investigated for interacting with displays at a distance (e.g., Robertson et al., 1996). Vogel and Balakrishnan (2004) developed a progression of interaction techniques for different distances from the display, since proximity to the screen changes the ways that people interact with visual information.

Data projectors are currently the most cost-efficient way to achieve large displays (e.g., Johnson and Fuchs, 2007), although resolution is limited unless several displays are combined. Researchers have explored several aspects of projection-based display, including automatic calibration and tiling (e.g., Ebert et al., 2010), perspective correction for different display surfaces (Nacenta, 2006), and using multiple projectors to provide more coverage and combine digital and real-world interactions (e.g., Welch et al., 2000). Commercial systems used in planetariums have also provided full coverage using a dome-shaped screen and a hemispherical projector lens to distribute an image to the interior of the dome.

The development of mobile and handheld projectors has recently enabled new kinds of interactions. For example, research into ‘structured light’ combines cameras or other sensing technologies with pico-projectors to create mobile interactive surfaces that can be used on any flat surface (e.g., Wilson and Benko, 2010). Similarly, steerable projectors have been used as general-purpose display mechanisms for ubiquitous computing and augmented reality (Pinhanez et al., 2003).

### Modeling the Space Between as Part of the Workspace

The idea of including the empty space between displays was first considered by Baudisch and colleagues (2004), who proposed “Mouse Ether” to solve problems in desktop multi-monitor setups, where the monitors are “stitched” together (e.g., moving the mouse out of the right side of one monitor warps the cursor to the left side of the other monitor). By taking into account the actual size of displays and the space between them, Mouse Ether provides a more accurate representation of the physical environment in motor space (i.e., visual space and motor space match better).

Mouse Ether has two main advantages over ordinary monitor stitching. First, it provides a better match between visual space and motor space – the computational workspace now makes explicit use of the physical arrangements of the different surfaces in the MSE – and avoids distracting jumps and trajectory inconsistencies. Second, cross-display movements are consistent with movement within a display, allowing for more natural cross-display transitions. However, Mouse Ether also has an evident drawback: the cursor is invisible when it is in displayless space, and the user lacks visual feedback on its position (Figure 3). We solve this problem by using a full-coverage display, as described below.

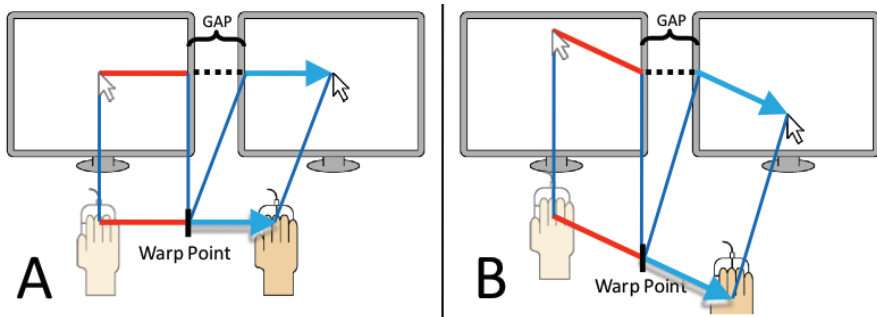


Figure 3. Inconsistency between motor space and visual feedback with Stitching.

- A) The gap is ignored in motor space (gap is compressed into the warp point).
- B) A diagonal motion is transformed into a multi-linear trajectory.

### Using a Full-Coverage Display to Fill the Space Between Surfaces

Low-Resolution Full-Coverage (LRF) displays are display systems that blanket an entire multi-display environment with addressable pixels. Large projector-based display systems have been seen before (e.g., Johnson and Fuchs, 2007), but ours is the first to cover an entire room with a single



static projector. In the LRFC we developed for the study described below, an ordinary data projector is beamed at a hemispherical mirror, which distributes the projector's light around the room (Figure 4, Left). The idea behind LRFC displays is that there are many display tasks in an MDE that are dependent on the physical environment, but that do not need a full-resolution display. Moving between physical displays that are located in different parts of the room is one example.

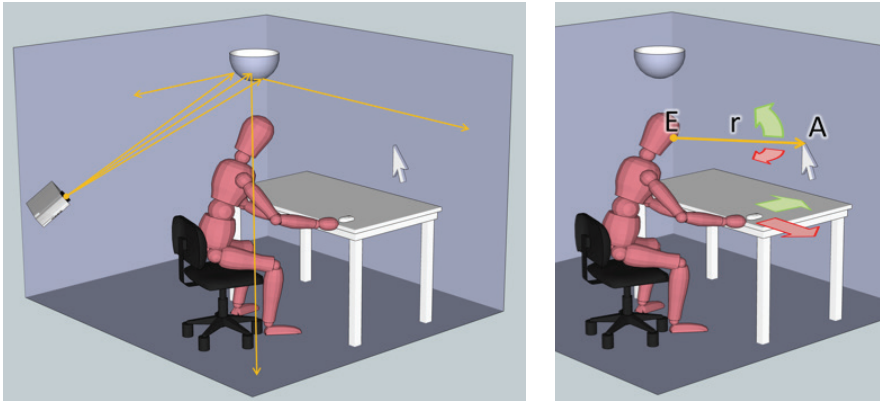


Figure 4. Left: schematic of the full-coverage display. By reflecting onto the spherical mirror, the projector can project onto almost any surface. Right: the movements of the mouse cause a change in the orientation of perspective cursor's defining ray.

Our particular interest in this system is the display of a cursor that can be used to interact with multiple displays arranged around the room. The algorithm to display a cursor has two phases: the calculation of the location of the cursor in physical space, and the reverse mapping of this position to projector coordinates.

To calculate the location of the cursor in the room we use the Perspective Cursor algorithm. The system calculates the ray ( $r$  in Figure 4, Right) that goes through the eye-position ( $E$ ) of the user and is oriented according to the movements of the mouse. Moving the mouse left to right will make the ray rotate clockwise around a vertical axis on the user's eye position (changes the azimuth angle – red arrows in Figure 5). Moving the mouse back to fore will rotate the ray to point more vertically (changes the zenith angle – green arrows in Figure 4).

The ray intersects a 3D model of the room that has been previously provided to the system. In our prototype, the 3D model includes all active displays, the tables, the floor and all the walls of the room.

The first intersection of the ray with one of the surfaces of the model determines the position of the cursor in physical space ( $A$  in Figure 4). If the cursor is located on an active surface, only this display will show it; if the

ray intersects a wall or a non-active table, the position of the cursor in the 3D physical space is passed to the next phase of the algorithm to enable projection on a non-active surface.

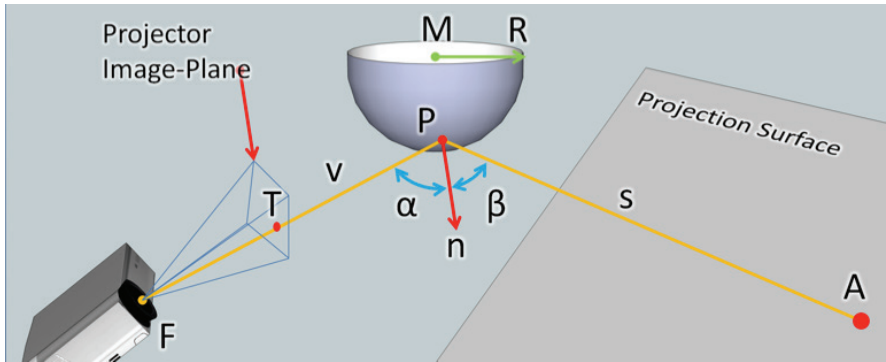


Figure 5. Graphical formulation of the reverse mapping problem.

Now that we know the physical location of the cursor, we need to know how to project onto it. The geometric problem of reverse mapping of the physical position into the image coordinates of the projector is solved by iterative Newtonian approximation. The graphical formulation is illustrated in Figure 5: to project on a given point A, we need to find a point P on the spherical mirror M of radius R such that the angles  $\alpha$  and  $\beta$  formed by lines v (passing through P and the projector's focal point F) and s (passing through P and A) are symmetric with respect to the normal n to the mirror at P. The intersection between the line v that connects F and the calculated P in the image plane of the projector (point T) determines the coordinates in the 2D image of the projector that will project onto A. These constraints are derived from the physical properties of light propagation and mirror reflection.

The process described above can be applied to multiple points to draw polygonal shapes such as the cursor. Unlike related approaches that use steerable projectors or laser pointers, our system can easily project several cursors. Any modern desktop computer can perform the calculations necessary to provide many cursors in real-time.

The size and brightness of the pixels in the room depend on the projector and size of the room. In our test setup, each pixel is approximately 10x7mm; due to differing distances from the projector, pixels are not exactly the same size all around the room. Because a single projector is used to cover the entire room, the brightness of the image is reduced. In our test setup, which uses an ordinary Sony VPL-CX11 1500-lumen projector in a low-light environment, the cursor is easily visible. A more powerful projector would easily be able to display the cursor in either a brighter or a larger room.

The control-display gain for perspective cursor in our system is fully

adjustable. For our experiment we set it so that 3000 mouse pixels translate into 180 degrees for either movement; in other words, the entire field of view has the same mouse resolution as a 3000x3000 pixel display.

We conducted our study with participants in a fixed location, so we were able to achieve perspective effects without real-time head tracking. In a real-world implementation, the location of the user's head must be tracked; this is now becoming possible with low-cost equipment (Nacenta, 2006).

## Evaluation

We compared the effectiveness of a full-coverage solution that showed a cursor between the surfaces to an indirect-feedback solution (modified Halos) and a warping technique (Stitching). We also tested a combined technique that used both UbiCursor and Halos. In the study, participants carried out simple cross-display pointing tasks in an MDE with five displays. We constructed a multi-display environment in a meeting room, using five displays. The room setup is shown in Figure 6.

Participants performed repeated aiming tasks, which always started on one display and ended on another (there were no within-display paths). We tested six paths as shown in Figure 4 (right): A→C, B→C, C→E, E→D, D→B, and A→E. Targets were presented in both directions for all paths (e.g., A→C and C→A). Paths were one of three types: coaxial movements across right-to-left and top-to-bottom seams (B→C, C→E), non-coaxial movements across right-to-top seams (A→C, E→D), and multi-hop movements across intermediate displays (D→B, A→E).

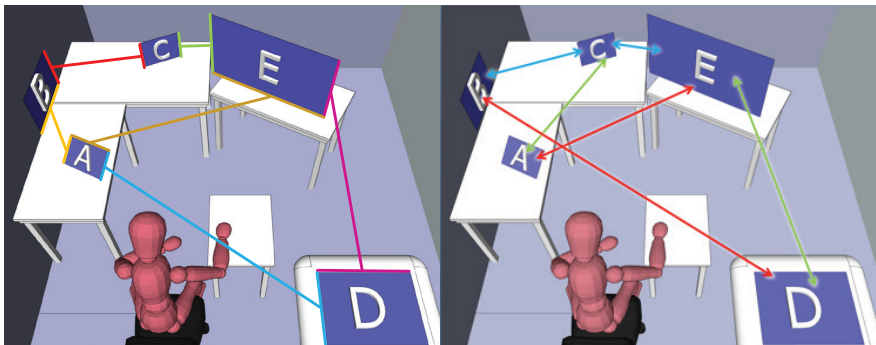


Figure 6. Stitching configuration of the displays (left) and experimental paths (right). Blue: coaxial, Green: non-coaxial, Red: multi-hop.

The aiming task was comprised of an initial selection of the source target, movement to the display containing the destination target, and selection of the destination target. The destination target of a trial and the source target of a subsequent trial were never presented on the same display, requiring participants to move the cursor to a different display between trials. The source target was always presented in the center of the monitor, and the destination target was presented either in the center or at the leading edge

of the display (see Figure 7).

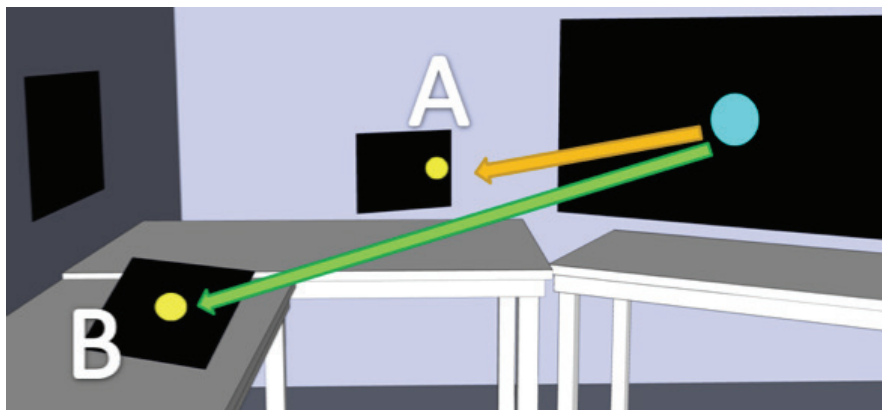


Figure 7. Leading edge target task (A) and Center target task (B).

Sixteen participants carried out the study, and we used a repeated-measures factorial design with three factors:

- Technique (UbiCursor, WedgeHalo, UbiCursor+Halo, Stitching)
- Path (six unique paths in both directions; see Figure 6)
- Target location (leading-edge or center)

The dependent variables, recorded by the study system, were trial completion time and number of errors. We also report on the results of the NASA Task Load Index worksheets completed after each interface condition, and the post-experiment questionnaire.

An omnibus ANOVA with three factors: technique (UbiCursor, WedgeHalo, UbiCursor+WedgeHalo, Stitching), path (12 levels, 6 different display combinations in both directions), and target type (centered on display, or in leading edge), and participant as random factor yielded significant differences on the log-transformed completion times for technique ( $F_{3,42}=8.7$ ,  $p<.001$ ,  $\eta^2=.39$ ), path ( $F_{11,154}=166$ ,  $p<.001$ ,  $\eta^2=.92$ ), target type ( $F_{1,14}=285$ ,  $p<.001$ ,  $\eta^2=.95$ ), and for all fixed factor interactions except technique\*target type ( $F_{3,42}=.52$ ,  $p=.67$ ,  $\eta^2=.04$ ). Logarithmic transformation of the data was required to comply with the normality assumption of the parametric ANOVA.

Post-hoc tests on the technique factor reveal that all average completion times between techniques were statistically different (all  $p<0.006$  after Tukey HSD multi-comparison correction) Averaged across all tasks and target types, UbiCursor is the fastest ( $\mu=1.83s$ ), followed by UbiCursor + WedgeHalo ( $\mu=1.92s$ ), WedgeHalo ( $\mu=1.98s$ ), and Stitching ( $\mu=2.04s$ ). See Figure 8.

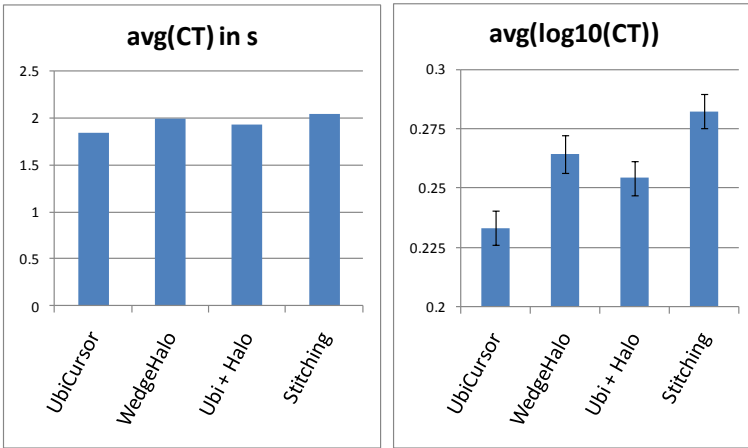


Figure 8. Average completion times by technique in a linear scale (left) and after a log10 transformation (right). Error bars indicate standard error. Note that the vertical scale starts at 0.2 for the log-transformed graph.

The results from the omnibus comparison of techniques generally support H1 (direct feedback is better than indirect feedback); the interaction between technique and path supports H2 (techniques perform differently on different paths). H3 is contradicted by the results since center targets were reached significantly faster than leading edge targets, even though the distance that needs to be covered is larger ( $\mu_{center}=1.8s$ ,  $\mu_{edge}=2.1s$ ).

### Path Analysis

To test H2, H2a, and H2b, we performed ANOVAs equivalent to the global test, but separately for each of the three a-priori groups of tasks (coaxial, non-coaxial, and multi-hop). The results are analogous to the omnibus test results (technique, path and target type  $p < 0.05$ ), except that the technique\*target type interaction was significant for the coaxial tasks (unlike the omnibus and the other task groups). The post-hoc comparisons between techniques yield the same ordering (UbiCursor, UbiCursor+WedgeHalo, WedgeHalo, Stitching), but with fewer statistically significant pairings because of the reduced power of the segmented data analysis (see Table 1).

	coaxial				non-coaxial				Multi-hop			
	UbiCursor	Ubi+Halo	WedgeHalo	Stitching	UbiCursor	Ubi+Halo	WedgeHalo	Stitching	UbiCursor	Ubi+Halo	WedgeHalo	Stitching
UbiCursor	x	0.56	<0.01	<0.01	x	<0.01	<0.01	<0.01	x	<0.01	<0.01	<0.01
Ubi+Halo	d	x	0.11	0.52	d	x	0.22	<0.01	d	x	0.4	<0.01
WedgeHalo	d	d	x	0.98	d	d	x	<0.01	d	d	x	<0.03
Stitching	d	d	d	x	d	d	d	x	d	d	d	x

Table 1. P values of the post-hoc multiple comparison tests for the different path groups (Tukey HSD multiple comparisons, significant if  $< 0.05$ ). Green cells indicate significant, red not-significant, yellow close to significance.

These results generally confirm H2a and contradict H2b (i.e., Stitching did not perform better in any path group) but, more importantly, provide evidence that the grouping of paths we determined a priori is not useful to further differentiate the performance of the different techniques. We address this issue in Section 5.2 (additional analyses).

### Errors

The error counts across participants (see Figure 9) reveal that participants missed the target many more times with Stitching (502 misses, 33 per participant average) than with any of the other techniques (UbiCursor: 354 misses, 23.6 per participant, WedgeHalo: 364 misses, 24.3 per participant, Ubi+Halo: 391 misses, 26 per participant). Notwithstanding the size of the overall differences, a non-parametric Friedman test revealed no significant difference in the number of errors between techniques ( $\chi^2(3)=.568$ ,  $p = .904$ ), possibly due to the large variability in number of errors between participants. Some participants made large numbers of errors with Stitching – up to 86 – whereas for two participants Stitching was the only technique with no errors.

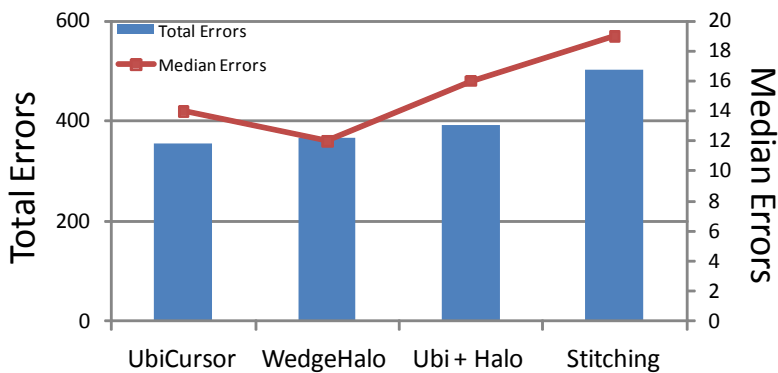


Figure 9. Total and median errors (excluding training).

### Discussion

Our study showed that a direct-feedback, perspective-based technique for supporting cross-display movement (Ubiquitous Cursor) was significantly faster than an indirect-feedback technique (WedgeHalo), a combination technique (Ubi+Halo), or a standard cursor-warping technique (Stitching). In the following sections we explain these results in terms of the main differences between these techniques (direct vs. indirect feedback; perspective vs. warping), and also discuss the limitations of this work and the ways it can be generalized for designers of MDEs.

The main goal of our experiment was to investigate the differences between direct and indirect feedback for mouse-based cross-display targeting. The results of our experiment provide solid evidence for our hypothesis that UbiCursor (a technique with direct targeting feedback) is better than indirect forms of feedback, such as wedges or halos. The difference between our direct and indirect conditions is underscored by the fact that the indirect

feedback technique we tested – WedgeHalo – was optimized for the study in ways that would cause difficulties in real use (e.g., it occludes many pixels on the displays and would be distracting in collaborative work environments).

In addition, the combination of direct and indirect feedback (UbiCursor+WedgeHalo) was not equivalent to UbiCursor alone. Adding indirect feedback appeared to impair performance, possibly due to the extra cognitive load of deciding which type of feedback to pay attention to. This result is relevant for the design of targeting techniques in MDEs because it indicates that, for targeting tasks, more information is not necessarily better.

The empirical study presented in this paper provides further evidence that using an input mapping that corresponds to the user's position (i.e., perspective techniques) is beneficial for performance. Our results tested an MDE where the displays were sparser than in the original Perspective Cursor study (Nacenta et al., 2006). Moreover, our results also help generalize the original findings to variants of perspective where feedback is direct, and to other forms of indirect feedback.

Although we expected our initial classification of paths to shed some light on the differences between techniques, it was only through a new regrouping that we could further learn about the specific strengths of each technique. Our results suggest that Stitching only has an advantage over perspective techniques if the displayless gap is large. In contrast to the planar dual-monitor setup studied by Nacenta and colleagues (2008), where Stitching was the fastest technique even with relatively small gaps between displays, the more complicated transitions between displays in our experiment made perspective mappings a better option, even for short transitions such as the C→E path.

Our additional analysis also suggests that perspective provides an advantage over Stitching for traveling from a small display to a larger display (e.g., C→E), but this advantage is reversed when targeting in the opposite direction (e.g., E→C) because of the 'funneling' effect created by stitching a large screen edge to a smaller one. In perspective techniques, traveling from large to small screens requires reaching the small display within its surrounding displayless space. We believe that this effect may be responsible for the asymmetry in results for paths B→D and D→B, and paths E→D and D→E.

Testing two kinds of target positions within the target display revealed that reaching targets that are close to the leading edge is harder than targets that are centered. This is not surprising for Stitching techniques, which are known to cause overshooting (Nacenta et al., 2008), but was unexpected with the perspective-based techniques (including UbiCursor, which provides direct feedback). This results contradicts linear and angular formulations of Fitts's law (i.e., by definition, leading-edge targets are closer to the starting point and should therefore be faster to reach). We speculate that the visual

transition from background display to foreground display may have caused people difficulty; however, this is a phenomenon that should be investigated in future work.

Combining our findings about displayless space with Nacenta et al.'s (2008) earlier results implies that the targeting geometry of complex MSEs is very different from that of small and large single displays. Designers of multi-display environment interfaces can take this into account: for example, commonly accessed interface elements could be placed at locations that are unlikely to be leading edges (e.g., top center of display E in our configuration), and displays that are frequently used in combination can be located so that they have only a small gap.

We designed our study to test a broad range of targeting transitions that represent a sample of many of the types of targeting tasks that could take place in complex MDEs. For example, the paths that we selected represent transitions from horizontal to vertical displays, from large displays to small, and between displays that are close or distant from each other. This provides a fairly generalizable set of tasks, but makes it difficult to quantify the specific contributions of factors to overall performance. It is therefore necessary to follow up with experiments that are designed to investigate the factors that our study highlighted as most relevant: the effect of angle difference between displays, the threshold at which displayless space becomes detrimental for performance, and the effect of display size differences on targeting. To further generalize the results, it would also be interesting to test tasks with different target sizes.

Finally, the focus of our study was on targeting feedback for mouse-based interaction. Although we believe that mouse interaction will still be predominant for future MDEs, new MDE control techniques from other emerging input paradigms such as multi-touch interaction and free-air gesturing should be designed and tested against perspective mouse interaction.

Although additional work needs to be done to replicate and extend our results, there are several principles and guidelines that can be generalized from our experiences. These ideas will help designers of MDEs understand the issues underlying cross-display targeting performance. First, our results show that stitching becomes problematic in complex MDEs. Although stitching is a simple solution for composing an MDE's workspace, and although stitching outperforms Ether-based approaches in simple setups, this technique becomes more difficult for users when paths do not map easily to a 2D plane. For highly complex MDEs, perspective-based approaches should be considered as a way to simplify cross-display movement. Second, our study shows that direct cross-display feedback works better than indirect feedback. In situations where perspective-based techniques are used, our study shows conclusively that direct feedback improves performance. The low-resolution full-coverage display system that we developed shows that



direct feedback can be provided simply and inexpensively. Third, stitching will still be faster if the real-world distance between displays is large. As distances between displays increases, eventually the advantage of cursor warping overshadows any benefits of perspective-based techniques. If an MDE's displays are very far apart, Stitching will likely be the best choice, although hybrid techniques are also possible.

## **Conclusions**

Multi-display environments present the problem of how to support movement of objects from one display to another. We developed the Ubiquitous Cursor system as a way to provide direct between-display feedback for perspective-based targeting. In a study that compared Ubiquitous Cursor with indirect-feedback Halos and cursor-warping Stitching, we showed that Ubiquitous Cursor was significantly faster than both other approaches. Our work shows the feasibility and the value of providing direct feedback for cross-display movement, and adds to our understanding of the principles underlying targeting performance in MDEs.

Our initial experiences with Ubiquitous Cursor suggest several directions for further research. First, we plan to test the UbiCursor technique with more realistic MDE tasks; in particular, we will explore the effects of having different C:D ratios in the projected display and the MDE displays. Second, we will further investigate the principles uncovered in our study (effects of angle differences between displays, performance thresholds for the different techniques, the effects of different display and target sizes, and the use of the technique with other input devices). Third, we will explore the other possibilities presented by the idea of a low-resolution full-coverage display, which can enable augmentation of and interaction with real-world objects inside the scope of the projected display.



## The Simple Multi-Touch Toolkit

*Kalev Sikes, Zachary Cook, Erik Paluka, Mark Hancock, and Christopher Collins*

### Introduction

The popularity of mobile devices and large interactive displays has brought the touch input paradigm into the limelight. Individuals from various domains are eager to take advantage of the benefits of this interaction style. The problem is that the differences from mouse and keyboard input often create barriers for non-expert programmers to prototype their ideas. The lack of familiarity of the unique requirements for surface application development has inhibited the proliferation of this platform as a medium for research, design, and art. To mitigate this problem, surface computing education needs to be incorporated into the curricula of programs in computer science (CS), information systems, and digital media. In order for this to happen we need tools which can be successfully used by people of different programming skill levels, and which support the rapid prototyping of applications. Existing toolkits for surface development tend to be too complex for non-CS majors to use. In addition, the time required to create a prototype using these toolkits prevents them from being integrated into high paced human-computer interaction courses. To solve this dilemma, we have created the Simple Multi-Touch toolkit (SMT).

With a focus on education and interdisciplinary use, the main goal of our open source toolkit is to simplify the prototyping process for people from differing domains whose programming skill levels range from novice to expert. As a library for the Processing programming language (Reas and Fry, 2006), our toolkit has a simplified syntax and an accessible graphics model. Its high-level nature makes surface development a more inclusive activity and less daunting for beginners. Novices are able to take advantage of its features without knowing CS concepts such as object oriented and event-driven programming. The toolkit is also beneficial for expert programmers since it is highly customizable, efficient, and provides access to low-level input data and graphical primitives.

To further reduce the knowledge and time required to develop surface

applications, SMT is device agnostic through the integration of many input bridges. People no longer have to spend a considerable amount of time customizing their application or use multiple toolkits to develop for different platforms. These design choices have resulted in a robust toolkit that has been used, with success, at multiple universities for developing research prototypes to full-fledged applications. The tool has also been integrated into HCI courses at two universities to facilitate the teaching of prototyping to non-programmers and multi-touch computing to CS students.

Our primary contribution is a simplified software toolkit that can reduce the amount of time required for prototyping by both programmers and non-programmers. We also briefly describe our experiences and resulting insights gained from using this toolkit over the span of two years in HCI courses, as well as for research and application development.

### **Related Work**

With the advent of computer vision frameworks (NUI Group, 2013; Gokcezade et al., 2010; Kaltенbrunner, 2009), the creation of multi-touch systems has become increasingly prevalent. With this rising popularity, researchers have been working on ways to reduce the difficulty of developing for these platforms (Kammer et al., 2010). As a result, multi-touch toolkits for different programming languages have been designed (Hansen et al., 2009; Khandkar et al., 2010; Laufs et al., 2010; Leftheriotis et al., 2012; Luderschmidt et al., 2010; Nebeling and Norrie, 2012). While reducing development complexity is important, supporting rapid prototyping is equally so, as it allows the evaluation of design decisions with minimal effort (Tang et al., 2011) resulting in an improved design process (Olsen, 2007).

To support rapid prototyping in post-WIMP design, König et al. created Squidy, which uses semantic zooming and visual dataflow programming to make development accessible to novices with the ability to provide advanced features when needed (König et al., 2010). T3 is an interactive tabletop toolkit meant for prototyping high-resolution (multi-projector) applications (Tuddenham and Robinson, 2007). To facilitate prototyping interfaces for shared interactive displays, such as interactive tabletops, Shen et al. (2004) developed the DiamondSpin toolkit, which works exclusively with DiamondTouch tables. Specifically focusing on gaming, Marco et al. (2012) created a software toolkit to ease the prototyping of tangible games for vision-based interactive tabletops. Hasen et al. (2009) present the PyMT toolkit, with a specific focus on a new event model to support flexible and creative design of multi-touch widgets and interactions in a post-WIMP environment. Our SMT toolkit similarly supports rapid prototyping of surface applications, but we focus on the Processing model of coding as sketching, and designed it to support teaching multi-touch programming in classroom environments as well as enabling digital media expressivity and creativity.

Pedagogical software toolkits have ranged from teaching students skills related to art (Ariga and Mori, 2010) to more traditional computer science

concepts and skills (Kobayashi et al., 2006; Murshed and Buyya, 2002). Toolkits have been shown to lower the skill barriers for entry and reduce development viscosity when creating user interface applications (Olsen, 2007). For example, Hornecker and Psik (2005) effectively used the ARToolKit to teach students how to prototype tangible interfaces. In this work, we target the Processing programming language to create a toolkit which is useful for both prototyping and education for multi-touch applications. Processing is a high level programming language and development environment designed to enable nontechnical people to use computational methods in the creation of their projects (Reas and Fry, 2006). Our toolkit augments Processing by providing the first comprehensive library of high level methods and features targeted at reducing the complexity of surface development and supporting educators in teaching the fundamentals of surface computing.

### **Design Goals**

The ability to use one's hands and fingers to interact with digital information is a promising technology for a variety of creative applications and interfaces. Hardware supporting collaboration, in the form of tabletop and wall displays, is becoming more common and significant continued growth is expected (Jain, 2014). For a variety of reasons, including variable content orientation, multiple simultaneous inputs, the prevalence of direct manipulation, and a need to support co-located collaboration, traditional WIMP (Windows, Icons, Menus, and Pointers) interfaces are undesirable for many multi-touch usage scenarios. We have designed SMT for non-programmers and programmers alike to be able to rapidly prototype creative and novel interfaces and techniques that make use of multi-touch interaction.

We chose Processing as our target language for several reasons. Processing supports teaching the fundamentals of computer programming, and has been used for this purpose in many different educational contexts around the world, including high school, university, and online courses in visual arts and computer science, and has been downloaded over two million times (Reas and Fry, 2015). The Processing platform already has many powerful graphical libraries, which support the rapid prototyping of beautiful, creative sketches.

It has an easy deployment pathway for installation of libraries directly in the IDE, and a wide variety (e.g., sound, networking, data, math, etc.) of libraries are already available. Processing is built around a flexible programming model supporting three levels of development (Reas and Fry, 2003):

**Simple:** single line programs

**Novice:** hybrid procedural/object-oriented style

**Expert:** full object-oriented (Java) style

In addition to supporting this multi-level coding flexibility, we built the SMT toolkit using the following design objectives, derived from our experiences in teaching modules on multi-touch computing in HCI courses:

*Multi-touch for the masses.* The toolkit was designed to allow people to rapidly create prototypes with little knowledge of programming. We focussed specifically on allowing access to touch interaction and common multi-touch components, without the need for an understanding of object-oriented programming (OOP) or events.

*Ability to sketch multi-touch ideas.* The toolkit was designed to allow for the sketching of multi-touch interfaces and interaction techniques. Specifically, we focussed on minimizing code required to have a working multi-touch interface that enables the testing of design ideas, rather than on polishing the look and feel of interface components or developing a robust application ready for deployment.

*Ability to code multi-touch in a one-hour lab session.* The toolkit was also designed to enable students to go from no experience with multi-touch programming to creating a simple multi-touch interface in a one-hour lab session. Specifically, the toolkit was designed with the intent of allowing courses to focus content on the design aspect of multi-touch, rather than the in-depth programming understanding required to make working multi-touch systems.

*Support for a variety of platforms and inputs.* The toolkit is cross-platform, running on Windows, Mac, and Linux. An Android version is also available but requires a custom build of Processing to use it. SMT was designed to support native (e.g., Windows) touch events, as well as popular input providers such as the TUIO protocol (Kaltenbrunner, 2009). SMT also supports touch emulation using a mouse.

*Support both novices and experts.* The toolkit was designed for use in teaching of HCI courses where students range from students in programs such as visual design or management (“novices”), to fourth year CS students (“experts”). Similarly, the toolkit was designed to support quick sketching of small ideas (e.g., lab assignments) as well as development of large projects (e.g., graduate student research or interactive artwork). This was achieved through a flexible syntax in which there are multiple avenues for achieving the same result.

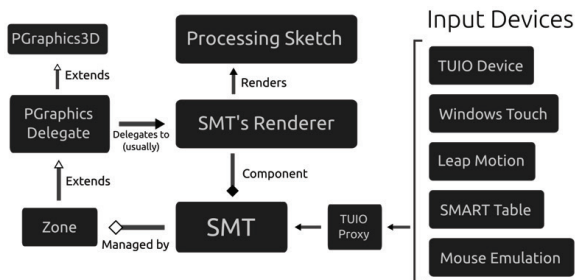


Figure 1. Overview of the architecture of the Simple Multi-Touch Toolkit. The Processing Sketch is written by the student or designer after importing the SMT library, which provides input handling and rendering capabilities.

## The Simple Multi-Touch Toolkit

Following our design guidelines, the SMT toolkit integrates with the styles of programming supported by Processing. The central construct of SMT is a new display and interaction primitive called the Zone. SMT also provides back-end support for a variety of input devices, handling touch events and providing them to applications using a common Touch construct. The accompanying website offers documentation, including complete JavaDoc and a full suite of tutorials and teaching materials.

### Zones

The Zone is the central concept of the toolkit (Figure 1, bottom left). Zones are similar to Windows or Panels from other windowing toolkits, with the important difference that, as graphical primitives, they are not limited to assumptions such a predefined direction/shape/scale or interaction through a single mouse and keyboard. Moreover, they are designed to be understandable without an in-depth understanding of object-oriented programming, messaging, or callback functions. Each zone defines a drawable and touchable artifact in the programmer's sketch. Zones can be customized to accomplish a variety of interface goals. The most important and common modifications, changing how a zone draws and what happens when it is touched, have special support from the toolkit. Zones can be nested, which permits the creation of more complex user interface elements, such as toolbars and menus (which SMT also provides).

```
1 |
2 |
3 | void setup() {
4 |   size(640, 360);
5 |
6 |   noStroke();
7 |   rectMode(CENTER);
8 | }
9 |
10 | void draw() {
11 |   background(51);
12 |   fill(255, 204);
13 |
14 |   rect(mouseX, height/2,
15 |        mouseX/2+10, mouseY/2+10);
16 |   fill(255, 204);
17 |
18 |   int inverseX = width-mouseX;
19 |   int inverseY = height-mouseY;
20 |   rect(inverseX, height/2,
21 |        (inverseY/2)+10,
22 |        (inverseY/2)+10);
23 | }
24 |

1 | import vialab.SMT.*;
2 |
3 | void setup() {
4 |   size(640, 360, SMT.RENDERER);
5 |   SMT.init(this);
6 |   noStroke();
7 |   rectMode(CENTER);
8 | }
9 |
10 | void draw() {
11 |   background(51);
12 |   fill(255, 204);
13 |
14 |   for (Touch t : SMT.getTouches()) {
15 |     rect(t.x, height/2,
16 |          t.y/2+10, t.y/2+10);
17 |     fill(255, 204);
18 |     int inverseX = width-t.x;
19 |     int inverseY = height-t.y;
20 |     rect(inverseX, height/2,
21 |          (inverseY/2)+10,
22 |          (inverseY/2)+10);
23 |   }
24 | }
```

Figure 2. An example from Processing's website (<https://processing.org/examples/mouse2d.html>) to demonstrate mouse use (left), converted to support multiple touches using SMT (right).

*Zone Methods.* There are two critical methods that must be implemented for each zone. These are the draw method (adapted from Processing) and the touch method (introduced in SMT). There are two different styles in which these methods can be written—procedurally and using object-oriented programming (OOP). These two styles mimic the approaches taken by

Processing and Java, respectively. We discuss how we incorporated both styles into SMT later in this report. To implement these methods using OOP, the traditional approach of overriding the methods in a class that inherits from the Zone class is used:

```
class MyZone extends Zone {
    void draw() {
        // ... drawing code
    }
    void touch() {
        // ... touch handling code
    }
}
```

To implement these methods procedurally, one would first create the zone with a string-based name:

```
void setup() {
    // ...
    SMT.addZone("MyZone");
}
```

And then define a method in the processing sketch by appending the zone's name. For example, to implement the draw method, one would write the method:

```
void drawMyZone(Zone zone) { ... }
```

To implement the touch method for same zone, one would write the method:

```
void touchMyZone(Zone zone) { ... }
```

When both a procedural and object-oriented implementation are detected for the same zone name and method, the procedural one is selected and invoked by the toolkit.

*Nesting.* An important principle in user interface design is the nesting of elements. SMT supports this principle by permitting zones to be nested in parent-child relationships. This is done by having the child zones inherit their parent's transformation matrix. If the parent is rotated, scaled, or translated, the child will be rotated, scaled, or translated along with it.

## Touch Input

SMT supports all the most common desktop touch input devices (Figure 1, right). This includes TUIO devices, Windows Touch, SMART Tables, and Leap Motion. Each of these touch event sources are optional and can be used in any desired combination. Since each of these devices provides events in a different way, they must be unified in some manner. SMT handles this by

converting all input into the TUIO protocol. SMT then wraps the underlying TUIO cursor object with a convenient Touch class which provides the user with an abstract handle to touches that is both easy to understand and use. For example, to make any Processing sketch touch-capable, one need only add a few lines of code (Figure 2).

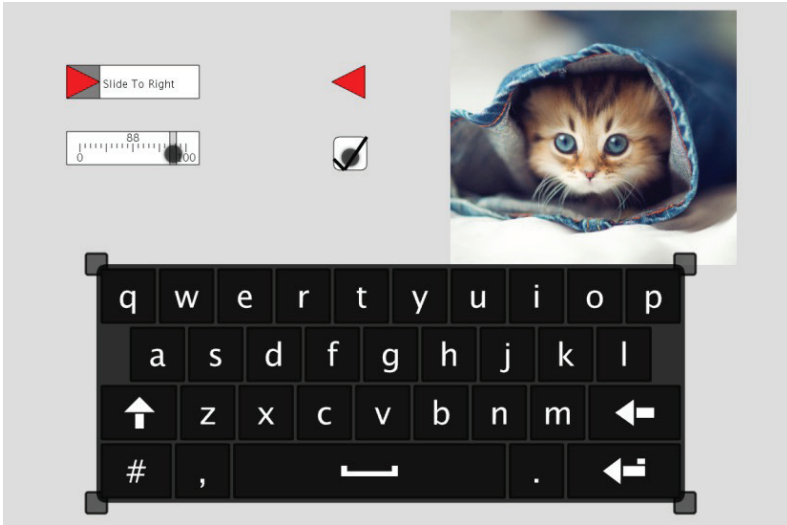


Figure 3. Various interface components provided as Zones in SMT.

While we have designed processing of touch events to closely resemble mouse handling in Processing, we have also provided several techniques for conveniently enabling common multi-touch interaction techniques, such as rotation, translation, and scaling. For example, to implement the common RST (rotate-scale-translate) method on any zone, one would write:

```
void touchMyZone (Zone zone) {  
    zone.rst ();  
}
```

*Implementation Details.* After conversion into the TUIO format, touches are assigned to zones using standard colour picking. The definition of picking bounds is actually done with a zone method in the same form as the draw and touch methods previously discussed. Special care has been taken in the development of SMT to prevent colour calls and similar erroneous call from being made within this picking method. After touches have been assigned to their zones, a group of methods that correspond to the main types of touch events are invoked. These methods can also be defined in the procedural or object-oriented form. Finally, the touch method is invoked, within which there are a number of predefined standard gestures that can be used, such as drag, pinch, rotate, and scale.

### Common Interface Components

In addition to providing support for programmer-drawn zones and low-level



touch handling, we provide several common interface components that can be added in the same way as any other zone. For instance, we provide support for tabs (`TabZone`), buttons (`ButtonZone`), sliders (`SliderZone`, `SlideRevealZone`, `PatternUnlockZone`), checkboxes (`CheckBoxZone`), menus (`PieMenuZone` and `LeftPopUpMenuZone`), keyboards (`KeyboardZone`), and many other common interface components (for a total of 21 zones). Figure 3 shows several of these components rendered in an SMT sketch.

Many of these components are made interactive through methods that can again be overridden in a child class (OOP) or directly in the sketch through a named method (procedural). For example:

```
void setup() {  
  // ...  
  SMT.add(new ButtonZone("My Button"));  
}  
void pressMyButton() {  
  // ... respond to button press  
}
```

### Development and Debugging Tools

*Multi-touch Emulation.* Not all development machines necessarily have touch input methods. In order to support the development and testing of SMT sketches on machines lacking such input devices, we implemented a convenient way of emulating multi-touch with just a mouse. The system is fairly simple: the left mouse button emulates a temporary touch, and the right mouse button emulates a touch that lingers. Touches created with the left mouse button will only stay as long as the mouse button is held down. Conversely, touches created with the right mouse button will remain after the mouse button is released. At this point, these lingering touches can either be moved around with the left mouse button, or removed by right-clicking them again. Any number of touches can be created, but only one can be moved at a time with the mouse.

*Procedural Programming Warnings.* The procedural-style zone methods must follow a fairly specific form in order to be detected and invoked properly. Since mistakes in following this form are easy to make, SMT provides a number of warnings to help guide the user. For example, when a method is detected with one of the zone method prefixes, but the rest of the method name does not match any known zones, it is likely that the user simply misspelled the name of one of their zones, so SMT prints a warning.

*Documentation.* In this vein, SMT's website covers most of the bases. In addition to recent release information, the website hosts SMT's JavaDocs as well as a full suite of tutorials, examples, and a Processing-style reference page. The tutorials start with the basic concepts, then covers all the important more advanced concepts, including various visual customizations, how to make viewports, and how to transition code from the procedural style to

the object-oriented style.

## **Programming with SMT**

In this section we demonstrate through examples how SMT supports both novice and expert coding styles in a manner which is harmonious with the norms in the Processing programming language. Many of these examples are also available in the tutorials section of the SMT website.

## **Supporting Different Programming Styles**

We support two main styles of development, novice and expert. Statements in each style can be interleaved in the same application, giving maximal flexibility to developers. The novice style is a hybrid of procedural and object-oriented programming (OOP), minimizing use of OOP concepts such as event processing, constructors, and object inheritance. The expert style is standard OOP. In addition, developers may use the Processing IDE (best suited for novices) or whichever development environment they prefer (e.g. Eclipse, best suited for experts). For example, the standard Java statement `SMT.add(new Zone("MyZone", 100, 200, 50, 60));` can be rewritten as `SMT.addZone("MyZone", 100, 200, 50, 60);` in the novice style. Note that due to the constraint that all Processing sketches must extend `PApplet` ("Processing Applet"), we are unable to make methods available to developers without requiring the SMT. prefix.

In the following example we demonstrate how to create a simple, highly responsive application which renders a custom "happy face" Zone that supports multi-touch rotate, translate, and scale. The code is written using the Processing hybrid procedural/object-oriented style for novices (Figure 4, left) and using traditional object-oriented style for experts (Figure 4, right).

In both examples, the sketch is initialized with the import statement from SMT, which is provided automatically by Processing when the library is included in the IDE. The setup method is common to all Processing sketches. In SMT it must include a call specifying the initial window size and selecting the SMT renderer, which inserts SMT zone management into the Processing rendering queue. SMT is then initialized. In this example, a single zone called "MyZone" is added to the sketch. In the novice style, the zone is added by naming it in the `addZone` call, and subsequent draw and touch methods reference the specified name using reflection. That is, the novice can create a method called `drawMyZone` and it will be invoked appropriately to render the zone. Some people, especially those used to Java and object-oriented programming, can find SMT's reflection-invoked methods non-intuitive. Thus, in the expert style, an inner class called `MyZone` is created using the `add` method and has its own draw and touch methods.

```

1 import vialab.SMT.*;
2
3 void setup(){
4 // Processing and SMT setup
5 size(800, 600, SMT.RENDERER);
6 SMT.init(this);
7 textFont(createFont(
8 "Droid Sans Bold", 64));
9
10 // add our zone to the sketch
11 SMT.addZone("MyZone",
12 300, 200, 200, 200);
13 }
14
15 void draw(){
16 background(220);
17 }
18
19
20
21
22
23
24 void drawMyZone(Zone myZone){
25 fill(100, 150, 60, 200);
26 ellipse(100, 100, 200, 200);
27 fill(10, 220);
28 textAlign(CENTER, CENTER);
29 textMode(SHAPE);
30 text("^_^", 100, 90);
31 }
32
33 void touchMyZone(Zone myZone){
34 myZone.rst();
35 }
36 }

```

```

1 import vialab.SMT.*;
2
3 void setup(){
4 // Processing and SMT setup
5 size(800, 600, SMT.RENDERER);
6 SMT.init(this);
7 textFont(createFont(
8 "Droid Sans Bold", 64));
9
10 // add our zone to the sketch
11 SMT.add(new MyZone());
12 }
13
14
15 void draw(){
16 background(220);
17 }
18
19 class MyZone extends Zone {
20 public MyZone(){
21 super(300, 200, 200, 200);
22 }
23
24 public void draw(){
25 fill(100, 150, 60, 200);
26 ellipse(100, 100, 200, 200);
27 fill(10, 220);
28 textAlign(CENTER, CENTER);
29 textMode(SHAPE);
30 text("^_^", 100, 90);
31 }
32
33 public void touch(){
34 rst();
35 }
36 }

```

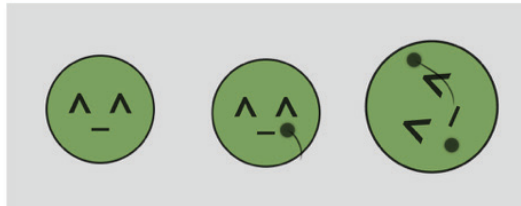


Figure 4. Code for creating the “happy face” example, using novice (i.e., more procedural) approach on left, and expert (OOP) approach on right, and the resulting sketch (bottom).

## Examples

In this section, we introduce Processing sketches built with the SMT toolkit. To support a learning-by-example style of learning, as requested by students in the first in-class deployment, the SMT library in Processing comes with more than 25 example sketches which illustrate each Zone type and method. In addition, we provide 4 sketches corresponding to online tutorials, and 12 fully realized demonstration applications, including a photo organizing application, a checkers game, a login screen, and a table hockey game. We will discuss the table hockey example below.

The table hockey demonstration application was made by an intern within their first week working with SMT. The 311-line sketch produces a simple two-player table hockey game, designed to be played on a multi-touch table display. Each of the pucks are SMT Zones. All the pucks could

theoretically be handled at the same time, as long as the touch devices being used can handle that many touches. Pucks can be tossed across the game board at variable velocities. To demonstrate Zone manipulation, a 160-line custom physics engine manages collisions between pucks and with board boundaries, but this could also be accomplished with a third party physics library.

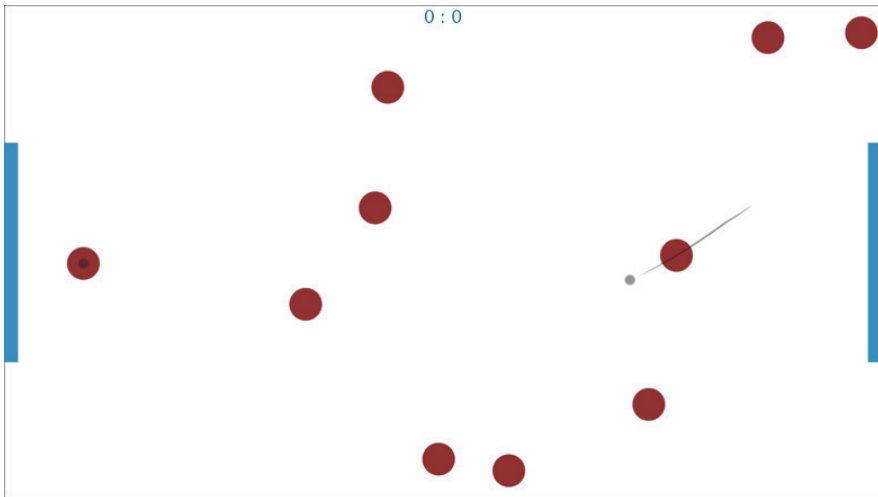


Figure 5. A table hockey game written with SMT.

### Initial Evaluation

After the initial phase of development on SMT, we deployed it in two HCI classes for students to use in laboratory activities and in the development of term-long group projects. We studied the deployment of the toolkit through student feedback surveys and analysis of completed student projects. The goals of the study were to investigate whether SMT was useful for prototyping multi-touch applications, accessible to novices, and powerful for experts. In particular, we sought to understand the speed of the development cycle and whether students became comfortable with rapid prototyping (sketching) using SMT during their brief exposure to it.

### Method

Participants were recruited from two HCI courses at two separate universities. At one of the universities, the HCI course was being taught to mainly management sciences students who had relatively little experience with programming ("novices"). At the other university, the students were fourth year computer science and software engineering students ("experts"). The idea behind this approach was to show separately how both novice and intermediate programmers responded to SMT.

After their first lab session using SMT, the students of these courses were asked to fill out a questionnaire on the toolkit. After their last lab session using SMT (6 weeks later), they were asked fill out the same questionnaire

again. Students were invited to grant permission to use the code and images of their project for the purpose of analysis of the toolkit. The questionnaire was based on “A Cognitive Dimensions Questionnaire” (Blackwell and Green, 2007), a standardized framework for analyzing the usability of information artifacts, in particular software systems (Blackwell, 2015). All data was collected by a third party and retained until after final grades were submitted to ensure separation of the study and the course outcomes.

We received a total of 22 responses to the first round deployment of the questionnaire, but only 1 response to the second round. At one of the universities, no students completed the questionnaire. Thus, all responses we received were from the “experts” group. This made the intended comparison between the four sets of responses infeasible. Results below refer only to the first administration at one university. Four (out of 14) groups in the computer science class gave unanimous permission to evaluate their projects for the purposes of this study.

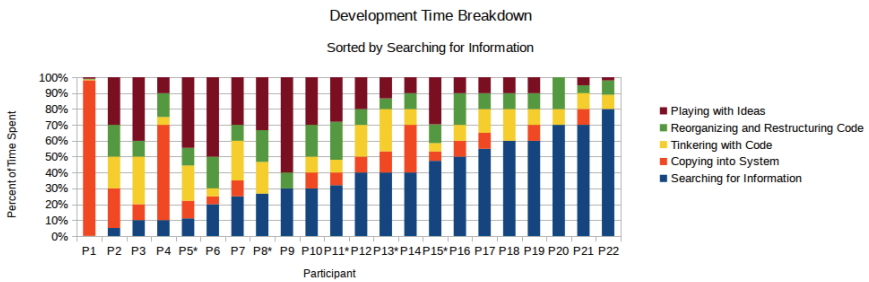


Figure 6. Breakdown of time spent, sorted by time spent searching for information.

### Questionnaire Results

Below we discuss the results of the three sections of the questionnaire: time using SMT, questions about usability of the API, and suggestions for improvement.

*Time.* Out of the 22 participants who completed the questionnaire, 19 had only spent 1-2 hours working with the toolkit. The other three participants all had spent 3-5 hours working with the toolkit.

A series of questions asked about fraction of time spent on each type of development activity that can occur while using a notation. It was intended, but not enforced, that the sum of each response would be 100%. Figure 6 shows how each participant estimated the time they spent on the various types of development activities they undertook while working with SMT. The participants are sorted by their answer to the first question. Participants whose responses did not add up to 100% have been normalized and are marked with asterisks.

The results show a marked variance in activities undertaken with the toolkit.

Given that these results come from after only a short time using SMT, it makes sense that, for many participants, searching for information and copying code examples into the system were dominant tasks. For eight participants, more than half the time was spent on the core prototyping activities of tinkering with code and playing with ideas.

*Questions about API Usability.* Table 1 shows the response breakdown for a series of questions related to the various features of SMT. First impressions of the students indicate that they thought SMT was easy to use (Q1, Q2, Q4), succinct (Q3), predictable and transparent (Q6, Q7, Q8, Q12), and flexible (Q9, Q10, Q11, Q13). Comments included “The concept of zones and sub zones does work well and provides an easy hierarchy to follow” (Q6) and “[It is] easy to have a short development cycle with save and run” (Q9). There is evidence that some students found it easy to make errors or slips (Q5), indicating our error checking and compile-time warnings could be improved. In particular, several participants lamented the lack of a standard debugger in Processing. Also, students indicated that they did not use the toolkit in new and different ways (Q14, Q15), which was likely due to the brevity of their experience with it.

	Question	Yes	No
Q1	Is it easy to see or find the various parts of your code?	16	6
Q2	Is it easy to make changes?	19	3
Q3	Is the toolkit succinct?	15	7
Q4	Do some things require hard mental effort? If yes, what things?	7	15
Q5	Is it easy to make errors or slips? If yes, what errors or slips?	13	9
Q6	Is the notation (API) closely related to the result?	19	3
Q7	When reading your code, is it easy to tell what each part is for?	16	6
Q8	Are dependencies visible?	10	12
Q9	Is it easy to stop and check your work so far?	18	4
Q10	Is it possible to play around with ideas?	21	1
Q11	Can you work in any order you like?	16	6
Q12	Is the toolkit consistent in the ways to use it?	16	6
Q13	Can you make informal notes (comments) to yourself?	18	4
Q14	Can you define new terms or features?	12	10
Q15	Do you use the toolkit in unusual ways?	2	20

Table 1. Responses to the questions asked in our questionnaire.

*Suggestions.* Twelve participants responded with specific suggestions for improvement of SMT. Seven of the responses in some way requested better documentation, often specifically requesting example-based documentation. Two of the responses recommended changing SMT to better follow object-oriented design. Two responses requested features from more a complete IDEs like Eclipse (which was related to the Processing environment and not SMT). Two responses were generally positive

comments, e.g. “nice and adequately built toolkit”. One response was a specific feature request for improvements to the zone rotation process.

*Student Projects.* The projects the students completed as part of their course mainly involved the design of a prototype user interface. Various methods of design were taught and encouraged, including sketches and storyboards, paper prototypes, and software prototypes (created in Processing). The software prototypes used SMT to manage the touch interactions in the user interface. Prototypes developed with SMT ranged across a wide variety of topics, including mobile workout coaching for a phone-sized device and transit planning for a wall display, demonstrating the flexibility of SMT across domains and hardware.

*Discussion.* There were pragmatic challenges in running a classroom-based study in our own classrooms. One issue was that we were not granted approval under research ethics to incentivize our participants in any way, including through means unrelated to the course, such as monetary remuneration. In addition, we did not allocate class time for the administration of the study. Thus, requesting students to complete an optional and anonymous questionnaire on their own time with no reward contributed to our low response rate. We also hypothesize that the specific design of some of the questions, based on the Cognitive Dimensions model, may have intimidated students due to unfamiliar language referring to “notations”.

The suggestions received in the questionnaire likely reflect participants’ enrollment in a traditional computer-science program: they expected a powerful IDE and object-oriented style. To respond to these, we improved the documentation and the curriculum to explicitly help advanced students work within Eclipse in an object-oriented style if they chose to do so. We improved SMT and its documentation based on student feedback and several months of refinement with our users through the open source deployment before offering one of the courses again, with a revised study, as discussed in the next section.

### **Follow-up Evaluation**

Based on experiences with the first use of SMT in teaching multi-touch for human-computer interaction, we made many improvements to the toolkit, including extensive documentation, online tutorials, and examples which illustrated each Zone type and method. Advanced examples and tutorials illustrated functionalities such as custom picking and viewports. In addition, we simplified our study method and conducted a second round of evaluation at one university (the second university was not offering the course).

### **Method**

Participants were recruited from a fourth year computer science course (the same course for which results of the first study are reported). At the end of the semester, after two laboratory activities using SMT, and after using SMT to create prototypes for their term project, a questionnaire which focused

on ease of learning SMT was administered. Demographic data on years of experience and self-rated programming skill was collected.

Again, students were invited to grant permission to use the code and images from their final group project, with optional acknowledgement to them, for the purpose of analysis of the toolkit. We received a total of 18 responses to the questionnaire and 7 groups provided unanimous permission to evaluate their projects for the purposes of this study.

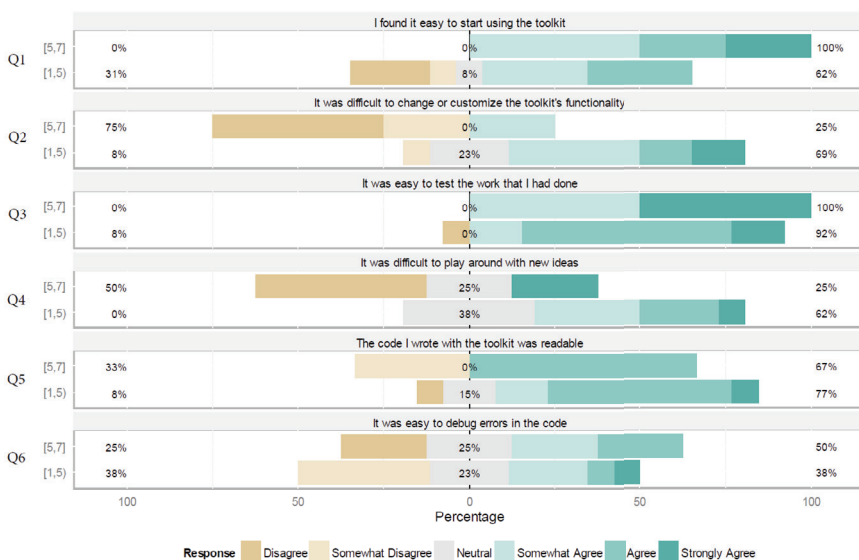


Figure 7. Ease-of-coding questionnaire results from follow-up evaluation. Participants are grouped by skill level as indicated on the left.

## Results and Discussion

Students indicated an average of 19 hours experience with SMT (min: 4, max: 80). The large spread is expected as they were using SMT as part of a large group project where greater coding responsibilities may have been delegated to some students. Our questionnaire contained a series of questions investigating how long it took to learn the toolkit (from one hour to several months). All but three students indicated “agree” or “strongly agree” with feeling comfortable using the toolkit after a few hours, and all students but one were comfortable after a day. The one remaining student (skill level=4, hours of use=40) indicated “neutral” for all time periods. In the analysis which follows, we divided students into two groups: novice (self-rated 1–4, n=8) and expert (self-rated 5–7, n=9). A summary of questionnaire results by skill level is found in Figure 7.

Q1 indicated that most students of all skill levels found it easy to start using the toolkit. Q2 shows a split, with novices expressing more challenge with customizing and changing the toolkits functionality. This is not concerning as 75% of experts did not find it difficult, and this is an advanced function



which normally would not be used by novices. Q3 showed that all students found it easy to test their work. Q4 again reveals a split between 25% of experts who had some difficulty playing with new ideas, and 64% of novices who had some difficulty. Both experts and novices found the code readable (Q5). The results on ease of debugging were similar between groups, with around 30% indicating some difficulty debugging. 14 students provided specific suggestions for improvement. Of these, 6 corresponded to the Processing IDE (e.g. desire for code completion). 8 comments related to feature and improvement suggestions for SMT, with 7 students suggesting further improvements to the online documentation, including coded examples for every method.

Students created a wide variety of prototype applications for multi-touch table and wall displays, including transit planning, digital board games, personal health monitoring, and a prototype public display providing access to outreach services for the homeless, pictured in Figure 8.



Figure 8. Example screens from a student term project created with SMT, showing a public kiosk interface to provide information about services for the homeless.

Overall, the results of our questionnaire indicate student satisfaction with SMT across skill levels. Concerns around ease of debugging likely relate to the use of runtime warnings (e.g. if a student creates a zone called "OKButtonZone" without the "drawOKButtonZone" method, a warning is generated at run-time instead of compile time). This is due to the use of reflection and the capabilities of the Processing IDE. Students self-rating as novice did also indicate some difficulty playing with new ideas, revealing that for this group, further improvements to code simplicity and training materials are needed. We leave this for future research.

### Real-World Use

SMT has been developed and actively supported for two years, during which it has been used in two human-computer interaction and interface

design courses at two different universities across multiple semesters (for a total of four classes) to help students learn to develop medium fidelity prototypes of their multi-touch designs. Our toolkit has also been used for the development of research prototypes in at least six graduate student projects.

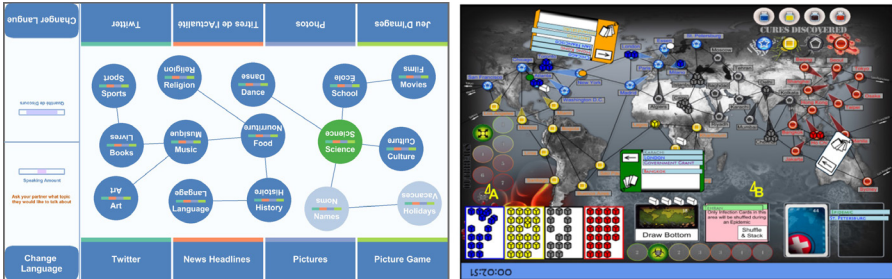


Figure 9. Graduate research projects TandemTable (left) and, Pandemic (right), created with SMT.

Feedback from the use in these real-world projects has been mostly positive, with students able to create interesting and sophisticated multi-touch designs, while not requiring significant in-class time to learn how to program. Students were instead able to focus their learning on design methods and evaluation techniques. Graduate students commented on the ease with which they could rapidly prototype, mentioning how in most cases the development took far less time than their previous experience with Application Programming Interfaces such as Windows Presentation Framework and C#.

We feature two graduate student research projects using SMT in Figure 9. The first is an assistive application for the tandem language learning method. It was developed in order to study how interactive tables can be used to augment the language learning process (Paluka and Collins, 2015). The second is a multi-touch implementation of the Pandemic board game (Chang et al., 2014). It was developed in order to study how knowledge of past game events may change people’s strategies and behaviors while playing turn-based games. SMT has also been used to develop and publish a multi-touch visualization application by a research group not affiliated with the SMT authors (Dai et al., 2015). While the graphical rendering capabilities of the Processing environment were helpful to these projects, specific features of SMT were also critical to their success.

SMT is a free and open source library. Its codebase is currently hosted on GitHub at <http://github.com/vialab/smt>. Being hosted in a public and easily accessible venue holds many benefits to a toolkit. One of these benefits is the feedback and input from people from all over the world, whom we would never otherwise have been given the opportunity to interact with. SMT’s GitHub page regularly receives 60+ unique visitors per week. Not including the authors, SMT’s GitHub page has been followed by 26 people,

and starred by 31. In addition to this, we have received and dealt with many bug reports and feature requests from users around the world.

### **Conclusion and Future Work**

We have created the Simple Multi-touch Toolkit, which is a simplified software toolkit for the Processing programming language. It is designed to reduce the amount of knowledge required and the complexity involved in programming multi-touch applications. Although our toolkit simplifies multi-touch programming, seasoned developers are afforded many advanced additional features, such as access to more low-level data structures and many customization features. By combining SMT with a mouse-based multi-touch emulator, users are able to develop their applications on machines without interactive surfaces, which then run seamlessly on touch-enabled surfaces. Cross-platform development is enabled through the integration of multiple input bridges and native TUIO support. SMT has been successfully used at multiple universities for developing research prototypes as well as full-fledged applications. The toolkit has also been used in courses at these universities for teaching concepts and skills related to HCI. Our web resources include tutorials and teaching materials for using SMT in the classroom and we will continue to support its use in teaching and research environments. In the future, we plan to incorporate additional support for more complex multi-touch gestures, to add automatic layout algorithms for creating interfaces with multiple Zones, and deploy SMT for Android, which is currently in private alpha development stage.





# **SURFACE APPLICATIONS**





## Surface Applications

*Frank Maurer, University of Calgary*

### **I**ntrouction

SurfNet's fundamental research was guided by the needs of industrial applications. Applications also provided test beds and case studies for the research conduct by the SurfNet team. The application areas were developed in collaboration with industry partners, and provided promising vertical markets for digital SurfNet researchers were working with industrial partners on the following application areas:

- Health Technologies
- Planning, Monitoring and Control Environments
- Learning, Gaming, New Media and Digital Homes
- Software Team Rooms

To illustrate the network's contributions to on the application side, we selected seven contributions for this book.

Innovative Health technologies promise to improve patient care while reducing service costs. Digital surface technologies can be used in all areas of health care, from hospital settings during treatments of conditions over telehealth applications to provide medical services over a distance to the prevention of illnesses through encouraging healthier behavior.

- *Radiology Image Scrolling*
- *Towards At-Home Physiotherapy: Next Generation Teleconferencing and Surface Based Interventions*

- *Discouraging Sedantry Behaviors Using Interactive Play*

Planning, monitoring, and control activities often require people to combine their expertise while working over a shared workspace. Surfaces allow the group to simultaneously view and manipulate large, multi-dimensional data. This improves collaborative decision-making and problem solving in complex, time- and safety-critical environments.

- *OrMiS: Use of Digital Surface for Simulation-Based Training*
- *TableNOC: Touch-Enabled Geo-Temporal Visualization for Network Operations Centers*

Digital surfaces create new opportunities for learning, gaming, and other new media applications. Some of our partners were interested in social gaming, serious games, and game play in MSEs. Others are concerned with educational applications for digital surfaces. Large digital surfaces are increasingly present in the home and public spaces.

- *Beyond Efficiency: Intriguing Interaction for Large Displays in Public Spaces*

Developing software is an artefact-driven and highly collaborative activity that is increasingly geographically distributed. MSEs create visual workspace surfaces linked together in spite of distance barriers. The promise is that distant-separated teams can collaborate more productively, and can thus develop better quality software. Initial industrial partners are Pyxis Technologies, Bederra, and CAE Professional Services.

- *Surface Applications for Security Analysis*





## Radiology Image Scrolling

*Louise Oram, Philippe Kruchten, and Karon MacLean*

### Introduction

To utilize the detailed information provided by today's high-resolution image capture technologies, such as Magnetic Resonance Imaging (MRI) and Computed Tomography (CT), radiologists must examine ever-larger image sets. It is not uncommon for multi-trauma CT scans or coronary CT angiograms to have data sets of 4000 images (Andriole et al., 2011). Diagnosis entails a complex, time-pressured visual search task, where target conspicuity, background clutter and other attentional factors can influence the radiologist's ability to detect anomalies (Andriole et al., 2011), and radiologists are put at substantial risk of repetitive strain injury (Goyal et al., 2009).



Figure 1. Sketch showing the idea of a image slices creating a stack.

Radiology images are currently mostly viewed as single 2D slices (Andriole et al., 2011, Atkins et al., 2009), arranged in a stack through which is scrolled through depthwise. The main interaction tool is generally a scrollwheel mouse, which is basically unchanged since 1995. Image stacks have evolved towards continuous media streams from their humble beginnings as single x-ray images. Efficient perusal demands fluid, controllable interaction akin to video scrubbing (Matejka et al., 2013), as has been demonstrated with a haptic scrollwheel (Snibbe et al., 2001).

Meanwhile, the daunting scope of the image-viewing task makes it a candidate for semi-automation, e.g. computer-aided detection (CAD) of anomalies in images (Doi, 2005). Such algorithms are tuned to find all real anomalies (true positives) at the cost of substantial rates of false positives, which radiologists must then distinguish. As it takes 5-7 seconds to re-evaluate a CAD-identified nodule (Rubin et al., 2005) there is clearly a cost to potential time and accuracy gains. Similar issues exist for annotations from other sources, e.g. other radiologists, in redundant procedures and peer reviews or training reviews.

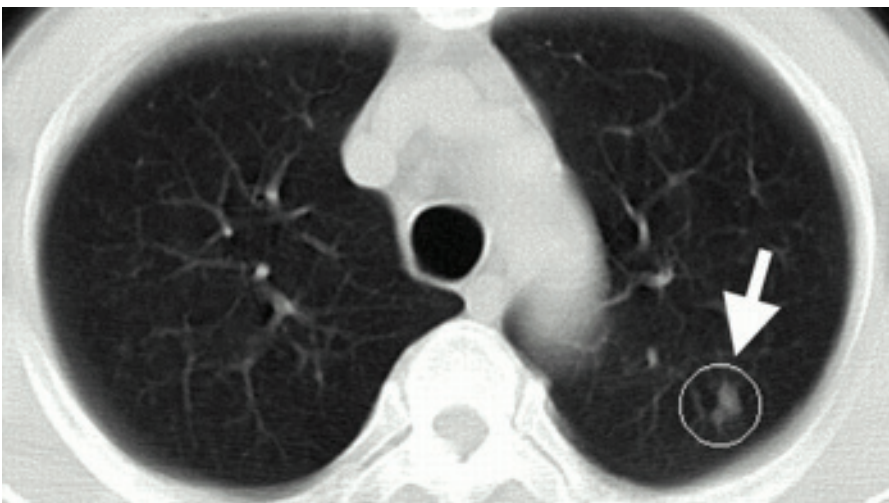


Figure 2. Image of lungs, with potential lung nodules detected (from: Armato S.G. et al., *Radiology* 225: 685-692, 2002).

Stack annotation can affect detection accuracy (Alberdi et al., 2004; Doi, 2005). Of concern is context bias (radiologists' diagnostic sensitivity depends on expected prevalence of a given anomaly (Eggin, 1996)); and automation bias (CAD misses particular cancer types), and learned dependency could lead the user to miss anomalies too.

How might alternative annotation presentation affect bias? CAD data is now presented as visual highlights, which may be more likely than another modality to influence what the radiologist sees at perceptual and attentional levels. If integrated with care haptic highlights might also avoid an identified

risk of degrading the decision process through simple sensory overload (Manning et al., 2005): highly tasked visual systems and the noisy hospital environment.

Radiologists, like most people, are creatures of habit and therefore adding a specialized device or compromising familiar mouse functions will likely not be accepted. They heavily use other manual tools (such as the keyboard & dictaphone), and transition swiftly between GUI pointing and stack strolling. The x-y mouse is best for pointing (Goyal et al., 2009), and its ease of use and familiarity make it favored relative to alternative input devices in this setting (e.g. (Sherbondy et al., 2005)).

In this work, we aimed to streamline the radiology image-scrolling task, investigating whether alternatives in user's input mobility (finger/hand movements used for scrolling control) can improve stack navigation; and how modality of annotation display and scrolling mechanism impacts signal detection patterns.

After analyzing 19 radiologists' work via observation and/or interviews, we prototyped augmentations to the standard mouse (Figure 3) which we hypothesized could support (a) more efficient image scrolling (with more fluid interaction) and (b) attentionally improved annotation display (in the haptic modality). We obtained qualitative feedback from our radiologists on these prototypes and the interactive techniques they support; and examined impact of interaction and display on detection rates in a controlled, abstracted study with non-radiologists (Oram et al., 2014).

### **The Radiologist' Work Environment and Constraints**

To view images, radiologists use two or three high-resolution LCD monitors, a mouse for stack navigation and GUI navigation, and keyboard and dictaphone to transcribe diagnoses. Data is provided via a Picture Archiving and Communication System (PACS): workstation, software, and network for image storage and retrieval according to industry standards. PACS are sourced by health authorities as major capital investments from a small number of medical imaging vendors, and have proprietary elements.

### **Viewing Images by Scrolling**

Scrolling is integral to image review. Computerized Tomography (CT) image consumption is faster with a stack than viewing as tiles (multiple images visible at once), probably due to eased perception of 3D structures (Mathie and Strickland, 1997). Radiologists must scroll at different speeds, stop, and reverse to compare or examine locations. They are trained to review specific anatomical structures, and make successive passes focusing on each of these in turn.

PACS workstations typically support two scrolling techniques: scrollwheel or click-&-drag. Both employ position control (scrolling distance is proportional to the position of mouse or angle traversed by scrollwheel). Atkins et al.

(2009) compared scrollwheel and click-&-drag techniques to a jogwheel (a rate control device: scrolling rate is proportional to input position), and found that most radiologists preferred the more familiar position control even though some were faster with rate control. Relative movement rates were generally fastest for the wheel/click-&-drag combination, slowest with wheel alone, and in between for jogwheel (Atkins et al., 2009). Sherbondy et al. used a tablet and stylus for scrolling, and found that position was faster than rate control for finding a target in a CT stack (2005).

### **Beyond the Mouse, and Direct-Touch Sensing**

Multi-touch sensing has become a ubiquitous manual control. In an early mouse example, Hinkley et al. explored touch sensing near the scrollwheel, and found it a useful discrete scroll alternative to the wheel, e.g. tapping to page up/down (Hinkley et al., 1999). Villar et al. considered multi-touch in five desktop mouse form factors, finding it could extend control degrees of freedom and support different input modes, mitigating need to switch between devices (Villar et al., 2009). They advised locating touch-sensed areas in easy reach of one hand posture, and cuing their location.

A pen and tablet solution showed decreased times relative to a mouse for the radiology task of outlining a region of interest (Dix et al., 2010). However, switching between different devices may hinder radiologists' workflow. Direct-touch reduces the need for device switching, but creates occlusion (Vogel & Baudisch, 2007) and fatigue from unsupported hands (Wang and Ren, 2009).

Other desk-supported variants have diversified interaction. The "Rockin' Mouse" adds a degree of freedom; while faster than a normal mouse in 3D, scrolling was not studied (Balakrishnan et al., 1997). Many other control movements could be used with a mouse-like device, but have not been explored in the radiology setting.

### **Haptic Feedback in Support of Scrolling**

Akamatsu et al. found that for a pointing task with a mouse, tactile feedback (pin pushing into fingerpad when on target) was quickest, and no feedback slowest for final positioning times (Akamatsu et al., 1995). Levesque et al. saw variable friction feedback speed target selection on a touch screen (Levesque et al., 2011). Tilting a mobile device to scroll was augmented so the user felt a vibrotactile (VT) buzz when they transitioned to the next item on the list. VT feedback lowered task completion time, and position was faster than rate control (Oakley et al., 2004).

These results suggest that haptic feedback on possible targets will give modest performance gains, even if the system does not know where the user is heading. The prevalence of detents on a mouse in a radiology setting indicates radiologists may be receptive to this.

## Computer Aided Detection (CAD)

Most CAD research focuses on validating that CAD information, provided as visual image annotations, improves radiologist detection sensitivity and/or speed (Doi, 2005). However, annotations overlaid on the stack affect what radiologists see. Even when biased towards finding everything CAD misses 20% (Doi, 2005), and also suffers automation bias. Radiologists attending to annotated areas are more likely to miss artifacts not found by the CAD. Alberdi et al. found a lower detection rate for users given CAD information in comparison to those who were not; here, the largest difference was seen in cancers not found by CAD. They hypothesized a bias effect, where users calibrate to the expected prevalence of cancers and expected proportion of cancers missed by CAD in the current data set (Alberdi et al., 2004). Additionally, a criticism of many CAD studies is that they contain an unrealistic proportion of cancers in their data sets, and radiologists know this (Alberdi et al., 2004). We have not seen studies that modified how CAD annotations are displayed; yet this may help mitigate the detection bias that CAD produces.

Rubin et al. (2005) saw CAD had a significantly higher sensitivity to finding lesions missed by a first human reader, in comparison to a second human reader. However, this comparison posits unrealistically that the user of the CAD annotations would accept all true positives and reject all false positive CAD detections.

In low-dose CT images, a CAD scheme detected 83 percent of lung nodule cancers (on images with on average 1-2 nodules), with 5.8 false positives per scan (Doi, 2005). Another scheme (run on different scans, containing some potentially more subtle cancers) detected 80 percent, with 2.7 false positives per scan. In our experiment we therefore manipulate annotation display assuming a detection ratio of 80% to align with current CAD performance.

### Table 1: Task Examples

1. *Identifying or finding a specific piece of anatomy:* The radiologist looks for an object or area of interest in one anatomical plane, looking through several slices to find and properly identify it. If unsure, or things are unusual, then s/he may look at the area in another plane (or several other planes if they are available). Can cross-reference a point between different planes, to see the location in other planes. Additionally, they may adjust the window/level to get better contrast between the object and its surrounds.
2. *Defining the edge / size of something:* The radiologist may want to know the size of an object, or if it is encroaching on the area of other anatomy. Window/level may be used to get better contrast of the object to its surrounds. After looking at the object in several planes, they choose a specific image, or multiple images, to outline, circle, or measure the diameter of the object.
3. *Tracking / connecting objects:* The radiologist follows a part of the

anatomy through several slices to check for abnormalities. The radiologist moves back and forth through the image slices while watching the area of interest. If they feel they have missed something, or loose track of the object they may slow down and watch more carefully for a subset of the image slices. This is repeated as many times as needed for different anatomical parts, usually by organ system but sometimes by area (such as in the brain).

4. *Comparing two images (old and new)*: The goal is to look for interval change: differences between the sets of image. Do new objects appear, have old objects enlarged? The radiologist brings up both sets of diagnostic images and looks at the same plane and area in each image side by side. They scroll back and forth in each set of images, comparing the areas of interest (can link the two images so they scroll together, but the slices may not land at exactly the same spots). They may re-measure objects that were found in the first diagnostic to see if they have changed in size.

5. *Identifying the makeup of something*: The radiologist may want to know what something abnormal is composed of. They look at the item in several planes, and see the attenuation of the item. They may adjust the window/level to get the best contrast with the surrounds, or to see colour differences within the object. To know the density of the item from the imaging they can select part or all of it and see the density number.

6. *Getting a second opinion*: If the radiologist is unsure of something, less familiar with it, or finds something unusual, they may ask the opinion of another radiologist. Another option is to look up papers on the topic to help confirm the diagnosis or learn about more nuanced aspects they cannot remember off the top of their head.

## **Approach**

Motivated by a general awareness that radiologists were not benefiting from 30 years of interaction advances, and were subject to ergonomic stressors, we observed radiologists and encapsulated their work in task-examples (Table 1). The subsequent analysis of these tasks (through questionnaire and interview) revealed that they were important and frequently performed tasks (Oram et al., 2014). These tasks rely heavily on stack scrolling, and as such we moved forward with that as our design space.

We iteratively prototyped and reviewed with domain experts several concepts for improving interaction. The scroll wheel mouse was used as the baseline for both scrolling with the wheel and click-&-drag scrolling (both position control scrolling techniques). A touch-scrolling mouse (also position control) was used to investigate if this method of scrolling works well for navigating stacks. Lastly, a mouse that could tilt back and forth was created and used to enable rate control scrolling as well as diversify the hand movements that can currently be performed on a mouse.



Figure 3. Prototypes. From left: Touch, Tilt, Wheel / Click+Drag.

Because data annotation is crucial to workflow scalability, yet there are many concerns about resulting bias, we wished to see if haptic and visual annotations differed in bias causation. Therefore all of the prototype devices had a pager motor under the top surface, so a haptic annotation could be given to the user through their hand.

We conducted an experiment with an abstracted detection task that utilized lay users in lieu of hard-to-access and time restricted radiologists. The abstracted task was created to mimic a simple search task in a radiology stack. More specifically, we tried to emulate the task of a trained radiologist scrolling through a lung CT image stack while looking for and marking potentially cancerous nodules (a case of Task Example 1). In real stacks, lung images exhibit bronchi (tubes feeding into small sacks called bronchioles). The bronchial tree can look similar to, but have slightly different characteristics, than cancerous nodules. We created a stack with randomly placed rectangles of varying grey colours, and the subject needed to find the perfect square (which was a grey in the middle of the colour range). This can be seen in figure 3, where the perfect square is outlined on the right image. To mimic CAD we correctly annotated 80% of the squares while the other 20% annotated an incorrect rectangle.



Figure 4. Images from abstracted task. Right image shows visual target annotation.

## Results & Discussion

The study was run on 12 lay subjects, and the ordering of the devices and annotation modality was counterbalanced. They scrolled through the image stack looking for the perfect square and hit a button when they found it. The subjects were instructed to complete the task quickly and accurately.

Completion time exhibited a broad and heavily skewed distribution: participants varied in the care they took (often trading accuracy for speed), with trials tending to go long if they did not find the square in the first pass. The targets were placed at different distances from the start point to mimic reality, and although each subject had the same set of distances this increases the variability in our data. Conventional models like ANOVA and GLM (general linear modeling) require normality. ANOVA can also only treat whether or not they got the trial correct as a variable, whereas a Cox model can use this factor to censor the data. Further, completion time and accuracy were not fully independent since with enough time a correct target could always be found in our abstracted task.

We therefore used a proportional hazards model (Cox regression (Andersen & Gill, 1982; Cox, 1972)) for completion time, which assumes that if given more time users could answer correctly. Non-error trials have all the information needed; error trials have partial information (we only know they did not find it up to a certain time).

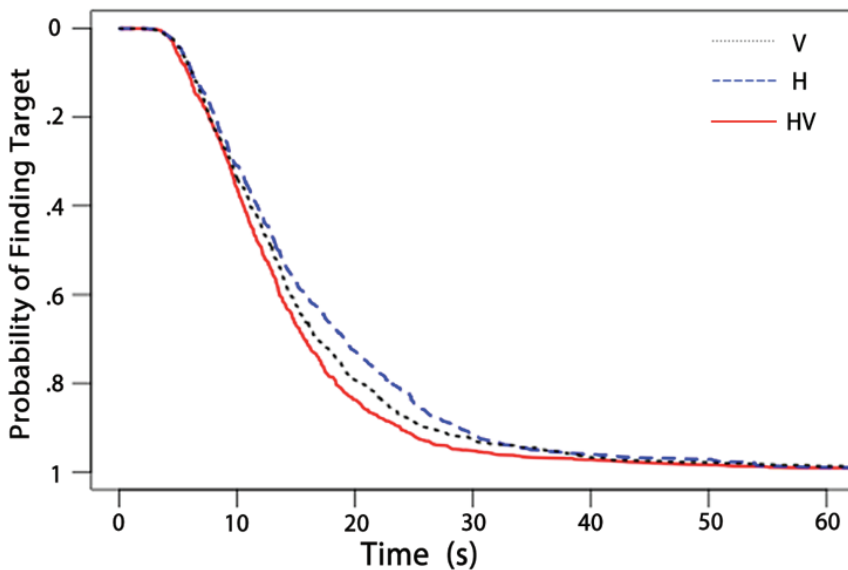


Figure 5. Survival likelihood (Cox regression) vs. projected completion time.

As you can see from figure 5 the combined haptic and visual annotation afforded faster detection, while haptic alone was slowest and visual alone was somewhere between the two. The traditional and most familiar scroll wheel supported the fastest task completion times by lay users, and was preferred. In most metrics, touch scrolling was ranked second. However, click-&-drag supported faster initial approach, even if it was to the wrong area. Further explanation of the results can be found in (Oram et al., 2014).

Speed of initial approach along with familiarity, is likely why the scroll wheel and click-&-drag work well together in the radiology environment. However,



novel input methods (e.g. a tilt or rocking motion associated with rate control scrolling) were disadvantaged by their newness and less optimized implementation. Because the scrollwheel has known ergonomic issues from excessive repetitive movement, alternate methods still need to be explored.

In the emerging practice of incorporating annotations (from CAD or other radiologists) into radiologists' workflow, we showed that multimodal cues are a promising approach, showing task speedup without error degradation, for a task abstracted to non-experts. Radiologists are heavily visually loaded, and may benefit from information provided through a less loaded modality, even when redundant.

Radiologists were interested in reducing the repetitive movements associated with the mouse that occur often with scrolling (e.g. clutching with the mouse wheel). This encourages us to continue to refine our Tilt implementation and test it following longer learning, as its rate control approach while continuing to support other functionality. Multi-touch would also allow many more potential improvements in radiology image interaction, via the mapping of gestures to different tools that could reduce the need for modal interaction with PACS workstations.

### **Final Prototype**

We improved and combined the best performing features found in the scrolling input and annotation types evaluated above, to create a prototype that worked as a conventional mouse with the added abilities to (a) touch-scroll, and (b) tilt backwards to access rate control scrolling. We began with a Microsoft Wedge mouse, added a rocking base (Polymorph™), and sensed tilt with a potentiometer (an accelerometer would confound translation with tilt). An Arduino relayed mouse signals, and a tacter was installed underneath the touch surface (Figure 6).



Figure 6. Modified prototype.

We then took the modified prototype to the workplaces of 3 radiologists (2 previously interviewed, 1 new), demonstrated its movement and haptic feedback (in context of our abstracted test task populated with radiology images) and informally discussed its potential usefulness with them.

Given existing customizability of PACS setups, radiologists reiterated their receptivity to the idea of a personalizable mouse. Their preferred speed of scrolling is highly personal and varies depending on the type of stack, so the rate control could have several preset speeds (potentially controlled via a slider on the side of the mouse). *"The goal should be to customize the mouse... in a perfect world once, and then to not have to fool with it after that"* [P1].

P2, an emergency radiologist, stated *"The way that I look at a large data set study is I fly through it once and get a birds eye view... I want to exclude any immediately life-threatening conditions"*. Further, in a diagnosis he needed to access multiple stacks, and felt the haptic feedback would help re-orient him when switching between them. He also indicated aesthetic appreciation: *"Ooh the haptic feedback I love"*.

Sometimes radiologists need to re-read other radiologist's image sets, e.g. with trainees, to ensure quality of care. The haptic annotations could help speed this review: *"You mark up the image in a peer review, and then I go through it to check whoevers work, and I can find immediately what they were looking at – that is valuable"* [P1].

P3 noted there might be *"a temptation to go really fast"*, and worried that the haptic cues would encourage this, resulting in missing anomalies. However, he further mused that it would be useful for very large data sets, such as the lungs. He generally felt that *"You have a problem and you are trying to find a solution to the problem, and here we have a potential solution to many problems"*.

Unsurprising was some mention of potential integration issues: *"Many of our workflows are so refined over the years... because we are just used to going through data sets in a certain way"* [P2].



## **Towards At-Home Physiotherapy: Next Generation Teleconferencing and Surface Based Interventions**

*Kody Dillman, Richard Tang, and Anthony Tang*

### **Introduction**

Hundreds of thousands of Canadians regularly sustain soft tissue injuries best suited for physiotherapy intervention, but many of these Canadians live in rural areas—away from the urban centres where most physiotherapists practice. This chapter describes two threads of work to address this problem: first, explorations of teleconferencing technologies to enable physiotherapy “visits” with remote practitioners, and second, explorations of at-home technologies that can support daily physiotherapy exercise. We discuss promising avenues of inquiry, and outline paths for ongoing future work.

For many injuries and movement disorders, physical therapy (physiotherapy), can increase mobility and decrease disability for patients receiving treatment (Tousignant et al., 2011). In the case of an injury like rotator cuff tendinitis, a physiotherapist guides patients through (and assigns as homework) exercises such as in Figure 1 in order to rehabilitate the patient. Those living in cities, where most physiotherapists operate (Canadian Institute for Health Information, 2011), tend to be served well by physiotherapy services. Yet, those who live in rural areas, where manual labour is an occupational norm (in Canada, over 18% of the population live in rural areas (Statistics Canada, 2012)), not only suffer a disproportionately large number of such injuries (Peek-Asa et al., 2004), but do not have easy access to physiotherapy professionals. As we learned from our design sessions with practicing physiotherapists, asking rurally based patients to travel into the city to access services can exacerbate many such injuries (e.g. sitting for hours during travel can worsen a back injury).

Our goal is to design technologies to allow patients to perform physiotherapy exercises from their homes. In particular, we envision near-future possibilities through commodity hardware already in people’s homes, for example with laptops equipped with web cameras, or in living rooms equipped with commodity depth cameras attached to gaming systems (e.g. the Xbox Kinect camera can model basic biomechanics of bodily movement). Using

these technologies, we envision patients speaking directly to professional physiotherapists to receive movement guidance, or smart video-based systems that can train, instruct, and correct patients when performing exercises.

We are guided by three central questions in this work: first, what are the communication practices in traditional face-to-face physiotherapy that must be preserved; second, what challenges does video media space present to these practices, and third, how can technologies be designed to overcome these challenges?

We explore these questions in this chapter through two explorations. In the first, we worked with physiotherapists to understand how to design tools to enable patients to work with physiotherapists live—for diagnosis and exercise training. In the second, we explored the ‘at-home’ case of doing exercises between physiotherapist visits.

Our explorations in this space have resulted in four sketch/prototype systems that point to useful directions for designers looking to support physiotherapy in future systems. As a group, the sketches reflect our understanding about how physiotherapists use the patient’s body and surrounding environment to communicate with patients, the role of mirrors, and home exercise.

We make two contributions in this work. First, we provide insights into a specific domain (physiotherapy) that can be used to guide design of video media spaces for remote work in this area. Second, from this work, we explore the concept of the body as a workspace, developing this idea through both sketches and critical reflection of our experiences. Our ongoing work involves designing tools for effective remote physiotherapy, though the findings should also support other domains where it is important to remotely teach activities that require specific movements (e.g. dance, personal training, martial arts, etc.).

### **Physiotherapy Process**

Physiotherapists work with patients through three phases of treatment: assessment, at-home exercise, and follow-up. Activities in these phases include teaching the patient exercises and correcting improper motions through movement guidance, as well as constantly performing assessments, since the physiotherapist must take measurements related to disability and function to create an effective treatment plan. The patient also performs exercises between sessions to build strength and/or flexibility. Assessment and movement guidance may require hands-on interaction, which requires collocation of the physiotherapist and patient. Follow-up sessions comprise exercise, manual therapy (e.g. the physiotherapist physically massages the shoulder), and discussions about home-treatment.

As a running example, we refer to a common exercise: external rotation (Figure 1). This exercise is commonly prescribed for patients with rotator

cuff tendinitis, a condition that commonly results from overhead reaching such as painting or window washing. In this exercise, the patient holds a resistance band, keeps the elbows tight against their sides, and pulls the band outward, their forearms pivoting around the elbows. While performing such an exercise, there are a number of pieces to consider: keep the elbows in tight, keep the forearms parallel to the ground, pinch shoulder blades together, stand upright and do not slouch, do not rush, only go to a certain extent, etc. This is a complex movement where performing any one of these parts incorrectly renders it far less effective.

**SHOULDER - 112 Resisted External Rotation: in Neutral  
- Bilateral**

Sit or stand, tubing in both hands, elbows at sides, bent to 90°, forearms forward. Pinch shoulder blades together and rotate forearms out. Keep elbows at sides.

Repeat 15 times per set.  
Do 2 sets per session.  
Do 1 sessions per day.

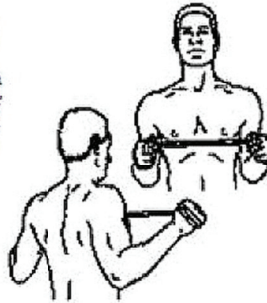


Figure 1. An example of a handout with an exercise that the physiotherapist might prescribe to the patient. This illustrates the external rotation exercise.

### Related Work

To set the stage, we discuss prior work that has demonstrated that telerehabilitation can be a viable and effective means of restoring bodily function. We then describe recent work that explored movement guidance through visual feedback, and finish by discussing the various roles bodies play in video media spaces.

### Efficacy of Telerehabilitation

Early pilot studies of telerehabilitation show promising objective and subjective results (Lai et al., 2004; Russell et al., 2011; Tousignant et al., 2011) with joint replacement and stroke therapy being common conditions for study (Rogante et al., 2010). Much of this pilot work employs considerable technology (e.g. sensors, haptics, and even virtual reality technologies) that is readily available in research labs, but far less likely to appear in patients' homes. Nevertheless, studies exploring the use of videoconferencing-based telerehabilitation following total knee replacement report positive results (Russell et al., 2011; Tousignant et al., 2011). For stroke rehabilitation, a community-based approach using videoconferencing tools demonstrated that patients showed significant improvement in all treatment measures, with additional mental and social benefits of group physical therapy (Lai et al., 2004). Furthermore, there seem to be high satisfaction levels for both patients and physiotherapists in spite of the lack of face-to-face time

(Tousignant et al., 2011). The literature suggests that assessments involving coarse-grained detail, such as gross movement or patient environment, are well suited for remote assessment (Cabana et al., 2010; Sanford et al., 2013). However, in cases where physiotherapists must use touch (e.g. feeling to check whether a joint is moving properly) remote assessment is not possible.

### **Solo Physiotherapy at Home**

*Home-based Physiotherapy.* Related to telerehabilitation works are home-based physiotherapy systems for self use. These allow the patient to exercise and receive feedback whenever they exercise, regardless of whether their physiotherapist is available. Some prior systems used wearable sensors to track patient limbs (Ananthanarayan et al., 2013; Ayoade & Baillie, 2014), but commodity depth sensors like the Microsoft Kinect are showing promise for at-home use (Doyle et al., 2010; Huang, 2011; Yeh et al., 2012). These systems use visuals on computer displays to provide feedback. The visuals range from pre-recorded video of a physiotherapist (Doyle, 2010; Huang, 2011) to stylized 3D representations of limbs (Yeh et al., 2012; Ayoade & Baillie, 2014). Work by Ananthanarayan et al. (2013) is unique in that the wearable sensor visually depicts the knee's bend angle.

Patients using these systems lack the immediate one-to-one communications of a physiotherapist either in-person or by telepresence. While this appears detrimental to the patient: early studies by Ayoade & Baillie (2014) on their prototype demonstrated that patients using such a system at home with basic 3D visuals to supplement routine physiotherapist visits improved more over patients using traditional methods.

*Movement Guidance.* Other recent research has explored teaching or guiding users through movements, and applications using ideas from such systems will likely prove useful for at-home exercise between sessions without the therapists. For example, LightGuide projects a movement guide onto the user's hand, and guides the user through specific, fine-grained gestures using feedback and feedforward cues (Sodhi et al., 2012). While this approach seems effective, it may be of limited use in a physiotherapy context, as many body parts are inappropriate for projection (and/or the projections may not even be visible). MotionMA provides visual feedback based on models of body and movement to guide a user in exercises (Velloso et al., 2013), though this specific approach provides very coarse-grained feedback, instructing the user to translate one or two bones of interest vertically or horizontally. While these tools focus on communicating through a visual channel, recent work has also made use of haptics to guide people through exercises (Alizadeh et al., 2014) by simulating the touch this person would receive from a collocated trainer or teacher.

### **Video Media Spaces for Physiotherapy**

In his conceptual reframing of video media space research, Buxton describes two fundamental conceptual "spaces" that bodies occupy in video media

spaces: people space, and reference space. People space is where one reads expression, trust, gaze, where the voice comes from, and where one looks when speaking to another—usually supported via an audio-video link that focuses on the participants’ faces. Reference space is where people use their bodies to reference the work, for instance by pointing and gesturing—usually supported via a video link that focuses on participants’ arms as they work over a flat, shared workspace (e.g. Tang et al, 1991). Thus in traditional video media spaces, the performs at least two functions: first, as a means through which people can communicate and express intention and ideas verbally (i.e. through spoken language), as well as non-verbally through facial expression; second, the body acts a means through which shared reference is established, by allowing people gesture using their hands—for example to point at things. Yet, in the case of physiotherapy application domain, a person’s body plays the role of a “workspace” in that conversation and communication occur about the body itself.

Thus, one of the principal challenges in designing video media spaces for physiotherapy is that the frame of reference is reflexive. That is, the workspace itself is one’s body, rather than an external entity. For instance, if one were speaking about movement pain in a joint, one would point to the joint, move to the angle where the pain begins, and point at the source of the pain. Yet, this kind of approach only works well for parts of one’s body that one can see; it does not work well for things that one cannot easily see (e.g. one’s back). These are new kinds of problems that we have not yet encountered in traditional video media space work.

## **Summary**

Prior literature has shown that telerehabilitation can help provide people with effective treatment for ailments, even when they are not co-present with a therapist (e.g. Tousignant et al., 2011; Russell et al., 2011). Yet, none of this work explores the specific communication challenges that arise as a consequence of physiotherapy.

Instead, considerable work has investigated how we can remove the therapist altogether, focusing primarily on the movements and training and teaching exercise (e.g. Anderson et al., 2013; Velloso et al., 2013). In our work, we address how the body needs to play a reflexive role in physiotherapy, because the discussion and communication in the media space is about one participant’s actual body.

## **Exploration 1: Design Sessions with Physiotherapists**

Physiotherapists teach patients strengthening and flexibility exercises, correcting improper motions through movement guidance, and providing hands-on manipulation for assessment and therapy. Yet, what kinds of support do patients and physiotherapists need if we are to design technology to enable this process remotely?

We recruited five actively practicing physiotherapists who participated

separately in design sessions that consisted of interviews about their practice, observation of their use of technology sketches (as we designed and implemented them) in mock physiotherapy sessions, and discussions about their experiences with the sketches to support further iteration. Our primary interest was in understanding and designing to support their communication practices when working with patients in a remote physiotherapy scenario.

The earliest meetings with physiotherapists were exploratory, and served to provide us with a basic understanding of how physiotherapists work in practice. This included: interviews about the types of treatment provided, what a typical session looks like, how health issues are assessed, and how treatment is delivered in person. After getting an understanding of the process, we engaged in collocated mock treatments with the therapists to experience physiotherapy from the patient's point of view. In these mock treatment sessions, one of the authors acted as the patient to experience the session first-hand.

### Technology Sketches for Live Physiotherapy

Sketching is an important part of the design process, and is a cheap and effective way to approach a new problem space (Buxton, 2010); where prototypes are meant to be didactic and refine an idea, sketches are evocative and allow for exploration. Rather than creating prototypes, we chose to create simple technology sketches through the course of our discussions with physiotherapists, which allowed us to explore the remote physiotherapy space without committing to any one solution.

We iteratively designed and built three different sketches: a mirror sketch, where the physiotherapist and patient are represented as if they were in a mirror together, an annotation sketch that allows physical therapists to draw on and around the body of the patient, and a targeting sketch that allows a physiotherapist to define a path of targets for the patient to move through. These sketches were built using C#/WPF, large projection screens, and the Microsoft Kinect camera. To mimic remote sessions with the physiotherapists, we created a dual setup to enable paired videoconferencing in our lab, and used these in our design sessions.

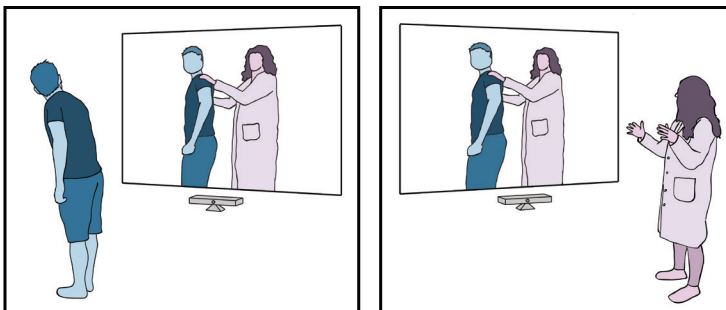


Figure 2. View of the physiotherapist's (right) and patient's separate physical workspaces, with shared workspace displayed on each participant's own display.



*Sketch 1: Mirror for Shared Discussion.* Figure 2 illustrates the first sketch, a videoconferencing environment where each participant is made to feel like they are sharing a mirror with remote participants (Morikawa & Maesako, 1998; Ledo, et al., 2013). The depth cameras respect the relative spatial relationships between participants as illustrated in Figures 2 and 3 (Ledo, et al., 2013). We based this first sketch on our own experiences in physiotherapy, where the physiotherapist stands with the patient in a mirror in order to show/teach exercises. Communication occurs through the mirror, where the physiotherapist can demonstrate an exercise alongside a patient's attempt. The physiotherapist can also gesture at parts of the patient's body if it is not moving or positioned correctly. Figure 3 illustrates client perspective.



Figure 3. Screen capture of mirror sketch. Inset image shows view of the patient's space (enhanced for clarity).

*Sketch 2: Annotation of the "Bodyspace".* Our second sketch focused on providing therapists with a means to annotate the patient's body and the area around it. A therapist can use this by freezing the video scene (with the patient's body in it), and the therapist can annotate the image using a variety of colours and brushes to illustrate different aspects of movement, or orderings (e.g. blue movement comes first, then red, etc.). As Figure 4 illustrates, the tablet provides the therapist (and/or patient) with a view of the video scene. The live video scene can also be annotated so that, for example, the patient can know the extents of a movement (i.e. the arm should not move further than point X, or lower than point Y).

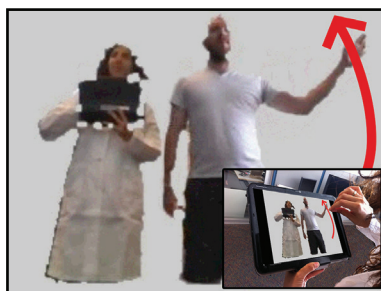


Figure 4. Illustration of the physiotherapist using annotations to guide the patient's hand. Inset image shows the physiotherapist's view of the tablet.

*Sketch 3: Target Paths for Movement at Home.* To support at-home exercises, we designed the third sketch to allow a physiotherapist to define a movement path through space (through a set of targets) that a patient could later “retrace” at home (Figure 5). Here, we drew on themes from prior work emphasizing notion of feed-forward and feedback in guiding movement through space (Bau & Mackay, 2008; Freeman et al., 2009; Sodhi et al., 2012). The 2D targets are displayed on-screen “in” the patient’s environment, with the size of the target representing its relative depth in the scene. The therapist places targets by physically moving her own limbs in space, and communicating with the system through voice commands. Once the therapist has placed the targets, the patient can then perform exercises by correctly moving through the targets, with visual feedback given if the target has been reached (Figure 5, middle and right).

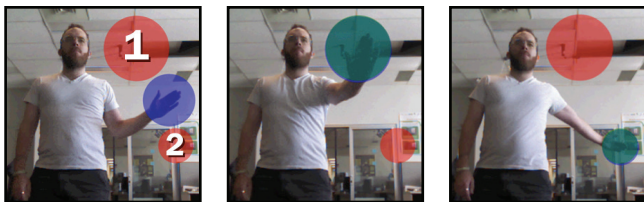


Figure 5. The patient interacts with targets that have been placed by the physiotherapist. Target 1 is closer to the screen/camera than target 2.

## Findings and Discussion

We summarise the findings from our design sessions with the physiotherapists in two categories here: communication and movement guidance, and assessment and progress tracking. In each, we discuss current practices and how the physiotherapists expected these practices to be augmented with the sketches. Finally, we provide our own thoughts about how to deal with these issues, while considering the body as a workspace.

*Communication and Movement Guidance.* Physiotherapists teach patients new exercises and movements first through demonstration, and second through gesture; if these fail, they fall back to physically guiding the patient through touch. The physiotherapist usually demonstrates the proper exercise to the patient so that he can see the entire form. Therapists will also use gesture, pointing at various body parts to indicate what should stay still, what should move, and how far. This often happens in front of a mirror, which makes it easier for a patient to see and understand how his body is positioned and how he moves. In collocated treatment, the physiotherapist can mark up the mirror to better train proprioceptive senses, or his awareness of his body’s position in space (Stillman, 2002).

Conventional videoconferencing technologies do not provide a patient with a view of himself, nor for the physiotherapist to meaningfully help guide motion. The physiotherapists encountered issues in conventional videoconferencing with the patient not understanding verbal instructions,

and the inability to point made clarification challenging. We also observed issues with the way the conventional videoconferencing setup presented different views for each person: the local view presented in the corner of the display sometimes occluded the image of the remote person, causing confusion. In contrast to the conventional setup, the mirror and annotation sketches worked extremely well for the therapists. Placing the patient next to the physical therapist in a mirror image (as in the mirror sketch), allows the therapist to easily model the ideal version of an exercise. The patient can then mimic the movement simultaneously, which is a way that people learn movements (Schmit, et al., 2005). Some of our physiotherapists instinctively stood beside the patient in the space. One thought it would be compelling to overlay the images, as the therapist's body could therefore act as an explicit visual guide so the patient could mimic the movement.

The mirror sketch also allowed the therapists to make and use the same gestures that they commonly use in collocated therapy to guide the patient (Figure 3). Interestingly, as much as exercises are about movement, they are also about keeping particular bodily parts still. To this end, the annotation sketch could be used to provide a reminder to keep a body part still. For instance, the external rotation involves proper positioning of the elbow, shoulder, and back, so being able to quickly reference and mark joints is necessary. For example:

*(Drawing a dot on the patient's shoulder.) So right there, I want you to try and keep that point still while you lift your arm up and come back down. (Patient's shoulder moves away from dot.) And you can see how it comes forward and comes up a little, so try and keep it more still in space as you lift. [P5]*

Similarly, the targeting sketch could be appropriated to help indicate to a patient that his arm has moved too far one way or another (since the target changes colour when the body part passes over the area).

Finally, there were multiple instances of the patient not being able to see certain parts of the body. For example, one therapist attempted to get her patient to perform a back exercise and asked him to turn his back to the camera. Upon learning that the patient could no longer see himself, she had him turn to the side as a next-best option. Incidents such as this prompted discussion about: pausing the video so the patient can see their back, being able to record and replay video, or having the patient hold a tablet to be able to turn their back to the camera and still see a view of the back.

*Discussion.* As illustrated in Figure 1, even physiotherapy exercises that seem simple are complex given the number of ways that they can go awry. While a basic audio-video link is clearly better than an audio-link alone, the mirror sketch added a new dimension to the interactions between therapist and patient as described above. Nevertheless, a major limitation of this communication is the inability of the physiotherapist to be able to guide the patient through touch. While there are some emerging solutions to this

problem that, for example, explore haptics (Alizadeh, et al., 2014), these typically require additional equipment and instrumentation. In the absence of touch, employing new configurations of the video space (i.e. as a mirror) may be the most straightforward way of addressing this communication gap.

Our design sessions revealed two additional challenges arising from the need to discuss parts of the patient's body, with the body acting as the workspace. First, the patient's body is frequently in motion. Annotations on the live video rapidly became out of sync with the patient's body and irrelevant. Second, the patient might not be able to see certain parts of his body that might need to be annotated (e.g. his back), or that might need to be discussed. We resolved this in our sketches through the addition of a "pause" feature, which addresses the latter problem, but less so the former (i.e. dealing with motion). Other possibilities could be to include a "playback the last 10 seconds" feature that could be annotated, multiple cameras, or bodily-tracked annotations (that follow the body even as it moves in the camera view).

*Assessment and Progress Tracking.* A therapist tracks a patient's progress through recovery using both experience (i.e. "reading" a patient through her hands), as well as with formal tools such as a goniometer (akin to a protractor). Common measures include strength, flexibility, as well as pain. Physiotherapists are trained to use touch to gain information and assess the patient, which presents a major issue when touch is not possible, as in remote physiotherapy. Visual inspection is also used by the physiotherapist for assessment: for example, the patient might demonstrate an exercise for the physiotherapist to assess visually, or she might also check for things like skin tone or hair growth. Patients will also communicate a lot of information through non-verbal cues, such as facial expressions and recoil: so-called "soft-signals", which might indicate pain or discomfort. The face, therefore, must be visible.

For precise range of motion assessment, our participants felt that being able to actively display joint angle information for patients would be valuable, particularly if it was an automatic feature (skeleton tracking can be used to approximate these values). When asked about the potential to do assessments, P3 agreed that she could use the mirror sketch to assess her back patients, though that she would "like to put sensors on them to have an objective measure" of range of motion automatically. For example, in the external rotation exercise, the physiotherapist may want to know how the patient is progressing by measuring the angle between the forearm and chest while pulling the resistance band.

*The numbers are really good for motivation, and they need that to stick with their therapy. They need to see that motivation. If they're thinking, "Oh my gosh, my numbers aren't getting any higher", they're going to be discouraged. [P2]*

*Discussion.* While assessment of certain variables traditionally assessed through hands-on interaction may never be practical or possible remotely, certain visual assessments may be possible remotely using the features afforded by the technology sketches. This should serve to decrease the number of face-to-face appointments necessary, in turn easing the burden on rural patients.

One of the major problems encountered with visual inspection and assessment in remote physiotherapy is the fact that the physiotherapist no longer has the space to work around the patient, and is limited to a single-angle view when using videoconferencing. In collocated therapy, the physiotherapist can get close to the patient for a “zoomed in” view, and can kind of walk and “pan” around the patient for different vantage points, and none of this is possible with a single-camera videoconferencing system. Multiple camera views can begin to address this issue, and allowing a therapist to remote control a video-capture drone in the patient’s space may be an interesting alternative.

To support some range of motion assessment, the annotation sketch could be used to mark the extents of a movement, and these annotations could be compared across time to show progress. As a visual charting tool, this would become immediately useful for the therapist and a useful motivational tool for the patient. Similarly, playback of past attempts over time (compared to one’s current progress) could be used.

*The Body as a Workspace.* Movement instruction is a complex and dynamic task even when co-located, with motions requiring proper placement of multiple joints and/or limbs at once. Current videoconferencing tools (e.g. Skype) allow for some demonstration, but the separation of space between the patient and physiotherapist makes discussion and movement guidance in the patient’s workspace difficult. This separation creates some added distance between patient and therapist, and cuts off their ability to gesture at or manipulate the patient’s body, which is relied on for communication in collocated therapy. Our exploration of physiotherapy shows us that when the body becomes the subject of conversation, Buxton’s three-space articulation of video media spaces (Buxton, 2009), is only useful conceptually, as all three spaces are all merged into one (i.e. the patient’s body is all of person-, task-, and reference- space). Retaining this unified presentation, as we saw in the mirror sketch, eases gestural interaction, as well as facilitating shared understanding of attention.

Yet, in general, having a body as a workspace in a video media space presents a number of challenges for both the “teacher” and “student” that need to be reconsidered due the fact that the subject of work and conversation is a participant in the media space rather than a separate, static entity that can be manipulated independently.

*Challenge: Visibility.* People cannot see certain parts of their bodies in

real life—we learn and receive feedback about muscles and movements on our back through tactile and kinesthetic feedback, or with mirrors. The traditional videoconferencing setup of one camera at one display is therefore not ideal in telerehabilitation and other configurations or hardware should be explored to allow areas of the body to be rendered visible. Patients straining and twisting to see the screen are usually not performing exercises correctly. Additionally, physiotherapists lose the ability to move freely around the space of the patient during remote therapy. Multiple camera and display configurations could address this issue (as in Physio@Home). Physiotherapists suggested also providing patients with a tablet so that the patient could always see the shared video feed regardless of the direction he is facing.

*Challenge: Annotations.* Annotations are semi-permanent mark-ups on the workspace that allow people to read/refer to ideas and information. Because the workspace here is a person's body and the space around it, these annotations need to be "connected" to those body parts and/or the space around it. For instance, our annotation sketch presented problems as soon as a person moved (even a limb) in the video scene—arrows would no longer point to the right body parts, or may even be pointing in the wrong direction. Furthermore, those annotations were in 2D space, many movements may be in the entire 3D space.

*Challenge: The "workspace" is non-static.* Particularly in relation to movement guidance, the "workspace" is a moving, living, and breathing entity. Because the patient can freely move about, and movements have a temporal element, gestures and annotations about these movements also need to have a temporal element. This is realized in YouMove (Anderson, et al., 2013) and ChoNo (Singh et al., 2011; Carroll, 2012), where annotations are layered as "tracks" that are only visible for specific durations. Yet, while this solution works for an asynchronous situation, how can we design these for real-time interactions when a remote physiotherapist is working with a patient?

*Challenge: Attention.* Specifically in the context of movement guidance, many body parts and joints may be in motion at the same time—how do we draw one's attention to the right point of interest? In mock sessions with the physiotherapists, we noticed sometimes that deictic references to body parts (i.e. "Move that upward"), if misinterpreted (e.g. moving the hand upward rather than the elbow), would lead to situations where the entire exercise would need to be reset. Thus, while annotation seems to be effective for supporting body movement discussion, and recording for playback (or slow-motion replay) and discussion should be explored further.

## **Exploration 2: Physio@Home for Exercising Between Sessions**

We previously described sketches to enhance video conferencing interaction between patient and physiotherapist, but these sketches still require the physiotherapist to be present and working with the patient,

albeit through video conferencing instead of in-person. The patient must still exercise at home on their own between routine physiotherapy sessions, but will no longer have the physiotherapist to guide and correct their exercise movements. The patient is now liable to forget their exercises, or to perform them incorrectly and risk slower recovery or re-injury.

To investigate this problem, we developed a prototype system called Physio@Home (Tang et al., 2015) to be used in patients' homes, where the patient will use the system while performing their exercises. The physiotherapist models exercises for the patient and gives them the recording files, and Physio@Home uses these files to guide and correct patient movements. The purpose of Physio@Home is not to replace the physiotherapist. Instead, it and similar systems supplement either regular in-person visits or telepresence sessions as in the previously described sketches to ensure patients are correctly performing their exercises while away from their physiotherapist. The physiotherapist is still required to diagnose their condition and provide exercises.

### Characteristics of Movement

To guide exercise movements without a physiotherapist, we needed to understand how physiotherapists describe movement and motion of the body and limbs. We analyzed commonly prescribed shoulder exercises and the ways physiotherapists taught them to develop a set of important characteristics that physiotherapists use to communicate. These characteristics are illustrated in Figure 6.

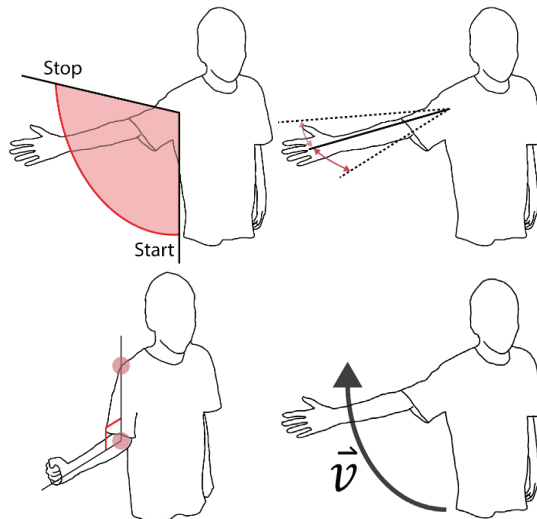


Figure 6. Characteristics of movement.  
(Top, Left to Right) Plane/range of movement, extent of movement,  
(Bottom, Left to Right) maintaining position/angle, rate of movement.

*Plane/range of movement.* This refers to the plane that the body part will move along during the exercise. The range refers to the “start point” and

“end point” of this movement. For instance, during non-angled shoulder abduction, the patient’s arm moves up along the frontal plane, starting from a resting position to where it is exactly aligned with the shoulder.

*Extent of movement.* This limits how a body part’s motion can and should deviate from the plane of movement. For example, during angled shoulder abduction, the arm must maintain its angle relative to the body’s sagittal plane.

*Maintaining position/angle.* For many exercises, certain joints need to be kept in a fixed position or at a fixed angle. In the case of abduction/adduction, the arm must be kept straightened, and the shoulder kept level with the ground. Other exercises are stricter—for example, with an external rotation exercise, the elbow needs to stay next to the body, and be bent at 90°.

*Rate of movement.* This refers to how fast a body part should move. For some exercises, performing them slowly ensures the right muscles are being used. This characteristic applies to a variation of the shoulder adduction where the arm must travel slower as it returns to the patient’s side. In many cases, an exercise does not have a set rate of movement and patients are free to proceed at their own pace.

### **Wedge Visualization**

We iteratively designed a visualization called the Wedge (Figure 7) using these characteristics of movement for use in Physio@Home. The Wedge consists of an arrow with a long stem to show movement path and an arc to show the plane of movement. The arc is divided into a completed section in green and the remainder of the movement in grey to show progress. This conveys both feedback and feedforward, and offers motivation for the user. When the patient is moving incorrectly from the recorded exercise, a red stick-figure arm appears to show the required position and posture of their arm.



Figure 7. Wedge visualization in Physio@Home.



## Multiple Cameras

In addition to the frontally facing camera view, Physio@Home also provides a secondary top-down view of the participant (Figure 8). We found during early pilots that the single frontal view was insufficient for showing movements in depth, often resulting in participants not knowing how far back to move or what angle to maintain.

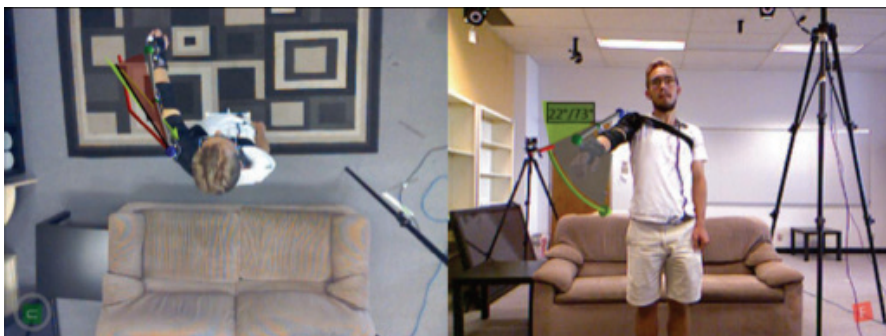


Figure 8. What the participant sees on-screen when using Physio@Home. (Left) View from ceiling-mounted camera. (Right) Mirror view from forward-facing camera.

We resolved this by mounting a camera in the ceiling. This allowed the participants to see themselves from above, and thereby see their depth alignment much easier. We can also draw the Wedge from this angle with an additional visualization to clearly denote their depth alignment. The rest of the Wedge's features are also visible from this perspective. We implemented the secondary view as just a top-down perspective for now. We imagine it also being used for details the frontal mirror view alone cannot show—such as close-ups of joints, exercises done behind the patient's back, etc.

## Findings and Discussion

We summarize our findings on Physio@Home and discuss the implications of the system's design features.

*Study.* To evaluate Physio@Home, we performed a laboratory study on 16 participants recruited from the local university. We evaluated how closely participants could follow pre-recorded exercises using the Wedge compared to simply watching and mimicking an exercise video, as is currently available for physiotherapy patients. We also evaluated the use of single and multiple camera views to see if they could benefit participant performance. Our early results showed participants being able to follow exercises the closest using the Wedge with multiple views. Overall, the Wedge outperformed the video conditions and allowed participants to follow the exercises closer.

*Discussion.* Physio@Home was designed to be used independently from a physiotherapist. We can also imagine it supplemented by live patient-therapist video conferencing in future work. Physio@Home's use of multiple

cameras may also benefit physiotherapists. One of the major problems encountered with visual inspection and assessment in remote physiotherapy is the fact that the physiotherapist no longer has the space to work around the patient, and is limited to a single-angle view when using videoconferencing. In collocated therapy, the physiotherapist can get close to the patient for a “zoomed in” view, and can kind of walk and “pan” around the patient for different vantage points, and none of this is possible with a single-camera videoconferencing system.

Because the exercises at home between sessions play such an important role in treatment outcomes, it is likely that supporting this activity well will prove most beneficial to patients in the end. Given that physiotherapy exercises are frequently dynamic (i.e. non-isometric), providing the patient with exercise recordings being properly performed is more effective than a static handout (Kingston, et al., 2013). These could be as simple as recordings made during meetings with the physiotherapists. Currently, Physio@Home only records the physiotherapist performing the exercises for the patient to follow, but we could use the patient as a model (e.g. by using a recording of the patient performing the motion correctly during a session with the physiotherapist). These recordings could double as a mechanism to track progress over time.

### **Conclusions and Future Work**

Physiotherapy is an effective treatment for common injuries, but remains difficult to access for many individuals. The work we present here represents a starting point for designing telerehabilitation tools for physiotherapy. Video conferencing tools need to be augmented to account for the fact the body is now a workspace, and that lessons from video media space work should be adapted here to support non-verbal communication (gesture, gaze), though the dynamic and complex nature of physical movement will need to be accounted for.

While the insight provided by physiotherapists regarding patient communication was incredibly valuable, the lack of actual patient participation is a limitation, and patients should be involved in future studies. Nevertheless, the findings have been helpful in informing our work moving forward, particularly as it relates to designing video media space systems where a participant’s body is the workspace, and we see this work as informing next steps for similar telerehabilitation tools.



## **Discouraging Sedentary Behaviors Using Interactive Play**

*Regan L. Mandryk and Kathrin M. Gerling*

(published in *Interactions*, vol. 22 no. 3, 2015)

### **Introduction**

Regular physical activity has many benefits, including to a person's physical, emotional, and cognitive well-being (Tremblay, 2010). Although adults should achieve 150 minutes of moderate- to vigorous- intensity physical activity per week, only 15 percent of adults meet these guidelines in at least 10-minute bouts, and only 5 percent of adults meet these guidelines in at least 30-minute bouts on five or more days per week (Colley, 2011a). For children, the statistics are even more discouraging. Although kids should get 60 minutes of activity per day, only 7 percent of Canadian youth accumulate 60 minutes per day six days a week (Colley, 2011b). The exercise habits adopted by children and pre-teens during this critical period can have lifelong consequences in physical health and self esteem. To encourage physical activity, researchers and developers in HCI have created a variety of "exergames," which encourage people to exercise by integrating exertion into the game mechanics (e.g., Mueller, 2010). Many exergames have focused on providing intense physical activity for players and have been shown to yield sufficient exertion to obtain the aforementioned benefits to a player's well-being.

However, recent work among health researchers has shown that there are also negative physiological consequences associated with sedentary behavior and that these consequences are distinct from those that result from a lack of physical activity (Tremblay, 2010). Although this may seem surprising, physical activity and sedentary behavior are not mutually exclusive. Even if a person is physically active (e.g., biking to work in the morning), she can also be sedentary (e.g., by primarily sitting for the remaining waking hours); the effects of too much sitting are physiologically distinct from too little exercise (Tremblay, 2010). The potential negative health outcomes are of particular relevance to populations who spend large parts of the day sitting, for example, schoolchildren who spend many hours a day sitting at their desks, and groups that struggle to gain access to opportunities for regular

physical activity, for example, people with mobility impairments and older adults in long-term care.

Because of the potential negative effects on health, researchers are now exploring the need for anti-sedentary guidelines to exist alongside guidelines for physical activity (see Mandryk, 2014). As researchers who design digital game-based interventions to promote health, we have been focused on designing games to promote physical activity; however, these exergames may or may not also work to combat sedentary behaviors. For example, a game designed to encourage a jogger to commit to and follow through with a daily jog will help a player meet the physical activity guidelines but will not help to combat sedentary behavior over the remaining waking hours. There has been little research into how the design of anti-sedentary exergames should differ from exergames that promote vigorous physical activity.

In a recent book chapter (Mandryk, 2014), we presented and contrasted the medical guidelines for physical activity and those for sedentary behaviors. We identified five design principles that need to be considered for anti-sedentary game design (see next section). We dub these anti-sedentary games *energames*—games that reduce sedentary time by requiring frequent bursts of light physical activity throughout the day. Here, we revisit the design principles for energames and show examples of how they have been used to design games that combat sedentary behavior in three at-risk populations: schoolchildren, people who use wheelchairs, and institutionalized older adults. Our work in this area is distinct in both intention and execution from much of the work on exergame design. Rather than designing for exertion experiences (e.g., Mueller, 2010), our goal is to use the motivational pull of games alongside interaction design to decrease sedentary time throughout the day.

### **Design Principles for Anti-Sedentary Energame Design**

Design principles for integrating physical activity into games while fostering player motivation include aspects such as the importance of providing feedback on activity levels, drawing awareness to past and current activity levels, providing feedback on goal achievement, leveraging social sharing, and integrating activity into a user's lifestyle. Based on these and other exergame design principles, we identified the following five design principles to foster energame design (Mandryk, 2014):

- Providing an easy entry into play. Lowering the barrier to foster physical activity can be accomplished by offering players an easy entry into play using accessible core game mechanics and controls.
- Implementing achievable short-term challenges to foster long-term motivation. To engage players over a longer period of time, achievable short-term goals can build self-efficacy and foster long-term player motivation.
- Providing users with appropriate feedback on their exercise effort. Providing players the opportunity to review their exercise efforts after

play or through in-game feedback can improve performance and foster motivation.

- Implementing individual skill-matching to keep players engaged. Adapting in-game challenges to match players' individual skill levels is one of the most important aspects of energame design, and is applicable both for player-versus-system and player-versus-player games.
- Supporting social play to foster interaction and increase exercise motivation. Supporting social play and fostering interaction between players is a core component when trying to increase long-term exercise motivation.

Here, we present some energame examples that follow these guidelines to help reduce sedentary lifestyles in three vulnerable populations.

### **Building Energames that Interrupt Sedentary Behaviors**

The goal of energames is to encourage people to break up sedentary time with movement. Three populations who are at risk of the negative consequences of sedentary lifestyles are schoolchildren who sit in desks for much of the day, people who use wheelchairs, and the elderly who reside in nursing homes. We have developed energames for each of these populations, and discuss their design and evaluation.

*GrabApple.* We initially developed GrabApple to explore the space of casual exergames—that is, computer games that players can learn easily and access quickly, using simple rules and special game mechanics, to motivate them to exercise at a moderate intensity for short periods of play (Colley, 2011a). Evaluated originally with young adults, we found that players were able to increase their heart rate during play, which helped them improve their performance on tests of attention and focus (Gao, 2012). This led us to consider the use of GrabApple for schoolchildren who could gain the acute cognitive benefits of breaking up sedentary time by playing a motion-based game (Gao, 2014).

*Gameplay.* The goal of GrabApple is to pick up falling apples and avoid touching the falling bombs (Figure 1). The avatar is controlled through the movement of the player's body, and the game uses the player's body weight as resistance to generate exercise through jumping, ducking, and movement. Score multipliers and game mechanics encouraged jumping, ducking, and periodically dashing to the keyboard.

*Game input.* The game used the Microsoft Kinect sensor to detect users' body movements. In the Kinect version, the player's position in space controlled the x and y location of the player's avatar. In a mouse-based version, avatar position was controlled using the mouse cursor.



Figure 1. Screenshot of GrabApple.

*User experience.* We compared the physical exertion, affective state, and player experience of children playing GrabApple with a sedentary version of the game and traditional physical exercise used for activity breaks to interrupt sedentary time at school (Gao, 2014). Our energame raised heart rates and perceived exertion levels significantly more than sedentary play, but not as much as traditional physical exercise. Players rated their arousal as higher after playing the energame (compared to sedentary play), and rated the game as more enjoyable than traditional exercise. Students also identified benefits to concentration from light exercise during a short break during the day and were interested in using a game to engage in movement-based activities during breaks.

Although GrabApple was successful as an energame, it is not accessible to players who use mobility aids such as wheelchairs. To address this design space, we implemented and evaluated Wheelchair Revolution, a game for people who use wheelchairs.

*Wheelchair Revolution.* We designed Wheelchair Revolution (Gerling, 2014b) with two goals in mind: First, we wanted to design a motion-based game accessible for people who use wheelchairs, and second, we wanted to support parallel competition between players who use wheelchairs and able-bodied players.

*Gameplay.* Wheelchair Revolution is a dancing game similar to Dance Dance Revolution, a popular motion-based game. The gameplay consists

of performing steps (indicated by falling arrows) synchronously to the beat of a song (Figure 2). The player aims to perform the move indicated by each arrow at the moment the arrow is in line with a target at the bottom of the screen and is awarded points based on how well each step is executed.



Figure 2. Wheelchair Revolution being played by a person in a wheelchair and an able-bodied player.

*Game input.* Players could use a dance mat, a game pad, or a wheelchair as input. The wheelchair mode emulates dancing by requiring players to move around with the wheelchair (forward, backward, and turning the wheelchair to the left and right). Wheelchair movements are captured by a Microsoft Kinect sensor. We implemented a variety of player-balancing mechanisms to ensure fair competition between various input types.

*User experience.* We had dyads of players (one able-bodied person, one person using a wheelchair) play the game in conjunction with the Canadian Paraplegic Association's wheelchair relay, an annual family sports event. Participants provided feedback on the game and their player experience. Our findings showed that players using wheelchair input showed heightened satisfaction of needs (e.g., competence, autonomy, and relatedness) compared with a neutral response; satisfaction of needs during play ultimately predicts a player's motivation and is indicative of a positive user experience. Players rated their enjoyment of our game significantly higher than a neutral response, and their comments demonstrated that they enjoyed how the game integrated the wheelchair (e.g., "It is nice to see my wheelchair in the game instead of being an object that stands between

me and the world”). Although our balancing mechanisms helped equalize the playing field between the different types of input, able-bodied players still outscored their opponents using wheelchairs, suggesting that better balancing approaches need to be investigated and implemented.

Our work on Wheelchair Revolution demonstrates how the wheelchair can be integrated into a game as an input device. This game was targeted at younger adults; however, we were curious to see whether motion-based play could also provide physical stimulation for older adults experiencing age-related changes. We conducted several studies, exploring the space of motion-based game design for the elderly.

*Hunting, Cooking, and Candy.* Our work on motion-based game design for the elderly has investigated various input controls (Gerling, 2013), including wheelchair-based control, and the use of motion-based games to foster relationships with caregivers (Gerling, 2014a). These research projects led to the design of a suite of motion-based games for use by the elderly, which we deployed in a long-term evaluation with seniors who lived in a care home (long-term care) and in a senior residence (assisted living) (Gerling, 2015).

*Gameplay.* In Candy Kids, candy moved across the screen and could be fed to a child by moving the player avatar (represented by a virtual hand) over the scrolling candy. Prairie Hunter invited players to hunt virtual animals by moving crosshairs over the animal using the motion of their hand. In Cooking Challenge, players prepared a salad by chopping, arranging, and mixing ingredients. Harvest Time invited players to cut down apples from a tree and hand the apple to a girl (Figure 3).

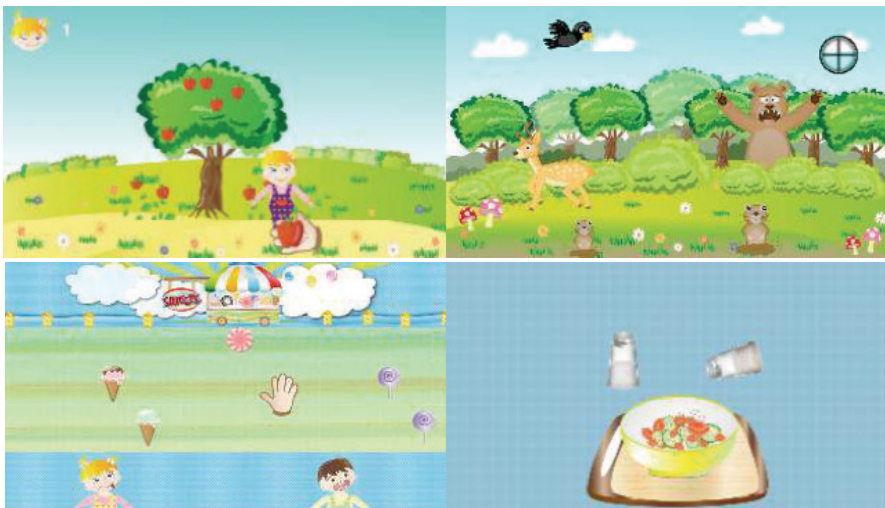


Figure 3. Screenshots of the four games designed for older adults (clockwise from top left): Harvest Time, Prairie Hunter, Cooking Challenge, and Candy Kids.



*Game input.* Both Candy Kids and Prairie Hunter used pointing input, where the player's hand was tracked using Microsoft Kinect to control an avatar within the game. Cooking Challenge and Harvest Time implemented gesture-based input that mimicked the real-world actions associated with the content of the games. Players used their strong hand to perform gestures and pointing actions. All games could be played in single-player or multiplayer mode.

*User experience.* A four-month deployment of the games in the two care facilities provided insights into the use of the games by the residents. Focusing on qualitative analysis of interview and observational data, we found that playing video games in the context of a weekly activity is enjoyable and empowering for independent older adults in a senior residence, but difficult when people experience complex age-related changes and impairments—as in the care home, for example—if these changes influence how older adults view the social context of play and how much assistance they require.

### **Reflections on Energame Design: Summary**

We have presented three examples of how energames designed according to a set of guidelines can motivate movement through playful interaction design. Our games were designed for three specific populations who are vulnerable to long periods of sedentary behavior. GrabApple was deployed in schools to break up long periods of sitting. In addition to raising heart rate and being an enjoyable experience, it also met the guidelines for energame design. The simple-to-learn game mechanics offered an easy entry into play, the in-game challenges were achievable in a short time, players received immediate feedback related to their exerted effort, the game difficulty adjusted to the player's skill through increasing challenge, and we provided a class-based aggregate leaderboard to provide motivation through social play without identifying individual players.

Wheelchair Revolution was designed to provide wheelchair-accessible motion-based play. By integrating the wheelchair as an input device, we gave players a way to break up sedentary periods of the day, and use the wheelchair as a tool to interact with a game while promoting movement. The guidelines for energame design guided development: The game provides easy entry into play by using accessible mechanics and controls; it provides short-term challenges to build self-efficacy; it provides users with feedback about how well they performed—which is directly tied to their physical effort; it balances play for players with different abilities and skills; and it allows players with different physical abilities to directly compete, offering a social play experience with other people who use wheelchairs or able-bodied players.

Finally, our suite of games for institutionalized older adults was created using the guidelines for energame design in combination with design recommendations for games for older adults. As such, we focused on energame design within the context of accessibility of games for older

adults experiencing age-related changes and impairments. Our results show that the nature of energames (easy entry into play, combination of short- and long-term challenges, playability in a social setting) makes them particularly suited for deployment in care-home settings, where sessions of play often need to fit in with other scheduled activities, but that their successful integration and older adults' engagement with them ultimately depends on their individual abilities and interests. However, if older adults do take ownership of energame play, our findings demonstrate that such games can be a valuable opportunity to provide mental and physical stimulation to combat sedentary behavior in late life, encouraging older adults to reintroduce challenge and competition into their leisure activities.

Our results suggest that energames can promote movement among very different populations— from schoolchildren to older adults living in care homes. Motivating physical activity in short bursts throughout the day can help to break up long periods of sedentary behavior; interactive play is a fun way of achieving this goal.



## **OrMiS: Use of a Digital Surface for Simulation-Based Training**

*Christophe Bortolaso, T.C. Nicholas Graham, Stacey D. Scott, Matthew Oskamp, Doug Brown, and Liam Porter*

(More details on OrMiS and its application can be found in our published papers. These include an overview of OrMiS and its design goals (Bortolaso et al., 2013). We have studied in detail the tradeoffs between lenses, radar views and tabletop-level zoom (Bortolaso et al., 2014). Finally, we have shown how a multi-surface map table can support a variety of terrain visualization techniques (Oskamp et al., 2015).)

### **Introduction**

The Canadian Army uses simulations to train officers in executing effective Command and Control (C2) at the formation headquarters and unit command post levels. In these exercises, the primary training audience (PTA) is composed of officers practicing tactical decision-making in a simulated command headquarters. Retired military officers (called interactors) act out the role of troops on the battlefield. Trainees operate in a mocked up command headquarters – a room with tables, maps, computers and communications equipment. Trainees use radio and chat programs to communicate with officers in the field, use battle management software to plan missions and operations and to maintain situation awareness, and use unmanned aerial vehicles (drones) to monitor the operation. In simulations there of course are no troops and vehicles in the field. Instead, the interactors use simulation software to carry out the orders they receive, for example using point and click mouse-based computer interactions to specify the routes that vehicles take as part of a convoy.

Simulation-supported exercises provide numerous advantages over exercises carried out in the field. Simulations are much cheaper than field deployments, enabling large-scale exercises at low cost. They enable actions, which would be cost prohibitive or dangerous in real-world training, such as blowing-up buildings. Simulation-based training therefore allows officers to be trained more frequently, at a lower cost, and in some ways more realistically. However, the quality of the training experience depends

on the ability of interactors to enact a realistic and educationally beneficial scenario. Modern simulation tools provide deep and rich functionality, but at the cost of complex user interfaces that interactors often find difficult to learn and to use.

As an alternative to current simulation tools, we developed OrMiS, a system for Orchestrating Military Simulation (figure 1). OrMiS provides users with a multi-display and multi-touch simulation interface based on a digital tabletop. OrMiS follows the conventions of traditional map tables where a small group of people can work together to observe the state of a battlefield. Unlike traditional map tables, OrMiS can also be used to control a simulation, allowing users to plot routes and positions for vehicles. OrMiS provides a touch interface, where dragging out a route with a finger moves units, and where the map can be panned and zoomed with pinch gestures. Lenses can be used for focused work; separate tablets can be used for private work, and radar views provide group context.



Figure 1. OrMiS supports military simulation by allowing small groups of people to collaborate around a shared touch surface. OrMiS is based on a large multi-touch table, handheld tablets, and a radar view display.

In this chapter, we report our experience analyzing interactors' practices and show how this informed the design of OrMiS. Through field observations and interviews with staff from the Command and Staff Training and Capability Development Center (CSTCDC), we identified that the quality of the exercises is constrained by a mismatch between existing simulation interfaces and interactors' expertise, collaborative practices, and workflow. Existing simulation tools are complex and difficult to learn. Days of training are required prior to each exercise to make interactors productive. Currently, interactors sit in front of a PC, making it difficult for them to coordinate their actions. In order to collaborate, interactors are forced to switch between their screen and a physical map when impromptu events occur during an exercise.

In this chapter, we present the design of OrMiS and show how its large table-based form factor and touch interface address these problems of ease of learning, coordination and support for planning. We first provide background in tabletop interaction in general and survey earlier efforts to use digital tabletop interfaces for planning and command and control. We then show how OrMiS was designed to be easy to learn, while helping with coordination and planning tasks. Finally, we report on enthusiastic feedback from the use of OrMiS by officer candidates.

## **Background**

Large tabletops naturally support collaborative work by enabling face-to-face communication, pointing and gestures, and seamless awareness of others' activities (Gutwin & Greenberg, 2002). These properties have led researchers to explore the benefits of digital tabletops for computer supported collaborative work in collocated situations. Decisions around how to position and orient the content displayed on a tabletop (Kruger, Carpendale, Scott, & Greenberg, 2004) are key to achieving fluid interaction and smooth collaboration. For example, objects oriented toward and close to an individual are understood by others as belonging to that person, whereas objects located in the middle of the table are often shared by the group (Scott, Sheelagh, & Inkpen, 2004). Similarly, an object intentionally occluded at the bottom of a pile is typically considered no longer relevant for the ongoing task, or stored for later use. Techniques have been proposed to move and rotate objects with only one finger (Hancock, Carpendale, Vernier, & Wigdor, 2006) and to manage occlusion between physical items resting on tabletop displays and virtual objects (Javed, Kim, Ghani, & Elmquist, 2011; Khalilbeigi et al., 2013).

Co-located collaboration around a tabletop also introduces problems of physically reaching parts of the table, leading to physical interferences (one person's arm getting in the way of another's). Doucette et al. have shown that people working around a table try to avoid physical touching as much as possible. This can lead them to fall back to turn-taking (Doucette, Gutwin, Mandryk, Nacenta, & Sharma, 2013), losing a primary benefit of a shared surface that it allows people to work at the same time. Similarly, conflicts can occur when two people try to simultaneously access the same elements. For example, if two people try to pinch-to-zoom a map on a digital surface at the same time, the result can be unpredictable and confusing. Previous research shows that relying solely on social protocols to prevent or resolve such conflicts is frequently insufficient (Morris, Ryall, Shen, Forlines, & Vernier, 2004). Tabletop interfaces should therefore provide support to limit both physical and interaction conflicts.

Finally, when collaborating, people frequently switch between working together and working separately. For example, when planning routes in a C2 tool, planners may focus separately on the units for which they are responsible, then discuss global goals, then return to individual planning. This type of collaboration is called mixed-focus collaboration (Gutwin &

Greenberg, 1998), and applies to activities such as brainstorming (Geyer, Pfeil, Höchtl, Budzinski, & Reiterer, 2011), route-planning (Tang, Tory, Po, Neumann, & Carpendale, 2006) and information analysis (Isenberg, Tang, & Carpendale, 2008). The challenge in the design of a tabletop tool to support this kind of work is to support both styles of work, and to provide seamless transitions between them so that people do not lose context or have difficulty returning to their focused work after collaborative discussions. Many interaction techniques such as personal viewports (Ion et al., 2013; Scott et al., 2010), lenses (Forlines & Shen, 2005; Tang et al., 2006) or sharable containers (Hinrichs, Carpendale, & Scott, 2005) have been designed and tested to support different levels of collaboration.

### **Tabletop Interfaces for Geospatial Content**

For centuries, people have used tabletops to collaboratively work with maps. With the widespread availability of Geographical Information Systems (GIS), digital tabletops have become a compelling medium for collaboratively interacting with maps. Digital maps support zooming and panning and dynamic update of the map's contents.

The first map-based tabletop systems provided simple interfaces, relying on social protocols and on the intrinsic properties of tabletops to ease collaboration and workspace sharing. For example, LIFE-SAVER (Nóbrega, Sabino, Rodrigues, & Correia, 2008) was designed to support flood disaster response operations. This system first displayed a 3D rendered map on an interactive table to allow experts to analyze flooding simulations in a collocated manner. Similarly, MUTI (Nayak, Zlatanova, Hofstra, Scholten, & Scotta, 2008) supports decision-making in disaster management through a zoomable digital map and a set of oriented controls. In these pioneering systems, little attention was paid to how best to support collaborative work.

When several users have to interact on the same space, an obvious solution is to provide personal viewports on the map, windows that allow each person to have and manipulate their own view. This avoids the possibility of physical awkwardness as people try to touch the same part of the map or need to reach around each other, and allows all users to zoom and pan their personal view as they choose. For example, uEmergency (Qin, Liu, Wu, & Shi, 2012) supports forest fire responders by providing real time geolocated information on a large tabletop. To support mixed focus collaborative tasks, uEmergency displays a shared interactive map as well as individual windows and widgets for each user. The same approach is also used in eGrid (Selim & Maurer, 2010), which provides multiple rotating views of the same map to support the analysis of a city's electrical grid. This approach of splitting the same map into multiple views on a tabletop display efficiently supports individual work while maintaining workspace awareness. However, much of the advantage of tabletops is lost, since people are no longer looking at the same shared display, and possibly lose awareness of what others are doing. This approach is therefore not suitable for tightly-coupled collaboration where users are attempting to discuss and manipulate a single part of the

map (Tang et al., 2006).

Finally, another emerging approach is to provide each user with a personal hand-held device (such as a tablet) showing a personal view of the map. This is another form of personal viewport, but where the private map appears on a separate physical device, not on the table. For example the Tangible Disaster Simulation System (Kobayashi et al., 2006) divides the output space by combining a tabletop display with two external screens showing a 3D first-person perspective of the map and charts describing the underlying disaster simulation. A more recent approach consists of physically splitting the input space by providing tablets to the users around a tabletop display. For example, the SkyHunter project (Seyed, Costa Sousa, Maurer, & Tang, 2013) enables geological exploration by providing a tabletop and multiple tablets to a group of users. Predefined gestures allow users to transfer part of the map from the table to a tablet and back, thus enabling individual and group work and transitions between them. Recent controlled studies showed that this combination of table and tablets is beneficial for teamwork (Wallace, Scott, & MacGregor, 2013) which makes this approach very promising.

### **Tabletop Interfaces in Military Training and Operations**

Despite the fact that the military has a rich history working with tables, few research projects have focused on using digital tabletops to support command and control activities. The Digital Sand Table that face-to-face work around a digital command and control application could strongly support collaboration. Similarly, the Comet project (Cerdec Comet Multitouch, <http://www.cerdec.army.mil/about/comet.asp>)—a collaborative project between the US Army's Communications-Electronics Research, Development and Engineering Center (CERDEC) and Microsoft showcased at the 2010 Army Science Conference—proposed a digital tabletop interface to enable collaborative access and manipulation of maps and videos to support command and control. Canadian naval simulation researchers at Defense Research and Development Canada (DRDC)-Atlantic in conjunction with SurfNet researchers proposed the ASPECTS system (Scott et al., 2010), which provided a digital tabletop system to support naval command and control by providing real-time monitoring of ships' locations. ASPECTS used personal viewports on the tabletop, and provided pie-menus and role-based interaction based on user identification with pens.

Companies specializing in defence and security have explored digital implementations of the traditional map table. In 2007, Northrop Grumman demonstrated TouchTable, an 84" digital tabletop supporting collaborative interaction with geospatial data. The FAA's Cyber Security Incident Response Center installed a TouchTable (Northrop Grumman's War table: [http://news.cnet.com/8301-17938\\_105-9773294-1.html](http://news.cnet.com/8301-17938_105-9773294-1.html)) to help cyber analysts identify and respond to cyber-attacks against the FAA's network ([http://www.irconnect.com/noc/press/pages/news\\_releases.html?d=125335](http://www.irconnect.com/noc/press/pages/news_releases.html?d=125335)). Around the same time, Northrop Grumman also demonstrated a 3-dimensional

digital map tabletop, called the TerrainTable (Northrop Grumman's TerrainTable: <http://blogs.walkerart.org/newmedia/2006/05/16/art-com-northrop-grumman-and-audiopad/>). Activating mechanical pins in the table to distort a silicone skin physically formed the shape of the terrain. As the terrain was formed, satellite pictures of the map were displayed through an overhead projector. This early work, along with recent advances in digital tabletop hardware platforms, however, paved the way for currently available product offerings, for example the iCommand (iCommand: <http://www.aaicorp.com/products/unmanned/icommand>) Table by AAI / Textron Systems, which provides a multi-touch based digital tabletop interface to a cloud services-based battlefield map data. The iCommand system offers a distributed interface across digital tabletop and other multi-touch devices, such as an interactive wall or smartphones, to visualize units' position in real time in the field or in command posts. Similarly, HDT Global (Command Table: <http://www.hdtglobal.com/products/command-control/audio-video-display/60-interactive-command-table/>) and Steatite Rugged Systems (Rugged Interactive Mapping Table: <http://www.rugged-systems.com/products/rugged-monitors/interactive-mapping-table.html>) currently offer portable (i.e. foldable) digital tabletop systems that can be deployed in the field to forward command posts. Both systems provide a multi-touch interfaces to existing C2 software systems.

Despite the above research and commercial products, there are still relatively few digital tabletop systems currently available in real-world military training or operational contexts. This chapter contributes to this domain by documenting the OrMiS interface, and providing lessons learned in designing a digital tabletop interface to support military simulation-based training exercises.

### **Designing for Simulation-Based Training**

When conducting simulations to help train staff officers in command and control techniques, the Canadian Command and Staff Training and Capability Development Center (CSTCDC) relies on retired military officers (called "interactors") to role-play officers in the field and to enact simulated troop actions (Roman & Brown, 2008). As shown in Figure 2, a standard approach locates trainees in a mocked-up command headquarters, communicating by radio or text chat with "officers in the field". The trainees use BattleView (BattleView: <https://www.thalesgroup.com/en/content/battleview-newly-integrated-canadian-armys-tactical-c2-system>) a command and control application on personal computers and paper maps to perform battle management and operational planning. The positions of units in the field are periodically updated on BattleView, whose main map view is displayed on a wall, making it visible to all the officers in the headquarters (see Figure 2A).

The officers in the field are role-played by interactors who relay observations to the trainees and carry out their orders. The interactors are, in fact, located at desks with personal computers located in a private room, and



use simulation tools that mimic battlefield troop movement and combat engagement (see Figure 2B). In the back of the interactors' room, a set of screens display a map showing the global state of the mission. In the middle of the room, a large paper map of the mission's area of interest is located on a table (called a "bird table", as it provides a bird's eye view), with small paper icons to represent the units' positions. The interactors primarily use this table to collaboratively plan the simulation before it begins. Because of the difficulty of keeping the table's paper markers updated, the table is rarely used after the exercise begins.

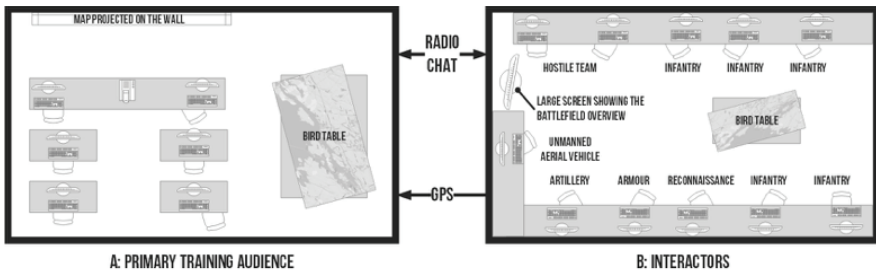


Figure 2. Physical layout of a typical simulation-based training session.

The simulation software allows interactors to mimic troop movement and combat engagement. Two popular simulation tools are ABACUS (Advanced Battlefield Computer Simulation - <http://www.raytheon.com/>) and JCATS (Joint Conflict and Tactical Simulation - <http://www.jtepforguard.com/jcats.html>). Simulation tool interfaces are composed of a full-screen map view with a large set of accompanying controls. The units are displayed directly on the map using standard military symbols. Interface controls allow operators to set the position, orientation, heading and rules of engagement of units, to organize units' hierarchy, to perform combat operations, and to create routes. Each interactor is in charge of a set of units, typically split according to the units' command hierarchy.

We visited the CSTCDC three times to observe live simulation exercises. These field observations, in conjunction with supplementary interviews with simulation experts, have revealed that the quality of the exercises is constrained by three main issues with the current infrastructure:

1. *Interface Complexity*: The interfaces of existing simulation tools are complex, requiring significant training and expertise to use. A lack of qualified personnel limits the number and size of simulated exercises that can be held.
2. *Weak Support for Coordinated Tasks*: Tightly coordinated actions between interactors are poorly supported by the existing tools. This is largely due to the physical setting, where interactors sitting at individual PCs have difficulty communicating with each other and maintaining a global awareness of other interactors' actions within the (digital) battlefield.

3. *Poor Flexibility When Plans Need to Change*: If the trainees perform unexpected actions, the interactors may need to adjust their training strategy. Re-planning requires intensive communication and requires reference to the state of the battlefield. The physical layout of the current PC-based infrastructure makes re-planning difficult, requiring interactors to leave their desks and move to the physical bird-table. But this is hindered by the fact that the physical markers on the bird table have become out of date with respect to the simulation. Once the re-planning is complete and the interactors return to their PCs, they no longer can see the new plan sketched out on the bird table, and must enact it from memory.

To solve these issues, we implemented the Orchestrating Military Simulation (OrMiS) system, which provides an interface for interactors based on a multi-touch tabletop surface and supplementary displays. OrMiS provides interactors with an efficient and easily learned way to perform simulations while supporting collaborative manipulation of units.

### **OrMiS: Bringing Multi-Touch to Simulation-Based Training**

OrMiS provides small groups of interactors an interface to move units and perform combat actions while sharing a common overview of the battlefield. OrMiS is a multi-display environment (MDE) composed of a multi-touch table, multiple tablets to provide personal views, and additional screens displaying an overview of the battlefield. Interactors can either work together on the table, or separately using the tablets. The devices are synchronized over the network, so actions performed on one device are immediately propagated to the others. This diversity of devices offers a range of possible configurations, detailed later in the chapter.

### **The Interactors' Interface**

As shown in Figure 3, OrMiS displays a topographic map from a top-down perspective. Operators can pan the map by dragging with two fingers and zoom the map with a pinch gesture. As with standard map applications, the resolution of the map display automatically increases with the zoom level, showing details that are not visible on the overview. The map can also be zoomed using bifocal lenses and personal viewports, as described in the following sections.

Units positioned on the map are depicted using standard military symbols. Interactors can tap on a unit to access specific controls such as to specify the unit's heading, rules of engagement or speed. Visibility and attack range are displayed by overlays on the map. A line of sight overlay is shown when an operator selects a unit or sketches a route. Operators can change a unit's heading by selecting and rotating it. Combat begins when opposing units move within range and visibility of each other, respecting the rules of engagement for each unit type.

Routes can be created, modified, or deleted using single finger gestures,

as detailed below. Two types of routes are supported: permanent routes, which are created from the map and can be used by multiple units at the same time and in any direction, and one-time routes, which are created from individual units and disappear when the associated unit reaches the route endpoint. A one-time route can be connected to a permanent route to drive units onto it.

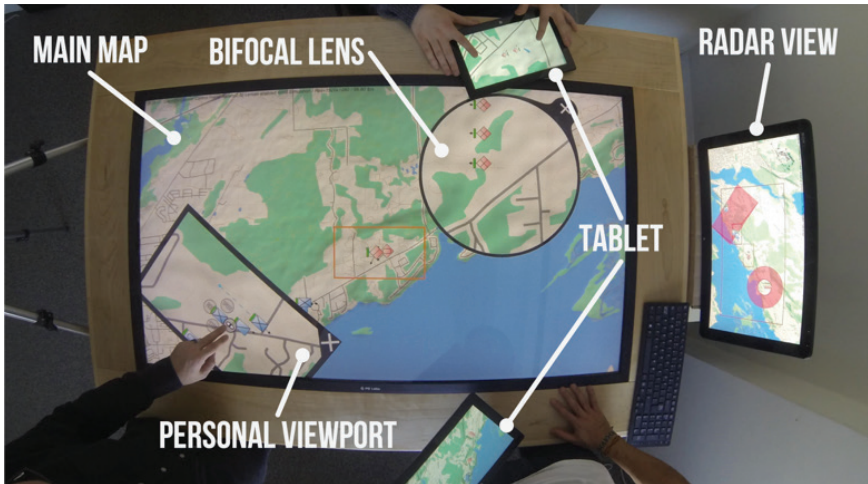


Figure 3. The OrMiS system combines a digital tabletop, a radar view display, and tablets for private work.

### **OrMiS Technical Setup**

OrMiS's interactive table is built from a PQ Labs G4S multi-touch frame and a 55" high-definition television housed in a custom-built wooden frame. The OrMiS software application was implemented in C# using the Unity game engine (<http://unity3d.com/>). This engine eases 3D programming and provides fast rendering and a very responsive interaction. OrMiS is compatible with Windows 8 and TUIO (Tangible User Interface Objects - <http://www.tuio.org/>) multi-touch inputs. The maps of OrMiS are generated using the InterMAPhics GIS (Kongsberg Gallium, 2013). Multiple surfaces are synchronized over a network using the Janus software toolkit (Savery & Graham, 2012).

Over all, OrMiS provides the features required to perform a simple but realistic exercise. With OrMiS, small groups of interactors can plan and then direct a scenario through a simple touch-based tabletop interface. OrMiS provides ways to work individually as well as in tight collaboration without having to switch between workstations.

### **Addressing Ease of Learning**

Our interviews with simulation center staff revealed a strong desire for simulation tools that were easy for interactors to learn and use. Most interactors are retired military officers who have high expertise in military command and control, but are not experts in simulation tools such as

ABACUS or JCATS. Interactors typically participate in simulation supported training exercises once or twice a year, and so need to be trained (or re-trained) prior to each exercise.

The interactor's interface in ABACUS or JCATS shows a map of the battlefield including the units under the interactor's control. A profusion of menus support actions such as plotting routes, operating vehicles, firing weapons, checking units' line of sight, and filtering which units and terrain features are displayed on the map. Interactors use two side-by-side computer screens, with one screen displaying the map and the other displaying the menus (Figure 4). Interactors need to become sufficiently proficient with all interface controls in order to work in the real-time of live simulated exercises.

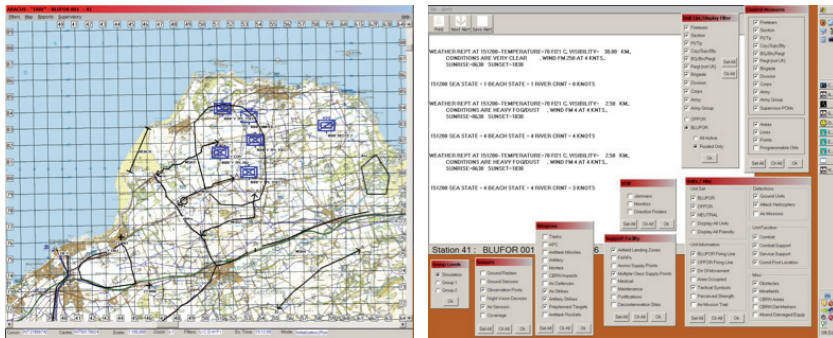


Figure 4. The interface of the ABACUS simulation tool displayed on two screens.

The interactor's interface in ABACUS or JCATS shows a map of the battlefield including the units under the interactor's control. A profusion of menus support actions such as plotting routes, operating vehicles, firing weapons, checking units' line of sight, and filtering which units and terrain features are displayed on the map. Interactors use two side-by-side computer screens, with one screen displaying the map and the other displaying the menus (Figure 4). Interactors need to become sufficiently proficient with all interface controls in order to work in the real-time of live simulated exercises.

### **Simple, Touch-Based Controls Improve Usability and Scalability**

We designed OrMiS to be easy to use. We applied traditional user-centered design methods, regularly evaluating the usability of our interface with military experts. We followed a parsimonious design process, adding features only when we could demonstrate that they were needed. This led our final design to be controllable with a small number of touch actions and controls.

Interactors can drag, tap, or long press (i.e.. touch and hold) elements to directly see the effects on the display. For example, a simple drag gesture originating from a unit icon automatically creates a route for the associated unit (see Figure 6A). Tapping on the first or last waypoints can extend a route. When a unit is driving along a route, the waypoints can still be modified. The unit will adapt in real time to new waypoint positions. This

allows interactors to easily specify routes, and to quickly react to situations, such as the need to escape from an enemy.

Similarly, a unit's line of sight can be shown by tapping on its icon, in the form of an isovist visualization (see Figure 6B). The heading of the unit can be modified with a circular widget. To hide the line of sight and circular control, the interactor simply taps the unit again. This visualization tool allows interactors to easily organize formations of units to cover a specific area.

Similarly, to limit the number of controls, feedback indicators are displayed automatically only as needed. For example, a small label indicating the terrain type (e.g., forest, road, water, land) is automatically displayed close to an operator's point of touch. This feature supports terrain exploration without the need of additional controls.

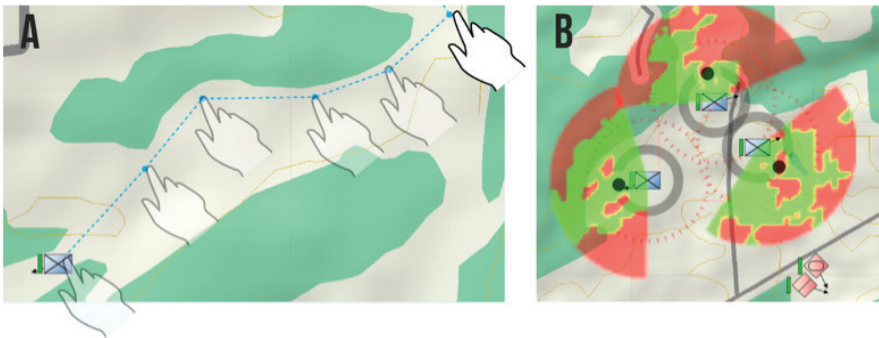


Figure 5. A) Routes in OrMiS are specified using a simple dragging gesture; B) Three units' isovist viewsheds.

In contrast to the existing simulation interfaces, all of OrMiS' controls (described above) have the advantage of being located directly in the context of the elements with which they are associated (e.g. unit, map, route) rather than on separate interface elements or in external windows. To interact with the system, interactors do not need to switch between controls and the map, but can directly apply their actions to the units themselves. As we describe below, both simulation experts and officer trainees have reported that the OrMiS interface can be learned in minutes. This is in sharp contrast to the equivalent features in the ABACUS and JCATS simulation tools, which require days of training before each exercise.

### **Supporting Coordinated Tasks & Awareness**

The current physical setting of the simulation room and the existing PC-based simulation tools hinder both explicit and consequential communication. Explicit communication involves planned, intentional behavior, such as verbal expression, or non-verbal actions such as pointing or gesturing (Baker, Greenberg, & Gutwin, 2001). For example, an interactor who calls across the room to initiate an attack is using explicit communication.

Consequential communication occurs when a person does not necessarily intend to communicate with another person, but nonetheless conveys information to an observer. For example, an interactor positioning his/her units in a specific formation may communicate the intent to attack to someone watching his/her actions. Consequential communication between interactors relies on their common understanding of military tactics and procedures, and on their ability to observe each other's actions.

OrMiS provides a shared physical and virtual workspace for interactors to perform their actions, and thus supporting both explicit and consequential communication.

### **PC-based Setting and Communication Issues**

Existing simulation tools poorly support both explicit and consequential communication. Interactors sit at their own desks, using a PC, possibly distant from other interactors with whom they are coordinating activities. This physical arrangement limits opportunities for explicit communication between interactors during an ongoing exercise. We have observed that rather than talking directly, interactors call to each other across the room. This does not work for extended or complicated conversations. When calling across the room, interactors cannot reference shared materials, such as pointing at a map. Instead, they need to turn or stand up and walk to another interactor's workstation. In practice, they are rarely willing to do so, and the quality of coordination suffers. The current physical arrangement makes it difficult to coordinate complex scenarios that involve dependencies between units being controlled by different interactors. For example, interactors using existing simulation tools find it challenging to move infantry units along a road while flanking a tank. This scenario requires the two interactors controlling the infantry and the armour units to look at each other's screens or to verbally communicate across the simulation room while performing their actions.

These scenarios are so difficult to perform with existing tools that in practice, the interactors typically change ownership of units so that the tightly coordinated units are under the control of only one person. This requires a high level of expertise with the simulation interface. As we will see, OrMiS improves explicit communication between interactors to directly enable high degrees of coordination, allowing such complex scenarios to be carried out without the need for interactors to change location, to call across the room, or to modify the order of battle.

The current physical setting and existing simulation tools also limit consequential communication between interactors. With JCATS and ABACUS, interactors share the state of the battlefield on their screens, and thus, theoretically can observe the actions of other interactors within the battlefield context. In practice, however, interactors typically filter out other interactors' units and zoom and pan to different parts of the map, as their current task requires. This means that other interactors' actions may not

be observable and interactors may not be aware of important movements executed by their colleagues. To help with global awareness, a large monitor in the back of the room displays a map of the complete battlefield (see Figure 2). However, interactors rarely look at this screen, since they are typically focused on their own PCs. When interactors are working on separate parts of the map, consequential communication is insufficient to maintain awareness of other interactors' actions.

### **OrMiS Supports Communication with Space-sharing Techniques**

OrMiS supports both explicit and consequential communication by allowing small groups of interactors to work together around a digital tabletop. The tabletop interface naturally improves awareness by providing a shared physical and virtual workspace and enabling face-to-face communication. Consequential communication is supported through peripheral vision around a shared tabletop and explicit communication is facilitated by the physical configuration of the group around a shared workspace.

However, relying solely on a shared tabletop is not sufficient to support activities where interactors need to view different parts of the map at different levels of detail. For example, two interactors may plan routes for different units on different parts of the map, both requiring a detailed view of their part of the map; this would be a form of loosely coupled coordination, as they are working to the same global objective, but at the moment are working separately. This first scenario requires little (if any) explicit communication, but consequential communication may be important to retain general awareness of the locations of the other interactors units.

Conversely, two interactors coordinating the passage of units through the lines need to see the same part of the map in detail, each controlling the units for which they are responsible. This latter scenario is an example of tightly-coupled coordination, where the interactors are working closely together and attending carefully to the other interactor's actions. In this scenario, both explicit and consequential communication is important.

To assist with the requirement to support both loosely and tightly-coupled collaboration and both consequential and explicit communication, we implemented in OrMiS a set of interaction techniques, each adapted to different situations:

1. *The main map* (Figure 6A) provides a shared space for interactors. The map can be zoomed using a standard pinch gesture. The main map is suitable, for example, for tasks where several interactors need to move units in a coordinated manner, or for the passing through the lines scenario described above. The main map supports explicit communication by providing interactors with a shared space that they can point to in discussions. It also supports consequential communication through the fact that it is visible to all interactors, providing ongoing awareness of the state of the simulation.

2. *Bifocal Lenses* (Figure 6B) provide a circular area that can be zoomed independently of the map itself. As the name implies, a bifocal lens magnifies the part of the map over which it is placed. The position of the lens shows others what part of the map is being used, fostering consequential communication. Lenses are particularly useful when two interactors need to maintain awareness while working with detailed views of different parts of the map, as with the scenario of two interactors planning routes for units in different parts of the map.

3. *Personal viewports* (Figure 6C) provide a rectangular area that can be panned and zoomed independently of the main map. Unlike bifocal lenses, personal viewports do not magnify the part of the main map where they are located, but are independent of the main map. Therefore, viewports provide support for explicit communication by enabling face-to-face communication. However, since they are decoupled from the main map, it can be difficult for a person to determine what part of the map someone else's viewport is showing, limiting consequential communication.

4. *Tablets* (Figure 6D) provide viewports on the shared map that are displayed on a separate hand-held device. Tablets allow people to work independently around the digital tabletop. Actions performed on a tablet (e.g., moving a unit) are directly propagated through the network to the other displays. Tablets are similar to personal viewports, but provide a higher degree of privacy, and do not take away screen real estate from the main map. Tablets provide poor awareness of others' actions, since it may not be easy to see what other people are working on. Tablets are best for individual work requiring a low level of awareness. Therefore, tablets are similar to personal computers in their support of explicit and consequential communication but are particularly useful for individual actions.

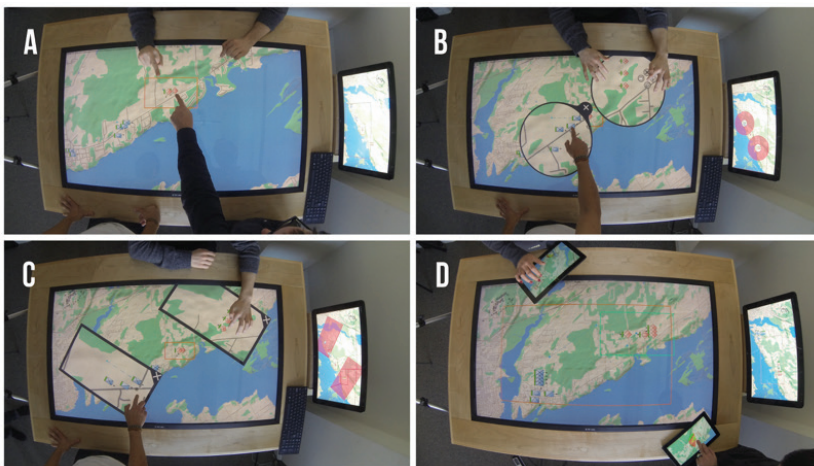


Figure 6. OrMiS in three different settings:

- A) Only the main map is shown, ideal for planning,
- B) Interactors with bifocal lenses working on close parts of the map,
- C) Interactors with personal viewports working on separated parts of the maps,
- D) Interactors with individual tablets around the table.



In addition to these techniques, OrMiS also provides a general overview of the battlefield on a separate screen. This radar view (see Figure 3) is synchronized over the network so that changes performed on the table or on the tablets are shown immediately. The radar view displays the entire battlefield at all times, providing general awareness information even when the main map is zoomed. The radar view shows the position and area shown by the main map, lenses, personal viewports and tablets within the battlefield. Similarly to the large monitor in the setup currently used by the CSTCDC (Figure 2), this view provides general awareness for interactors throughout the simulated exercise.

These four space sharing techniques and the radar view support a continuum of collaboration scenarios, from the main map for tightly coordinated actions to individual work on tablets around the tabletop. In addition, the use of each technique conveys different information about interactors' work and position on the map. With OrMiS, interactors can choose whichever interaction technique best suits the current collaborative scenario, and as a result provides the level of support for consequential and explicit communication required by the given situation. In the next section, we address the third and final issue identified in the existing simulation environment: flexibility to plan ad-hoc or impromptu changes.

### **Flexibly Supporting Changes in Plans**

A typical military training exercise is organized around four major steps: planning, battle management, battle updates and after-action review. First, interactors plan their movement based on trainees' orders. This usually includes war-gaming on a large map table as depicted in Figure 2. Then, interactors execute the plan using the simulation tool on their PCs. During the plan's execution phase, interactors regularly provide updates to the trainees. When the exercise is finished, interactors and the trainees gather and perform an after-action review to confirm how training objectives were met and to discuss lessons learned. In practice, unforeseen events occur, forcing the trainees and interactors to reconsider their plans.

### **Re-planning and Workflow**

During simulations, unexpected events may arise. For example, the officer trainees might change their plan after receiving updates from the interactors and provide truly unexpected orders. Reasons for such changes are various and related to the strategy adopted by the trained officers in the headquarters. We observed that the interactors' reaction to unforeseen events depends on the impact of the event on the original plan. If the event requires minor re-planning, the lead interactor verbally communicates the changes to other interactors. Because interactors are retired officers with significant experience in command and control, this type of minor re-planning is usually performed without problem. On the other hand, if major re-planning is needed, interactors usually gather around the bird table to re-plan. Because the paper map on the bird table is not automatically updated, interactors have to manually position the units on the table before

proceeding to the planning phase. Meanwhile, one interactor is left in charge of monitoring all the units while the others are re-planning. Therefore, only automated movements (e.g. moving along a defined road or performing a pre-programmed patrol) can be performed, potentially impacting the realism of the simulation. For example, units' reactions to an attack may be delayed or orders sent by the PTA can be missed.

### **A Diversity of Co-located Setups**

As described in the previous section, OrMiS provides a set of interaction techniques to support both individual and collaborative work on and around the digital tabletop. These techniques enable interactors to work together on the table at different levels of coordination or to work independently on tablets. For example, in the early phase of the exercise, the main map on the tabletop provides a shared space to a small group of people, enabling those people to communicate face-to-face, using speech, pointing and gestures. During battle management, the lenses, personal viewports and tablets allow interactors to work in different ways depending on the level of coordination and awareness required. For example, two interactors can work closely using the tabletop while the others perform independent work on their tablets.

Because these techniques are located directly on or around the interactive tabletop, the effort for transitioning between them is low. When performing the exercise, if unexpected events occur, interactors can immediately switch to a re-planning phase by looking at the tabletop display in front of them. During re-planning, interactors can place their tablets on the table's edge to ease collaborative work over the table itself (see Figure 3 and Figure 7D). During collaborative planning, interactors can monitor their own units directly on their tablets, through a personal viewport or by looking at the radar view. For example, if an unexpected attack happens, the event appears directly on the tabletop display and on the radar view. Concerned interactors can then immediately respond without interrupting the planning phase. Finally, the repositioning of the units on the table is avoided since the state of the battlefield is automatically updated by the system. Once the plan has been changed, the transition to battle management can be achieved in the same way. Thus, OrMiS' physical organization around a table and tablets ease transitions between different work styles and activities.

### **User Feedback about OrMiS**

When designing OrMiS, we solicited regular feedback from military officers and simulation experts to understand the required features and to get feedback on OrMiS' interface. We also assessed the usability of OrMiS with a group of officer candidates. We now report on their feedback.

We invited six pairs of officer candidates from a nearby military university to perform a simple but realistic scenario with OrMiS. There were 12 male participants, between the ages of 18 and 30 years old. All participants held the Basic Military Officer Qualification–Land (BMOQ-Land), requiring

knowledge of the topographical standards used in military maps, as well as basic troop deployment strategies. Each pair was asked to perform the scenario illustrated in Figure 7. The scenario was introduced to the participants as follows:

“Infantry units (1B, located to the west) and armour units (1A, located to the east) have been operating separately. The commander has ordered a new mission involving a platoon of infantry and armour elements. Your task is to move the infantry and armour to the rendezvous point (2) and then proceed towards the objective (3). There is a high risk of enemies located in the wooded area flanking the main road. Send your armour with infantry escorts to sweep the forest in order to avoid ambush.”

This scenario was designed in collaboration with senior military officers. In the scenario depicted in Figure 7, one participant controls the armoured units located at 1A, and the other controls the infantry units located at 1B. Their first task was to rendezvous at position 2. They were then to move through hostile territory to the objective position 3, with the infantry flanking the armour in order to flush out enemies located in the woods.

Participants were first trained in the OrMiS system, and allowed as much time as they wished to become familiar with the application and the interaction techniques. The version of OrMiS presented to participants was limited to the use of the main map, bifocal lenses and radar view; the personal viewports and tablets were not available. Training time typically lasted 15 minutes. Participants had no time limit and on average spent 9 minutes to complete the scenario ( $M=9:12$ ,  $SD=2:00$ ). After completed the exercise, participants were asked to complete a usability questionnaire based on the System Usability Scale standard (Brooke, 1996) including questions related to the main features, the lenses, main map and radar view. Participants were then interviewed.



Figure 7. Collaborative scenario used during the study.

## Results

All participants completed the task without encountering significant usability issues. In interviews, participants were positive, reporting that they found the interface easy to use and appreciated using the table to collaborate and to enact their plans. One participant stated: "I really liked the table, it was very intuitive". Participants also liked the labels indicating the terrain type. One participant said: "when we clicked it would tell us if it was water, road, etc. and that was really handy. I liked that." Similarly, when asked about the usefulness of OrMiS, one participant said "...for planning the route, I found it was actually pretty good!". These results indicate that operators enjoyed the OrMiS's interface when performing the scenario.

In terms of collaboration, participants successfully took advantage of the different interaction techniques to split their work. All the groups used lenses for the first part of the scenario (from 1A/1B to the rendezvous at 2 in Figure 8) where no specific coordination was required. Participants expressed strong positive feelings about the lenses because they allowed users to work simultaneously without disturbing each other. The majority of the groups switched to the main map in the second part of the scenario (from 2 to 3 on Figure 8) where units had to be tightly coordinated. Prior to switching to the main zoom, most users quickly discussed which way to proceed to coordinate their units. As expected, the tabletop setting eased face-to-face communication. Participants also noticed the limitation of both interaction techniques. Several participants experienced overlapping problems between the lenses when working physically closely on the table: "when we are close, the lenses stack together even if there is a lot of terrain between the two lenses". This shows the importance of providing the zooming feature in the main map so that collaboration is possible around closely located points.

The scores obtained with the SUS questionnaires confirmed this feedback and revealed interesting differences between the features. Lenses and main map respectively obtained an average SUS score of 65.4% (SD=3.2) and 67.5% (SD=5.1) indicating a high level of usability for both techniques. However, the radar view was perceived as less usable, obtaining only a 19% (SD=3.58) usability score. During the interviews, participants reported that they did not use the radar view much. We believe that since there were only two participants and four units, participants did not require the radar view to maintain a global view of the battlefield.

Over all, these results confirm that OrMiS enables a pair of people to perform a simple but realistic scenario with minimal training, allowing the pair to complete their task, and communicate in both explicit and consequential forms. This is in a sharp contrast to the current setting using simulation tools like ABACUS or JCATS, which require days of training and significant effort to maintain awareness and perform tightly coupled movements.

## **Lessons Learned**

In addition to these results, the participants provided us with insightful feedback helpful to the design of multi-touch systems supporting simulation-based training. Two participants reported ergonomic and orientation issues: "The table should be higher or angled ... there is clearly one side that's better". One participant complained about pain in his neck at the end of the study, indicating the importance of making the height of the table comfortable for extended touch interaction. As participants were working face to face, one member of each pair saw the map upside-down, and had to make an additional cognitive step to correctly interpret cardinal references. We believe that the introduction of tablets and personal viewports that can be oriented will solve this problem.

Participants reported that they had to verbally communicate to avoid conflicts when working together on the main map: "[We] had to create a seniority of who was allowed and who was in control of the board, because at some points I would go touch something and it would screw him up, ... so we had to have one person who would say don't touch it until I'm done". This result is in line with previous findings in digital tabletop research showing the importance of social protocol when working on shared spaces. Simple interaction techniques like using two fingers for panning (instead of the more traditional one-finger panning) can reduce unintentional actions and consequently conflicts.

## **Conclusions**

In this paper we first provided an overview of the state of the art in tabletop research for collaborative work and more specifically for map-based applications. Through this literature review, we illustrated that collaboration around tabletop requires specific support to the various collaborative work styles.

We presented OrMiS, a multi-display environment dedicated to military simulation based-training. OrMiS combines the best of existing space-sharing techniques dedicated to interactive surfaces. The OrMiS system provides a simple interface combining zoom, lenses, personal viewports, tablets and radar views to provide maximum flexibility during the exercises. We showed how features of OrMiS solve important usability, coordination and communication issues encountered by interactors during simulations. To assess the usability of OrMiS, we reported on feedback from officer candidates at a military university. Our results show that users are able to perform a simple but realistic scenario with minimal training with OrMiS, and they overwhelmingly enjoyed using the tool. We also highlighted some interesting limitations of OrMiS such as orientation issues of the map or the usefulness of the radar view when few units have to be monitored.



## TableNOC: Touch-Enabled Geo-Temporal Visualization for Network Operations Centers

*Pierre Bastianelli, Theodore D. Hellmann,  
and Frank Maurer*

### Introduction

Many modern telephony networks are spread across very wide areas, routing different kinds of information via different data pipelines. Because so many systems depend on these networks, problems can be extremely expensive. Considering this, it is critical for network operations center (NOC) staff to understand the state of a network quickly and easily. However, large, distributed networks can be quite complex.

In order to understand the state of a network, NOC operators poll various information about the system – such as server load statistics, memory usage levels, and log files. This information tends to be conveyed through existing tools as static charts or text. While this information is useful for debugging serious errors in detail or understanding network performance in retrospect, it is difficult to interact with or use for understanding the state of the network in real time. Additionally, it is even more difficult to use for proactively addressing issues that may become serious in the future, and does not easily convey the distributed nature of the network.



Figure 1. TableNOC running on a SMART board and Evolve One digital table.

This paper describes TableNOC, how it was implemented and evaluated within the business context of Ivrnet, a small telephony-service provider in Calgary, Canada. TableNOC is a touch-enabled tool for visualizing and exploring information about the status of a telephony network in a way that reflects this information's temporal, spatial nature.

TableNOC presents visualizations of large amounts of call-related data and allows users to browse through the data as a time series to observe the evolution of a situation. Further, TableNOC includes visualizations that update in real-time in order to reduce the latency between the moment a problem occurs, and when it is detected by an operator.

The pilot study that we ran validates how well the tool addresses issues encountered by various roles at Ivrnet. We present the different use cases we envision TableNOC being used for and the results of a design critique conducted with potential users. We found that the users were enthusiastic about the possibility of using such a tool, and that the integration of interactive surfaces added capacity to its data exploration abilities and collaboration capabilities.

TableNOC's core capabilities include geospatial visualization of real-time and historical network data, data aggregation for different time intervals, custom visualization builder and touch-enabled visualization exploration.

## **Context**

The NOC is the core of a telecommunication company. It is the link between the physical entities: servers, routers, phones, phone lines, SIP-lines and other Voice over IP equipment, Primary Rate Interfaces. All of these hardware pieces are linked together across vast geographic areas, while the information carried by the network is consumed by end users, data centers, or call centers. The role of the NOC and its operators is to monitor the status of all the equipment on the network and the data that is going across it. Equipment not owned by the service provider but that is connected to the internal network has to be monitored as well, in order to ensure that all the services and applications – including telephony, conferencing, Interactive Voice Responders (IVRs), texting, automated phone calls, and various web services and applications – have as little downtime as possible.

The network and services that rely upon it generate an incredibly large amount of data. The challenge for NOC staff is to be able to make sense of all of this data, including network load, software errors, hardware failures, specific configurations and re-configurations, etc. Further, this data is often displayed as text (log files or databases) in the worst case, or charts in the best case. Neither of these forms really conveys the essentially geographic nature of the data they represent. The operators have to rely on their experience to use this data to understand the state of the network.

With no global overview of the situation, it is close to impossible to

recognize emergencies or recurrent patterns that would allow a much quicker identification of the problem or better long-term planning. There are many limitations that make it difficult to comprehend and even more difficult to predict in this scenario.

At the same time, in spite of the high quantity of information available, little is being presented in real time in a comprehensible form in the NOC. The collection and consolidation of the data has to be done on a personal computer and then shared via the traditional information channels and is not readily available for collaborative work. Therefore, there is a clear need for a collaborative environment where:

- The NOC operators can discuss and view the same data at the same time
- The data is presented as a set of comprehensible, interactive visualizations

In this way, we envision operators can either work on concurrent issues impacting different areas of the system and visualize the impact of their actions across the whole network; or, work together on the same issue, while discussing and interacting on the same data.

As tabletops and multi-touch environments have been shown to enhance collaboration (Scott, Grant, & Mandryk, 2003), their use with TableNOC makes perfect sense, in addition to the fact that surface and touch environments are very well adapted to the world of GIS (geographic information systems). As the usage context differs, we made TableNOC platform-independent in order to have it running and tested on various types of surfaces: tablets, tabletops, interactive walls, etc. Therefore, client-side web technologies such as JavaScript and Ajax were chosen for its development.

Finally, TableNOC was developed using user-centered design (UCD) methods, with an iterative process, along with an agile development cycle. This way, the industry related concerns (costs, delays, resources and especially specification change) could still remain at the core of the project and allow a smooth tradeoff with all the functionality that was designed.

## **Solution**

TableNOC is a tool for visualizing and interacting with telephony network data implemented as a research and development partnership between Ivrnet and the Agile Surface Engineering group at the University of Calgary, funded by an NSERC Collaborative Research and Development Grant. TableNOC's goal is to better support decision-making within a NOC by providing automatically-generated visualizations of the past and current state of the network that can be explored for greater understanding.

In order to facilitate collaboration and increase ease of use, TableNOC is implemented as a touch-enabled application. Further, TableNOC is



implemented as a web application in order to enhance cross-platform compatibility. While we found this to be beneficial, it is worth noting that supporting touch uniformly across a variety of platforms and browsers can be a resource-intensive proposition.

At a very high level, TableNOC generates visualizations either from static files or from polling web services and displays these visualizations to users. This is accomplished by tying together a variety of different systems. A description of how TableNOC works for real-time data is shown in Figure 2.

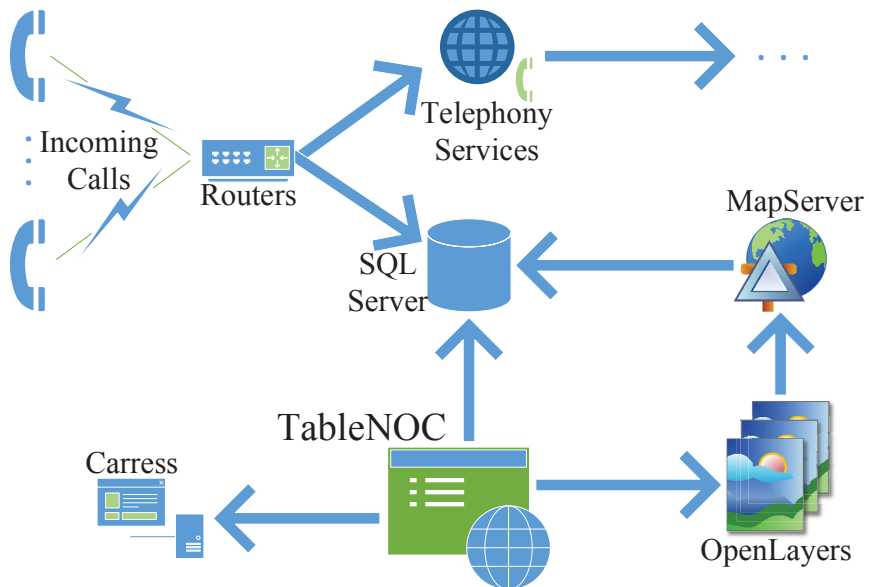


Figure 2. Architecture of TableNOC.

Phone calls and texts are initially handled by Ivrnet’s routing hardware, which forwards these as appropriate to other hardware within the system, where the calls are redirected to further hardware or software systems for handling. The routers also create a record of the call in a centralized SQL Server database. This database, or replications thereof, can be used for storing and geolocating (finding the geographic location of the source of a call or text) data.

Next, we use MapServer (<http://mapserver.org/>) to pull data from the SQL Server and create map layers – mostly-transparent images showing some form of visualization, such as lines representing provincial boundaries or points representing the origin of a phone call. It is important that these images are mostly transparent in that this makes it possible to stack them on top of each other without losing the ability to see underlying layers. Because of this, it is possible to build visualizations out of discrete layers, such as a base map of Canada below a layer showing the point of origin of calls received by Ivrnet. We use OpenLayers (<http://www.openlayers.org/>) in

order to keep track of, and appropriately stack these layers. The TableNOC web application then displays the layers provided by OpenLayers for users to interact with.

In order to avoid issues with touch input that will perform differently on various operating systems with multiple web browsers, TableNOC makes use of Caress (<http://caressjs.com/>), a JavaScript-based touch and gesture toolkit. This allows us to consistently handle touch input independent of individual browsers and systems.

### **Design for Multiple Form Factors**

We expect TableNOC being used in two main use cases: information radiation and data exploration.

An information radiator is a large, easy-to-read display posted in a public area so that people can quickly understand information of value to them (Cockburn, 2004). TableNOC is intended to be displayed in a highly-visible, collaborative space, so people can get an idea of what is going on with the whole network with a glance.

In order to achieve this, TableNOC was designed to display a high-level overview of the status of a network using different symbols, colors, sizes, and links between elements. In the near-real time visualization mode (explained in more detail below), for example, TableNOC denotes call centers using a telephone icon; denotes the number of calls coming out of a given area code as a circle with increasing size and increasing redness indicating higher call volume; and draws a link between the source of a call and the call center that handles that call. This link will grow larger as the number of calls increases. In this way, we aim to communicate through high-level visualizations so that people spend less time digging for details. If network operators need to obtain more detailed information about a smaller geographic region, a specific call center or call source, we have designed TableNOC to also work on personal touch-enabled devices such as tablets. As described in the section below on TableNOC's touch-enabled web interface, users are able to quickly navigate TableNOC's visualizations.

### **Visualizing Historical Data**

One of the major use cases for TableNOC is the visualization of large amounts of data covering various slices of time. This can be useful for answering questions about how the customer base of a service behaves as a whole. For instance, in one service for a customer who only did business within Canada, we became curious as to why we were getting calls from outside of Canada and how we should handle these calls.

We created a heatmap visualization of over 300,000 call records from a single month. In this sort of visualization, the size and color of circles on the map is tied to the number of calls coming from that region – similar to Figure 3. This visualization allowed us to understand a good amount about

our customers – including, where the highest volume of calls was coming from. For understanding historical trends, we added an additional feature to allow us to view the data in more discrete time slices. Similar to the concept of a time series, we added a feature to allow users to restrict the heatmap to only data collected within certain time periods. This enabled us to see that calls from outside of Canada were only occurring later in the day – perhaps symptomatic of customers on vacation. This implied to us, that these calls still needed to be handled even though they were coming from outside the expected business area. By stepping through different time slices, we help users gain an understanding of how the network functions during different discrete intervals rather than just as a whole, allowing more fine-grained understanding of the network.

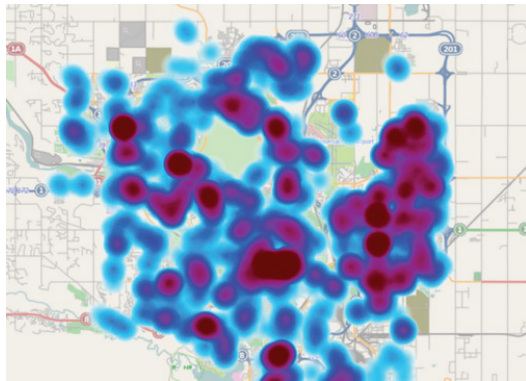


Figure 3. A heatmap visualization.

### **(Near) Real-Time Visualization**

Beyond the first major use case for TableNOC, it can also be used to visualize the network in near real-time (NRT). This is beneficial to keep track of a situation as it evolves. For example, we can use the system to monitor how the quality of service provided by a call center changes over time.

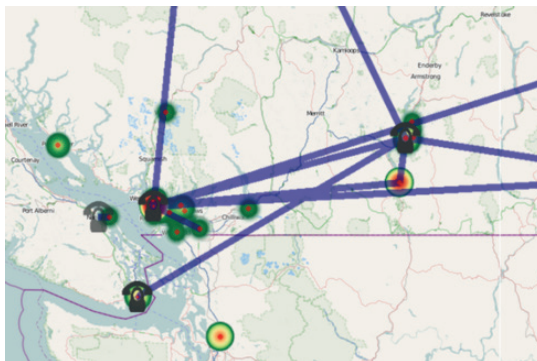


Figure 4. (Near) Real-time visualization.

The NRT visualization differs from the standard heatmap visualization in that it also provides a link between the source of a call and the call center at

which it is ultimately answered. This link increases in size as more calls are directed into a given call center, providing a visual representation of call volume (see Figure 4). Importantly, network operators can visually identify inefficient call routings, just by viewing a graphic link between source and sink for calls. While individual calls are not expensive on their own, the large volume of calls handled by Ivrnet means that even small savings add up quickly, so identifying when sub-optimal routing paths are being used is a valuable feature.

Further, we can monitor characteristics of calls – such as whether the caller hangs up before a service representative picks up or how long on average a caller has had to wait before being served – in order to gain an understanding of call quality. This allows network operators not only to quickly understand if a problematic situation has occurred, but also gives them more context about where problems are occurring compared to the traditional system (Figure 5).



Figure 5. The existing NOC.

### Detailed Summaries

The visualizations described thus far have been focused on providing a high-level view of the network so that problems can be spotted across the network as a whole. However, we also included features to gather a summary of network data from within TableNOC itself to enable users to access specific details about the network without having to access the data directly.

First, TableNOC presents a summary of all data in currently active layers. This summary is accessible through a tab on the main interface and, for

historical data, shows the number of calls represented by currently-enabled visualizations.

Second, it is also possible to single-tap on points within a heatmap to get a summary of data pertaining to that location. If a point representing a call center is selected, a summary will pop up showing the number of calls currently in progress, the number of calls that disconnected before they were answered, the average length of a call, and the maximum length of any call answered by that call center. This allows us to alert customers to issues pertaining to their own services – for example, average call center wait time.

### **Touch-Enabled Web Interface**

TableNOC's interface was designed to be as cross-platform as possible in order to cater to clients with different preferred operating systems. However, this proved to be significantly challenging as we designed this application to be touch-enabled, running on various types of devices. Initially, we ended up writing custom code for handling touch interaction in various popular browsers. However, this quickly became impractical. In order to keep touch interactions working consistently, we turned to a tool designed for this purpose: Cares.

Cares is a JavaScript implementation of the TUIO protocol that allows us to directly receive and interpret touch events in a cross-platform way. By using Cares, we can bypass the idiosyncrasies of how individual web browsers handle touch events and focus on implementing touch interactions.

In TableNOC, we use touch as a method to help users explore data. Users can perform standard map-based interactions such as pinch/zoom and pan actions. The entire interface is also designed to be large enough to interact with via touch on a tablet such as a Microsoft Surface or iPad. Further, on maps with regions defined by shapes (school districts, provincial boundaries, electoral districts, etc.), we have experimented with the ability for users to select specific regions in order to narrow down the data that is presented in summaries.

### **Data Import System**

Currently, users can specify either static, local files or web services as data sources. For files, users do not need to know how the file is delimited and which columns contain geo-locatable information, whereas for web services, the schema will need to be known. Once the data source is determined, users then create layers in order to determine what sort of visualization they want to use. Finally, users are able to add filters to layers in order to visualize only data from the data source that meets the given criteria. The system guides its users through the process of uploading data and creating visualizations in an understandable manner so that technical expertise is not required to perform these tasks.

## **Pilot evaluation: Design Critique**

As opposed to a focus group, a design critique is a meeting with users where the discussion is not oriented towards new features or items but instead to decide if a prototype, with specific features, is fit for use and how it can be improved. The goal of a design critique is to refine the existing application rather than create new elements of design. It has been defined as “a process of discourse on many levels of the nature and effects of an ultimate particular design” (Blevis, Lim, Roedl, & Stolterman, 2007). It can be focused on certain aspects of the product to make it more efficient: in our case, we had three major questions to answer:

- What missing functions would enhance TableNOC?
- Why were users not drawn to use the software yet?
- How can TableNOC’s usability be improved to make users more efficient?

A design critique appeared to be a good choice given that we wanted to find out if people would actually use TableNOC and its available resources. The design critique has the major advantage of being a rather fast method for collecting information, which is a precious point given the fact that our users are – due to Ivrnet’s business model – not likely to be available for research purposes. Therefore, it is a great alternative to a more traditional survey that would have taken more time and resources from our users.

## **User Profiles**

We had the opportunity to run our pilot evaluation with participants with three different roles within Ivrnet:

- P1, the operation/support team manager. Runs daily reports on the system health and manages the NOC. In charge of dealing with problems with the network.
- P2, the network manager. Sets up Ivrnet’s network architecture, and is in charge of monitoring hardware settings and health. If a piece of hardware fails, he is responsible for replacing and reconfiguring it.
- P3, a senior manager. Keeps track of situations to communicate efficiently with customers or the rest of the management team.

These roles are different but similar: the business needs (from a TableNOC perspective) are the same but with a different scale. Each user is interested in visualizing the NOC status from a different level of abstraction. P2 is interested in hardware statuses and the availability of equipment, P1 is positioned a bit higher and needs to see the statuses of services and log file sizes, whereas P3 needs to have a complete overview of the network.

## **Protocol**

In order to be able to run a design critique, our users first had to be exposed to TableNOC. This was done in two ways. First, it was left running on a large flat-screen TV in the development/operations meeting room. The goal of

this location was: 1) to make people aware of the application and to draw attention to it; 2) to make key information available to operation people, especially during meetings in collaborative spaces.

Second, we demoed TableNOC in detail to each participant and provided him or her with access to the system. The goal with this phase was to provide the application as-is to the users, giving them the opportunity to use TableNOC in their own work and on their own time. This was intended to evaluate the relevance of the use cases we identified and see if there were use cases we did not address. Another expected effect was to favor serendipity, and determine if our users would interact with the application to discover unpredicted new ways of working with the tool.

### **Findings and Analysis**

Even at a prototype stage, TableNOC proved useful from a business perspective in a broad range of situations. First, it has been used for data representation and analysis. Ivrnet has used it for the visualization and analysis of network data on several phone surveys. TableNOC was used to display this data in a graphical manner and to visualize trends across geographic regions. Another example is the texting project wherein users text into the service to get information about upcoming services at specific locations. As a result, using TableNOC's visualizations, it was possible to infer information about demand-for-services at specific locations. This was visualized using a heat map representation, similar to Figure 3.

Second, TableNOC was also running at a research facility at the University of Calgary during several events involving unusual and very high call volume into services. TableNOC was set up to monitor in real-time. During these events, TableNOC immediately displayed a change from the nominal situation due to higher phone call traffic and failed phone calls. Further, the researchers noticed this activity very close to its start and, upon getting in touch with Ivrnet staff, found that they had in fact used TableNOC to detect an unusual network situation.

From our interviews with our users, we had mostly positive feedback on the software. Critics were not oriented towards wrongly targeted features and were very constructive. Most of feedback was about additional features that would make the software perfectly suitable for each user's needs. The critics can be summarized from high-level interface issues to low-level hardware-related issues.

We sort the feedback into three categories: Missing Features, Usability Issues, and Technical issues. Features include all the items that would make TableNOC really useful to whoever makes the comment, whereas Usability addresses usability issues and suggested improvements.

*Missing Features.* The main missing feature of TableNOC is the call center detail display. Two of our users mentioned that they would like to see

everything that is related to the different call centers that Ivlnet interacts with across the country:

- Detailed call center information: Total call count, average call length, total call time, number of calls per queue and per call center, etc.
- Concurrency graphs of phone calls
- Static graphs showing capacity vs. load
- Cost of common call routings
- Show the main contributors in the traffic at and between call centers

Another request was to display more information about the phone calls themselves: What are the status of the current phone calls, especially if the call fails, and the ability to trace the phone call not only geographically but also on a traditional (symbolic) network map between different pieces of equipment at Ivlnet's office.

The last item is about the different pieces of hardware that are in the call centers and in the NOC: TableNOC should be able to display the current health of the different pieces of equipment and to show important information from the logfiles.

*Usability Issues.* Although a lot of time has been spent on paper prototyping, some usability issues remain. TableNOC's response time was most annoying to users. Load and reload operations are quite long and tedious, bringing the user to a halt in his or her work experience with the software. Some of this issue has been addressed by introducing cacheing during zooming operations, preventing the screen from going blank while waiting for images and visualizations to reload.

Some of the controls received bad reviews, too: their size appeared to be too small for touch interaction on certain devices. The layer management widget lacked feedback on which layer groups are active or not, making layers nested within a folder structure difficult to use.

*Technical Issues.* Most of the technical issues encountered by our users were browser-related. Chrome seems to have difficulties supporting the application, especially on MacOS X or when handling touch interactions.

*Conclusion.* As a conclusion, users were really pleased and enthusiastic with TableNOC, and it appears to be a success from a user experience point of view. However, the missing features were responsible for the fact that it was not used as much as we hoped. Using surfaces for supporting NOC operations in an industrial context seems to be the right move, but our pilot study shows that without a very good integration with the business' equipment and servers, such a tool remains incomplete. A fully functional application tied to the data coming from the core of the NOC, will be of great interest for all the users interacting with the NOC and fully support



their day-to-day activities, allowing them to focus on what is important about the NOC.

### **Related Work**

TableNOC is at its core, a network visualization tool. Several related tools have been produced, and research publications have also taken several approaches similar to ours. In order to restrict the amount of work discussed in this section, we focus on tools and publications that present geospatially-focused visualizations.

### **Related Tools**

In terms of tools, two are important to note: Ni3, by DocsLogis (DocsLogic Healthcare Solutions, 2013); and the Vodafone Fixed Network-Visualization Tool for ArcView GIS (FVT) (Belfqih, 2003).

Ni3 is noteworthy in that it presents a geospatial view of a network including standard GIS features such as panning and zooming while also including functionality for obtaining detailed information about specific nodes in the network. For example, it is possible to overlay various charts describing the characteristics of a network – similar to TableNOC’s ability to tap on a node to bring up data. However, this tool is not optimized for real-time visualizations and is not designed for easy, touch-based interaction.

In addition to providing visualizations similar to Ni3, the FVT is interesting in that it includes a data upload tool similar to the one presented in TableNOC. This feature allows users to configure FVT to pull data from given databases and generate reports from within the tool itself. However, unlike the version presented in TableNOC, FVT is not geared towards non-technical users.

### **Related Research**

There is a large body of existing work on network visualizations. In order to narrow the field considered in this paper, we restrict our discussion here primarily to geo-temporal visualization systems. Within this area most existing research focuses on either trend analysis or interactive data exploration.

Tools that focus on trend analysis provide some sort of automated pattern recognition to better help users understand what aggregate data means (Roe, Murphy, & Schmidt, 2009), (Eick, Eick, Fugitt, Heath, & Ross, 2008), (Behnisha, 2010), (Malik, Maciejewski, Hodgess, & Ebert, 2011). This can be done for example using statistical processes – like correlation coefficients – in order to understand if the current state of the data is abnormal (Malik, Maciejewski, Hodgess, & Ebert, 2011). However, this is difficult to visualize directly and can involve displaying the data in a non-spatial format – such as a correlation coefficient displayed alongside a geospatial map. While this is useful for understanding the behavior of a system, it does not take advantage of the essentially geographic nature of the data, leaving room for future work in this direction.

Tools that focus on visualization exploration tend to be systems that promote user interaction with the data (Roe, Murphy, & Schmidt, 2009), (Behnisch, 2010), (Andrienko, et al., 2010), (Malik, Maciejewski, Hodgess, & Ebert, 2011), (Zhang, Korayem, You, Erkang, & Crandall, 2012), (Jänicke, Heine, & Scheuermann, 2013), (Patroumpas & Sellis, 2012), (Nagel, Duval, & Moere, Interactive exploration of geospatial network visualization, 2012). These systems aim at helping users analyze information through the process of exploring visualizations – by examining different areas of a map at different zoom levels and at different times, for example (Hoerber, Wilson, Harding, Enguehard, & Devillers, 2011), or by allowing users to easily switch between time series interactively (Malik, Maciejewski, Hodgess, & Ebert, 2011). The end goal is often to help users intuitively understand a network without resorting to automated trend analysis – instead presenting the data through visualizations that allow humans to more easily perceive patterns.

Of course, some systems make use of both of these approaches – presenting trend analyses in a form that users can explore to gain additional insight into a system (Behnisch, 2010), (Malik, Maciejewski, Hodgess, & Ebert, 2011), (Zhang, Korayem, You, Erkang, & Crandall, 2012). In (Zhang, Korayem, You, Erkang, & Crandall, 2012), for example, data is automatically clustered based on spatial and temporal patterns, but then presented in such a way that users can explore the data further to find additional trends on their own.

Some research has also focused on taking an essentially collaborative approach to visualization of real-time geospatial temporal data. In (Nagel, Duval, & Moere, 2012), the tool was designed to run on a digital tabletop – a good environment for encouraging multiple people to interact with an application simultaneously. Similarly, (Nagel, Duval, & Heidmann, 2011) focuses on analyzing co-authorship data through a digital, table-based application. Both of these methods encourage groups to engage with each other in the exploration and understanding of data.

### **Limitations of our Approach**

In spite of all the features provided by TableNOC and the encouraging results of the pilot study presented in this paper, some weaknesses remain. First, the software is limited in various ways. The current version does not support multi-screen or multi-device interaction, preventing multiple users to interact with TableNOC at the same time. This restriction to single-user, single-device interaction strongly limits the collaborative aspects of the software and will need to be corrected in the near future.

### **Pilot Study**

The pilot study itself also suffers from limited participation. Although the user profiles selected make sense in the business perspective of TableNOC, interviewing only three users is slightly restrictive, and the quality of the data collected in the presented pilot study would greatly benefit from additional participants. With more users operating TableNOC during a greater time

interval, this would allow us to integrate different data collection techniques such as shadowing, interviews, and questionnaires, improving the iterative design process with quality data.

### **Connectivity**

Functionality-wise, TableNOC is far from being perfect, which is understandable given that it is still a prototype. The current implementation only supports very specific kinds of data and frequently requires customization. Connecting TableNOC to Ivrnet's call centers not only requires development on the software side, but requires Ivrnet to expose specific services and APIs for retrieving the right information. Finally, the data import feature is restricted to the .csv file type and does not support other formats, which requires the user to be able to manipulate different data types, potentially requiring a good knowledge of data management or powerful equipment dedicated to handling large amounts of data.

### **Information Visualization**

Lastly, the visualizations used also need to be reviewed. We are still searching for the correct visualizations to represent different elements of the NOC and the additional data that might overlap it. The issue here is that we need to adapt the representation not only to the data, but also to the quality of the data. For example, displaying the calls will only cause one dot to appear over the city of Calgary, as there is only one routing point that the calls transit to, for this city (that is accessible to Ivrnet). The first reaction might be to increase the accuracy of our geo-location, as we currently limit precision to just the area code of a phone number. However, as the precision of our geo-location of calls increases, there will be additional privacy concerns to consider.

### **Future Work**

The ultimate goal with TableNOC is to make it suitable for use within a networking / communications business such as Ivrnet. We clearly see that TableNOC missing features and usability flaws make this difficult at present.

Future improvements will include adding additional static data types (excel, database extracts, etc.) and data sources (web services discovery, database connection) in order to expand the range of data displayed. It will also include expanding the set of symbols representing the different phone call elements: allow the display of call direction, infrastructure type (PRI, SPI, etc.), length, etc. It would also be possible for the user to determine which symbols and colors to use when importing data and building layers.

We also hope to add in artificial intelligence support in order to help identify trouble situations and optimize future network performance. Performing retrospective analysis of past trouble situations and the solutions used to overcome them will do this. On a basic level, we hope to be able to notify network operators that a sequence of events which indicates a problem is about to occur has happened, allowing these personnel time to prepare for

– and ideally prevent – such a situation from occurring.

We also envision a multi-surface environment, where multiple users would view TableNOC on their own personal or collaborative devices, but sharing the same underlying data model. This would allow easy sharing of data layers or views and improve communication especially between the different types of users. This would beautifully tackle the situation where a manager is monitoring the evolution of a situation through the same tool as his operations team is using for troubleshooting, but with a different perspective on the data.

Finally, we would like to run a more extended evaluation of TableNOC. Ideally, this would involve more than one business, as we aim at having a greater overview of what the needs are for geo-located network monitoring.

## **Conclusion**

This paper presents TableNOC, a tool that allows the geographic monitoring of large-scale, country-wide networks and its application within Ivrnet. It is meant to be used at the heart of the communications business using it, in its NOC. Its goal is to provide a geographic overview of network activity.

The results of the pilot study we conducted were encouraging and showed that there is a need for such a tool. People usually deal with a lot of raw data in the form of automatic reports and log files, and displaying all this data in a geographic way to encourage exploration seems to be of great value for the network operators. Therefore, there is no doubt left about the need of such an application.

This tool is expected to have a significant impact on the way the network businesses work. Although there is a certain reluctance to using it, explained by the fact that TableNOC is only at a prototype stage, the features offered support key elements of the user's work: efficiency, detection of failures of equipment or interfaces, prediction of crisis situations, etc.

One major difficulty is that the domain is specific to every company and use case, and so is the data displayed. As a consequence, the use of TableNOC is subject to heavy development work in order to customize it to the architecture of the company before it can be used. Once TableNOC reaches a certain maturity, it is expected to offer sufficient configuration options to accommodate a large variability of domain uses.



## **Beyond Efficiency: Intriguing Interaction for Large Displays in Public Spaces**

*Uta Hinrichs, Alice Thudt, Lindsay MacDonald, Miguel Nacenta, John Brosz, and Sheelagh Carpendale*

### **Introduction**

The requirements for interfaces and interactions in public spaces vary considerably from desktop interfaces. This chapter will discuss three pieces of research that took place in public spaces. One, the first in this chapter, is a study of an existing installation, the other two are installations we created. Over the duration of SurfNet we have installed and studied several installations of large displays in many different public spaces ranging from art galleries to libraries and museums. Each installation is unique in its goals and aspirations. We will discuss factors that affect the degree to which an installation is attractive and intriguing, being able to gather and hold attention, including the use of concepts from serendipity and complex adaptive systems to move towards possibilities of endlessly fascinating interaction.

This chapter draws primarily from these three papers. We will discuss them in order of publication.

- Uta Hinrichs and Sheelagh Carpendale. Gestures in the Wild: Studying Multi-Touch Gesture Sequences on Interactive Tabletop Exhibits. In CHI '11: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, pages 3023–3032, 2011.
- Alice Thudt, Uta Hinrichs and Sheelagh Carpendale. The Bohemian Bookshelf: Supporting Serendipitous Book Discoveries through Information Visualization. In CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, pages 1461—1470, 2012.
- Lindsay MacDonald, John Brosz, Miguel A. Nacenta and Sheelagh Carpedale. Designing the Unexpected: Endlessly Fascinating Interaction for Interactive Installations. In TEI '15: Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction, ACM pages 41–48, 2015.

In this chapter, we draw attention to three factors that we have found particularly important for large displays in public spaces:

1. Gestures and/or touch interaction, as used in real world situations (commonly referred to as 'in the wild' in our literature), are not simply a replacement for menus and buttons. Instead these types of interactions, because they involve our hands and bodies in situations where people, both friends and strangers, can see us play additional, often more important roles than triggering a system response. In these situations gestures form an important part of our social interaction. Thus gestures, while they do start interaction response, must fit seamlessly into the social activities that are currently taking place (Hinrichs et al., 2011).
2. Serendipity can be far more important than the occasional introduction of randomness. Serendipity is linked to several factors that can be programmed for. We describe an example interface that makes use of serendipity. However, these factors can be embedded in other interfaces (Thudt et al., 2012).
3. As more information gets placed on large screens in public spaces, much concern is being focused on attracting and maintaining people's attention. The third section of this chapter presents work towards, what might seem like an unrealistic goal, achieving endlessly fascinating interaction (MacDonald et al., 2015).

### **Gestures in the Wild:**

#### **Studying Multi-Touch Gesture Sequences on Interactive Tabletop Exhibits**

In this section, we present the findings from a field study that was conducted at the Vancouver Aquarium. We use the phrase 'in the wild' to refer to the fact that these interactions were taking place in an uncontrolled environment where people were choosing to interact with these displays of their own volition. In this study we explored how people make use of multi-touch gestures on an interactive walk-up-and-use tabletop exhibit. Our findings show that multi-touch gestures are conducted in a temporal sequence. That is, gestures are deeply embedded in an interaction context where previous gestures influence the choice of subsequent gestures. In addition, the choice of gestures is influenced by a social context that includes impact from the age of visitors, the visitors' personal opinions on the content shown, and the social encounters that are happening around the display. In observing these multi-touch gestures in the wild, we saw:

- a large variety of multi-touch gestures for actions,
- interaction context as an important concept that implies the need for fluid gesture sequences,
- differences between adult's and children's use of gestures, and
- that choice of gestures is influenced by the social context.



Figure GW1. These images provide examples of the types of interactions we observed.

*Background.* Multi-touch technology has been around for some years now and we starting to have commercial products that support multi-touch gestures. The ability to touch a display surface and directly interact with content directly has been found to be pleasurable and fun.

In public environments, such as museums, there has been a clear increase in the use of direct-touch technologies that allow the public to explore exhibition content in direct and playful way. This has many advantages, including:

- there is still a certain novelty effect for direct-touch technology, particularly when used on large displays,
- no external input devices are required, and
- the interaction is very visible which can draw the attention of other visitors, and at the same time teach the visitors how to interact.

Applications in these spaces are becoming more and more sophisticated and often now make use of multi-touch gestures for exploring digital information. However, the audience in such spaces is broad and their experience with technology might differ quite a bit. Furthermore, visitors usually interact for very brief periods of time with each exhibit. This can be extremely brief, for instance, two minutes is considered a long time. Given this environment, people are unlikely to read through elaborate instructions on how interact with a touch display exhibit. Thus it is important to design multi-touch gesture sets that can be used in a walk-up-and-use manner. Public scenarios, such as these, really demand that the gestures used can be applied in an intuitive walk-up-and-use way without prior practice or elaborate instructions.

In our study, we were interested in characterizing the multi-touch gestures that people actually apply in walk-up-and-use scenarios. For instance, there are many factors that influence the choice of gestures. These may well include factors such as general preferences and whether the people are children or adults or elderly.

This study took place in the Vancouver Aquarium's Arctic Exhibit.



Figure GW2 (top) & 3 (bottom). The Vancouver Aquarium Arctic Exhibit.

While we were interested in how people interacted with the table top displays, these displays were in a public setting – the Vancouver Aquarium’s Arctic Exhibit. The Arctic Exhibit included a large aquarium with many graphic and digital displays. Figure GW3 shows the two interactive tables in the foreground. Both the displays and the interactive software for these displays were built by Ideum.

The interactive software for each of these tables was quite different. One was map-based with relatively formal controls. The other, the Collection Viewer, included much more freeform interaction. For this analysis, we concentrate on the Collection Viewer.



Figure GW4. The Collection Viewer Media items, both images and videos could be freely moved around the display, could be resized, and the videos could be played at will.

To collect data, we positioned two cameras as shown in Figure GW5, left. Figure GW5, right shows the camera views from the top and from the side.



We also sat nearby and took field notes.

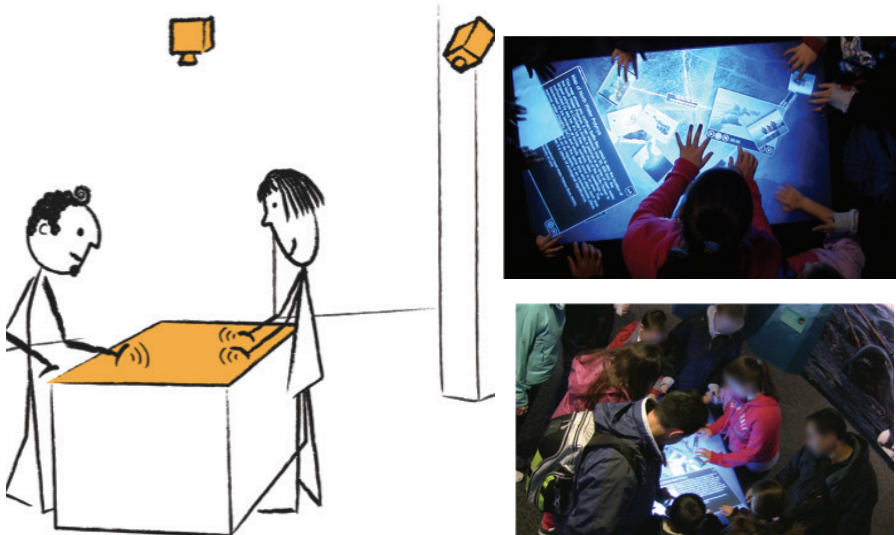


Figure GW5. Camera positions and the views seen from them.

We conducted in-depth video analysis, coding 926 gesture instances. When coding the gestures, we included such factors as the number of hands touching, the numbers and types of fingers touching, the hand postures and the hand motions. Figure GW6 shows a sample of coded gestures.

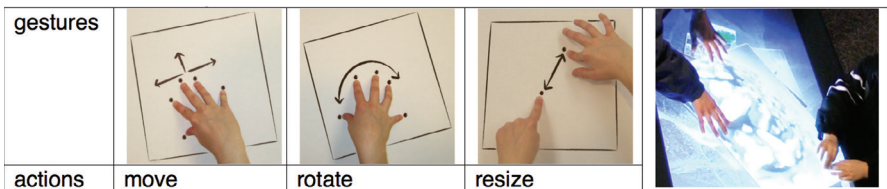


Figure GW6. A few sample gestures and their associated actions.

Figures GW7 and GW8, respectively show a variety of gestures with (Figure 7) drag/move intentions and Figure 8, resize actions.

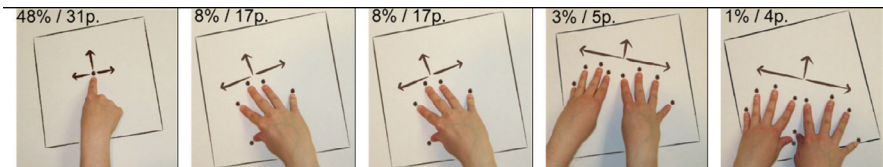


Figure GW7. Gestures for drag/move actions.

Note, in Figure GW8, the great amount of gesture variety for a single action intent: resize. Note that people also sometimes use one and sometimes two hands.

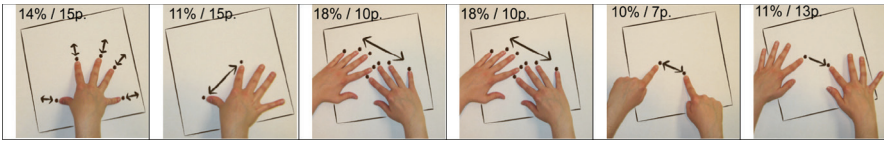


Figure GW8. Resize actions.

### Gestures in Context

People combined gestures into fluid gesture sequences. Hand postures and touch points remained relatively stable with only hand motions being adjusted. In Figure GW10 you can see that after using an expanding pinch gesture with thumb and middle finger, that same touch position and hand posture is maintained but now the gesture is used for dragging. In other circumstances this posture is quite uncommon for drag/move actions.

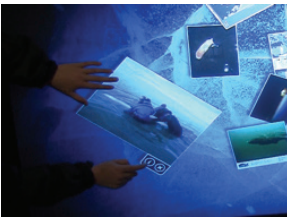


Figure GW9. Gestures had some similarity to gestures with physical objects such as asymmetrical use of dominant non-dominant hands.



Figure GW10. The same hand posture is used for both expand and drag/move.

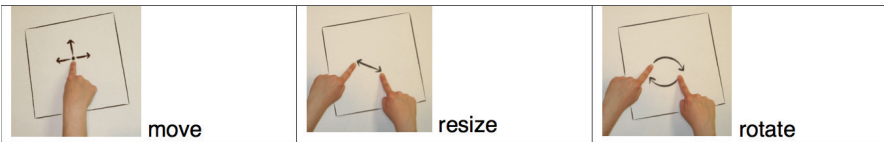


Figure GW11. This is an illustrative sequence of actions, showing how a fluid interaction gesture sequence works. This fluid gesture sequence starts with a one finger, one touch move action. Next, keeping the first hand posture and touch the same, the person adds a second one finger touch to provide the two touches needed for first resize and then rotate.

### Children Visitors



Figure GW12. Many of the visitors were children.

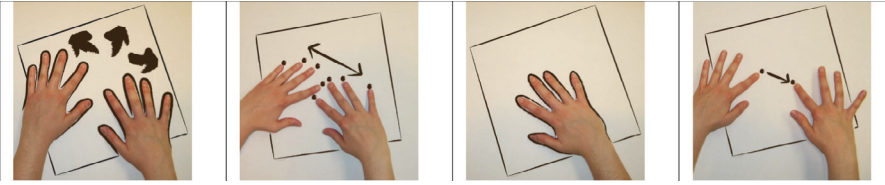


Figure GW13. Children visitors used more content-independent actions such as sweeping display clear and flicking media items across the display. Their gestures tended to be more coarse-grained and imprecise.



Figure GW14. Some of children’s interactions were competitive such as large-scale gestures to maintain control of the display space.

## Adult Visitors



Figure GW15. Adult visitors tended to use more content-oriented exploration. This included more single finger and single handed gestures and more use of rotate and tap actions to explore the content.

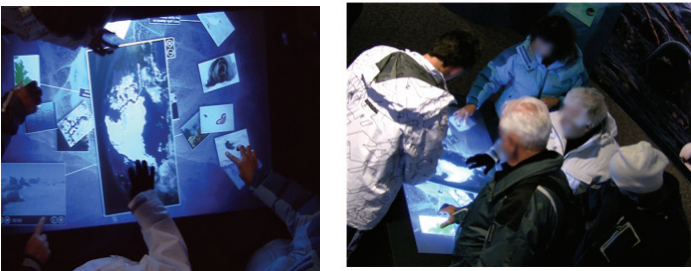
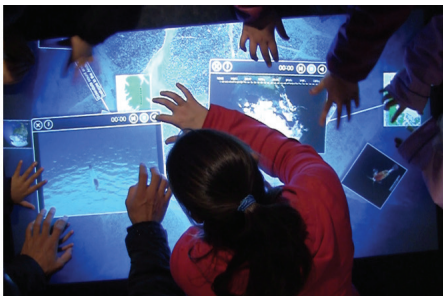


Figure GW16. Adults tended to use small scale gestures on crowded tables, showing respect for other visitors’ interaction space.

## Social Context

This discussion is about the social context of the gestures at the tabletop displays, not the full social context of the event.



*Mentoring.* People demonstrated gestures to each other. Most often this was between family members, but sometimes strangers also showed each other gestures.



*Imitating Observed Gestures.* Here the person in orange gathers a group of media items and then the person in purple tries the same gesture.



*Gestures for Personal Expression.* The girl in red has a strong dislike for bugs. When she see an image of a bug says "No Bugs" and pushes the image of the bug away with a very expressive gesture.

## Gestures in the Wild Summary

*Huge Gesture Variety.* People used a large variety of multi-touch gestures. This included using a large variety of gesture for a single type of intended action.

*Temporal Context.* The interaction context, that is what a person had just

done or said or the position of their hands, greatly influenced the gesture they would take next.

*Social Context.* A person's choice of gestures is influenced by the social context, by who they are with and by how the motion of their hands will harmonize with what they are saying. People did not want to make awkward gestures. They would attempt to get the action response they wanted with a fluid transition from their previous position. This speaks to a need for the design of fluid gesture sequences.

### **The Bohemian Bookshelf: Supporting Serendipitous Book Discoveries through Information Visualization**

Since the possibility of serendipity when searching for books on a traditional library shelf is so important, we explored the possibility of including support for serendipity in a digital bookshelf. The main message of this research is that serendipitous discoveries can be facilitated through information visualization, by leveraging the factors that encourage serendipity beyond just coincidence and luck.



Figure BB1. Switching to digital libraries involves many changes but does not have to include the loss of serendipity. Traditional libraries have dense racks of bookshelves (left). Digital libraries, where the search for books has become digital, focus more on meeting and working spaces (right).

For this research, our goal was to better understand how to support open ended searches and explorations and, if possible, to support serendipitous discoveries. We start by talking a closer look at the concept of serendipity. The OED (Oxford, 2011) defines serendipity as “the faculty of making happy and unexpected discoveries by accident”. This definition would lead one to think that serendipity happens by coincidence, luck, chance or other such factors. In fact, it could be suggested that the introduction of mere randomness might promote serendipity. However, a deeper look into the literature including the original fairy tale, the “Three Princes of Serendip” and literary discussions around this concept point out that serendipity can be supported in a variety of ways (Andre et al., 2009; Erdelez, 1999; Foster and Ford 2003; Liestman 1992; Remer 1965; Toms 2000). We group these methods into those that pertain to the personality traits of the information seeker and those that pertain to factors from other people and systems they have developed.

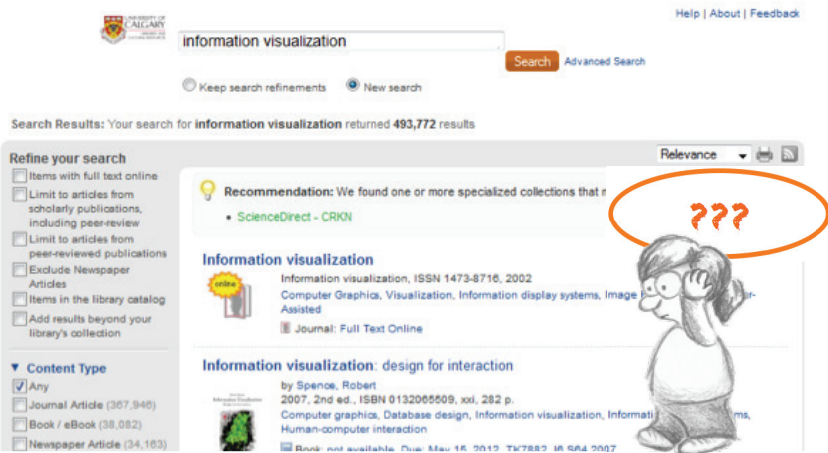


Figure BB2. Digital search can be great if you know what you are looking for. However, it is much more difficult if you do not know what you are looking for. For example, searching for something open ended such as 'I want to find something I will find exciting' or 'I want to find something my mother-in-law will like' is not really supported in current digital search. It is hard to use digital search for open ended explorations.

The personality traits of an information seeker that promote serendipity are:

- knowledge – having some prior knowledge can make it more likely that a person will find what they want;
- open-mindedness – being willing to think open mindedly can help a person discover the usefulness of surprising factors; and
- perseverance – being willing to keep on looking.

While we individually can culture these practices and probably promote serendipity in our own lives, they are not the type of factors that can be programmatically include in a computer system. To consider how these factors work in one's personal life, consider the well known story of Sir Alexander Fleming (Rosenman, 1988). It is said that he discovered penicillin serendipitously. He kept a very messy laboratory and one day he discovered that one of his petri dishes was contaminated by a strange mould. He was knowledgeable enough to recognize a possibly interesting occurrence, and open-minded enough to run further experiments with this mould. By persevering with this direction, he discovered penicillin, which revolutionized medicine, and he was awarded with a Nobel Prize (Rosenman, 1988).

The factors related to other people and their systems are more hopeful factors to consider for algorithmic inclusion. Think about a bookstore. It provides a rich environment for browsing books. It will contain multiple bookshelves and tables upon which books have been organized in various ways, such as alphabetically, by theme, by age appropriateness, etc. Libraries and bookstores provide environments where serendipity is more likely because librarians and bookstore owners have put considerable

thought into how to organize the books. An an example what this could look like (Book store with multiple shelves and tables. Image © A. Thudt.):



From this second group of factors that relate to other people and their systems, we have gleaned five approaches that we can introduce algorithmically: multiple access points, highlighting adjacencies, enticing curiosity, flexible pathways, and playful exploration. These five approaches for supporting serendipity are diagrammatically illustrated in Figure BB3.

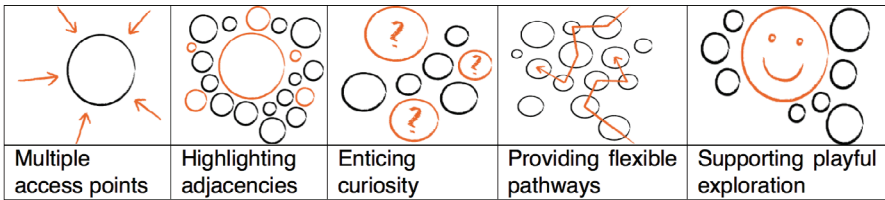


Figure BB3. Five methods that systems can include to support serendipity.

Offering multiple access points provides people with different ways of entering into the search. These different access points offer different points of view or different perspectives as the information can be approached from different angles.

Highlighting adjacencies provides reinforcement that may help people make connections. For instance, one type of adjacency is topic. People often find interesting books unexpectedly while browsing through library shelves where books have been organized by topic. Books in close proximity on the shelf often capture attention. However, while bookshelves because of physical limitations can only offer one type of adjacency, digital visualizations do not have to rely on physical proximity to provide indications of adjacencies, and thus can offer many different types of adjacency.

Serendipitous discoveries have also been attributed to curiosity, which is a factor in open-mindedness. Visualizations, visual metaphors and visual aesthetics can be used to entice viewers and to promote curiosity thus initiating further exploration. Providing flexible pathways lets people explore at will, encouraging their own curiosity. Offering playful interactions can further extend investigation by making interactions more fun.

Using these five factors makes it possible for a visualization to be designed to provide a series of unusual perspectives on a data collection. Together

they can offer support of open-ended, exploratory search strategies that go beyond querying, and can support navigation of information collections in open-ended ways. Non pre-determined pathways can constantly offer new crossroads through the collection and multiple interactive overviews can act as visual guides, offering adjacent information as visual signposts. In this way play can be a facilitator for creative search and may also facilitate serendipity. Making information playful and pleasurable may also encourage people to persevere leading to longer explorations.



Figure BB4: Bohemian Bookshelf installation. Image © A. Thudt.

### The Bohemian Bookshelf

To provide an example of how these guidelines can be used, we developed the Bohemian Bookshelf prototype. It works with a small dataset of books and can be used on a touch interactive display in a library space as well as on a website. Figure BB4 shows the Bohemian Bookshelf installed at a local library. Figure BB5 provides a view of the combined visualizations that make the Bohemian Bookshelf.



Figure BB5. Bohemian Bookshelf showing the five visualizations and one selection.



The Bohemian Bookshelf interface consists of five visualizations that each focus on different book attributes (Figure BB5). Some represent aspects that are common in digital search interfaces, such as keywords and author names. Others focus on visual and tangible characteristics that can be experienced in physical libraries, such as colour and pattern of the cover, or the thickness of the book.



The *Cover Colour Circle* shows the distribution of cover colours in the book collection. As you can see in the adjacent image, in this dataset there are more orange books than green or blue books. It is possible to browse through the Cover Colour Circle by moving your finger or mouse across the visualization. The covers that bubble up can easily be selected. The colourwise adjacent books are shown adjacently to the selected book.



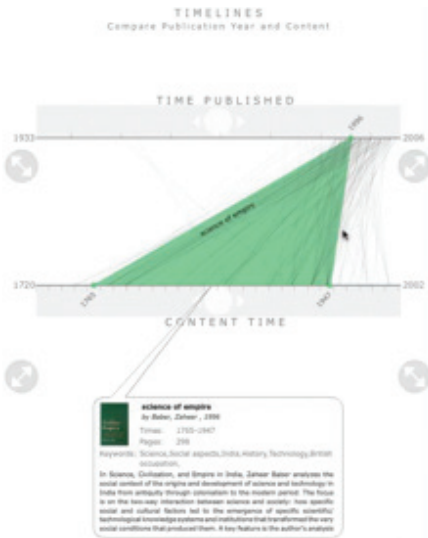
The *Keyword Chains* visualization shows keyword connections between different books. The selected book is shown in the center and all its keywords branch out from it and form connections to other books that share a keyword with it. The keyword chains can be stretched out to facilitate reading. When an adjacent book is selected, it moves to the centre and new keyword chains form around it. In this way, people can navigate through the collection by following keyword connections between the books.



The *Author Spiral* orders the books alphabetically by author name. We used a parchment metaphor to visually represent this author list. People can fluidly browse through this list by scrolling up and down in the list. Due to the alphabetical ordering, books from the same author appear next to each other.



The *Book Pile* visualizes the page count of the books in the library. Thicker books are represented by larger squares and are located on the top of the pile, while the books that have fewer pages trickle down to the bottom of the pile. It is possible to fluidly browse through the pile. For the selected book, the actual number of pages is shown and books with a similar page count are highlighted by showing their covers.



The *Timelines* focus on temporal aspects of the books. The top timeline shows the publication year, while the bottom timeline shows the time period the book is about. Books are represented by the thin lines that connect both timelines. The book that is selected here for example is published in 2001 and is about a time period in the 17th century. People can browse through the visualization by moving their finger across these connection lines. Furthermore, it is possible to zoom into the timelines to take a closer look at a certain time period of interest.

### Bohemian Bookshelf

The interface shows all five visualizations at the same time. The visualization in the center is shown slightly larger. People can switch between the visualizations by using the arrow buttons. All 5 visualizations are interactive no matter what location they are in and they are all interlinked with each other. When a book is selected in one visualization, the others change accordingly to bring this book into focus across all of the visualizations.



We installed the Bohemian Bookshelf in a local library, as shown above, and studied how library visitors experienced this way of browsing book collections. We interviewed 11 visitors who had used the Bohemian

Bookshelf for the exploration of its book collection. We asked about their thoughts on the differences between this and other search interfaces they were familiar with, about the role of visualizations, visual aesthetics, and the large display technology for browsing book collections.

In general people were very positive about the Bohemian Bookshelf and as they talked to us it was apparent that they were able to understand the representations: "I'd say like 90% of the understanding is the visual component. I read the labels, but after looking at the visuals." [V8]

People liked having the multiple visual access points together in one interface: "It gives you more options. So if you have more information, it is easier to have a starting point." [V8]. "I'm sure each element works differently for different people. I like having it all together. It kind of promotes curiosity." [V11]

The highlighted adjacencies helped people to see relations between the books and thus follow up on certain areas of interest: "The Bohemian Bookshelf is a cool tool to discover something new through different associations." [V5]

People commented many times about how the interface is pleasant to look at and they especially noted that the colours and the cover images evoked curiosity. "First of all the Bohemian Bookshelf catches interest. I don't know what the Cover Colour Circle is for exactly, but it makes it more interesting and then if you stumble upon something, you might want to read it. And that's a good way to get people to actually want to read." [V5]

From our observations, we could see people switch back and forth between visualizations following up on highlighted books and using the visual overviews to steer their explorations. "The current way of searching for a book is, you have to know what it is or just browse through an alphabetical author list. But with the Bohemian Bookshelf you can kind of branch off by keyword and find similar books on the same topic." [V4] During these explorations the links between the visualizations were used as crossroads that drove the explorations in new directions.

People really liked the playful approach of the visualizations and commented several times on how they are fun to use. "You have the touch screen with all the different covers that open up and you can just pick them. That's sort of like browsing. It's more satisfying than sitting on the computer clicking through a whole bunch of stuff." [V9]

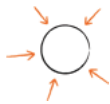
We were also very interested in whether or not people actually made serendipitous discoveries when using the Bohemian Bookshelf interface. Although we had not informed people about the purpose of the interface, a lot of people mentioned it could help them to make unexpected book discoveries: "I think that the Bohemian Bookshelf would be a good way of

finding new books. You get to see more different books that you might find interesting later, which you otherwise would never see, because you wouldn't be looking for them." [V4]

People also reported concrete book discoveries they made with the interface. More than half of the participants were able to name books that they discovered while browsing, and that they would like to check books these out from the library. They also explained how they found these books: "I had no expectations and I just saw an author name that seemed familiar to my language, and then I thought, well, why not check it out." [V11] and "I picked my favourite colour. I picked pink and then I found a book that I liked." [V7]

### The Bohemian Bookshelf Summary

The main message is that serendipitous discoveries are not just triggered by luck, chance or coincidence. We can actually facilitate serendipity through information visualization by leveraging aspects that can encourage serendipity such as:



Providing multiple access points



Highlighting adjacencies



Enticing curiosity



Providing flexible pathways



Supporting playful interactions

Our example of Bohemian Bookshelf is just one example of an interface that shows how these guidelines can be realised. Visualizations can be designed to foster serendipitous discoveries.

### Designing the Unexpected:

#### Endlessly Fascinating Interaction for Interactive Installations

One of our goals for creating installations in public spaces is to make them endlessly fascinating. This goal sounds difficult, if not impossible, to achieve. In the following we describe one approach to creating an installation that offers endlessly fascinating interaction, or EFI.

We postulate that these conditions must be present in an interactive installation in order for it to be endlessly fascinating: first, it must be interesting at any given time; second, the content should not be repetitive; third, it should present the viewer with multiple possible storylines.

First, here are some of the inspirations for our interactive art installation, or, some things that we find fascinating. These installations in particular inspired us because they can all be said to have some degree of EFI.

Krueger's Videoplace set up an actively dynamic and playful experience that attracted viewers and encouraged them to explore how the work would respond to their actions (Krueger, 1977).

Hill's Tall Ships engages viewers' interest by having them exchange intense gaze with projected ghosts on the walls of a corridor (Hill, 1992). Ghosts approach when viewer is on sensor in front of wall, stay while the viewer is still there, and leave when viewer steps off sensor. Gaze functions as a means of getting viewers to relate to the ghosts' sense of longing with experiences in their own lives, creating the illusion of an emotional bonding experience.

Gonsalves' installation Chameleon sustains viewers' interest by reading their facial expressions and having the projected faces reflect them back, like an emotional contagion (Gonsalves, 2009).

Another thing we find fascinating is liminal spaces. Broadly speaking, a liminal space is an in-between space, neither here nor there, such as a doorway, or a hallway (Thomassen, 2009). Liminal spaces can force us to adopt behaviour that we see as being socially acceptable in order to avoid being uncomfortable. For example, in an elevator, we resort to all kinds of things to cope with being in a confined space with a stranger.



Examples of liminal spaces.

From this we decided to explore the possibilities for creating EFI in the liminal space of an elevator. Our interactive installation, A Delicate Agreement (ADA), is a false elevator with peepholes in the doors.

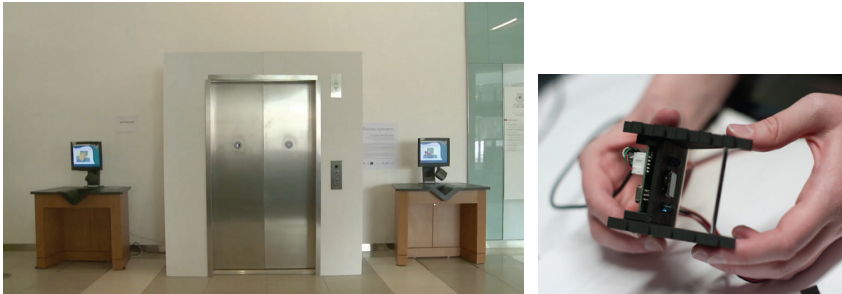


Figure ADA: A Delicate Agreement (left); gaze tracker (right).

Since we did not want the viewer to have to discover difficult actions to trigger a response from our piece, we made use of incidental (Dix, 2002) and passive interaction (Nakatsu, Rauterberg, & Vorderer, 2005). Viewers can affect the unfolding story in this installation with their gaze simply by looking through the peepholes in the doors. Behind each door, we custom made a low-fi gaze tracker from a modified webcam and a hot mirror. The details about this tracker can be found in MacDonald et al., 2015 (MacDonald, Brosz, Nacenta, & Carpendale, 2015).

A Delicate Agreement (Figure ADA) is a site-specific installation that explores concept of EFI. It offers viewers a rich interactive narrative of encounters between people and viewers. Externally it is a false elevator with peepholes.

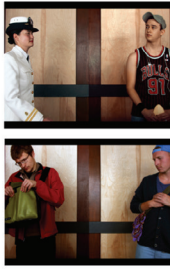
1. Challenging setting: A Liminal Space.
2. Create storylines with a Complex Adaptive System.
3. Design interactions with characters based on art and social theory.

Upon looking into the peepholes, the viewer will see a stop motion animation composed of up to two characters riding in the elevator, performing behaviours, and getting on and off at appropriate floors. Here are three possible scenes:



There are sixteen characters and they each have multiple behavior image sequences available to them, totaling more than 40000 still images. Since each character stays on one side of the elevator for one trip, but may use the other side for a different elevator ride, the combinatorics of all the possible combinations is quite large.

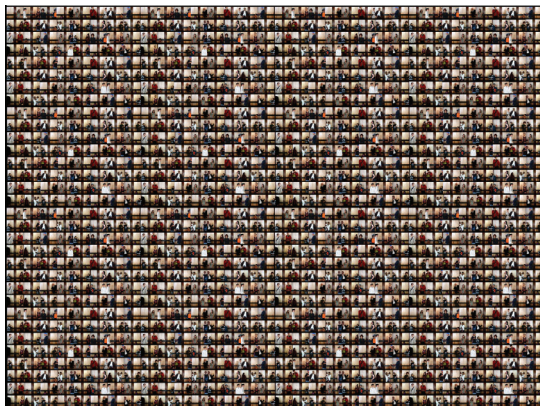
Here are 4 scenes (left) and 16 scenes (right):



Here are 64 scenes:



And here are many more:

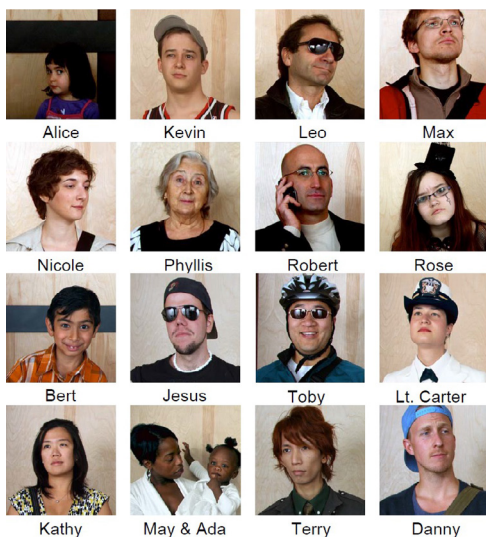


## Complex Adaptive System (CAS)

We decided to build the sequencing of scenes, the interaction between the characters who inhabit the elevator using a complex adaptive system (CAS) (Miller and Page, 2009; Waldrop, 1992; Zeeman, 1976). A CAS is not totally constrained—actions and interactions are developed based on local awareness.

A CAS is a system that is neither fully constrained nor chaotic. That is, there is considerable freedom, but yet there are some rules to be followed. Individual characters are only locally aware, having no overview of whole system. An example of this is an economy with individual people as agents (Waldrop, 1992), or an immune system. Another example is Conway's Game of Life (Gardner, 1970), (Marek Fiser, 2013 <http://www.marekfiser.com/Projects/Conways-Game-of-Life-on-GPU-using-CUDA>).

Conway's Game of Life is a prevalent example of a CAS. It uses three simple rules about living, reproducing and dying, and can produce extremely complex behaviours. Our agents are our characters, and we have 16 of them, each with their own set of rules, or personalities. Our agents are our characters. They each have flexible, responsive storylines, which results in emergent behaviours. CAS allows us to have a flexible story.



Our simple rule is as follows. To find out what a character, currently in the elevator will do next, we take their current behaviour, and add to that:

- their impression factor, which is a combination of the current behavior of the other character in the elevator (if there is one) and the viewer(s) gaze,
- their expression factor, which is the emotional direction they will head given their impression, and
- the character's predilection, which is based on each character's emotional map.

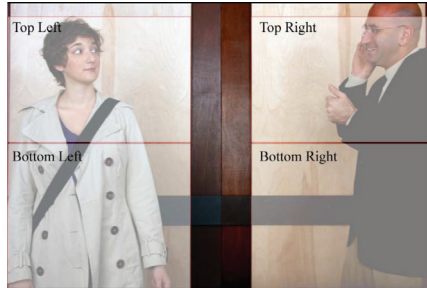




Third, upon introduction of the extra eight characters and the absence of fear caused by the gangster character, a new behaviour emerged that we were not ok with - characters were starting to flirt with the little girl character. Unfortunately, we had to fix this by making the little girl less interesting.

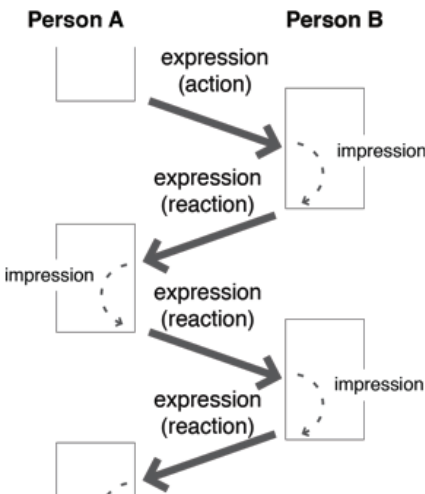
Taking a brief look inside our CAS:

*Incidental interaction triggered via gaze.* Depending on what area the viewer gazes at affects the behaviour of the characters in the elevator, depending on their personality as well. There are 4 possible areas that can trigger a reaction - the 5th is not looking at all:



### Using Narrative and Social Theory

To help us address the interaction challenge of creating a story that unfolds and is non-repetitive and endlessly fascinating, we made use of interactive narrative (Bang, 1993) and ideas about interaction expression and impression from Goffman (Goffman, 1959). We implemented this through our CAS.

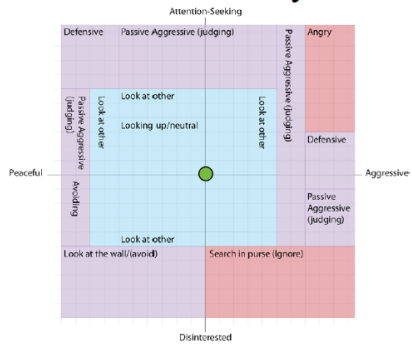


To create a character engine for our CAS, we created a character map for each character. The personalities look like this.

For more complex characters, there are more behaviours, and more complex patterns of squares on the maps. Simpler characters have simpler maps. The axes of aggression and attention are based on an idea from Nass (Nass et al., 1995) about personality being defined by 2 meaningful dimensions, extraversion and agreeableness. There are 26 possible behaviours and each character has a subset of these (MacDonald et al., 2015).



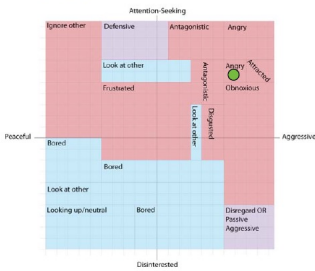
Each one of these regions corresponds to a sequence of images representing a behaviour. At any given time, she can only be in one spot on this map. There are three ways to change her behaviour state, which is represented here by the green dot in the middle of this map. Here is Phyllis in the elevator, and her personality or behaviour map.



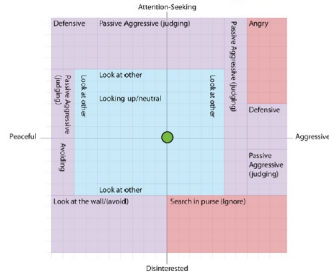
The next thing that happens is the entrance of another character. At first she is neutral, and he is angry:



Leo



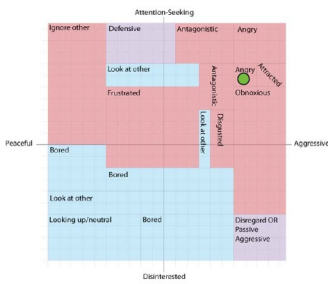
Phyllis



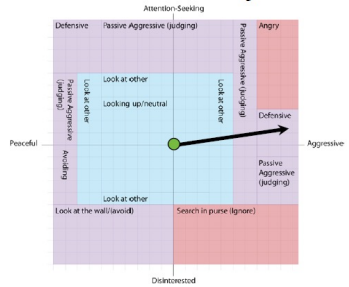
Leo's actions here will trigger a reaction in Phyllis:



Leo



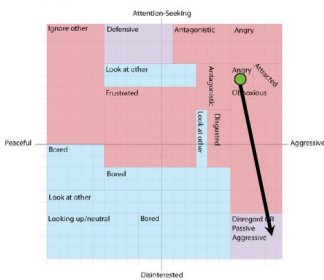
Phyllis



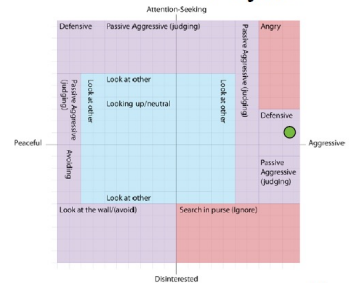
In turn, Phyllis' reaction will determine Leo's subsequent behaviour.



Leo



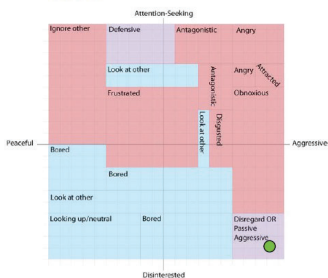
Phyllis



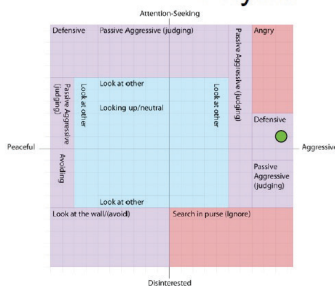
So here they are now in their new states:



Leo

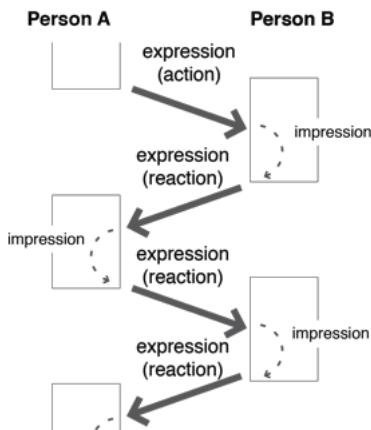
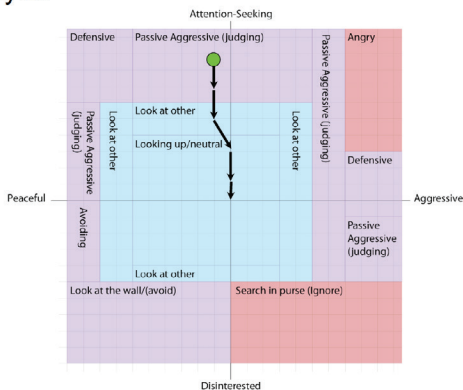


Phyllis



The third way a character's behaviour state can be changed is the progression of time. As time moves forward, if nothing else happens, Phyllis will gravitate back to her default behaviour at the center of her map.

Phyllis



Behaviors are not limited to the selection of visual output. Behaviors also affect the emotional state of other characters in different ways. For example, the aggressive anger behavior of Leo causes other characters to become more aggressive. This is the basic mechanism of interaction between characters: Leo's anger behavior is his expression and triggers the other character's impression. Therefore, the personality grids are a representation of the visual output of the character.

The viewer affects this by staring at one of the characters. This can affect behaviour, and can also trigger a special acknowledgment behaviour. This acknowledgment behavior changes according to character.



We chose the liminal space of the elevator as a challenging setting and to produce endlessly fascinating interaction, we created multiple story lines with a complex adaptive system. We design interactions between characters based on art and social theory.

### Summary for Endlessly Fascinating Interaction

1. The observed reactions of the piece are both understandable and intriguing;
2. The viewer is not required to discover difficult or obscure actions to trigger a response;
3. The viewer does not need to be aware of the effect of their own interaction; and
4. The story that unfolds is non-repetitive, and endlessly fascinating.

### A Delicate Agreement



### Conclusions

In this chapter we have presented the results from an 'in the wild' study on natural use of a multi-touch table in the Vancouver Aquarium Artic Exhibit. We outlined our exploration into the possibility of programming for serendipity in a visualization based search interface and we described our use of a complex adaptive system to create a continually changing story line with emergent behaviours that we hope is a step towards developing endlessly fascinating interaction.



## Surface Applications for Security Analysis

*Judith Brown, Jeff Wilson, Peter Simonyi, Miran Mirza, and Robert Biddle*

### Introduction

SurfNet Theme 2 concerned software development for surface applications. There were two perspectives, one being the utilization of surface technology to support the development process, and the other being development processes that arise in surface applications. Much of the work of our HotSoft group at Carleton has concerned the latter. Other research of ours concentrates on human factors in computer security, so we decided to explore how surface technology might support security analysis. This specific domain allowed us to investigate how study surface application design and development in an established context, and thus learn how the real needs of the domain might best be supported. We were fortunate to also have partners in industry and government working in the domain, and therefore able to give us advice and feedback. A number of projects were conducted over the span of SurfNet, each one offering findings that informed later projects. In this chapter, we provide an outline our work, summarizing the each of the main projects, and their findings. Each project is documented more extensively in other publications elsewhere, and we provide references to those papers throughout this overview. We conclude with a summary of our main findings and how they inform the development of surface applications in general. The main sections of this paper are therefore as below. We conclude the paper with some discussion about the general themes that emerged from our work.

- Review of Surface Computing for Collaborative Analysis
- Field Studies of Security Analysts at Work
- ACH Walkthrough: Software to Support Security Analysis
- Ra: Support for Web Application Interaction History
- Strata: Annotation for Web Applications

### Review of Surface Computing for Collaborative Analysis

Our first step in this sequence of projects was to conducted an extensive survey of the area. We covered a wide range of topics, covering not only the



literature specifically on the topic, but also on relevant theory and interaction design, as well as the underlying technologies and development platforms. The survey was published as 140 page book by Morgan and Claypool, J. M. Brown et al. (2013). Within the broader context of collaborative analysis work we particularly discussed co-located analysis work in the security domain which is typically either network security or intelligence work.

Surface computing is likely to become commonplace in some domains such as entertainment and education. However, we also expect large surfaces will serve a primary role in supporting collaborative work. Meeting rooms and team environments will be designed to feature large surfaces. These large surfaces, while being a key to enabling more collaborative computing environments, will typically work in concert with other display devices in mixed-display environments, where both individual and team devices are used together to support collaborative work. We believe large displays and mixed display environments (combinations of large displays, tablets, smart phones and other types of surfaces) will become ubiquitous in office environments of the future. In this emerging and novel context, application software, and especially application interfaces, must be explicitly and purposefully designed and developed to support surface computing for collaborative work. This book described current research in this space, and provided a perspective on it based on our own experiences and expertise. We first reviewed the underlying technology for surface interaction, especially looking at large surfaces and novel methods for interaction. We identified research on surface technology issues that are particularly important to analysis work. Document flow issues may impact work Individual work on digital artifacts using laptop and workstation computers in theory should be compatible with digital tabletops and digital wall displays, since digital artifacts don't have to be transformed into analog (paper) artifacts to be taken to a meeting.

In practice, however, the problem of moving digital artifacts seamlessly between surfaces has not yet been resolved. Also important is that human communication is rife with indexical references, i.e., pointing, which involves verbally or physically indicating something. In artifact-rich environments pointing behaviour is common and saves much time. Other issues arise with multiple display environments including both small and very large displays, with diverse kinds of displays being used together. Interaction design for surface computing presents novel challenges that are not easily solved by mechanisms used for traditional desktop interaction design. Menus and scrollbars may become things of the past, and new approaches to pointing, selecting, and hovering are required. Gesturing is the emerging approach, and is still evolving. Easy text entry and interactor identity remain challenges.

Research has clearly shown there are many advantages for large displays for individuals. These include cognitive benefits, increased productivity, reduced errors, and greater satisfaction. We believe these benefits to individuals often carry over into collaborative situations. Research

on groups and teams, however, is much newer. Early results are by and large very positive, but also indicate that it is very important that surface applications be carefully designed. For example, to increase situation awareness in contexts where groups are collaborating loosely, the research shows that it is very important to reduce the amount of information that is shared to no more than what is required. Other research indicates the positioning and arrangement of displays can impact collaboration. In mixed-display environments the research shows that it would be important to be clear about the most important objectives of the collaboration so that choices about display devices and functionality can be made with these considerations in mind.

Understanding analysis work is not easy. Designers and developers of tools often have undeclared assumptions about what analysis work is, and these assumptions can easily become embedded in the tools, resulting in a rupture between the work at hand and the tools to accomplish the work. Theories have been applied to aid understanding of individual analysis work, primarily based on understanding cognition. However, increasing amounts of data and larger and more complex analyses emphasize the need for collaborative analysis. Collaborative artifact-mediated work can be understood from a variety of theoretical perspectives. In particular, we reviewed Distributed Cognition, Evolutionary Psychology, Attention mechanisms, Group situation awareness, and Cultural-Historical Activity Theory. However, collaborative work, such as complex collaborative work in specialized domains, can be challenging to understand and predict, particularly where new technology presents unfamiliar opportunities.

With respect to software architecture and development there is and will continue to be some turbulence as technology standards and design best practices emerge and become established. It is very important for designers to understand this, as the challenges for developers are much greater than those long understood relating to WIMP (windows, icons, menus and pointer) interfaces. Diversity of toolkits and libraries may make cross-platform development problematic until the advantages of interoperability influence the market. Similarly, heterogeneity of data sources and formats may present challenges. One lingering issue is that few multi-touch surfaces have the means to identify the source of gestures when several collaborators are interacting with a single screen. However, the novel ubiquity and low cost of tablets and smartphones offers an excellent opportunity to provide the identity of collaborators in mixed-surface environments, as well as additional modes of interaction beyond touch. Further, tablets and smartphones as additional devices for collaboration can offer opportunities for private exploration, offline data manipulation and preparation, and interactions requiring personalization or authority. Some technologies have already dealt with multiplicity and diversity at the infrastructural level, particularly web-based frameworks, and seem to be a good starting point for collaborating across multiple devices. However, there remain differences in how gestures are shared with browsers within

each of the main handheld operating systems, and so it may be worth designing for a mix of browser-based and native code. Moreover, there are also deeper issues. The challenge of sharing application state across multiple devices gives rise to an important question of the identity and “ownership” of objects. It is important to draw appropriate distinctions between actual objects and inferred or proxy objects. Mutability (the ability of objects to be changed) of shared objects must follow logic that meets mutually shared goals of participants.

Our survey left us optimistic that large surfaces and mixed-display environments seem well poised to support co-located collaborative analysis work. However, it was clear that design for surface applications in the analysis domain requires a system perspective. Surface computing is only as useful as its application software, and applications for collaborative analysis work need careful study of the domain, and carefully designed interfaces and software. Further, surface computing environments need appropriate accommodation and infrastructure, which also needs to be designed. In this context it is important to design with an eye to end-user interaction, end-user experiences, and the broader environment, which would include team interactions and the physical aspects of the workplace.

### **Field Studies of Security Analysts at Work**

The next step in our research program was to conduct field studies. Especially in the domain of security analysis, access to professionals can be very difficult to obtain, and our partnerships with industry and government organizations were critical.

We conducted a number of studies in two related domains. In a first set of studies, we carried out observations and interviews of operations centres. In this set of studies, there were 7 sites involved, in a variety of industry and government contexts from financial transaction processing to healthcare support, each involving many hours and days of observations, and interviews across a range of workers and stakeholders. Our analysis of the data used Grounded Theory, and the results showed new patterns of work that have evolved similarly across the workplaces we studied, offering new insight about how these workplaces might be better supported with technology.

In a second set of studies, we focused on analyst teams conducting in-depth projects to explore specific issues of interest. Our main study in this work involved a team of 10 professional analysts over a 4 day project. The project itself was a proxy based on open data, rather than on real and therefore sensitive data, but was designed by one of the senior analysts as representative of their real work. Our analysis of the data used Culture-Historical Activity Theory, especially the work of Engeström on collaborative work Engeström (2000; Engeström, 1992).

We will elaborate here more on the second set of studies, because of the impact they had on later stages of our research. Our study found that in the

early stages of the analysis process, the analysts collaborated closely. Later on, despite them working on the same general topic, and using the same data set, we principally saw work done side-by-side, but independently. We illustrate this in Figure 1, using phases of collaboration based on Engström’s work.

The most common stronger form of collaboration we saw later was when one analyst requested technical help from another. Interestingly an exception to the pattern occurred when one analyst produced a large poster showing results of her work, whereupon others were keen to comment and find connections to their own work. Around the same stage of the process, we observed one analyst applying a structured analysis technique called “Analyses of Competing Hypotheses” (ACH). This is a technique developed by Heurer, and supported by software. The main idea is that an analyst considers several alternative hypotheses that might explain a set of evidence. They assess the data for credibility, relevance, and then consistency with each hypothesis. These factors are then used to build a mathematical model which facilitates identification of anomalies and reflection.

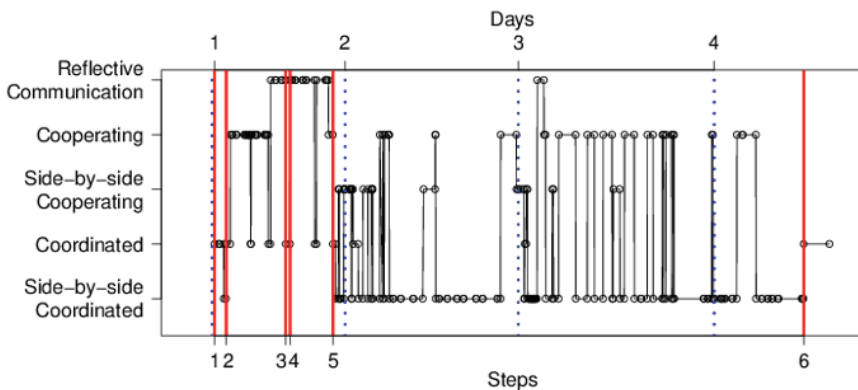


Figure 1. The process of collaboration across time. Solid lines show steps in the analysts’ process. Dashed vertical lines show days. Note the absence of reflective communication in step 5, selecting & analyzing issues. Source: Brown et al. Brown, Wilson, and Biddle (2014).

Our analysis of the data, both observations and interviews, led to a number of interesting findings. One relates to “Process Productivity”. As noted above, analysts collaborated more in early stages, and much less so later on. We observed that the early stages were done with whiteboards, posters, and brainstorming, where collaboration was explicitly supported. In the later stages there was much less support, and we felt that better support would facilitate and encourage more collaboration. Another finding related to “Process Outcomes”. This was related to the first finding, but we also realized that with only low levels of collaboration, the process involved little cross-checking, discussion of coverage, and comparison of results. We speculated that better support for collaboration would not only improve the

productivity of the team, but also the quality of their outcomes. Finally, we identified possibilities for better “Learning within the Process”. In activity theory it is well-established that important learning occurs in cycles of externalization and internalization as team members interact. In the activity we observed, more support could have been put in place to increase the likelihood of individual and team benefits, two secondary outcomes of strong collaborative practice. In the collaborative event we observed, very few team benefits ensued except when team members shared and reflected on their techniques during their presentations at the end of the project. There were also minimal individual benefits (although a few analysts learned new tools on their own, individuals seldom explicitly learned from each other).

Throughout the study, we had identified use of all kinds of surface-like artifacts, including whiteboards, posters, and notebooks, and well as certain software applications. Our conclusion was that there were important opportunities for surface computing to improve the collaborative analysis process. In particular, we felt that application software for large touch-surfaces might well support analysis techniques used later in the process, such as ACH. This kind of support might thus improve Process Productivity, Process Outcomes, and Learning within the Process.

### **ACH Walkthrough: Software to Support Security Analysis**

In this section we report on the design and implementation of a surface application to support co-located security analysis. The field study of a team of security analysts suggested that the “Analysis of Competing Hypotheses” process (ACH) would benefit from collaborative support because the consideration and judgement would both be assisted by team discussion. We found calls for increased collaboration by authorities in the intelligence analysis world. Heuer and Pherson suggest that their collection of structured methods Heuer Jr. and Pherson (2010) can support collaboration and reduce cognitive bias. Hackman Hackman (2011) concurs and emphasizes that collaboration both improves outcomes and contributes to the development of individual and team skills.

We reviewed other versions of ACH software, namely the version developed for individual analysts at PARC Palo Alto Research Center (2010), and two versions designed for collaboration, namely Globalytica Think Suite’s Team ACH Globalytica (n.d.), and Open Source ACH Burton (n.d.). We created extensive requirements for a collaborative version of ACH using surface technologies. The main requirements were that:

1. A collaborative version of ACH should enable part of a larger process where analysts alternate between individual work on an ACH and collaborative work on an ACH;
2. Analysts should be able to easily view evidence documents while working on an ACH analysis, and we speculated that a mixed-display environment would support this best;

3. Collaborative ACH work should be enabled by a walkthrough process whereby members of the team take on roles that would strengthen the analysis, while they walked through all aspects of the analysis and checked or extended its content.

We focused on requirement 3. We saw the walkthrough support as an important part of the tool, given evidence that a fair number of users were new to ACH. Our requirements also introduced both a new collaborative practice as well as a surface application. Together these would aim to improve an ACH analysis by enabling face-to-face discussions about the attributes of the analysis, e.g., its completeness, its correctness, and so on. Our application software, "ACH Walkthrough" accomplishes all these goals as a functional prototype. Figure 2 shows the software in use. The data set we use to illustrate the software is from publicly available material to investigate the collapse of ENRON Corporation Contributors (2011).



Figure 2. ACH Walkthrough in use: Running with synchronized data on a large multi-touch screen, on a laptop computer, and on a tablet.

While ACH Walkthrough can be used for ACH analysis generally, we especially intend for it to be used for a collaborative review, where a small team of analysts work together. In particular, we suggest an approach similar to that suggested by Wharton et al. called the "Cognitive Walkthrough" (Wharton et al. (1992)), where a team walks through steps, discussing and executing each step together, each team member contributing from their perspective. Recall that in our field study, we saw a need for reflective communication. We suggest that our walkthrough technique will provide strong support

for reflective communication. In particular, when reflecting, analysts should discuss the overall direction of the work, the quality of the work, and the methods they are using to achieve their common goal.

As well as a collaborative review, we suggest that ACH analysis involves some work best done by analysts working independently. For example, this might be most appropriate for searching through documents and identifying evidence, and even for many initial assessments of credibility, relevance, and consistency with hypotheses. Accordingly, we suggest that the best overall strategy for ACH is to alternate between independent work and collaborative reviews facilitated by ACH Walkthrough. The software allows analysis data to be transferred back and forth with spreadsheets.

The collaborative walkthrough is structured into a series of steps, where each is a step in an ACH analysis, together with discussion relevant to that step. To increase the value of the discussion, we suggest that team members adopt roles. For example, one analyst could play the role of a particular expert or organization, and represent that perspective in the discussion. This facilitates a diversity of perspectives in the discussion, and increases the possibility that critical issues will be identified. Heuer Jr. and Pherson (2010) discuss the advantages of role-play in intelligence analysis, along with related techniques such as devil's advocacy and "red team" analysis.

In the walkthrough, our multiple device architecture also supports multiple perspectives on the data. As illustrated in Figure 2, several devices can be used simultaneously with different views (each view is on a different 'tab' in a traditional tabbed display), and any changes made to the data are instantly synchronized. It would be possible, for example, to have two large touch displays, so that one could be used to consider consistency ratings (explained below), and the other could be used to browse related evidence documents. At the same time, individual analysts could check other tabs on the analysis using tablets or smartphones.

## **UI Design**

ACH Walkthrough is a client-server web application, and it requires login with a userid and password on a project basis. Within a project, the software supports many ACH analyses, each with hypotheses, evidence items, and the scoring of these following the model of Heuer. The UI presents several tabs, where each tab supports one functional aspect of the ACH process. We felt that a tabbed design was consistent with Heuer's step-wise process whereby the user's attention is deliberately tunneled through a structured process.

In addition to the basics of ACH Analysis, the software provides several innovative features to leverage surface computing to support collaboration. These include large-scale touch controls, suitable for small groups, some innovative touch controls we call "fretboards", and a visualization technique called "parallel coordinates" applied to ACH. We also provide

“Walkthrough” facilitation to help groups systematically review an ACH analysis. Finally, we use an innovative multi-device approach which allows several devices to be used simultaneously.

*Fretboards.* In ACH, there are several steps that involve entry of a quantitative score: credibility and relevance of evidence items, and consistency of evidence with hypotheses. Instead of using numeric entry, we designed a new touch control, the Fretboard. The name refers to the fingerboard on a stringed instrument, with lines that mark positions for certain musical notes. Our fretboards allow touch and drag interaction to position an indicator, showing the appropriate quantity. This makes the entry highly visible to the group, and allows spatial reasoning (see Figures 3 and 4).

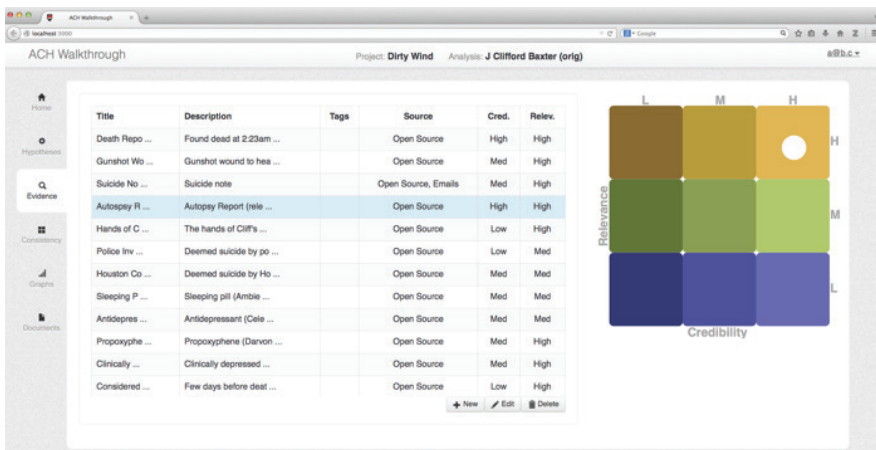


Figure 3.

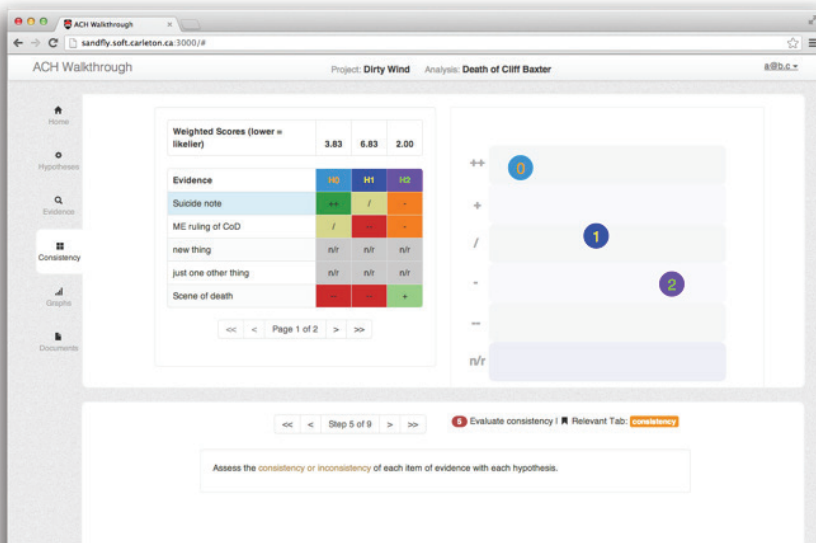


Figure 4.



*Walkthrough Advice.* In our field study, we identified a need to better facilitate strong collaborative activities such as those involved in joint review. To support this, we leverage ideas from a kind of software inspection technique called the “Cognitive Walkthrough” (Wharton et al. (1992), hence the name of our tool being ACH Walkthrough. The technique involves members of the group selecting roles to play in the review, and then the group stepping through the analysis together discussing each step. This supports a diversity of ideas, and avoids “groupthink”. To support this, our tool has “walkthrough notes” that appear and give guidance, as seen in Figure 4.

*Parallel Coordinates Visualization.* In our field study and in later exploration of ACH analysis, we found that people wanted to consider the overall patterns in rating evidence for credibility and relevance, and in scoring of hypotheses for consistency. To support this in our tool, we added a visualization of the ACH analysis using the visual formalism known as a “Parallel Coordinates”. We considered alternatives Wilson, Brown, and Biddle (2014), but settled on this visualization for its fit to task. Parallel Coordinates is an established visualization technique (Inselberg and Dimsdale (1990) to aid exploration of diverse data, and the technique has been advocated especially in the context of cyber-security Conti (2007). See Figure 5 for an example.

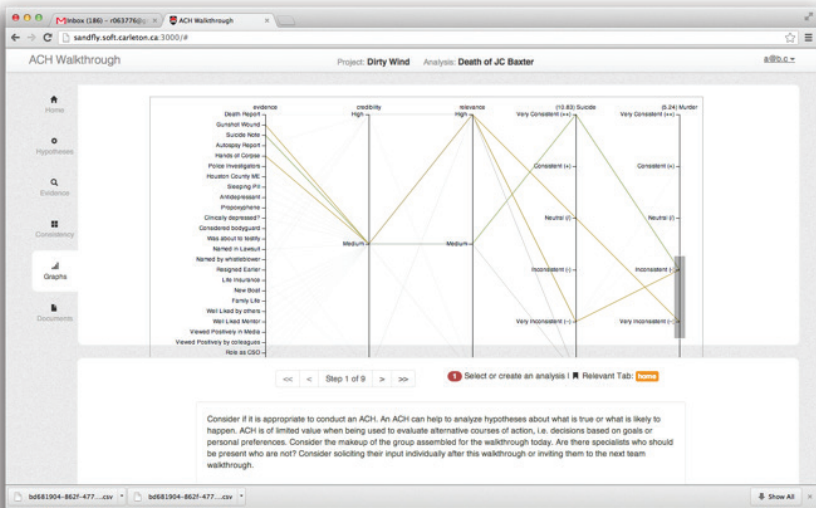


Figure 5.

*Multiple Devices.* One important collaborative characteristic in our software does not involve any specific element in the UI. By leveraging Meteor’s automatic synchronization of data across connected clients, multiple screens/users are updated in near real-time, as illustrated in Figure 5. This means that, at a meeting, several screens can be used for the same ACH analysis, where changes made on any device are reflected on them all.

Multiple large screens may be used, or small tablets. This facilitates parallel work in a collaborative context.

### Software Implementation

The web-based approach was taken to enable deployment across many platforms with sufficiently powerful and standards-compliant web browsers, and without any need for complex software installation. As with most web applications, the overall system depends on a central server, with a certain amount of code loaded onto the browser (client) while the software is running. The ACH-W software relies, however, on processing that occurs on both the server and the client. This client application is delivered and updated without interruption or the need for client-side installation. In many cases the server can even be modified and restarted without the client application losing its place. This approach enables many useful features, such as no data being stored on the client machine when the program is not in use, and the ability for simultaneous use of the software for the same analysis by different devices. The software is written primarily in the JavaScript programming language, using standards compliant language software running in both servers and clients. We use several important external but open-source libraries in our implementation.

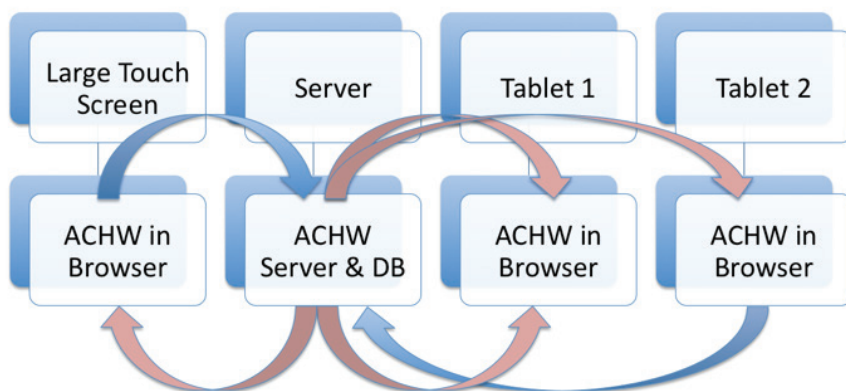


Figure 6. Multiple Device Flow: In ACH Walkthrough, any number of devices of various kinds can be used to work on the analysis, and to make changes simultaneously and independently. The changes flow to the server, and thence to any other devices working on the same analysis.

### Evaluation

For our evaluation of Ach Walkthrough we had access to two vital resources. The first resource consisted of senior members of the group from the field study ('the client'), and the second resource was a panel of professors at an American university ('the demo panel') for whom the client requested we give an extended hands-on demo. Both groups provided extensive and helpful feedback.

One set of issues identified concerned the visualization elements in ACH

Walkthrough, especially the parallel coordinates display, and the interaction afforded by “brushing” on the axes.

Figure 7 shows the first version of a plot for ACH-W, and an important issue should be immediately apparent. The problem can be seen when examining the number of lines between the first and second axes (left to right) and the apparent loss of detail as lines in subsequent gaps overlap. This problem results from the fact that the data points are not floating point values but instead are categorical (the first axis) and ordinal (the remaining axes). This loss of information can be corrected by using curved lines, as shown in Figure 7.

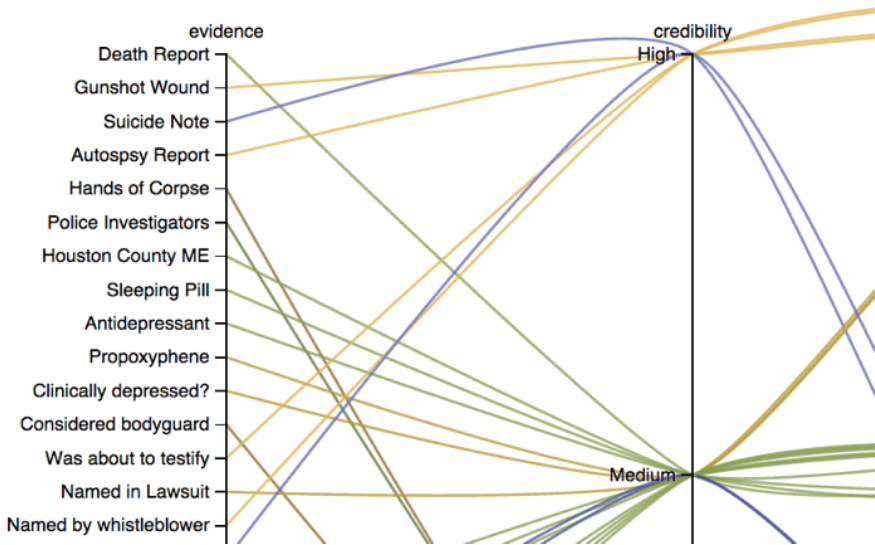


Figure 7. ACH-W Parallel Coordinates showing improvement with curves.

Brushing supports a surprising range of interaction tasks, especially as users become familiar with the meaning of the graph’s dimensions. Users new to parallel coordinates graphs might at first be drawn to visual clusters and reinforcing trends across the display, and indeed in many domains this is a strength of parallel coordinates in general. In the particular case of analysis work like ACH, the real power comes from drawing one’s attention to individual evidence items that fall within meaningful regions of the graph and then taking the time to consider one’s evaluations from fresh perspectives.

Interaction with parallel coordinates supports this kind of diagnostic reasoning by making it easy to select items that help rule out a given hypothesis. The user can create a brush that selects for ratings of inconsistent or very inconsistent along a particular hypothesis axis and then draw their attention to the evidence associated with only the highlighted lines. If they had previously defined brushes for high credibility and relevance, they would quickly find evidence items requiring the greatest consideration.

Our informal usability sessions revealed opportunities for refinements of interactive features. Our internal testing using brushing and parallel coordinates had shown it offered powerful analytic value, but in user testing we learned that it does depend on some prior awareness of the brushing as well as a certain level of patience and attention to detail. We had assumed that most users would have encountered interactive displays in web forms, but for several users (particularly those new to parallel coordinates), the availability of brushing was not immediately obvious. It may have gone against their expectations if they assumed that the visualization was merely a static aggregation of data.

Without cues from experienced users, our testers did not attempt to apply any brushes. In our current implementation of ACH-W there aren't any obvious interaction cues for newcomers. In fact there is only one type of discoverable affordance and it is offered to mouse users when hovering the pointer over an axis. Unfortunately this feature assumed that hovering could even take place. Users of touch interfaces lack the ability to hover, and so they miss out on interaction cues altogether.

This issue became apparent through a usability test where the participant was helpfully thinking aloud and found himself stuck on one of the walkthrough steps. It was only the novelty of the technique that caused a problem. Once he was shown the availability of the brush feature, its meaning was readily apparent. Even when users understood brushing, they did not immediately grasp its ability to help seek evidence that could disprove their favourite hypothesis, a task that is central to reducing cognitive bias. One possible enhancement for first-time users might be to introduce the feature of brush-based filtering by offering a list of pre-set selections based on ACH-specific tasks (e.g. filter irrelevant items, confirm diagnostic items for hypothesis  $n$ , then  $n+1$ , find counter-evidence for hypothesis  $n$ , etc.) and then instruct the user to walk through each of these presets. Also, the initial rendering of the parallel coordinates graph could briefly show animated selections on each axis that quickly unfold until they encompass their full range and then leave behind affordances for the user to adjust (see mockup in Figure 8).

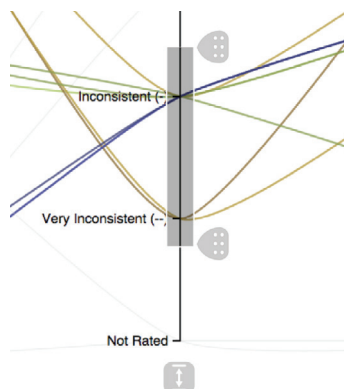


Figure 8. Mockup of possible affordances.

A number of senior researchers from the demo panel expressed concerns with the process of ACH in its present form. Their concerns fell into two broad categories: psychological, especially whether ACH avoids cognitive bias, and mathematical, about the nature of the model. These issues are both interesting, as they do not relate specifically to our software, but we will not elaborate further here.

Comments on the software features of ACH-Walkthrough, however, are our concern. One commenter was concerned that the two digits of precision used in presenting scores against hypotheses in the Consistency Tab and the Graphs Tab might mislead the user into perceiving a mathematical distinction between closely ranked scores. The scores use a formula developed with Heuer in the construction of the Xerox PARC version of ACH, and we chose to reproduce this formula. Future versions will represent a more coarse representation of score, or may eliminate the numeric score entirely and instead use a visualization that fosters appropriate attention to the similarity rather than the minor differences between hypotheses.

Similar to this concern was a comment on the immediate feedback of the change in score provided while manipulating the ratings on the Consistency tab. The reviewer believed the immediate feedback might actually encourage confirmation bias rather than fight it. This was an interesting concern that could form the basis of a future experimental review. Design of such an experiment could prove difficult to achieve however, particularly given the various other natural sources of confirmation bias present. It would also be difficult to produce a baseline from which to establish the presence of an effect. This was left as another potential avenue for future research.

Overall, our experience with ACH Walkthrough was positive, but the interaction design and software are still at the stage of functional prototype. The next steps should be a more controlled usability study, ideally professional analysts, and a real problem suitable for analysis. At the same time, our early feedback was often accompanied by suggestions for new features. The most commonly requested features are in the list below. The first two items on this list inspired the next project in our work, which we present in the next section:

1. Versioning and merging of versions
2. Roll-back and play-back functions
3. Improved support for integration with external data sources
4. Bidirectional links between evidence and precisely tagged supporting documentation
5. Voice input for hypotheses and evidence
6. Colour customization for rating system (from a colour-blind evaluator)

## **Ra: Support for Application Interaction History**

In the previous sections, we have described how our field study suggested that collaborative security analysis would be assisted by large surface tools, and we then presented such as tool, ACH Walkthrough. Both when observing usage of ACH-W, and when seeking feedback, two additional features seemed especially worthwhile exploring: Versioning and merging of versions, and Roll-back and play-back functions. We therefore set out to explore how such features might be provided. We developed an add-on for web applications, such as ACH-W, to support interaction history, and present our prototype in this section.

All users of complex software make decisions that they may later wish to change. Software can support this need to revisit past decisions by keeping past versions of the application's state that the user can go back to. There are several mechanisms for maintaining and presenting this history. Since early in the history of desktop computing in the 1970s and 80s, most user applications have provided users with an "undo" command to revert the most recent change. But not all uses of "undo" occur because of mistakes. Kirsh and Maglio (1994) divide (non-erroneous) user interactions into two categories: pragmatic actions are those that actually move the user closer to their goal, and epistemic actions are those that help the user learn about their situation, exploring to gather information that is either "hidden or hard to compute mentally". So interaction history systems should be designed to support epistemic interaction as well as error recovery.

Touchscreens make epistemic interaction more compelling. Lee et al. (2012) argue that touchscreens enable a kind of directness even more direct than the Direct Manipulation described by Shneiderman (1981), since Shneiderman was assuming the use of a mouse and keyboard. Large touchscreens also enable new kinds of co-located collaboration possibilities J. Brown et al. (2013). Sharing a touchscreen is much easier than sharing a keyboard and mouse. Touch interfaces are changing the kind of software we make, and the new types of applications need to support epistemic interaction.

Tools like ACH improve analysis work by reducing the impact of analysts' cognitive biases. ACH in particular is meant to reduce confirmation bias, where analysts will unknowingly focus on the evidence that supports their pet hypotheses rather than evaluating all evidence fairly. Another cognitive process that can interfere with effective analysis is satisficing Simon (1956), in which an analyst will stop when they have reached an answer that seems "good enough". On its own, this is rational and acceptable as long as the threshold is set right. The problem is that software may impose additional costs to further exploration – at worst, further exploration requires starting all over again – and that lowers the "good enough" threshold. This is related to the problem of premature commitment from the Cognitive Dimensions of Notations framework Blackwell and Green (2003). If you have reached a solution but want to try something else, you must decide whether it's worth the effort to just get back what you had if the "else" isn't any better.

Without a system for storing interaction history, the user is constrained to repeat the steps to achieve the old solution, or else execute the inverse of all actions taken since then. This may be a significant cost to exploration.

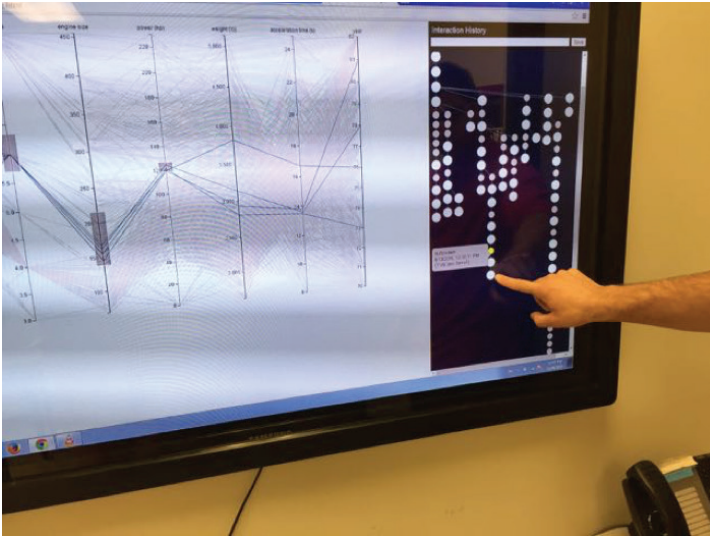


Figure 9. A study participant using our prototype software Ra (the dark blue sidebar) with an interactive data analysis tool.

We wanted to develop a system to provide users with access to all their historical interaction states, including those that would be discarded by a traditional stack-model undo system. Such a system should encourage more epistemic interaction by allowing users to return to known-good states after exploring and reduce premature commitment and the urge to satisfice by freeing users from the risk of losing good work while investigating other options. We want to make software tools better support data analysis and other kinds of nonlinear tasks that are hard to automate; we want risk-free exploration.

In furtherance of these goals, we developed a prototype library called Ra, pictured in Figure 9.

### **Visualizing History**

Software has many different methods for handling interaction history and exposing it to users. We reviewed a large number of approaches, but the ones that seemed most general were those from software source code version management. Some modern version control software such as Git and Mercurial store the history of a project as a directed acyclic graph (DAG). Tools for working with these systems will often display the history as a DAG as well. In software projects, the structure is a DAG because most work ends up in the final product. When developers work in parallel, they are usually not working on alternatives; they will merge both lines of development together.

In Ra, we represent the history as a tree. (This is also the data structure used internally.) We expected that the exploratory behaviour we want to encourage would result in mostly dead ends, or multiple different results for presentation or comparison, rather than some unification of most of the work. Merges seem like a desirable feature in some cases, but their usefulness may not be worth the extra complexity. It is unlikely that Ra could perform merges automatically, and there is no obvious way for a user to direct the merge of two snapshots of application state. This is in contrast to merging source code, which can often be done automatically, and manual merges of source code requires understanding the code, whereas merging application state would require understanding the (usually not human-readable) representation of that state.

The tree visualization in the Ra sidebar is inspired by the visualizations for version control systems, and the traditional visualization of trees in computer science. New nodes are added below, and if necessary to the right of, the parent node (which represents the state that happened immediately beforehand). This also, happily, matches normal English reading order.

## **Implementation**

The general technical goal is to capture snapshots of the running state of a Web application, and then be able to load snapshots without too much delay. There are several ways this could be accomplished, each with its own drawbacks.

For ease of prototyping, we chose to implement Ra as a JavaScript library, to be included in the Web application with some (but preferably minimal) supporting application changes. We wanted Ra to be non-invasive enough that it can be added to an existing application without restructuring the whole thing.

The central part of keeping required changes to the host application localized is the use of objects. The newly-finalized ECMAScript 2015 Language Specification (“ECMAScript 2015 Language Specification” 2015) (ECMAScript 6) introduces them, though prominent JavaScript engines such as SpiderMonkey in Firefox Mozilla Developer Network (n.d.) implemented versions specified in drafts of the specification well before the final publication. A *proxy* imitates an existing object, but it can intercept almost all interaction with that object. In the specification (“ECMAScript 2015 Language Specification” 2015), a *proxy* is defined as an “Exotic Object”, meaning that it is not required to display normal JS object semantics. For example, immediately after setting a property on a regular object, retrieving the same property must return the previously stored value (unless an exception was raised); *proxies* are not required to act this way. The *proxy* has a special handler function that can override the normal object semantics. Following the same example, retrieving a property value as would call a function provided when the *proxy* was created, and the expression would evaluate to the return value of that function. The function can usually return any



value it chooses, although there are some more complicated edge cases requiring the semantics of certain features such as non-writable properties to be respected (“ECMAScript 2015 Language Specification” 2015; Mozilla Develop Network, n.d.).

An application using Ra substitutes objects created by Ra for the objects that hold its state. When all objects that hold state in the application are actually objects managed by Ra, the application code continues to interact with Ra implicitly when it uses those objects, yet all other code can continue to use the objects as if they were the real state objects. This allows us to update state objects on demand, wherever they may be inside the application at the time, and whoever may have references to them. From the perspective of the application code, when the user loads a saved snapshot, the state objects immediately become the saved values, without requiring the application to actually make any changes. This is accomplished by setting all the traps in the to return the value from the current saved state object instead of the original. If the application was already written in an object-oriented style following the Model-View-Controller (MVC) pattern Krasner, Pope, and others (1988), with state stored as properties of long-lived objects, then the state objects do not have to be tracked through their entire lifecycle; to support Ra, changes are needed only where state objects are created. Additionally, since the “objects storing application state” that Ra needs correspond to objects in the Model component of MVC, all the state objects are already identified and ready to be replaced by proxies.

Our proxy-based approach corresponds very nicely to traditional MVC or three-tiered application architectures, since the Model component keeps the state objects isolated from the other code. However, the state object requirements are impractical in some programming paradigms and architectural styles (or lack thereof) used in JavaScript. Storing state in the web application domain object model (DOM) is a common technique that is problematic for Ra. Trying to recover the important state from the DOM from Ra’s position would be complicated and error-prone at best. Some applications keep state in closures, in variables local to a function but available to any other functions that are lexically inside the function. Programs written in the functional paradigm generally rely on this rather than mutating objects. It is common to use closures to avoid adding properties to the global object in top-level code, and some store state in variables in that scope. There is also a well-known pattern for “private members” in code trying to emulate Java-style object-oriented programming by using closures to restrict access to variables, since variables cannot be updated from outside their scope.

We found that the parallel coordinates application we used had several of these problems. It uses the D3 framework Bostock, Ogievetsky, and Heer (2011), which maintains listeners on the state objects in the model component, and the code was written in a partly functional style with significant state variables in closures. Restructuring the application to meet

Ra's requirements would have been a large undertaking comparable to rewriting the application.

We developed a mediation mechanism to allow an application to use Ra without significant restructuring when it can't meet the state object requirements. It puts more responsibility for managing state on the application, so this may be of limited practical value in an application with complex state. However, it was sufficient for the parallel coordinates application. The Ra API is extended to include priests, which are special objects provided by the application that act as interpreters between Ra and the application state. Objects that store state but do not meet Ra's requirements are still marked with a call to on creation, but a priest name can be supplied as well, as in the call . Instead of returning an object wrapping, this will return itself. Ra will then delegate responsibility for monitoring, saving, and restoring that object to a priest registered with the given name.

### Ra User Experience

The parallel coordinates application (showing the months Ottawa had at least some snow and the temperature never dropped below zero – a rare occurrence). The balloon popup for a node is showing the label and timestamp. The user has already returned to that state and started a new branch; the "Load this" button would let them do so again.

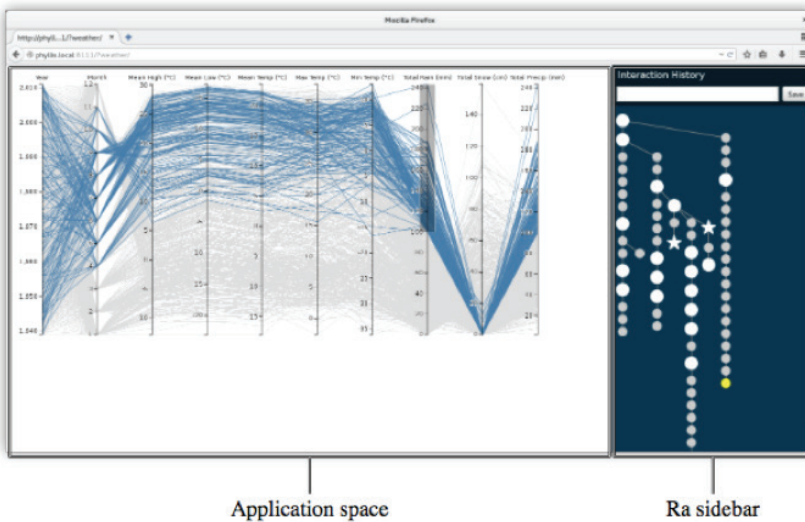
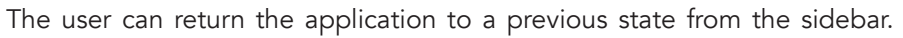



Figure 10. Ra divides the page to make room for a sidebar.

When Ra is part of a Web application, it adds a sidebar, shrinking the available application space, as shown in Figure 10. Ra does not try to intercept or manage user interaction with the application part, so aside from being narrower, the application works exactly as it would without Ra.

As the user uses the application, Ra records the state of the application

when it changes. We call the recorded state a “snapshot”. These are shown as nodes in the tree visualization in the sidebar, where each snapshot follows from its parent in the tree (“time flows down”).

The user can return the application to a previous state from the sidebar. Tapping or hovering with the mouse brings up a balloon popup for each node, as shown in , from which the saved state can be loaded. The node in the tree that represents the current application state is marked in yellow. When the user returns to a previous state, they can still interact with the application – making different decisions this time. Instead of replacing the history from that point forward, as a traditional undo system would, Ra starts a new branch in the tree, as shown in , so both timelines are available.

Ra can record states for different reasons, and these get different glyphs for the nodes in the tree visualization. There are three types of nodes:

- Automatic snapshots are shown as small dots. They are all given the same default label (“autosave”) because no semantic information about the state is available. Ra creates them automatically when it detects that the state has changed, though this is rate-limited to at most one per second, and the other snapshot types supersede automatic snapshots.
- App-suggested snapshots are shown as bigger, brighter dots. The application can tell Ra to create one of these snapshot when it is in a state the user is likely to want to return to, which is why these nodes are more prominent. The application provides the label for these snapshots. This kind of snapshot depends on the support of the application, and some applications (such as our simple maze) may not create any. Our parallel coordinates application uses these to mark the creation of new brushes (selections).
- Starred snapshots are shown as stars. They are created explicitly by the user. They may have a label set, also provided by the user. To make a starred snapshot, the user enters the label in the textbox in the sidebar, then presses the “Save” button. Note that in previous versions of Ra, starred snapshots had the same appearance as app-suggested snapshots.

We performed a usability evaluation in which participants used a simple puzzle and an interactive data analysis tool with Ra. Our experience implementing Ra and our observations from the study revealed several important themes. Perhaps most interestingly, we observed three kinds of history tasks. This categorization is not directly about the user’s intent, for which there would surely be more than three categories, but the relationship between the state the user was in (old) and the state to which the user went (new).

### **Correcting Mistakes**

In the “oops-undo” case, the user has made a mistake recently, or tried to perform an action but the computer did something unexpected. The old

state was clearly wrong; the user does not expect to need it again, and perhaps it should be hidden from view. This is the case that traditional undo was designed for, and it is reasonably well-suited to it.

### **Trying Alternatives**

In the “undo-retry” case, the user wants to try some alternative, usually starting from further back in history than the oops-undo case, or from a parallel timeline. New work will be based on the new state, but the user may not be certain that the old state should be discarded; the old state may still be useful.

Ra was intended to support this task in particular. Traditional undo mechanisms force the user to give up one branch to work on another, which requires the user to commit to a decision before they see the result; they may have to resort to manual version control (saving the file separately for each experiment) or make a decision with incomplete information. Ra allows the user to keep any number of parallel alternatives without the extra costs of saving and managing alternatives in files.

### **Comparing Versions**

In the “undo-review-redo” case, the user just wants to look at a previous version of their work. It might be to compare two alternatives, or to copy a particular piece of a previous solution, or even to remind themselves of what not to do. The old state is still the working copy where new editing work will happen; the new state is not something the user wants to keep.

Using traditional undo for this task is particularly dangerous because any accidental edit will discard the redo stack, leaving the user in a state they intended to abandon. Ra happens to support it better, since all versions are accessible, but there are features that could improve the experience, such as some way to keep track of the current working branch separately from the version being viewed. We did not think of this task when initially designing Ra, so it is an interesting outcome of the study that participants did this anyway.

Reviewing sequences of past states without editing them may also be able to help other people understand the final state. For example, Farah and Lethbridge Farah and Lethbridge (2007) developed a linear timeline for reviewing the development of software engineering models. In the field of intelligence analysis, the system could be used for a kind of traceability, allowing analysts to review the decisions that led them to a conclusion. Ra could emphasize this capability by making it easy to explore the path from a state to the root of the tree – that is, the work that went into a selected state, ignoring parallel timelines.

### **Strata: Web Application Annotation**

In several steps of our research, we have observed professionals in co-located collaborative security analysis work. One common kind of behaviour

is annotation, whether of documents, or on whiteboard diagrams. Where an application was displayed on a large screen, whether television or projection, it was common to see people using paper or whiteboards to make sketch duplications of key parts, and then annotate these. Often users tended to point and gesture to elements on the screen, as if they were marking up the content on the display itself. We therefore decided to explore explicit support for this behaviour, and developed an add-on for web applications to support annotation and easy screen capture. This would allow the users collaborating over the display to annotate the web application which they are interacting with, in addition to saving and retrieving previously saved annotations. In this paper we present the technology choices and interaction design of our prototype, “Strata”, see Figure 11.

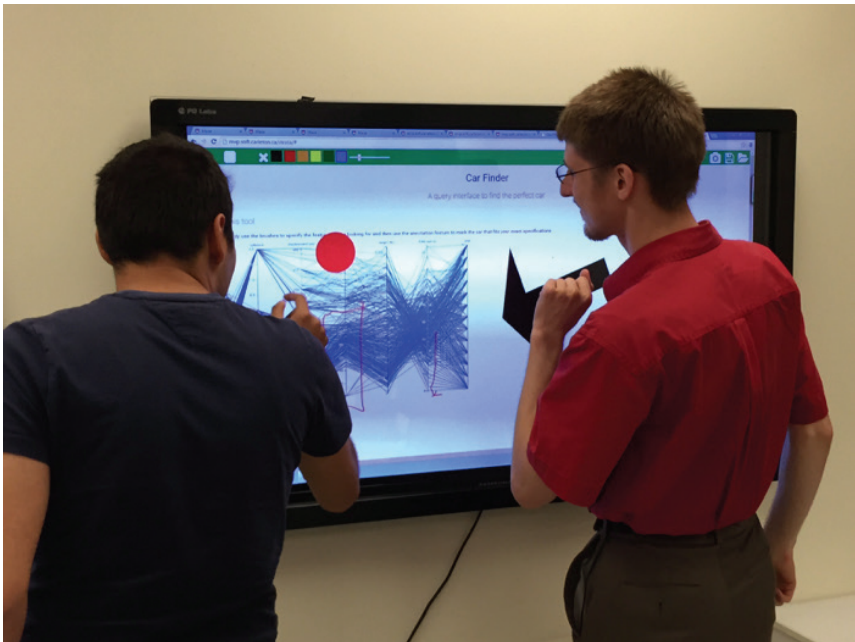


Figure 11. Co-located users collaborating over a large touchscreen display using Strata to mark up a sample “car finder” web application.

The value of markup on documents has been included in various contexts including in modern PDF reader applications such as Adobe’s Acrobat and Apple’s Preview which provide users with annotation capabilities on PDF documents. Annotation capabilities have also been included in the web browser by Microsoft’s Edge browser in their Windows 10 operating system, which provides markup tools for web pages such free hand drawing, highlighting, and text based notes. There are also various web browser extensions which allow for marking up webpages, such as the Hypothesis extension Hypothes.is (n.d.).

Annotations have also been the subject of various studies in academia. Denoue and Vignollet proposed a very simple implementation by storing

annotations on the client using extended URLs and avoiding the server all together (Denoue and Vignollet, 2002). Alternatively, Sodhi Chatti describes a “transparent white board” overlay approach to creating and storing annotations; the annotations would be formed so that it is self-contained which would therefore allow the annotations to be stored anywhere (either server or client) Chatti et al. (2006). Finally, Beryl Plimmer explores putting the web page into an iframe and then overlaying Adobe Flash based annotations on top of the frame and tagging annotations with metadata associated with the user. The annotations could then be stored in a database and thus retrieved at anytime and even be shared with various users Plimmer et al. (2010).

In summary, earlier work focused largely on the value of annotations on documents using a mouse driven interface using regular computer monitors, our focus is on the value of annotations on web applications (and not documents) as a mechanism for facilitating collaboration amongst users over large touchscreen based displays. The importance of annotations for interactive systems has long been suggested by Thomas Green as “secondary notation” in his identification of “cognitive dimensions” of complex systems Blackwell et al. (2001).

Our goal was to implement a JavaScript based add-on that can be included in any web application to provide users with mark up capabilities. Therefore when we developed the prototype of the Strata system, we decided that the project should meet the following requirements:

1. The system should enable creating annotations on top of web applications.
2. Annotations generated should not obstruct the content on the display. The content should remain accessible and manipulatable even if there is markup overlaid on top.
3. Both touch and mouse based input should be supported by the system, since there may be times when users would prefer using a mouse even on a touchscreen computer.
4. Multitouch drawing capabilities and gestures must be included in order to enable multi-user collaboration.
5. The system should leverage multitouch gestures in order to ease usability and facilitate exploratory interaction within the system.
6. The system should be able to save and load annotations so that users can revisit them at a later time or share them with other users.
7. The system should integrate seamlessly with a web application without requiring several re-writes by a web application creator wishing to include the system.

Strata was designed as a library that can be added on to any web applications, rather than creating an extension which requires users to

modify their web browser. This allows web application developers to easily integrate the Strata system with any existing web application in order to gain access to markup capabilities and enhance the collaborative aspect of their web application.

Initial prototypes of Strata were developed using HTML Canvas. However, this technology was abandoned in favour of Standard Vector Graphics (SVG), because the HTML5 Canvas element would overlay over the content and would thus prevent the user from interacting with the elements underneath. Our design is that users can toggle between interacting with the application itself, or with the annotation as a “layer” (hence the name “Strata”).

Since the system is intended to work on large touchscreen devices, the system should support a multitude of features including multi-touch input and gesture recognition. Strata is designed with both of those features in mind, it leverages the Interact.js library which provides unified mouse and touch events thus allowing for development on both touch screens and mouse based personal computers. In order to support multitouch Strata leverages Interact.js’s (Adeyemi, n.d.) “pointerly” attribute to assign a newly created pencil object to each finger thereby mapping each pencil object to a finger therefore allowing for drawing using multiple fingers. Multitouch is essential not only because a single user would expect it but also since the system is intended to be used on large touch screens, it would be intended to be used by multiple users and would therefore require multitouch to foster collaboration between the users interacting with the system. Another advantage of using Interact.js is because it provides support for gestures; any object with that matching class would recognize gestures including pinching to resize an element and a rotate gesture. Initially the system was designed with explicit state switching in mind where a user would have to switch between drawing and gesturing which was a detriment to usability, Strata was then redesign to support implicit state switching whereby tapping an element would switch to a gesture mode and upon completion drawing more would resume. This greatly improved the usability of the software and allowed for a much more natural and streamlined user interaction. By implementing both of these features, the system encourages explanatory interaction on the part of the user which is complemented by the exploratory nature of large touchscreen displays and should thus facilitate collaboration and user engagement.

To demonstrate and conduct preliminary usability testing, included the add-on on a car finder web application, which allows users to explore choices for cars based on economy, power, etc. This application uses parallel coordinates visualizations (Inselberg, 1997) and we use the implemented based on the D3 visualization library (Heer, Bostock, and Ogievetsky, 2010; Bostock (n.d.)). This is one of the primary design elements in our ACH Walkthrough application, introduced above. This visualization shows data attributes on several parallel axes and allows individual elements to be selected by “brushing” on the axes.

The application interface is shown in Figure 12, where the car finder is in the main part of the screen and the Strata add-on interface is shown as a toolbar across the top. The toolbar contains the pencil tool, options for stroke and colour. Document annotation tools such as those found in PDF viewers provide a similar interface for interaction. Users can begin interacting with the strata system by enabling drawing mode by pressing the pencil icon. Once the drawing mode is triggered, the web application will no longer become accessible so that the users can mark up the web application without fear of accidentally selecting text or interacting with the web application. The web application's functionality can be resumed once the drawing mode is disabled.

Once in drawing mode, the user has various options on the toolbar including clearing the paper, setting the colour and the stroke size options for the freehand "pencil" tool and a rectangle canned shape option. These options can also be changed at any time by using context menu options.



## Car Finder

A query interface to find the perfect car

### How to use this tool

To use this tool simply use the brushes to specify the features you're looking for and then use the annotation feature to mark the car that fits your exact specifications

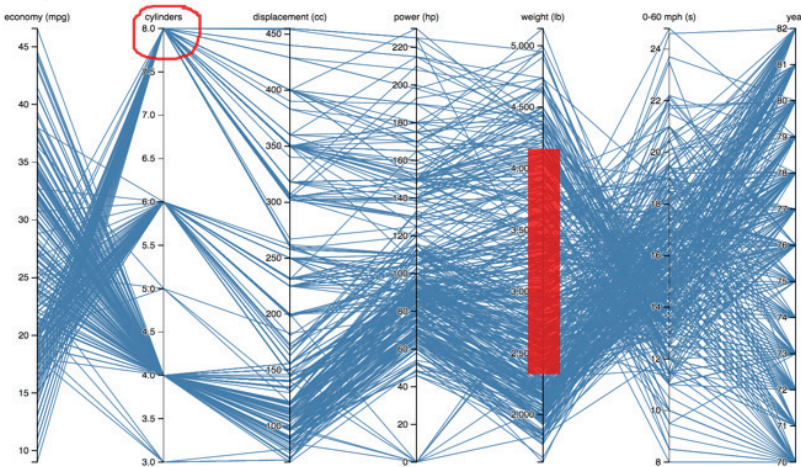


Figure 12. Strata annotations have been used to mark up the sample web application using freehand "pencil" drawings and a canned rectangle shape.

The strata add-on provides users with various ways of annotating web



applications they have access to both free hand drawings as well as canned shapes including arrows, rectangles and ellipses. A context menu is used to create new shapes as demonstrated, since it is a contextual menu it changes the options based on where it was triggered, triggering the context menu over an element will bring up the styling options for that particular element, otherwise triggering it over the web application will bring up element creation options which allows users to add rectangles, circles. In order to support both touch and mouse for triggering the context menu the context menu click (typically a right click) brings up the contextual menu, alternatively the menu can be brought up by using a “hold” gesture in order to support systems without a mouse.

The Strata toolbar presents users with various options, including saving the annotations as a JSON file once they are done annotating the web application, the resulting JSON file can then be loaded at any later time by invoking the load function through the Strata toolbar or shared with other users who can then load the annotations, view them and possibly add or remove elements from them. The system also includes the ability to save a screenshot of the annotations which can be invoked by clicking the camera icon in the Strata toolbar, once the screenshot functionality is invoked a screenshot of the web application (including any markup) will be taken and sent to a private image gallery.

Future work on the project includes formal usability testing, and in particular we need to do it in an ecologically valid context where we have people collaborating on real work using the system. Moreover, the possibility of semantic annotations should be explored, by which we mean a mechanism that can allow applications to present hooks so that Strata can do smart annotations using those hooks (delegating the markup to the web application). Finally, we believe that it would be important to investigate the value of a web extension based architecture in the future in order to be able to utilize the strata overlay in any web application without the developers having to include support themselves.

## **Discussion and Conclusion**

In this chapter we have reviewed a number of our projects on surface computing for security analysis. We began with a survey of related work, and then conducted field studies. We developed ACH Walkthrough, a surface computing application to support analysis work, and then two add-ons, one to support interaction history, and another to support annotation. This work was carried out over several years, and involved several projects we did not address in detail here.

Reviewing the work as a whole, several themes stand out. One suggested by our literature review, and confirmed in our field study, is simply that large surfaces, whether whiteboards, posters, or large computer displays, really do facilitate collaboration. Small surfaces are hard for multiple people to see, and are perceived as personal, making joint use seem invasive.

A second theme is more specific to security analysis. The work involves large amounts of data that is typically incomplete, unclear as to relevance, and can even be intentionally deceptive. Yet making determinations and recommendations must still be done, because security always involves risk. Together, this has led to analysis processes that have several kinds of filtering, assessment, and iteration, for example as described by Pirolli and Card (2005). Our field study of professional analysts suggested that this process would be improved in several ways by better collaboration, for example using large surfaces. This is also consistent with results found by Isenberg et al. (2010) in their study of students as doing intelligence analysis. However, we also learned that it was unrealistic, and almost certainly unhelpful, to expect analysts to work in close collaboration all the time. Much of the work required intense focus and concentration, and was best done alone for periods of time.

In our work developing and testing our surface computing tool for security analysis, ACH Walkthrough, a cluster of themes emerged. One was the importance of guided collaboration, where our walkthrough steps helped users follow the ACH process. Another was that it became clear that the work involved “epistemic” interaction. This has been identified by Kirsch and Maglio (1994) as supporting not actions intended have direct consequences, but rather speculative actions, done to explore possibilities. We realized this was the principle underlying our fretboards and parallel coordinates visualizations. At the same time, we appreciated the need for analysts to take away results, work alone, and bring back new ideas.

All this led to identification of new ways to better support this kind of work. Interaction history support, such as provided by Ra, can help epistemic interaction because it frees the analysts to explore alternative, while allowing easy return to previous states. Annotation of application states can be supported by software like Strata, which allows collaboration around application software, while making notes on the results for later review.

In summary, we found that surface computing has a strong relevance for security analysis, especially in how it can support collaborative epistemic interaction, and this can be improved by support for guidance, interaction history, and annotation. These are promising new directions for software design.





# **BIOGRAPHIES**

**TULLIO DE SOUZA ALCANTARA** is a UX developer for Computer Modelling Group Ltd. since 2013 and has been involved with software development since 2002. He completed his MSc in Computer Science at the University of Calgary, under the supervision of Dr. Frank Maurer in 2013. During his MSc, he specialized in human-computer interaction and prototyping of touch based applications.

**BON ADRIEL ASENIERO** is currently a PhD student at the Interactions Lab (iLab), University of Calgary. He is a member of the Innovations in Visualization Lab (InnoVis), supervised by Dr. Sheelagh Carpendale and Dr. Anthony Tang. His research topic is in the intersection of Information Visualization (InfoVis), Personal Informatics, and Human-Computer Interaction (HCI). He has also done research in augmented reality under Dr. Ehud Sharlin in the past. He has received a Bachelor's degree in Computer Science: Human-Computer Interaction, and a Master's degree in Computer Science: Information Visualization.

**ALAA AZAZI** is a graduate research assistant at the Agile Surface Engineering Group at the University of Calgary. He received his BSc in Computer Science & General Mathematics in 2014, and was awarded the Alberta Innovates Graduate Student Scholarship to join the MSc program at the University of Calgary. His work spans areas of ubiquitous computing, human-computer interaction, and software engineering.

**PIERRE BASTIANELLI** is an Interaction Designer at VizworX, Inc., a company that was born off the ASE Lab and some of the SurfNet technologies. Pierre initially graduated as a Computer Science Engineer with an MSc in Human Computer Interaction from the National Civil Aviation University (ENAC) in Toulouse, France. He then joined the ASE lab for half a year as part of the LEIF initiative, and wrote his Master's thesis there. After that, he worked at Ivrrnet, Inc., where the TableNOC project was being piloted, and did extensive design and prototyping on the matter, and submitted a related paper. At 3D-P and then VizworX, he then stayed close to the tabletop, webapp and mapping technologies that are now an essential part of his skillset.

**GUILLAUME BESACIER** is an Assistant Professor of Computer Science and Information and Communication Sciences at Université Paris 8 Vincennes-Saint-Denis (France), in the team Cybermedia, Interactions, Transdisciplinarity, Ubiquity (Citu). He was a SurfNet Postdoctoral Fellow from 2011-2013 at the University of Waterloo, where he managed the Surface Ghost projects discussed in the "Cross-Device Content Transfer in Table-Centric Multi-Surface Environments" chapter. His current research interests include augmented environments, connected objects, and digital mediation of cultural heritage... around an interactive tabletop, of course.

**ROBERT BIDDLE** is a Professor in the School of Computer Science and Institute of Cognitive Science at Carleton University in Ottawa, Canada. His research is in Human-Computer Interaction and Software Design. His recent research projects are on usable security, especially authentication and security decision-making, and on large-scale multi-touch devices, especially environments for collaborative design and visualization. Robert has Bachelors and Masters degrees in Mathematics and Computer Science from the University of Waterloo, a PhD from the University of Canterbury, and has diplomas in both childhood and adult education.

**CHRISTOPHE BORTOLASO** holds a Master's in Human-Computer Interaction from the University of Toulouse. During his PhD (also from the University of Toulouse),

his performed research in design methods for mixed reality systems, lead to an installation in the Toulouse Museum of Natural History. He worked as a SurfNet-sponsored post-doctoral fellow at Queen's University, during which time he led the development of the OrMiS mixed-surface environment for simulation-based training. He served as demonstrations co-chair at the 2014 Interactive Tabletops and Surfaces (ITS) conference.

**JOHN BROSZ** is the Research Data and Visualization Coordinator at the University of Calgary's Taylor Family Digital Library. In this capacity he manages the Visualization Studio, a research space with a high-resolution wall display that supports researchers from across campus. His past experience includes time as a post-doctoral researcher supervised by Professor Sheelagh Carpendale, and a PhD, MSc, & BSc in Computer Science (focusing on 3D Computer Graphics) at the University of Calgary. John's research interests include information visualization, multi-touch interfaces, large displays, and 3D modeling and rendering.

**DOUG BROWN** is the Senior Synthetic Environment Integrator in the Army Simulation Center in Kingston Ontario. He graduated from the University of Western Ontario in 1981 with an HBSc in Computer Science. He then spent 15 years as a Defence Scientist in Military Operational Research holding positions with the Canadian Navy and the Canadian Army. He also had a posting as an Operational Research Scientist at the SHAPE Technical Centre in The Hague. In 1996 he moved to the then infant Joint Command and Staff Training Centre in Kingston Ontario as the principle modeling and simulation expert. He has remained there over the past 14 years. He has been responsible for the design, construction and execution of synthetic environments for hundreds of Canadian Army and Canadian Forces exercises. He has several publications to his credit.

**JUDITH BROWN** is a postdoctoral fellow in the School of Computer Science and Institute of Cognitive Science at Carleton University in Ottawa, Canada. Her research is in Human-Computer Interaction, and in particular collaboration and the practices and tools that enable it. Her recent research projects have investigated the use of large displays in various domains including digital Cardwalls for agile development teams, incidence response teams in IT operations centres, collaborative analytic work, and collaborative gesturing for pairs working with visualizations. Judith has a PhD in Human-Computer Interaction/Psychology (Carleton University), a Masters in Computer Science (Queen's University) and undergraduate degrees in Psychology and Statistics (Queen's University).

**SHEELAGH CARPENDALE** is a Professor at the University of Calgary where she holds a Canada Research Chair in Information Visualization and an NSERC/AITF/SMART Industrial Research Chair in Interactive Technologies. She directs the Innovations in Visualization (InnoVis) research group and founded the interdisciplinary graduate group, Computational Media Design. Her research on information visualization, large interactive displays, and new media art draws on her dual background in Computer Science (BSc and PhD from Simon Fraser University) and Visual Arts (Sheridan College, School of Design and Emily Carr, College of Art).

**FRANK CENTO** is currently completing a BSc in Mathematical Physics at the University of Waterloo. He worked as an undergraduate research assistant in the Collaborative Systems Laboratory in 2014 under the supervision of Dr. Stacey Scott at the University of Waterloo as a SurfNet student researcher working on the Surface Ghost Project described in the "Cross-Device Content Transfer in Table-Centric

Multi-Surface Environments” chapter.

**EDWIN CHAN** is a Master’s Candidate at the University of Calgary, supervised by Dr. Frank Maurer. He did an internship with the NSERC SurfNet Strategic Network, and completed his bachelor’s degree with a concentration in Human-Computer Interaction. Currently he is researching the design and deployment of wearable devices for field responders in the emergency response domain.

**ANDY COCKBURN** is a Professor of Computer Science at Canterbury University in Christchurch, New Zealand. His research interests cover a wide range of topics in Human-Computer Interaction, including input modeling, spatial memory, cognitive biases, and interaction techniques.

**CHRISTOPHER COLLINS** is the Canada Research Chair in Linguistic Information Visualization and an assistant professor of Computer Science at the University of Ontario Institute of Technology (UOIT), where he leads the Vialab research group. His work has been published in many venues including IEEE Transactions on Visualization and Computer Graphics, and has been featured in popular media such as the Toronto Star and the New York Times Magazine. Dr. Collins received his PhD in Computer Science from the University of Toronto.

**ZACHARY COOK** was a SurfNet co-op intern from 2012-2013 and graduated with a BSc (Hons.) in Computer Science from the University of Ontario Institute of Technology in 2014.

**KODY DILLMAN** is an MSc student studying Computer Science at the University of Calgary as a member of the Interactions Lab. He completed his BSc in Computer Science with a concentration in Human-Computer Interaction (HCI) in 2014, and has been a member of the RICELab (Rethinking Interaction, Collaboration and Engagement Lab) research group since 2012. Kody became interested in HCI while taking an undergraduate course on the topic, learning just how common bad design is, and saw a challenge in creating usable systems to improve peoples’ lives. In his free time, he enjoys video games, comic books, and board games, and has also been known to play a musical instrument or two. Kody currently lives in Cochrane, AB with his wife, Sarah, and their doggie daughter, Niko.

**ANDRE DOUCETTE** is Product Director at Push Interactions Inc. in Saskatoon, Saskatchewan. He completed his PhD in the department of Computer Science at the University of Saskatchewan in 2014. His main research interests involve interaction design, visual design, multi-user systems, and mobile interaction.

**SHAHBANO FAROOQ** is a freelance software engineer and researcher with over 4 years of prior experience in industry developing and designing software. She attained an MSc from University of Calgary. Her research interest is in Information visualization design and collaboration.

**KATHRIN M. GERLING** is a Senior Lecturer in Computer Science at the University of Lincoln, where she is part of the Interactive Technologies Lab (INT LAB) and the Games Research Group. Her main research areas are human-computer interaction and accessibility; her work examines interactive technologies with a purpose besides entertainment. She is particularly interested in how interfaces can be made accessible for audiences with special needs, and how interactive technologies can be leveraged to support healthy lifestyles. She holds a PhD in Computer Science

from the University of Saskatchewan, Canada, and a Master's degree in Cognitive Science from the University of Duisburg-Essen, Germany. Before joining academia, she worked on different projects in the games industry, and still enjoys thinking about issues related to game usability and player experience.

**YASER GHANAM** is a software engineering academic and practitioner currently working as a Project Leader at Schneider Electric - Canada. His experience and interests span a number of areas including agile software development, project management, and usability engineering. Dr. Ghanam holds a doctoral degree in Software Engineering, a bachelor's degree in Computer Engineering, and a minor in Engineering Management.

**NIPPUN GOYAL** is currently completing an MASc in Systems Design Engineering at the University of Waterloo. His research focuses on human-computer interaction, interface design and information visualization. With passion in user-experience design, he wishes to pursue his career as a user researcher and enhancing the world around him. He is also a freelance photographer and enjoys travelling new places.

**NICHOLAS GRAHAM** is a Professor at the School of Computing at Queen's University. He performs research in the design and development of next-generation digital games, with a focus on exergaming, simulation-based training games, and digital tabletop games. Together with researchers at the Holland Bloorview Kids Rehabilitation Hospital, he has developed the Liberi exergame that provides opportunities for exercise and social interaction for children with CP. With the Canadian Forces' Army Simulation Centre, he developed the OrMiS interactive tabletop that uses gaming technology to train military officers in command positions. Graham has transferred technology, both to the public through open source development initiatives, and directly to industry through the co-founding of Namzak Labs Inc. He is a member and former chair of IFIP Working Group 2.7/13.4 on user interface engineering, and member of the steering committee of the ACM SIGCHI Symposium on Play in Human-Computer Interaction. He has received five best paper and honorable mention awards in the last six years.

**SAUL GREENBERG** is a Full Professor and Industrial Research Chair in the Department of Computer Science at the University of Calgary. While he is a computer scientist by training, the work by Saul and his talented students typifies the cross-discipline aspects of Human Computer Interaction, Computer Supported Cooperative Work, and Ubiquitous Computing. He and his crew are well known for their development of toolkits, innovative system designs based on observations of social phenomenon, articulation of design-oriented social science theories, and refinement of evaluation methods. He is a Fellow of the ACM, received the CHCCS Achievement award, and was elected to the ACM CHI Academy for his overall contributions to the field of Human Computer Interaction. Together with Nicolai Marquardt, Sheelagh Cappendale and Bill Buxton he is the co-author of 'Sketching User Experiences: The Workbook' (Morgan-Kaufmann 2012) as well several other books on Human Computer Interaction. <http://saul.cpsc.ucalgary.ca>

**CARL GUTWIN** is a Professor of Computer Science at the University of Saskatchewan. His main research areas are Human-Computer Interaction and Computer-Supported Cooperative Work, including interests in mobile-device interaction, performance analysis and modeling, and spatial memory.

**MARK HANCOCK** is the director of the Touchlab at the University of Waterloo. He is



an assistant professor in the Department of Management Sciences in the Faculty of Engineering and the Associate Director of Research Training for the Games Institute at the University of Waterloo. He is also cross-appointed in both the Cheriton School of Computer Science and the Department of Systems Design Engineering. Dr. Hancock received his PhD from the University of Calgary.

**THEODORE D. HELLMANN** received his PhD from the University of Calgary in 2015 for work covering test-driven development of graphical user interfaces after being a member of the Agile Software Engineering lab since 2008. His interests range from agile methodologies and testing through user-centered design, informative workspaces, software process management, and tech startups.

**UTA HINRICHS** is a Lecturer at the University of St Andrews, Scotland, UK in the SACHI research group. She holds a PhD in Computational Media Design from the University of Calgary. Uta's research is at the intersection of visualization, HCI, design, the humanities, and art. Her work focuses on designing and studying the use and experience of interactive systems that facilitate the exploration and analysis of (cultural) data collections from academic, leisurely, and artistic perspectives.

**SAMUEL HURON** is an Associate Professor at Mines Telecom Paris Tech in Design and Information Telecommunication Technologies. He graduated his PhD in 2014 from the university Paris Sud in collaboration with INRIA. He was then invited as a Post doctorate researcher at the university of Calgary. Before, he was the lead designer of the Institute of Research and Innovation of the Pompidou Center in Paris. His experience is based on fifteen years in interactive media industry where he worked for a broad range of civic, cultural and corporate clients.

**PHILIPPE KRUCHTEN** is Professor of Software Engineering in the Department of Electrical and Computer Engineering of the University of British Columbia, in Vancouver, Canada. He joined UBC in 2004 after a 30+ year career in industry, where he worked mostly with large software-intensive systems design in the domains of telecommunication, defense, aerospace and transportation. His current research interests are in software architecture and the phenomenon of technical debt, which slows down the evolution of large software.

**BEN LAFRENIERE** is a Human-Computer Interaction researcher at Autodesk research in Toronto, Canada. His research interest is in the areas of interactive help systems, the design of novel interaction mechanisms, and the usability of feature-rich software. While at the University of Waterloo, he developed the idea of task-centric user interfaces, in which high-level tasks and goals are used as the central organizing principle for a user interface.

**LINDSAY MACDONALD** is a PhD candidate in the Computational Media Design Interdisciplinary Graduate Group at the University of Calgary in Canada, co-supervised by Sheelagh Carpendale and Jean-René Leblanc. Her approach to research and creative production combines methodologies from computer science, design and art. She is interested in investigating creating site-specific interactive art installations in liminal places, along with the design and interaction challenges inherent in the process. Additionally, she is examining and documenting processes and practices in interdisciplinary art/computer science collaborative projects.

**KARON MACLEAN** is Professor in Computer Science at UBC (B.Sc. in Biology and Mechanical Engineering from Stanford [1986]; M.Sc. and Ph.D. in Mechanical

Engineering from MIT [1996]), with industry experience in robotics and interaction design. Her research interests are in situated haptic and multimodal interfaces, and affective, therapeutic human-robotic interaction. Karon received the Charles A. McDowell Award, 2008; is the Assoc. Editor of IEEE Transactions on Haptics; and was the co-chair of the 2010 and 2012 IEEE Haptics Symposium.

PHILLIP MCCLELLAND completed his BAsC in Systems Design Engineering at the University of Waterloo in 2011. He completed an MASc from the same department in 2013 as part of the SurfNet program. He led the software design and development of the Dominion digital tabletop game and the cross-device transfer methods use in the initial study discussed in the “Cross-Device Content Transfer in Table-Centric Multi-Surface Environments” chapter as part of his Master’s thesis research. Previously, he also served as an undergraduate research assistant in the Collaborative Systems Lab at the University of Waterloo. He now works as a User Experience Designer in Kitchener, Ontario, Canada.

SYLVAIN MALACRIA is a research scientist at Inria Lille, in the Mjolnir group. His research interests are in Human-Computer Interaction, particularly on understanding and improving the transition from novice to expert mode in graphical interfaces, and on identifying which type of resources (software and hardware) can be used to enrich the input bandwidth.

REGAN L. MANDRYK is an Associate Professor of Computer Science at the University of Saskatchewan. She pioneered the area of physiological evaluation for computer games in her PhD research on affective computing at Simon Fraser University with support from Electronic Arts. With over 100 papers that have been cited over 4000 times, she continues to investigate novel ways of understanding players and player experience in partnership with multiple industrial collaborators, but also develops and evaluates persuasive games, exergames, games for special populations including children with neurodevelopmental disorders, games that foster interpersonal relationships, and ubiquitous games that merge the real world with the game world. She has been the invited keynote speaker at two international game conferences, led the Games theme in the Canadian GRAND NCE, was the papers chair for the inaugural CHI PLAY conference, and is leading the new games subcommittee for SIGCHI.

NICOLAI MARQUARDT is a Lecturer in Physical Computing at University College London. At the UCL Interaction Centre he is working in the research areas of ubiquitous computing, physical user interfaces and interactive surfaces. In particular, his research of Proxemic Interactions focuses on how to exploit knowledge about people’s and devices spatial relationships in interaction design. He graduated with a PhD in Computer Science from the Interactions Lab at the University of Calgary, and joined Microsoft Research in Cambridge and Redmond as an intern during his graduate studies. Together with Saul Greenberg, Sheelagh Cpendale and Bill Buxton he is co-author of ‘Sketching User Experiences: The Workbook’ (Morgan-Kaufmann 2012). <http://www.nicolaimarquardt.com>

FRANK MAURER is a professor of Computer Science, and Principal Investigator of the NSERC SurfNet Strategic Network, at the University of Calgary. He also serves as Associate Vice-President (Research) at the University. Dr. Maurer is the co-founder and CTO of a start-up company, VizworX. He has authored and co-authored over of 170 peer-reviewed publications on agile methods, multi-surface systems, analytics technologies and knowledge management.

**MIGUEL NACENTA** is a senior lecturer at the University of St Andrews, Scotland, UK. He is a co-founder of the St Andrews Human-Computer Interaction group (SACHI) and the Scottish Informatics and Computer Alliance (SICSA) HCI theme co-leader. Miguel has been part of NSERC's SurfNet since its inception, when he was a PhD student under the supervision of Carl Gutwin. His interests rotate around multi-display environments, multi-touch interaction techniques, perception in visualization and computational typography.

**LOUISE ORAM** is a scientific programmer at the Oslo University Hospital, with a group that researches image-guided surgery. She graduated from UBC with a Master's of Science from the Department of Computer Science (specialization in Human Computer Interaction) in 2014.

**MATTHEW OSKAMP** recently completed his MSc Degree from Queen's University where he researched and developed military applications for interactive tabletops. He created the TerraGuide system for analyzing virtual terrain using a multi-surface environment, and co-developed the OrMiS multi-surface environment for simulation-based training. He is now a software developer with Stantive Technologies working on the Salesforce platform.

**ERIK PALUKA** completed his BSc from the University of Ontario Institute of Technology in 2013. His MSc (2015) from the University of Ontario Institute of Technology focused on mid-air gesture techniques for interacting with off-screen content.

**JULIAN PETFORD** is a PhD student at the University of St Andrews in Scotland. His research involves exploring the use of full-coverage display systems (FCDs) and their potential use in office and home environments. This includes designing and building software and hardware that supports full coverage applications that are inexpensive to build and easy to program.

**DAVID PINELLE** is a Senior Business Analyst at the Saskatchewan Health Quality Council, and has also held positions as an associate researcher at the National Research Council of Canada and as an assistant professor in computer science at University of Nevada Las Vegas. His research interests in Computer-Supported Cooperative Work involve support for loosely coupled collaboration, coordination over surfaces, and groupware evaluation.

**SYDNEY PRATTE** is an MSc candidate in the Computer Science department at the University of Calgary and is a member of the Agile Surface Engineering lab headed by Dr. Frank Maurer. Sydney completed her BSc in Computer Science with a concentration in HCI and a BA in Italian Studies. Sydney enjoys interface design and iOS development. In her spare time Sydney likes to read and hike with her puppy Bentley.

**JOEY SCARR** is a senior developer at Google in Sydney, Australia. He completed his PhD at the University of Canterbury in Christchurch, New Zealand in 2014. His research interests involve the design of novel interaction techniques, primarily based on the principles of spatial memory and spatial stability.

**STACEY D. SCOTT** is an Associate Professor of Systems Design Engineering, with a Cross Appointment in English, at the University of Waterloo. She was an early pioneer of surface computing in her PhD research on collaborative digital tabletops

at the University of Calgary, and has impacted the surface computing community through ongoing scholarship and community service. With over 100 scientific articles that have been cited over 3000 times, including the seminal papers, "System guidelines for co-located, collaborative work on a tabletop display", and "Territoriality in collaborative tabletop workspaces", she continues to investigate the needs of co-located collaborators and to develop novel surface computing interfaces and interaction techniques to support these needs. She has organized workshops covering such topics as collaborative tabletop systems (ACM CSCW 2002), interactive walls and tabletops (ACM UBICOMP 2002), social theories and interactive surfaces (SurfNet Annual Workshop 2014), and multi-surface systems to support co-located collaboration (ACM CSCW 2015). She has twice served as a Program Co-Chair for the ACM ITS conference, regularly serves on its Program Committee, and currently serves on its Conference Steering Committee. In 2010, she co-directed an academic exchange program that enabled Canadian and European students to receive state-of-the-art international research training on interactive surfaces from 2010-2013.

**TEDDY SEYED** is a designer, developer and PhD Student at the University of Calgary. His primary research focus is designing new wearable technologies and interactions for multi-surface environments.

**PETER SIMONYI** recently graduated from Carleton University with a Master's degree in Computer Science. His main focus was on interaction history management and undo-like features that support people using software to explore multiple solutions. He has also made brief forays into other topics such as gesture recognition on shared touchscreens and language design.

**KALEV SIKES** was a SurfNet co-op intern from 2013-2014.

**ANTHONY TANG** is an Assistant Professor in the Department of Computer Science at the University of Calgary. He directs the RICELab (Rethinking Interaction, Collaboration and Engagement) research group. His research interests are situated in Computer Supported Cooperative Work (CSCW), the study of how people work together using technology—with a twist of Ubiquitous Computing technology—in everyday scenarios. His current research investigates the integration of mobile devices in large display environments, exploration into personal informatics, and telecommunication technologies for collaborative work.

**RICHARD TANG** is a member of the Interactions Lab and RICELab at the University of Calgary. He completed his BSc in Computer Science at the University of Calgary and stayed to do his MSc under Anthony Tang. He completed his MSc in 2015 after developing and writing his thesis on *Physio@Home*. Outside of research, he is a military history buff with a specific interest in armoured vehicles. He currently lives in Calgary playing video games, watching anime, and collecting mechanical keyboards and fountain pens.

**ALICE THUDT** is currently a PhD student in Computational Media Design. She is pursuing here research at the InnoVis Group at the University of Calgary, Canada under the supervision of Prof. Dr. Sheelagh Cpendale. She graduated in Media informatics at the University of Munich (LMU), Germany in 2012. Her main areas of interest are information visualization, human computer interaction and interaction design. In particular, here research focuses on personal visualization that supports individuals in using their personal data collections for reminiscing and helps them to

integrate their digital possessions into their everyday lives.

**JULIE TOURNET** completed her BSc in Engineering in Télécom Bretagne (France) and was offered the opportunity to spend a year in the Collaborative Systems Laboratory under the supervision of Dr. Stacey Scott at the University of Waterloo (Canada) as part of the transatlantic LEIF exchange program. Integrated to the SurfNet consortium, her work focused on the Surface Ghost Project, aiming to improve T-MSE's user awareness by providing them with a visual feedback of their actions on the shared surface. After the end of her exchange, she pursued her studies at the University of Waterloo with an MAsc in Electrical and Computer Engineering. She is now a Ph.D. candidate at the University of Montpellier in France and kept close contacts with the SurfNet community.

**JAGODA WALNY** is a PhD Candidate at the Innovations in Visualization (InnoVis) research group in the Interactions Lab (iLab) at the University of Calgary, supervised by Dr. Sheelagh Carpendale. Her research focuses on enabling interactive visual thinking in information visualization interfaces — both in terms of representation and interaction — to support people in better engaging with and understanding information.

**YUXI WANG** is a 4th-year undergraduate student in the Computer Science department at the University of Calgary. He is also a software developer with the Agile Surface Engineering lab led by Dr. Frank Maurer. Yuxi is enthusiastic about using computer technologies to solve problems. Javascript, C#, and Swift are Yuxi's most recently used programming language. Yuxi likes to discover things to do in life. Recently, Yuxi has discovered traveling is a fun thing to do, and he strives to save up for future travel plans.

**JEFF WILSON** is a recent graduate with a Masters in Computer Science from Carleton University and an undergraduate degree in Computer Science with a minor in Psychology (highest honors). Over the course of his studies he co-authored several publications on several topics relating to collaborative decision support using interactive visualizations on large surface multi-touch displays. Jeff's current focus is on employing the latest web-based resources to help individuals and teams overcome cognitive biases while making challenging decisions.

# REFERENCES

## HUMANIZING THE DIGITAL INTERFACE

- Agrawala, M. and Stolte, C. Rendering effective route maps: improving usability through generalization (2001). In Proceedings of SIGGRAPH, ACM.
- Anderson, P.A., Leibowitz, K. The Development and Nature of the Construct Touch Avoidance. *Nonverbal Behaviour*. 3(2). 1978. 89-106.
- Anslow, C., P. Campos, et al. (2014). Collaboration Meets Interactive Surfaces: Walls, Tables, Tablets, and Phones (CMIS). Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 495-498.
- Anslow, C., S. Marshall, et al. (2013). SourceVis: Collaborative software visualization for co-located environments. *Software Visualization (VISSOFT)*, 2013 First IEEE Working Conference on.
- Azazi, A., T. Seyed, et al. (2014). Investigating inertial measurement units for spatial awareness in multi-surface environments. Proceedings of the 2nd ACM symposium on Spatial user interaction. Honolulu, Hawaii, USA, ACM: 152-152.
- Bachl, S., Tomitsch, M., Kappel, K. and Grechenig, T. (2011). The Effects of Personal Displays and Transfer Techniques on Collaboration Strategies in Multi-touch Based Multi-Display Environments. Proceedings of the IFIP TC13 Conference on Human-Computer Interaction (INTERACT), Lisbon, Portugal, 373–390.
- Baglioni, M., Lecolinet, E. and Guiard, Y. JerkTilts: using accelerometers for eight-choice selection on mobile devices. *Proc. ICMI 2011*, 121-128.
- Bailenson, J.N., Blascovich, J., Beall, A.C., Loomis, J.M., Interpersonal Distance in Immersive Virtual Environments. *PSPB '03*. 29, 7. 819-833.
- Bailly, G., Lecolinet, E. and Nigay, L. Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. *Proc. AVI 2008*, 15-22.
- Bailly, G., Müller, J. and Lecolinet, E. Design and Evaluation of Finger-Count Interaction: Combining Multitouch gestures and Menus. *IJHCS 70(12):673-689*.
- Ballendat, T., Marquardt, N. and Greenberg, S. (2010) Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In Proceedings of the ACM Conference on Interactive Tabletops and Surfaces - ACM ITS'2010. (Saarbruecken, Germany), ACM Press, 10 pages, November 7-10.
- Banovic, N., Li, F., Dearman, D., Yatani, K. and Truong, K. Design of unimanual multi-finger pie menu interaction. *Proc. ITS 2011*, 120-129.
- Bau, O., and Mackay, W., OctoPocus: a dynamic guide for learning gesture-based command sets. *Proc. UIST 2008*, 37-46.
- Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B. and Zierlinger, A. (2003). Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch-and pen-operated systems. Proceedings of the IFIP TC13 Conference on Human-Computer Interaction (INTERACT), Zurich, Switzerland, 57–64.
- Besacier, G., Rey, G., Najm, M., Buisine, S. and Vernier, F. (2007). Paper metaphor for tabletop interaction design. J. Jacko (Ed.). *Human-Computer Interaction: Interaction Platforms and Techniques* Springer-Verlag Berlin Heidelberg, LNCS 4551, 758–767.

- Beyer, H. and Holtzblatt, K. (1998). Contextual design: defining customer-centered systems. San Francisco, CA, Morgan Kaufmann Publishers Inc.
- Bhaskar, R. K., J. Paredes, et al. (2014). VACI: Towards visual analytics for criminal investigation. Visual Analytics Science and Technology (VAST), 2014 IEEE Conference.
- Boring, S., S. Greenberg, et al. (2014). "The Dark Patterns of Proxemic Sensing." Computer 47(8): 56-60.
- Bortolaso, C., T. C. N. Graham, et al. (2014). From Personal Computers to Collaborative Digital Tabletops to Support Simulation Based-Training. Proceedings of ICCRTS 2014: 19th Annual International Command and Control Research and Technology Symposium, Alexandria, VA.
- Brosz, J., Nacenta, M. A., Pusch, R., Carpendale, S., Hurter, C. Transmogrification: Casual Manipulation of Visualizations (2013). In Proceedings of UIST, ACM.
- Brudy, F., D. Ledo, et al. (2014). Is Anyone Looking? Mitigating Shoulder Surfing on Public Displays through Awareness and Protection. Proceedings of The International Symposium on Pervasive Displays. Copenhagen, Denmark, ACM: 1-6.
- Carroll, J.M., Rosson, M. B. Paradox of the active user, Interfacing thought: cognitive aspects of human-computer interaction, MIT Press, Cambridge, MA, 1987.
- Carroll, J.M., Thomas, J.C. Metaphor and the Cognitive Representation of Computing Systems. IEEE Trans. on Systems, Man and Cybernetics. 12(2). 1982. 107-116.
- Chang, Y. L. B., S. D. Scott, et al. (2014). Supporting Situation Awareness in Collaborative Tabletop Systems with Automation. Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 185-194.
- Cheung, V. and S. D. Scott (2015a). A laboratory-based study methodology to investigate attraction power of large public interactive displays. Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing. Osaka, Japan, ACM: 1239-1250.
- Cheung, V. and S. D. Scott (2015b). Studying Attraction Power in Proxemics-Based Visual Concepts for Large Public Interactive Displays. Proceedings of ITS 2015: ACM Conference on Interactive Tabletops and Surfaces. Madeira, Portugal, ACM Press.
- Cheung, V., D. Watson, et al. (2014). Overcoming Interaction Barriers in Large Public Displays Using Personal Devices. Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 375-380.
- Chokshi, A., T. Seyed, et al. (2014). ePlan Multi-Surface: A Multi-Surface Environment for Emergency Response Planning Exercises. Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 219-228.
- Cockburn, A., Gutwin, C., Scarr, J., and Malacria, S, Supporting Novice to Expert Transitions in User Interfaces. ACM Computing Surveys 47, 2, Article 31 (November 2014), 36 pages.
- Cohen, J. Eta-squared and partial eta-squared in communication science. Human Communication Research 28(56):473-490.
- Collomb, M. and Hascoët, M. (2008). Extending drag-and-drop to new interactive environments: A multi-display, multi-instrument and multi-user approach. Interacting with Computers, 20(6), 562-573.



- Collomb, M., Hascoët, M., Baudisch, P. and Lee, B. (2005). Improving drag-and-drop on wall-size displays. *Proceedings of Graphics interface 2005*, 25–32.
- Craik, F., and Lockhart, R. 1972. Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior* 11, 671–684.
- Dandekar, K., Raju, B. and Srinivasan, M. 3-D finite-element models of human and monkey fingertips to investigate the mechanics of tactile sense. *Journal of Biomechanical Engineering*, 125, 5, 2003, 682-691.
- Dingler, T., M. Funk, et al. (2015). Interaction Proxemics: Combining Physical Spaces for Seamless Gesture Interaction. *Proceedings of the 4th International Symposium on Pervasive Displays*. Saarbruecken, Germany, ACM: 107-114.
- Diaz-Marino, R. and Greenberg, S. (2010) The Proximity Toolkit and ViconFace: The Video. *Proc. ACM CHI Extended Abstracts*, ACM.
- Doeweling, S. and Glaubitt, U. (2010). Drop-and-drag: easier drag & drop on large touchscreen displays. *Proceedings of the 6th Nordic Conference on Human-Computer Interaction*, 158–167.
- Doucette, A., R. L. Mandryk, et al. (2013). The effects of tactile feedback and movement alteration on interaction and awareness with digital embodiments. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Paris, France, ACM: 1891-1900.
- Fitts, P. M. and Posner, M. I. 1967. *Human Performance*. Belmont, CA, Brookes/Cole.
- Fei, S., Webb, A.M., Kerne, A., Qu, Y. and Jain, A. (2013). Peripheral array of tangible NFC tags: positioning portals for embodied trans-surface interaction. *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces*, St. Andrews, UK, 33-36.
- Fu, W. and Gray, W. Resolving the paradox of the active user: Stable suboptimal performance in interactive tasks. *Cognitive Science* 28, 6 (2004), 901–935.
- Genest, A. and C. Gutwin (2012). Evaluating the effectiveness of height visualizations for improving gestural communication at distributed tabletops. *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*. Seattle, Washington, USA, ACM: 519-528.
- Genest, A. M., C. Gutwin, et al. (2013). KinectArms: a toolkit for capturing and displaying arm embodiments in distributed tabletop groupware. *Proceedings of the ACM Conference on Computer Supported Cooperative Work*, ACM: 157–166.
- Gray, W., Sims, C., Fu, W. and Schoelles, M. The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. *Psychological Review* 113 (2006), 461--482.
- Greenberg, S., Boring, S., Vermeulen, J. and Dostal, J. (2014). Dark patterns in proxemic interactions: a critical perspective. *Proceedings of the 2014 conference on Designing interactive systems*. Vancouver, BC, Canada, ACM: 523-532.
- Greenberg, S. and Kuzuoka, H. (1999) Using digital but physical surrogates to mediate awareness, communication and privacy in media spaces. *Personal and Ubiquitous Computing* 3,4, 182-198.
- Greenberg, S., K. Hornbæk, et al. (2014). "Proxemics in Human-Computer Interaction (Report from Dagstuhl Seminar 13452)." *Dagstuhl Reports* 3(11): 29–57.
- Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R. and Wang, M. (2011) Proxemic Interactions: The New UbiComp?. *ACM Interactions*, 18(1):42-50. ACM,

January-February.

Greenberg, S., Marwood, D. Real Time Groupware as a Distributed System: Concurrency Control and its Effect on the Interface. CSCW '94. 207-217.

Gutwin, C., A. Cockburn, et al. (2015). Testing the rehearsal hypothesis with two FastTap interfaces. Proceedings of the 41st Graphics Interface Conference. Halifax, Nova Scotia, Canada, Canadian Information Processing Society: 223-231.

Ha, V., Inkpen, K.M., Mandryk, R.L., Whalen, T. Direct Intentions: The Effects of Input Devices on Collaboration around a Tabletop Display. TABLETOP 2006. 177-184.

Hall, E.T. The Hidden Dimension. Doubleday, Garden City, N.Y, 1966.

Haller, M., Leitner, J., Seifried, T., Wallace, J.R., Scott, S.D., Richter, C., Brandl, P., Gokcezaade, A. and Hunter, S. (2010). The NiCE discussion room: Integrating paper and digital media to support co-located group meetings. Proceedings of the international conference on Human factors in computing systems (CHI 2010), 609-618.

Hart, S. and Staveland, L. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. in Hancock, P. and Meshkati, N. eds. Human Mental Workload, 1988, 139-183.

Hajizadehgashti, S. (2012). Investigating the impact of table size on external cognition in collaborative problem-solving tabletop activities. Systems Design Engineering. Waterloo, Ontario, Canada, University of Waterloo.

Hascoët, M. (2003). Throwing models for large displays. Proceedings of the International Conference on Human Computer Interaction (HCI'03), Bath, UK, 73-77.

Haubner, N., Schwanecke, U., Dörner, R., Lehmann, S. and Luderschmidt, J. (2013). Detecting interaction above digital tabletops using a single depth camera. Machine Vision and Applications, 24(8), 1575-1587.

Hayduk, L.A. Personal Space: Where We Now Stand. Psychological Bulletin. 94(2). 1983. 293-335.

Henrik Soerensen and J. Kjeldskov (2013). Moving Beyond Weak Identifiers for Proxemic Interaction. Proceedings of International Conference on Advances in Mobile Computing and Multimedia. Vienna, Austria, ACM: 18-22.

Heslin, R, Nguyen, T.D., Nguyen, M.L. Meaning of touch: The case of touch from a stranger or same sex person. Nonverbal Behaviour. 7(3). 1983. 147-157.

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. Proceedings of ACM Symposium on User Interface Software and Technology, Victoria, BC, 139-148.

Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P. and Smith, M. (2004). Stitching: pen gestures that span multiple displays. Proceedings of the working conference on Advanced visual interfaces (AVI), Gallipoli, Italy, 23-31.

Hinrichs, U., S. Carpendale, et al. (2013). "Interactive Public Displays." Computer Graphics and Applications, IEEE 33(2): 25-27.

Hinrichs, U., N. Valkanova, et al. (2011). Large displays in urban life - from exhibition halls to media facades. CHI '11 Extended Abstracts on Human Factors in Computing Systems. Vancouver, BC, Canada, ACM: 2433-2436.

Hornecker, E., Marshall, P., Dalton, N.S., Rogers, Y. Collaboration and interference: awareness with mice or touch input. CSCW 2008. 167-176.

- Hsu, S. C., Lee, I. H. H., and Wiseman, N. E. Skeletal Strokes (1993). In Proceedings of UIST, ACM.
- Huron, S., Y. Jansen, et al. (2014). "Constructing Visual Representations: Investigating the Use of Tangible Tokens." *Visualization and Computer Graphics, IEEE Transactions on* 20(12): 2102-2111.
- IDC. IDC Forecasts Worldwide Tablet Shipments to Surpass Portable PC Shipments in 2013, [www.idc.com/getdoc.jsp?containerId=prUS24129713](http://www.idc.com/getdoc.jsp?containerId=prUS24129713), 2013.
- Isenberg, P., B. Lee, et al. (2015). Workshop on Data Exploration for Interactive Surfaces (DEXIS). Proceedings of ITS 2015: ACM International Conference on Interactive Tabletops and Surfaces. Madeira, Portugal.
- Ishii, H., Kobayashi, M., Grudin, J. Integration of interpersonal space and shared workspace: ClearBoard design and experiments. *T. Inf. Syst.* 11, 4, '93, 349-375.
- Izadi, S., Brignull, H., Rodden, T., Rogers, Y., Underwood, M., Dynamo: A public interactive surface supporting the cooperative sharing and exchange of media. *UIST '03.* 159-168.
- Jakobsen, M. R. and K. Hornbaek (2015). Is Moving Improving?: Some Effects of Locomotion in Wall-Display Interaction. Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. Seoul, Republic of Korea, ACM: 4169-4178.
- Jeffrey, P., Mark, G. Navigating the virtual landscape: co-ordinating the shared use of space. In *Social Navigation of Information Space*, 112-131.
- Jourard, S.M. An Exploratory Study of Body-Accessibility. *J. Soc. Clin. Psych.*, 5, 1966. 221-231.
- Major, B., Heslin, R. Perceptions of cross-sex and same-sex nonreciprocal touch: It is better to give than to receive. *Nonverbal Behaviour.* 6(3). 1982. 148-162.
- Ju, W., Lee, B.A., and Klemmer, S.R. Range: exploring implicit interaction through electronic whiteboard design. *Proc. ACM CHI (2008)*, 17-26.
- Kin, K., Hartmann, B., and Agrawala, M., Two-Handed Marking Menus for Multitouch Devices. *ToCHI 18(3):* article 16.
- Kobayashi, K., Narita, A., Hirano, M., Tanaka, K., Katada, T. and Kuwasawa, N. (2008). DIGTable: A Tabletop Simulation System for Disaster Education. Proceedings of the 6th International Conference on Pervasive Computing (Pervasive2008), 57-60.
- Kurtenbach, G. and Buxton, W. Issues of Combining Marking and Direct Manipulation Techniques. *Proc. UIST 1991*, 137-144.
- Kurtenbach, G. and Buxton, W. User Learning and Performance with Marking Menus. *Proc. CHI 1994*, 258-264.
- Kurtenbach, G., Fitzmaurice, G., Owen, R. and Baudel, T. The Hotbox: efficient access to a large number of menu-items. *Proc. CHI 1999*, 231-237.
- Kurtenbach, G., Sellen, A. and Buxton, W. An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus. *Human-Computer Interaction* 8(1):1-23.
- Kurtenbach, G.P. The design and evaluation of marking menus, PhD Thesis, University of Toronto, 1993.
- Kuzminykh, A., S. D. Scott, et al. (2015). How to Measure Social Presence: The Role of Speech Patterns. Poster Presentation at Graphics Interface 2015, Halifax, NS, Canada.

- Ledo, D., B. A. Aseniero, et al. (2013). OneSpace: shared depth-corrected video interaction. CHI '13 Extended Abstracts on Human Factors in Computing Systems. Paris, France, ACM: 997-1002.
- Ledo, D., S. Greenberg, et al. (2015). Proxemic-Aware Controls: Designing Remote Controls for Ubiquitous Computing Ecologies. Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services. Copenhagen, Denmark, ACM: 187-198.
- Ledo, D., Greenberg, S., Marquardt, N., Boring, B. (2015) Proxemic-Aware Controls: Designing Remote Controls for Ubiquitous Computing Ecologies. In Proceedings of MobileHCI 2015. ACM, New York, NY, USA.
- Lepinski, G., Grossman, T. and Fitzmaurice, G. The design and evaluation of multitouch marking menus Proc. CHI 2010, 2233-2242.
- Malacria, S., Bailly, G., Harrison, J., Cockburn, A., Gutwin, C. Promoting Hotkey use through rehearsal with ExposeHK, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, April 27-May 02, 2013, Paris, France
- Marquardt, N. (2013). Proxemic interactions with and around digital surfaces. Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces. St. Andrews, Scotland, United Kingdom, ACM: 493-494.
- Marquardt, N., Ballendat, T., Boring, S., Greenberg, S. and Hinckley, K. (2012) Gradual Engagement between Digital Devices as a Function of Proximity: From Awareness to Progressive Reveal to Information Transfer. In Proceedings of the ACM Conference on Interactive Tabletops and Surfaces - ACM ITS 2012. (Boston, MA).
- Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011). The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. Proceedings of the 24th annual ACM symposium on User interface software and technology. Santa Barbara, California, USA, ACM: 315-326.
- Marquardt, N. and S. Greenberg (2012). "Informing the Design of Proxemic Interactions." IEEE Pervasive Computing 11(2): 14-23.
- Marquardt, N. and Greenberg, S. (2015) Proxemic Interactions: From Theory to Practice. (Series: Synthesis Lectures on Human-Centered Informatics, John M. Carroll, Ed., Ed.) 199 pages Morgan-Claypool, February 24.
- Marquardt, N., K. Hinckley, et al. (2012). Cross-device interaction via micro-mobility and f-formations. Proceedings of the 25th annual ACM symposium on User interface software and technology. Cambridge, Massachusetts, USA, ACM: 13-22.
- McClelland, P.J. (2013). Bridging Private and Shared Interaction Surfaces in Collocated Groupware. M.A.Sc. Thesis, Systems Design Engineering, University of Waterloo, Waterloo, ON, Canada.
- Mckinlay, J. D., Robertson, G. G., and Card, S. K. The perspective wall: Detail and context smoothly integrated (1991). In Proceedings of CHI, ACM.
- Morris, M., Ryall, K., Shen, C., Forlines, C., Vernier, F. Beyond "Social Protocols": Multi-User Coordination Policies for Co-Located Groupware. CSCW '04. 262-265.
- Mostafa, A. E., S. Greenberg, et al. (2013). Interacting with microseismic visualizations. CHI '13 Extended Abstracts on Human Factors in Computing Systems. Paris, France, ACM: 1749-1754.
- Mueller, F., S. Stellmach, et al. (2014). Proxemics play: understanding proxemics for designing digital play experiences. Proceedings of the 2014 conference on Designing interactive systems. Vancouver, BC, Canada, ACM: 533-542.

- Nacenta, M.A., Aliakseyeu, D., Subramanian, S. and Gutwin, C. (2005). A comparison of techniques for multi-display reaching. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Portland, Oregon, USA, 371-380.
- Nacenta, M.A., Gutwin, C., Aliakseyeu, D. and Subramanian, S. (2009). There and Back Again: Cross-Display Object Movement in Multi-Display Environments. *Human-Computer Interaction*, 24(1-2), 170-229.
- Nacenta, M.A., Pinelle, D., Gutwin, C., Mandryk, R.L. Individual and Group Support in Tabletop Interaction Techniques. TABLETOP 2010, 303-333.
- Nacenta, M.A., Pinelle, D., Stuckel, D., Gutwin, C. The effects of interaction technique on coordination in tabletop groupware. *GI 2007*. 191-198.
- Nguyen, T., Heslin, R., Nguyen, M.L. The meanings of touch: Sex differences. *Comm.* '73. 25(3). 92-103.
- Oskamp, M., C. Bortolaso, et al. (2015). TerraGuide: Design and Evaluation of a Multi-Surface Environment for Terrain Visibility Analysis. Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. Seoul, Republic of Korea, ACM: 3553-3562.
- Parhi, P., Karlson, A. and Bederson, B. Target size study for one-handed thumb use on small touchscreen devices. *Proc. Mobile HCI 2003*, 203-210.
- Park, Y., Han, S., Park, J. and Cho, Y. Touch key design for target selection on a mobile phone *Proc. Mobile HCI 2008*, 423-426.
- Patterson, M.L., Powel, J.L., Lenihan, M.G. Touch, compliance, and interpersonal affect. *Nonverbal Behaviour*. 10(1). 1986. 41-50.
- Pauchet, A., Coldefy, F., Lefebvre, L., Louis dit Picard, S., Perron, L., Guérin, J. Tabletops: Worthwhile Experience of Collocated and Remote Collaboration. *Proc. TABLETOP. 2007*. 27-34.
- Pinelle, D., Nacenta, M., Gutwin, C., Stach, T. The effects of co-present embodiments on awareness and collaboration in tabletop groupware. *GI 2008*. 1-8.
- Rosenfeld, L.B., Kartus, S., Ray, C. Body Accessibility Revisited. *Communication*. 26(3). 1976. 27-30.
- Pryeskin, D., Hancock, M. and Hoey, J. (2012). Comparing elicited gestures to designer-created gestures for selection above a multitouch surface. Proceedings of ITS 2012: ACM International Conference on Interactive Tabletops and Surfaces, Cambridge, MA, 1-10.
- Raedle, R., H.-C. Jetter, et al. (2014). Demonstrating HuddleLamp: Spatially-Aware Mobile Displays for Ad-hoc Around-the-Table Collaboration. Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 435-438.
- Rekimoto, J. (1997). Pick-and-drop: a direct manipulation technique for multiple computer environments. Proceedings of the ACM Symposium on User interface software and technology (UIST), Banff, AB, 31-39.
- Rekimoto, J. (1998). A multiple device approach for supporting whiteboard-based interactions. Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems, Los Angeles, CA, 344-351.
- Rekimoto, J. and Saitoh, M. (1999). Augmented Surfaces: A spatially continuous workspace for hybrid computing environments. Proceedings of CHI'99: ACM Conference on Human Factors in Computing Systems, Pittsburgh, PA, pp. 378-385.

- Reynaga, G., S. Chiasson, et al. (2015). Exploring the Usability of CAPTCHAS on Smartphones: Comparisons and Recommendations. Proceedings of NDSS Workshop on Usable Security (USEC), Internet Society.
- Roy, Q., Malacria, S., Guiard, Y., Lecolinet, E. and Eagan, J. Augmented letters: mnemonic gesture-based shortcuts Proc. CHI 2013, 2325-2328.
- Scarr, J., Cockburn, A., Gutwin, C. and Bunt, A. Improving command selection with CommandMaps Proc. CHI 2012, 257-266.
- Scarr, J., Cockburn, A., Gutwin, C. and Malacria, S. Testing the Robustness and Performance of Spatially Consistent Interfaces. Proc. CHI 2013, 3139-3148.
- Scarr, J., Cockburn, A., Gutwin, C., Quinn, P. Dips and ceilings: understanding and supporting transitions to expertise in user interfaces, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, May 07-12, 2011, Vancouver, BC, Canada.
- Scott, S. D., G. Besacier, et al. (2014). Surface Ghosts: Promoting Awareness of Transferred Objects during Pick-and-Drop Transfer in Multi-Surface Environments. Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces. Dresden, Germany, ACM: 99-108.
- Scott, S.D., Besacier, G. and McClelland, P. (2014). Cross-Device Transfer in a Collaborative Multi-Surface Environment without User Identification. Proceedings of CTS 2014: International Conference on Collaboration Technologies and Systems, Minneapolis, MN, 219-226.
- Scott, S.D., Besacier, G., McClelland, P., Tournet, J., Goyal, N. and Cento, F. (2015). Cross-Device Content Transfer in Table-Centric Multi-Surface Environments. Technical Report, CSL2015-01, Department of Systems Design Engineering, University of Waterloo, Waterloo, ON.
- Scott, S.D. and Carpendale, S. (2010). Theory of Tabletop Territoriality. C. Müller-Tomfelde (Ed.). Tabletops - Horizontal Interactive Displays, Springer, 375-406.
- Scott, S.D., Carpendale, S.T., Inkpen, K.M. Territoriality in collaborative tabletop workspaces. CSCW '04. 294-303.
- Scott, S. D., T. C. N. Graham, et al. (2015). "Local Remote" Collaboration: Applying Remote Group Awareness Techniques to Co-located Settings. Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work & Social Computing. Vancouver, BC, Canada, ACM: 319-324.
- Scott, S.D., Mercier, S., Cummings, M.L. and Wang, E. (2006). Assisting Interruption Recovery in Supervisory Control of Multiple UAVs. HFES 2006: 50th Annual Meeting of the Human Factors and Ergonomic Society, San Francisco, CA, USA, 699-703.
- Schmidt, R. Frequent augmented feedback can degrade learning: Evidence and interpretations. Requin, J. ed., *Tutorials in motor neuroscience*, 1991, 59--75.
- Serrano, M., Lecolinet, E. and Guiard, Y. Bezel-Tap gestures: quick activation of commands from sleep mode on tablets Proc. CHI 2013, 3027-3036.
- Seto, M., S. Scott, et al. (2012). Investigating menu discoverability on a digital tabletop in a public setting. Proceedings of the 2012 ACM international conference on Interactive tabletops and surfaces. Cambridge, Massachusetts, USA, ACM: 71-80.
- Simon, H. Satisficing. in Eatwell, J., Millgate, M. and Newman, P. eds. *The New Palgrave: A Dictionary of Economics*, Stockton Press, 1987, 243--245.

- Simonyi, P. T. (2015). Ra: Better support for "what-if" collaboration. School of Computer Science. Ottawa, ON, Canada, Carleton University. Master of Computer Science.
- Slater, M., Steed, A. Meeting People Virtually: Experiments in Shared Virtual Environments. *The Social Life of Avatars: Presence and Interaction in Shared Virtual Environments*. 2002.
- Smith, M.A., Farnham, S.D., Drucker, S.M. The Social life of Small Graphical Chat Spaces. CHI '00. 462-469.
- Smallman, H.S. and St. John, M. (2003). CHEX (Change History EXplicit): New HCI Concepts for Change Awareness. Proceedings of HFES 2003: 46th Annual Meeting of the Human Factors and Ergonomics Society Santa Monica, CA, 528-532.
- Stephen, R., Zweigenhaft, R.L. The Effect on Tipping of a Waitress Touching Male and Female Customers. *Social Psychology*. 126(1). 1986. 141-142.
- Streitz, N.A., Tandler, P. and Müller-tomfelde, C. (2001). Roomware: Towards the Next Generation of Human-Computer Interaction Based on an Integrated Design of Real and Virtual Worlds. J. A. Carroll (Ed.). *Human-Computer Interaction in the New Millennium*, Addison-Wesley, 553-578.
- Tang, A., Neustaedter, C., Greenberg, S. VideoArms: Embodiments for Mixed Presence Groupware. *People and Computers*, in Proc of HCI '06. 2-18.
- Tang, A., Pahud, M., Inkpen, K., Benko, H., Tang, J.C., Buxton, B. Three's Company: Understanding Communication Channels in Three-way Distributed Collaboration. CSCW2010. 271-280.
- Tang, A., Tory, M., Po, B., Neumann, P., Carpendale, S. Collaborative Coupling over Tabletop Displays. CHI 2006. 1181-1190.
- Tang, J.C. Findings from Observational Studies of Collaborative Work. *JMMS*, 34(2). 1991. 143-160.
- Thayer, S. History and Strategies of Research on Social Touch. *Nonverbal Behaviour*. 10(1). 1986. 12-28.
- Thudt, A., U. Hinrichs, et al. (2012). The bohemian bookshelf: supporting serendipitous book discoveries through information visualization. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. Austin, Texas, USA, ACM: 1461-1470.
- Thudt, A., U. Hinrichs, et al. (2015). A Modular Approach to Promote Creativity and Inspiration in Search. Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition. Glasgow, United Kingdom, ACM: 245-254.
- Tufte, E. R. *Envisioning Information* (1990). Graphic Press.
- Tse, E., Histon, J., Scott, S.D., Greenberg, S. Avoiding interference: how people use spatial separation and partitioning in SDG workspaces. CSCW 2004. 252-261.
- Tuddenham, P. Robinson, P. Distributed Tabletops: Supporting Remote and Mixed-Presence Tabletop Collaboration. TABLETOP 2007. 19-26.
- Voelker, S., Weiss, M., Wacharamanatham, C. and Borchers, J. (2011). Dynamic portals: a lightweight metaphor for fast object transfer on interactive surfaces. Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, Kobe, Japan, 158-161.
- Vogel, D. and Balakrishnan, R. (2004). Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple

users. Proc. ACM UIST, 137-146.

Wagner, J., Huot, S., and Mackay, W. BiTouch and BiPad: Designing bimanual interactions for hand-held tablets. Proc CHI 2012, 2317-2326.

Wallace, J. R., J. Pape, et al. (2012). Exploring automation in digital tabletop board game. Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion. Seattle, Washington, USA, ACM: 231-234.

Wallace, J.R., Scott S.D., Lai, E., and Jajalla, D. (2011). Investigating the Role of a Large, Shared Display in Multi-Display Environments. Computer-Supported Cooperative Work (CSCW), 20(6), 529-561.

Wallace, J. R., S. D. Scott, et al. (2013). Collaborative sensemaking on a digital tabletop and personal tablets: prioritization, comparisons, and tableaux. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. Paris, France, ACM: 3345-3354.

Wallace, J.R., Scott, S.D., Stutz, T., Enns, T. and Inkpen, K. (2009). Investigating teamwork and taskwork in single- and multi-display groupware systems. Personal Ubiquitous Computing, 13(8), 569-581.

Wang, M., Boring, S. and Greenberg, S. (2012) Proxemic Peddler: A Public Advertising Display that Captures and Preserves the Attention of a Passerby. In Proceedings of the International Symposium on Pervasive Displays. (Porto, Portugal), ACM Press, 10 pages, June 4 - 5.

Weiser, M., 1991. The Computer for the 21st Century. Scientific American 265, 94-104.

Wu, M., and Balakrishnan, R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. Proc. UIST 2003, 193-202.

Yee, N., Bailenson, J., Urbanek, B., Chang, F., Merget, D. The Unbearable Likeness of Being Digital: The Persistence of Nonverbal Social Norms in Online Virtual Environments. Cyberpsychology & Behaviour. 10(1). 2007. 115-121.

Zhao, S., Agrawala, M. and Hinckley, K. Zone and Polygon Menus: Using Relative Position to Increase the Breadth of Multi-Stroke Marking Menus. Proc. CHI 2006, 1077-1086.

Zhou, H., V. Ferreira, et al. (2015). "Exploring Privacy Notification and Control Mechanisms for Proximity-Aware Tablets." Int. J. Mob. Hum. Comput. Interact. 7(3): 1-19.

## **IMPROVING SOFTWARE TIME TO MARKET**

Alcantara, T. d. S., Denzinger, J., Ferreira, J., & Maurer, F. (2012). Learning gestures for interacting with low-fidelity prototypes. Paper presented at the Proceedings of the First International Workshop on Realizing AI Synergies in Software Engineering.

Alter, Adam L. and Daniel M. Oppenheimer. 2009. "Uniting the Tribes of Fluency to Form a Metacognitive Nation." Personality and social psychology review : an official journal of the Society for Personality and Social Psychology, Inc 13(3):219-35. Retrieved October 19, 2015.

Arias-Hernandez, Richard, Linda T. Kaastra, Tera Marie Green, And Brian Fisher,



2011. Pair Analytics: Capturing Reasoning Process In Collaborative Visual Analytics. Volume System Sciences (Hicss), 44th Hawaii International Conference, IEEE, Pp. 1-10.
- Arnheim, R. (1980). A plea for visual thinking. *Critical Inquiry*, 6(3), 489–497.
- Axure RP: Interactive wireframe software and mockup tool. Retrieved from <http://www.axure.com/>
- Bailey, B. P., Konstan, J. A., & Carlis, J. V. (2001). DEMAIS: designing multimedia applications with interactive storyboards. Paper presented at the Proceedings of the ninth ACM international conference on Multimedia.
- Balsamiq. Retrieved from [www.balsamiq.com](http://www.balsamiq.com).
- Barnum C. Usability Testing and Research [Book]. - New York : Pearson Education, 2002.
- Bayer, J., Gacek, C., Muthig, D., and Widen, T., "PuLSE-I: Deriving Instances from a Product Line Infrastructure", Proceedings of the 7th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems, 2000, pp. 237 - 245.
- Bernsen Niels Ole, Dybkjær Hans and Dybkjær Laila Wizard of Oz Prototyping: How and When [Journal] // CCI Working Papers in Cognitive Science and HCI. - 1994.
- Bertin, J., 1981. Graphics And Graphic Information Processing. Volume Degruyter Press, Berlin.
- Bier, E. A., Stone, M. C., Pier, K., Buxton, W., & DeRose, T. D. (1993). Toolglass and magic lenses: The see-through interface. SIGGRAPH '93 Proceedings of the 20th annual conference on Computer graphics and interactive techniques (pp. 73–80). New York, New York, USA: ACM.
- Bostock, Michael, Vadim Ogievetsky, And Jeffrey Heer, 2011. D<sup>3</sup> Data Driven Documents. Volume Visualization And Computer Graphics, IEEE Transactions On 17, No. 12., Pp. 2301-2309.
- Brandl, P., Haller, M., Oberngruber, J., & Schafleitner, C. (2008). Bridging the gap between real printouts and digital whiteboard (pp. 31–38). In Proceedings of the working conference on Advanced visual interfaces.
- Branham, S., Golovchinsky, G., Carter, S., & Biehl, J. T. (2010). Let's go from the whiteboard: supporting transitions in work through whiteboard capture and reuse (pp. 75–84). In Proceedings of the 28th international conference on Human factors in computing systems.
- Brosterman, N., K. Togashi, and E. Himmel. 1997. "Inventing Kindergarten." Retrieved June 12, 2013 (<http://www.getcited.org/pub/100128705>).
- Browne, J., Lee, B., Carpendale, S., Riche, N., & Sherwood, T. (2011). Data analysis on interactive whiteboards through sketch-based interaction (p. 154). In ITS '11: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, New York, New York, USA: ACM.
- Buhrdorf, R., Churchett, D., and Krueger, C., Salion's Experience with a Reactive Software Product Line Approach, Revised Papers of the 5th International Workshop, PFE 2003, Siena, Italy, November 4-6, 2003.
- Buxton Bill Sketching User Experiences [Book]. - [s.l.] : Morgan Kaufmann, 2007.
- Buxton, B. (2010). Sketching user experiences: getting the design right and the right design: getting the design right and the right design: Morgan Kaufmann.

- Cairo, Alberto. 2014. "The Island of Knowledge and the Shorelines of Wonder." Keynote IEEE VIS 2014. Retrieved October 19, 2015 (<http://ieevis.org/year/2014/info/overview-amp-topics/keynote-and-capstone>).
- Carbon, R., Lindvall, M., Muthig, D., and Costa, P., Integrating PLE and agile methods: flexible design up-front vs. incremental design", The 1st International Workshop on APLE, 2006 - SPLC.
- Card, S. K., Mackinlay, J. D. & Shneiderman, B., 1999. Readings In Information Visualization : Using Vision To Think. Volume Morgan-Kaufmann: San.
- Chapman, Michael. 1988. Constructive Evolution: Origins and Development of Piaget's Thought. Cambridge University Press. Retrieved August 23, 2013 (<http://books.google.com/books?hl=en&lr=&id=7WgCnXmdX1MC&pgis=1>).
- Cheema, S., Gulwani, S., & LaViola, J. (2012). QuickDraw: improving drawing experience for geometric diagrams (p. 1037). In CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: ACM.
- Cherubini, M., Venolia, G., DeLine, R., & Ko, A. J. (2007). Let's go to the whiteboard: how and why software developers use drawings (pp. 557–566). In CHI '07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, New York, USA: ACM.
- Clegg, K., Kelly, T., McDermid, J., Incremental Product-Line Development, International Workshop on PLE, Seattle, 2002.
- Clements, P., and Northrop, L., Software Product Lines: Practice and Patterns, Addison-Wesley, 2001.
- Constantine, L. (2004). Beyond user-centered design and user experience: Designing for user performance. Cutter IT Journal, 17(2), 16-25.
- Cooper, K., and Franch, X., "APLE 1st International Workshop on Agile Product Line Engineering", SPLC, 2006.
- Chul Kwon, Bum, Brian Fisher, And Ji Soo Yi., 2011. Visual Analytics Roadblocks For Novice Investigators. Volume Visual Analytics Science And Technology (Vast). Ieee Conference On. Ieee., Pp. 3-11.
- Cockburn, Alistair, And Laurie Williams., 2000. The Costs And Benefits Of Pair Programming.. Volume Extreme Programming Examined, Pp. 223-247.
- de Souza Alcantara, T. (2013). Designing Tabletop and Surface Applications Using Interactive Prototypes.
- de Souza Alcantara, T., Ferreira, J., & Maurer, F. (2013). Interactive prototyping of tabletop and surface applications. Paper presented at the Proceedings of the 5th ACM SIGCHI symposium on Engineering interactive computing systems.
- Derboven, J., De Roeck, D., Verstraete, M., Geerts, D., & De Grooff, D. (2010). Low-Fidelity Prototyping for Multi-Touch Surfaces. status: published.
- Dix, A., 1998. Evaluation Techniques. Volume Human-Computer Interaction. Europe, Prentice-Hall, Pp. 405-442.
- Dreamweaver CS4. Retrieved from <http://tv.adobe.com/show/learn-dreamweaver-cs4/>.
- ForeUI: Easy to use UI prototyping tool. . Retrieved from <http://www.foreui.com/>
- Forlines, Clifton, And Chia Shen, 2005. Dtlens: Multi - User Tabletop Spatial Data Exploration. Volume Proceedings Of The 18th Annual Acm Symposium On User

Interface Software And Technology. *Acm.*, Pp. 119-122.

Forlines, Clifton, And Ryan Lilien., 2008. Adapting A Single-User, Single-Display Molecular Visualization Application For Use In A Multi-User Multi-Display Environment. Volume Proceedings Of The Working Conference On Advanced Visual Interfaces. *Acm.*, Pp. 367-371.

Gerken, J., Jetter, H.-C., Schmidt, T., & Reiterer, H. (2010). Can touch get annoying? Paper presented at the ACM International Conference on Interactive Tabletops and Surfaces.

GestureWorks, a multitouch application framework for Adobe Flash and Flex. Retrieved from <http://gestureworks.com/>.

Ghanam, Y., and Maurer, F., Linking Feature Models to Code Artifacts using Executable Acceptance Tests, to appear in the proceedings of the 14th International Software Product Line Conference (SPLC 2010), South Korea, September 2010.

Gammel, Lars, Chris Bennett, Melanie Tory, and Margaret-anne Storey. 2013. "A Survey of Visualization Construction User Interfaces." [webhome.cs.uvic.ca 1–5](http://webhome.cs.uvic.ca/~mtory/publications/VisToolSurvey-eurovis13-short.pdf). Retrieved June 27, 2013 (<http://webhome.cs.uvic.ca/~mtory/publications/VisToolSurvey-eurovis13-short.pdf>).

Gammel, Lars, Melanie Tory, and Margaret-Anne Storey. 2010. "How Information Visualization Novices Construct Visualizations." *IEEE transactions on visualization and computer graphics* 16(6):943–52. Retrieved September 18, 2013 (<http://www.ncbi.nlm.nih.gov/pubmed/20975131>).

Hanssen, G., and Fægri, T., Process Fusion: An Industrial Case Study on Agile Software Product Line Engineering, special Issue of *Journal of Systems and Software (JSS)*, 2008.

Heer, Jeffrey, Stuart K. Card, And James A. Landay., 2005. Prefuse: A Toolkit For Interactive Information Visualization. Volume In Proceedings Of The Sigchi Conference On Human Factors In Computing Systems. *Acm.*, Pp. 421-430.

Heer, J. Et Al., 2008. Creation And Collaboration: Engaging New Audiences For Information Visualization.. Volume In *Information Visualization*. Springer Berlin Heidelberg., Pp. 92-133.

Heer, J. and B. Shneiderman. 2012. "Interactive Dynamics for Visual Analysis." 1–26. Retrieved July 3, 2013 (<http://dl.acm.org/citation.cfm?id=2146416>).

Hellmann T.D. [et al.] An Exploratory Study of Automated GUI Testing: Goals, Issues, and Best Practices [Report] : Technical Report. - Calgary : University of Calgary, 2014. - 2014-1057-08.

Hellmann Theodore D. Automated GUI Testing for Agile Development Environments [Report] : PhD Thesis. - Calgary : University of Calgary, 2015.

Hellmann Theodore D., Hosseini-Khayat Ali and Maurer Frank Supporting Test-Driven Development of Graphical User Interfaces using Agile Interaction Design [Conference] International Workshop on Test-Driven Development . Paris: [s.n.], 2010.

Hellmann Theodore D., Hosseini-Khayat Ali and Maurer Frank Test-Driven Development of Graphical User Interfaces: A Pilot Evaluation [Conference] // *Agile Processes and eXtreme Programming*. Madrid: [s.n.], 2011.

Hesselmann, T., Boll, S., & Heuten, W. (2011). SCIVA: designing applications for surface computers. Paper presented at the Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems.

- Hinrichs, U., & Carpendale, S. (2011). Gestures in the wild: studying multi-touch gesture sequences on interactive tabletop exhibits. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.
- Holzmann, C., & Vogler, M. (2012). Building interactive prototypes of mobile user interfaces with a digital pen. Paper presented at the Proceedings of the 10th asia pacific conference on Computer human interaction.
- Hosseini-Khayat Ali Distributed Wizard of Oz Usability Testing for Agile Teams [Report] : Master's Thesis / University of Calgary. - Calgary : [s.n.], 2010.
- Hosseini-Khayat Ali, Hellmann Theodore D. and Maurer Frank Distributed and Automated Usability Testing of Low-Fidelity Prototypes [Conference] // Agile Conference. - Orlando : [s.n.], 2010.
- Huron, Samuel, Sheelagh Carpendale, Alice Thudt, Anthony Tang, and Michael Maurer. 2014. "Constructive Visualization." Pp. 433–42 in Proceedings of the 2014 conference on Designing interactive systems - DIS '14. New York, New York, USA: ACM Press. Retrieved October 19, 2015 (<http://dl.acm.org/citation.cfm?id=2598510.2598566>).
- Huron, Samuel, Yvonne Jansen, and Sheelagh Carpendale. 2014. "Constructing Visual Representations: Investigating the Use of Tangible Tokens." IEEE transactions on visualization and computer graphics 20(12):2102–11. Retrieved October 19, 2015 (<http://www.ncbi.nlm.nih.gov/pubmed/26356924>).
- iPlotz: Wireframes, mockups and prototyping for websites. Retrieved from <http://iplotz.com/>.
- Isenberg, Petra, And Carpendale, Sheelagh, 2007. Interactive Tree Comparison For Co-Located Collaborative Information Visualization. Volume Visualization And Computer Graphics, IEEE Transactions On 13, No. 6., Pp. 1232-1239.
- Isenberg, Petra, And Danyel Fisher., 2009. Collaborative Brushing And Linking For Co-Located Visual Analytics Of Document Collections. Volume Computer Graphics Forum. Vol. 28. No. 3. Blackwell Publishing Ltd., Pp. 1031-1038.
- Isenberg, P. Et Al., 2011. Collaborative Visualization: Definition, Challenges, And Research Agenda. Volume Information Visualization 10, No. 4., Pp. 310-326.
- Isenberg, P. Et Al., 2010. An Exploratory Study Of Co-Located Collaborative Visual Analytics Around A Tabletop Display.. Volume In Visual Analytics Science And Technology (Vast), IEEE Symposium, Pp. 179-186.
- Jansen, W. 2009. "Neurath, Arntz and ISOTYPE: The Legacy in Art, Design and Statistics." Journal of Design History 22(3):227–42. Retrieved January 21, 2013 (<http://jdh.oxfordjournals.org/cgi/doi/10.1093/jdh/epp015>).
- Jansen, Yvonne and Pierre Dragicevic. 2013. "An Interaction Model for Visualizations Beyond The Desktop." IEEE Transactions on Visualization and Computer Graphics 19(12):2396–2405. Retrieved (<http://hal.inria.fr/hal-00847218>).
- Jeffries, Robin and Desurvire, Heather. Usability Testing vs. Heuristic Evaluation: Was There A Contest? [Journal] ACM SIGCHI Bulletin. - 1992. - 4 : Vol. 24. - pp. 39-41.
- Jeffries, Ron and Melnik, Grigori. Guest Editors' Introduction: TDD - The Art of Fearless Programming [Journal] IEEE Software. - May-June 2007. - 3 : Vol. 24. - pp. 24-30.
- Chris Johnson, Robert Moorhead, Tamara Munzner, Hanspeter Pfister, Penny Rheingans, Terry S. Yoo. 2006. NIH/NSF Visualization Research Challenges Report. Retrieved October 19, 2015 (<http://citeseerx.ist.psu.edu/viewdoc/>)

summary?doi=10.1.1.72.5285).

Ju, W., Lee, B. A., & Klemmer, S. R. (2008). Range: exploring implicit interaction through electronic whiteboard design (pp. 17–26). In Proceedings of the ACM 2008 conference on Computer supported cooperative work.

Kang, K., Cohen, S., Hess, J., Novak, W., and Peterson, A., FODA Feasibility Study, SEI Technical Report, 1990.

Khandkar, S. H., & Maurer, F. (2010). A domain specific language to define gestures for multi-touch applications. Paper presented at the Proceedings of the 10th Workshop on Domain-Specific Modeling.

Kin, K., Hartmann, B., DeRose, T., & Agrawala, M. (2012). Proton: multitouch gestures as regular expressions. Paper presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.

Kirsh, D. (2010). Thinking with external representations. *AI & Society*, 25(4), 441–454.

Kolodner, Janet L., David Crismond, Jackie Gray, Jennifer Holbrook, And Sadhana Puntambekar, 1998. Learning By Design From Theory To Practice. Volume In Proceedings Of The International Conference Of The Learning Sciences.

Krippendorff, K. (2006). The semantic turn: a new foundation for design. *ARTIFACT-ROUTLEDGE-*, 1(11), 51.

Kruger, C., Easing the Transition to Software Mass Customization, in Proceedings of the 4th International Workshop on Product Family Engineering, Germany, 2002.

Landay, J. A. (1996). SILK: sketching interfaces like crazy (pp. 398–399). In CHI '96: Conference companion on Human factors in computing systems, ACM.

Lao, S., Heng, X., Zhang, G., Ling, Y., & Wang, P. (2009). A gestural interaction design model for multi-touch displays. Paper presented at the Proceedings of the 23rd British HCI Group Annual Conference on People and Computers: Celebrating People and Technology.

Lee, B., Kazi, R. H., & Smith, G. (2013). SketchStory: Telling More Engaging Stories with Data through Freeform Sketching. *IEEE Transactions on Visualization and Computer Graphics*, 19(12).

Lee, B., Smith, G., Riche, N. H., & Karlson, A. (2015). SketchInsight: Natural data exploration on interactive whiteboards leveraging pen and touch interaction. *IEEE Symposium on InfoVis*, 199–206.

Leffingwell, D., *Scaling Software Agility: Best Practices for Large Enterprises*, Addison-Wesley Professional, 1st edition, 2007.

Lim, Y.-K., Stolterman, E., & Tenenberg, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 15(2), 7.

Lin, J. (1999). A visual language for a sketch-based UI prototyping tool. Paper presented at the CHI'99 extended abstracts on Human factors in computing systems.

Long Jr, A. C., Landay, J. A., & Rowe, L. A. (1999). Implications for a gesture design tool. Paper presented at the Proceedings of the SIGCHI conference on Human Factors in Computing Systems.

Lloyd, David, And Jason Dykes, 2011. Human-Centred Approaches In Geovisualization Design: Investigating Multiple Methods Through A Long-Term Case Study. Volume Visualization And Computer Graphics, *Ieee Transactions On* 17, No. 12, Pp. 2498-2507.

- Lyons, K., Brashear, H., Westeyn, T., Kim, J. S., & Starner, T. (2007). Gart: The gesture and activity recognition toolkit Human-Computer Interaction. *HCI Intelligent Multimodal Interaction Environments* (pp. 718-727): Springer.
- Mackinlay, J. D., Hanrahan, P. & Stolte , C., 2007. Show Me: Automatic Presentation For Visual Analysis. Volume *Visualization And Computer Graphics*, IEEE Transactions On 13, No. 6., Pp. 1137-1144.
- Mangano, N., LaToza, T. D., Petre, M., & van der Hoek, A. (2014). Supporting Informal Design with Interactive Whiteboards. In *CHI '14 Proceedings of The 32nd Annual ACM SIGCHI Conference on Human Factors in Computing Systems*, 1–10.
- Manning, JP. 2005. "Rediscovering Froebel: A Call to Re-Examine His Life & Gifts." *Early Childhood Education Journal*. Retrieved June 12, 2013 (<http://link.springer.com/article/10.1007/s10643-005-0004-8>).
- Mark, G., Kobsa, A. & Gonzalez, V., 2002. Do Four Eyes See Better Than Two? Collaborative Versus Individual Discovery In Data Visualization Systems. Volume *Information Visualisation*. Proceedings. Sixth International Conference On. IEEE., Pp. 249-255.
- McCurdy, M., Connors, C., Pyrzak, G., Kanefsky, B., & Vera, A. (2006). Breaking the fidelity barrier: an examination of our current characterization of prototypes and an example of a mixed-fidelity success. Paper presented at the Proceedings of the SIGCHI conference on Human Factors in computing systems.
- McGregor, J., *Agile Software Product Lines, Deconstructed*, Journal of Object Technology, 7(8), 2008.
- Memmel, T., Gundelsweiler, F., & Reiterer, H. (2007). Agile human-centered software engineering. Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI... but not as we know it-Volume 1.
- Microsoft Sketchflow. Retrieved from [http://www.microsoft.com/expression/products/sketchflow\\_overview.aspx](http://www.microsoft.com/expression/products/sketchflow_overview.aspx).
- Microsoft Surface User Experience Guidelines. Retrieved from <http://www.microsoft.com/en-ca/download/confirmation.aspx?id=19410>.
- Mockingbird: Wireframes on the fly. Retrieved from <https://gomockingbird.com/>.
- Moggridge, B., & Atkinson, B. (2007). *Designing interactions* (Vol. 14): MIT press Cambridge.
- Moran, T. P., & Van Melle, W. (2000). Tivoli: Integrating structured domain objects into a freeform whiteboard environment (p. 21). In *CHI'00 extended abstracts on Human factors in computing systems*.
- Morris, M. R., Wobbrock, J. O., & Wilson, A. D. (2010). Understanding users' preferences for surface gestures. Paper presented at the Proceedings of graphics interface 2010.
- Mynatt, E. D. (1999). The writing on the wall. In *Proceedings of the 7th IFIP Conference on Human-Computer Interaction*.
- Mynatt, E. D., Igarashi, T., Edwards, W. K., & LaMarca, A. (1999). Flatland: new dimensions in office whiteboards (pp. 346–353). In *CHI '99: Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, New York, New York, USA: ACM. <http://doi.org/10.1145/302979.303108>.
- Neurath, Marie. 2009. *The Transformer: Principles of Making Isotype Charts*. Princeton Architectural Press. Retrieved September 16, 2013 (<http://books.google>).

com/books?id=k6lmNAAACAAJ&pgis=1).

Neurath, Otto and Red Vienna. 2009. "Isotype Representing Social Facts Pictorially." (June).

Norman, D. A. (2007). *The design of future things: Author of the design of everyday things*.

Norman, D. A., & Nielsen, J. (2010). Gestural interfaces: a step backward in usability. *Interactions*, 17(5), 46-49.

North, C., Dwyer, T., Lee, B., Fisher, D., Isenberg, P., Robertson, G., & Inkpen, K. (2009). Understanding multi-touch manipulation for surface computing Human-Computer Interaction–INTERACT 2009 (pp. 236-249): Springer.

Novak, Joseph D., And Alberto J. Cañas., 2008. *The Theory Of Underlying Concept Maps And How To Construct And Use Them*. Volume Florida Institute For Human And Machine Cognition Pensacola FL, Www. Ihmc. Us., P. 284.

O'Brien, Z., & Martens, J.-B. (2011). Sketching interactive systems with sketchify. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 18(1), 4.

Papert, S. and I. Harel. 1991. "Situating Constructionism." *Constructionism*. Retrieved June 17, 2013 (<http://namodemello.com.br/pdf/tendencias/situatingconstructionism.pdf>).

Paige, R., Xiaochen, W., Stephenson, Z., and Phillip J., *Towards an Agile Process for Building Software Product Lines*, XP 2006, 198 – 199.

Pencil: Add-on for Mozilla Firefox. Retrieved from <https://addons.mozilla.org/en-US/firefox/addon/pencil/>.

Perin, Charles, Pierre Dragicevic, and Jean-Daniel Fekete. 2014. "Revisiting Bertin Matrices: New Interactions for Crafting Tabular Visualizations." *IEEE Transactions on Visualization and Computer Graphics* 20(12):2082–91. Retrieved October 19, 2015 (<https://hal.inria.fr/hal-01023890>).

Piaget, Jean. 1989. *Six études de Psychologie*. Ed. Deno[ë].

Plimmer, B., Blagojevic, R., Chang, S. H.-H., Schmieder, P., & Zhen, J. S. (2012). RATA: codeless generation of gesture recognizers. Paper presented at the Proceedings of the 26th Annual BCS Interaction Specialist Group Conference on People and Computers.

Pretorius, J. A. & Wijk, V. J., 2009. What Does The User Want To See? What Do The Data Want To Be?. *Volume Information Visualization* 8, No. 3, Pp. 153-166.

Price, Richard H., and Dennis L. Bouffard. "Behavioral appropriateness and situational constraint as dimensions of social behavior." *Journal of Personality and Social Psychology* 30.4 (1974): 579.

Proto.io : Silly-fast mobile prototyping. Retrieved from <http://proto.io/>.

Reas, Casey and Ben Fry. 2007. *Processing: A Programming Handbook for Visual Designers and Artists*. Retrieved October 19, 2015 (<https://books.google.com/books?hl=fr&lr=&id=tqW75bfJkxIC&pgis=1>).

Reas, C. & Fry, B., 2014. *Processing: A Programming Handbook For Visual Designers And Artists*. S.L.:Mit Press.

Robertson, S., & Robertson, J. (2012). *Mastering the requirements process: Getting requirements right*: Addison-wesley.

- Rudd, J., Stern, K., & Isensee, S. (1996). Low vs. high-fidelity prototyping debate. *interactions*, 3(1), 76-85.
- Schmid, K., and Verlage, M., The Economic Impact of Product Line Adoption and Evolution, *IEEE Software*, 19 (4), pp. 50-57, 2002.
- Sedlmair, M., Meyer, M. & Munzner, T., 2012. Design Study Methodology: Reflections From The Trenches And The Stacks. Volume Visualization And Computer Graphics, *IEEE Transactions On* 18, No. 12, Pp. 2431-2440.
- Sefelin, R., Tscheligi, M., & Giller, V. (2003). Paper prototyping-what is it good for?: a comparison of paper-and computer-based low-fidelity prototyping. Paper presented at the CHI'03 extended abstracts on Human factors in computing systems.
- Segura, V. C., Barbosa, S. D., & Simões, F. P. (2012). UISKEI: a sketch-based prototyping tool for defining and evaluating user interface behavior. Paper presented at the Proceedings of the International Working Conference on Advanced Visual Interfaces.
- Sinha, A. K., & Landay, J. A. (2003). Capturing user tests in a multimodal, multidevice informal prototyping tool. Paper presented at the Proceedings of the 5th international conference on Multimodal interfaces.
- Shalloway, A., Beaver, G., and Trott, J., *Lean-Agile Software Development: Achieving Enterprise Agility*, Addison-Wesley Professional, 1st edition, 2009.
- Smith, J. D., & Graham, T. (2010). Raptor: sketching games with a tabletop computer. Paper presented at the Proceedings of the International Academic Conference on the Future of Game Design and Technology.
- Statistics, C., 2013. Canada's Disaster Data. [Online] Available At: [Http://Www.Statcan.Gc.Ca/Start-Debut-Eng.Html](http://www.Statcan.Gc.Ca/Start-Debut-Eng.Html)
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research: Grounded theory procedures and techniques* (2nd ed.). Newbury Park, CA: Sage publications.
- Stuart K. Card, Jock D. Mackinlay, Ben Shneiderman. 1999. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann. Retrieved September 18, 2013.
- Stusak, S., 2009. Collaboration In Information Visualization. Volume Trends In Information Visualization., P. 46.
- Sutherland, I. E., & Sketchpad, A. (1963). A man-machine graphical communication system (Vol. 23, pp. 329–346). In *Proc. of AFIPS Spring Joint Comp. Conf.*
- T.I.B.C.O, 2014. *Spotfire-Business Intelligence Analytics Software & Data Visualization*. [Online] Available At: [Http://Www.Http://Spotfire.Tibco.Com](http://www.Http://Spotfire.Tibco.Com) [Accessed 20 11 2013].
- Tang, A. Et Al., 2006. Collaborative Coupling Over Tabletop Displays. Volume In Proceedings Of The Sigchi Conference On Human Factors In Computing Systems., Pp. 1181-1190.
- Tang, A., Lanir, J., Greenberg, S., & Fels, S. (2009). Supporting transitions in work: informing large display application design by understanding whiteboard use. In *Proceedings of the ACM 2009 international conference on Supporting group work* (pp. 149-158). ACM.
- Tobiasz, Matthew, Petra Isenberg, And Sheelagh Carpendale., 2009. Lark: Coordinating Co-Located Collaboration Wih Information Visualization. Volume Visualization And Computer Graphics, *IEEE Transactions On* 15, No. 6., Pp. 1065-



1072.

Tufte, Edward R., And P. R. Graves-Morris., 1983. *The Visual Display Of Quantitative Information*. S.L.:Vol. 2. Cheshire, Ct: Graphics Press.

Tversky, B. (2008). Making thought visible In *Proceedings of the International Workshop on Studying Design Creativity*. The Netherlands: Springer (Vol. 1, No. 2.1, pp. 2-2).

Unger, R., & Chandler, C. (2012). *A Project Guide to UX Design: For user experience designers in the field or in the making*: New Riders.

Van den Bergh, J., Sahni, D., Haesen, M., Luyten, K., & Coninx, K. (2011). GRIP: get better results from interactive prototypes. Paper presented at the *Proceedings of the 3rd ACM SIGCHI symposium on Engineering interactive computing systems*.

Van Wijk, J. J., 2006. Bridging The Gaps. *Volume Computer Graphics And Applications*, IEEE 26, No. 6., Pp. 6-9.

Victor, Brett. 2013. "Drawing Dynamic Visualizations." Retrieved (<http://worrydream.com/#!/DrawingDynamicVisualizationsTalk>).

Virzi, R. A., Sokolov, J. L., & Karis, D. (1996). Usability problem identification using both low-and high-fidelity prototypes. Paper presented at the *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

Walny, J., Bongshin Lee, P. Johns, N. H. Riche, and S. Carpendale. 2012. "Understanding Pen and Touch Interaction for Data Exploration on Interactive Whiteboards." *Visualization and Computer Graphics*, IEEE Transactions on 18(12):2779–88.

Walny, J., Carpendale, S., Riche, N. H., Venolia, G., & Fawcett, P. (2011). Visual Thinking In Action: Visualizations As Used On Whiteboards. *Visualization and Computer Graphics*, IEEE Transactions on, 17(12), 2508–2517.

Walny, J., Haber, J., Dörk, M., Sillito, J., & Carpendale, S. (2011). Follow that sketch: Lifecycles of diagrams and sketches in software development (pp. 1–8). IEEE.

Walny, J., Huron, S., & Carpendale, S. (2015). An Exploratory Study of Data Sketching for Visual Representation. *Comput. Graph. Forum* (), 34(3), 231–240. <http://doi.org/10.1111/cgf.12635>

Walny, J., Lee, B., Johns, P., & Riche, N. H. (2012). Understanding pen and touch interaction for data exploration on interactive whiteboards. *IEEE Trans Vis Comput Graph*, 18(12), 2779–2788.

Wiethoff, A., Schneider, H., Rohs, M., Butz, A., & Greenberg, S. (2012). Sketch-a-TUI: low cost prototyping of tangible interactions using cardboard and conductive ink. Paper presented at the *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*.

Williams, L., Kessler, R., 2001. *Pair Programming Illuminated*, Addison-Wesley Professional

Windows Presentation Foundation. Retrieved from <http://msdn.microsoft.com/en-us/library/ms754130.aspx>,

Wobbrock, J. O., Morris, M. R., & Wilson, A. D. (2009). User-defined gestures for surface computing. Paper presented at the *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*.

## BUILDING INFRASTRUCTURE FOR DIGITAL SURFACES

- T. Ariga and K. Mori. Sensory vision—development of a course for physical interaction and graphics. *Comput. & Graph.*, 34(6):800–810, 2010.
- Ashdown, M., Oka, K., and Sato, Y., Combining head tracking and mouse input for a GUI on multiple monitors. *Proc. CHI 2005*, 1188-1191.
- Ballendat, T., Marquardt, N. and Greenberg, S. (2010). Proxemic interaction: designing for a proximity and orientation-aware environment. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces – ITS '10* (pp. 121-130). New York, NY, USA: ACM Press.
- Baudisch, P., Cutrell, E., Hinckley, K. and Gruen, R., Mouse Ether: Accelerating the Acquisition of Targets Across Multi-Monitor Displays. *Proc. CHI 2004*, 1379-1382.
- Baudisch, P., Good, N., and Stewart, P. Focus plus context screens: combining display technology with visualization techniques. *Proc. UIST '01*, 31-40.
- Baudisch, P., and Rosenholtz, R., Halo: A Technique for Visualizing Off-Screen Locations. *Proc. CHI 2003*, 48-488.
- Benko, H. & Feiner, S., Pointer Warping in Heterogeneous Multi-Monitor Environments. *Proc. Graphics Interface*, 2007, 111-117.
- Benko, H. & Feiner, S., Multi-Monitor Mouse. *Proc. CHI 2005*, 1208-1211.
- Biehl, J. and Bailey, B. ARIS: an interface for application relocation in an interactive space. *Proc. Graphics Interface 2004*, 107-116.
- Birnholtz, J., Reynolds, L., Luxenberg, E., Gutwin, C., and Mustafa, M., Awareness Beyond The Desktop: Exploring Attention And Distraction With A Projected Peripheral-vision Display, *Proc. Graphics Interface 2010*, 55-62.
- A. Blackwell. Cognitive dimensions of notations resource site. <http://www.cl.cam.ac.uk/afb21/CognitiveDimensions/>. Last accessed January, 2015.
- A. Blackwell and T. Green. *A Cognitive Dimensions Questionnaire*, 5.7 edition, 2007.
- Bolt, Richard A., "Put-that-there": Voice and gesture at the graphics interface. *Proc. SIGGraph 1980*, 262-270.
- Brumitt, B., Meyers, B., Krumm, J., Kern, A. and Shafer, S. A. (2000). EasyLiving: Technologies for Intelligent Environments. In *Proceedings of the 2nd international symposium on Handheld and Ubiquitous Computing - UbiComp '00* (pp.12-29). London, UK: Springer-Verlag.
- Y.-L. Chang, S. D. Scott, and M. Hancock. Supporting situation awareness in collaborative tabletop systems with automation. In *Proc. ITS*, pages 185-194, ACM Press, 2014.
- Chen, X. A., Grossman, T., Wigdor, D. J. and Fitzmaurice, G. (2014). Duet: exploring joint interactions on a smart phone and a smart watch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14* (pp.159-168). New York, NY, USA: ACM Press.
- Chi, P. and Li, Y. (2015). Weave: Scripting Cross-Device Wearable Interaction. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15* (pp. 3923-3932). New York, NY, USA. ACM Press.
- A. Dai, R. Sadana, C. D. Stolper, and J. Stasko, Hands-on, large display visual data exploration, In *Poster Proc. of IEEE InfoVis*, 2015.

- Ebert, A., Thelen, S., Olech, P.-S., Meyer, J., Hagen, H. Tiled++: An Enhanced Tiled Hi-Res Display Wall. *IEEE TVCG* Jan/Feb 2010, 120-132.
- A. Gokcezade, J. Leitner, and M. Haller. LightTracker: An open-source multitouch toolkit. *Comput. Entertain.*, 8(3):19:1–19:16, 2010.
- Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R. and Wang, M. (2011). Proxemic interactions: the new ubicomp? interactions, (Vol.18, pp. 42-50). New York, NY, USA. ACM Press.
- Gustafson, S., Baudisch, P., Gutwin, C, and Irani, P., Wedge: Clutter-Free Visualization of Off-Screen Locations, *Proc. CHI 2008*, 787-796.
- T. E. Hansen, J. P. Hourcade, M. Virbel, S. Patali, and T. Serra. PyMT: A post-WIMP multi-touch user interface toolkit. In *Proc. ITS*, pages 17–24. ACM Press, 2009.
- Hamilton, P. and Wigdor, D. J. (2014). Conductor: enabling and understanding cross-device interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14* (pp.2773-2782). New York, NY, USA. ACM Press.
- Hinckley, K. (2003). Synchronous gestures for multiple persons and computers. In *Proceedings of the 16th annual ACM symposium on User interface software and technology – UIST'03* (pp. 149-158). New York, NY, USA. ACM Press. doi: 10.1145/964696.964713
- Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M., Stitching: pen gestures that span multiple displays. *Proc. AVI 2004*, 23-31.
- E. Hornecker and T. Psik. Using ARToolkit markers to build tangible prototypes and simulate other technologies. In *Proc. INTERACT*, pages 30–42. Springer-Verlag, 2005.
- Houben, S. and Marquardt, N. (2015). WatchConnect: A Toolkit for Prototyping Smartwatch-Centric Cross-Device Applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'15* (pp.1247-1256). New York, NY, USA. ACM Press.
- M. Jain. Multi-touch technology, applications and global markets. Technical Report SMC088A, BCC Research, 2014.
- Johnson, T., and Fuchs, H. A Unified Multi-Surface, Multi-Resolution Workspace with Camera-Based Scanning and Projector-Based Illumination, *Proc. IPT-EGVE Symposium*, 2007.
- Jota, R., Nacenta, M., Jorge, J., Carpendale, S., and Greenberg, S., A comparison of ray pointing techniques for very large displays. *Proc. Graphics Interface 2010*, 269-276.
- M. Kaltenbrunner. reactIVision and TUIO: A tangible tabletop toolkit. In *Proc. ITS*, pages 9–16. ACM Press, 2009.
- D. Kammer, M. Keck, G. Freitag, and M. Wacker. Taxonomy and overview of multi-touch frameworks: Architecture, scope and features. In *Workshop on Eng. Patterns for Multi-Touch Interfaces, EICS*, 2010.
- S. H. Khandkar, S. M. Sohan, J. Sillito, and F. Maurer. Tool support for testing complex multi-touch gestures. In *Proc. ITS*, pages 59–68. ACM Press, 2010.
- S. Kobayashi, T. Endo, K. Harada, and S. Oishi. GAINER: a reconfigurable I/O module and software libraries for education. In *Proc. NIME*, pages 346–351, 2006.
- König, A., Rädle, R. and Reiterer, H. (2009). Squidy: a zoomable design environment for natural user interfaces. In *Extended Abstracts on Human Factors in Computing*

Systems - CHI'EA'09 (pp.4561-4566). New York, NY, USA. ACM Press.

W. A. König, R. Rädle, and H. Reiterer. Interactive design of multimodal user interfaces. *J. on Multimodal User Interfaces*, 3(3):197–213, 2010.

U. Laufs, C. Ruff, and J. Zibuschka. MT4j - a cross-platform multi-touch development framework. In *Workshop on Eng. Patterns for Multi-Touch Interfaces*, EICS, 2010.

I. Leftheriotis, K. Chorianopoulos, and L. Jaccheri. Tool support for developing scalable multiuser applications on multi-touch screens. In *Proc. ITS*, pages 371–374. ACM Press, 2012.

Lucero, A., Keränen, J. and Korhonen, H. (2011). Collaborative use of mobile phones for brainstorming. In *Proceedings of the International Conference on Human Computer Interactions with Mobile Devices and Services – MobileHCI'10* (pp.337-340). New York, NY, USA. ACM Press.

J. Luderschmidt, I. Bauer, N. Haubner, S. Lehmann, R. Dörner, and U. Schwanecke. TUIO AS3: A multi-touch and tangible user interface rapid prototype toolkit for tabletop interaction. In *ITG/GI Workshop on SENSIBLE*, pages 21–28. Shaker Aachen, 2010.

J. Marco, E. Cerezo, and S. Baldassarri. ToyVision: A toolkit for prototyping tabletop tangible games. In *Proc. EICS*, pages 71–80. ACM Press, 2012.

Marquardt, N., Ballendat, T., Boring, S., Greenberg, S. and Hinckley, K. (2012). Gradual engagement: facilitating information exchange between digital devices as a function of proximity. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces – ITS'12* (pp.31-40). New York, NY, USA. ACM Press.

Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011). The proximity toolkit: prototyping proxemic interactions in ubiquitous computing ecologies. In *Proceedings of the symposium on User interface software and technology – UIST'11* (pp.315-326). New York, NY, USA. ACM Press

Marquardt, N., Hinckley, K. and Greenberg, S. (2012) Cross-device interaction via micro-mobility and f-formations. In *Proceedings of the symposium on User interface software and technology – UIST'12* (pp.3-22). New York, NY, USA. ACM Press.

M. Murshed and R. Buyya. Using the GridSim toolkit for enabling grid computing education. In *Int. Conf. on Commun. Networks and Distributed Syst. Modeling and Simulation*, pages 27–31, 2002.

Myers, B., Bhatnagar, R., Nichols, J., Peck, C., Kong, D., Miller, R., and Long, A., Interacting at a distance: measuring the performance of laser pointers and other devices. *Proc. CHI 2002*, 33-40.

Nacenta, M., Gutwin, C., Aliakseyeu, D., and Subramanian, S., There and Back again: Cross-Display Object Movement in Multi-Display Environments, *JHCI*, 24, 1, 2009, 170-229.

Nacenta, M., Mandryk, R., and Gutwin, C, Targeting across displayless space. *Proc. CHI 2008*, 777-786.

Nacenta, M., Sakurai, S., Yamaguchi, T., Miki, Y., Itoh, Y., Kitamura, Y., Subramanian, S. and Gutwin, C., E-conic: a Perspective-Aware Interface for Multi-Display Environments. *Proc. UIST 2007*, 279-288.

Nacenta, M., Pinelle, D., Stuckel, D., and Gutwin, C., The Effects of Interaction Technique on Coordination in Tabletop Groupware. *Proc. Graphics Interface*, 2007, 191-198.

- Nacenta, M., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C., Perspective cursor: perspective-based interaction for multi-display environments. Proc. CHI 2006, 289-298.
- Nacenta, M., Computer Vision approaches to solve the screen pose acquisition problem for Perspective Cursor, Report HCI-TR-06-01, Dept. of Computer Science, University of Saskatchewan, 2006.
- Nebeling, M., Mints, T., Husmann, M. and Norrie, M. (2014). Interactive development of cross-device user interfaces. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems – CHI'14 (pp.2793-2802). New York, NY, USA. ACM Press.
- M. Nebeling and M. Norrie. jQMultiTouch: Lightweight toolkit and development framework for multi-touch/multi-device web interfaces. In Proc. EICS, pages 61–70. ACM Press, 2012.
- Nebeling, M., Teunissen, E., Husmann, M. and Norrie, M. C. (2014). XDKinect: development framework for cross-device interaction using kinect. In Proceedings of the SIGCHI symposium on Engineering interactive computing systems – EICS'14 (pp.65-74). New York, NY, USA. ACM Press.
- NUI Group Community. Community Core Vision (CCV). <http://ccv.nuigroup.com/>. Last accessed on June, 7 2013.
- D. R. Olsen, Jr. Evaluating user interface systems research. In Proc. UIST, pages 251–258. ACM Press, 2007.
- E. Paluka and C. Collins, TandemTable: Supporting conversations and language learning using a multi-touch digital table. In Proc. Graphics Interface, 2015.
- Pinhanez, C., Kjeldsen, R., Levas, A., Pingali, G., Podlaseck, M., and Sukaviriya, N., Applications of Steerable Projector-Camera Systems, Proc. ICCV Workshop on Projector-Camera Systems (PROCAMS'03), 2003.
- Pratte, S., Seyed, T., Maurer, F. (2015). Projected Pixels: Exploring Projection Feedback in Multi-Surface Environments. In ITS 2015 Workshop on Collaboration meets Interactive Surfaces – CmIS'15.
- C. Reas and B. Fry. Processing 2: Overview. <https://processing.org/overview/>. Last accessed on Oct 1, 2015.
- C. Reas and B. Fry. Processing: A learning environment for creating interactive web graphics. In ACM SIGGRAPH 2003 Web Graphics, New York, NY, USA, 2003. ACM.
- C. Reas and B. Fry. Processing: Programming for the media arts. *AI Soc.*, 20(4): 526–538, 2006.
- Rekimoto, J. (1997). Pick-and-drop: a direct manipulation technique for multiple computer environments. In Proceedings of the symposium on User interface software and technology – UIST'97 (pp.31-39). New York, NY, USA. ACM Press.
- Robertson, S., Wharton, C., Ashworth, C. & Franzke, M., Dual Device User Interface Design: PDAs and Interactive Television. Proc. CHI 1996, 79-86.
- Schreiner, M., Rädle, R., Hans-Christian, J. and Reiterer, H. (2015). Connichiwa: A Framework for Cross-Device Web Applications. In Extended Abstracts on Human Factors in Computing Systems - CHI'EA'15 (pp.2163-2168). New York, NY, USA. ACM Press.
- Seyed, T., Azazi, A., Chan, E., Wang, Y., and Maurer, F. (2015). SoD-Toolkit: A Toolkit for Interactively Prototyping and Developing Multi-Sensor, Multi-Device

Environments. In Proceedings of the International Conference on Interactive Tabletops and Surfaces – ITS'15 (pp.171-180). New York, NY, USA. ACM Press.

Seyed, T., Burns, C., Sousa, M. C., Maurer, F. and Tang, (2012). A. Eliciting usable gestures for multi-display environments. In Proceedings of the International Conference on Interactive Tabletops and Surfaces – ITS'12 (pp.41-50). New York, NY, USA. ACM Press.

C. Shen, F. D. Vernier, C. Forlines, and M. Ringel. DiamondSpin: An extensible toolkit for around-the-table interaction. In Proc. CHI, pages 167–174. ACM Press, 2004.

Su, R., and Bailey, B., Put Them Where? Towards Guidelines for Positioning Large Displays in Interactive Workspaces, Proc. Interact 2005, 337-349.

L. Tang, Z. Yu, X. Zhou, H. Wang, and C. Becker. Supporting rapid design and evaluation of pervasive applications: Challenges and solutions. *Personal Ubiquitous Comput.*, 15(3):253–269, 2011.

P. Tuddenham and P. Robinson. T3: Rapid prototyping of high-resolution and mixed-presence tabletop applications. In TABLETOP, pages 11–18. IEEE Comput. Soc., 2007.

Vogel, D., and Balakrishnan, R., (2004). Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. Proceedings of the 17th annual ACM symposium on User interface software and technology, 137-146.

Waldner, M., and Schmalstieg, D., Experiences with Mouse Control in Multi-Display Environments, Proc. Workshop on coupled display visual interfaces at AVI 2010, 6-10.

Weiser, M. (2001). The computer for the 21st century. IEEE Pervasive Computing (Vol.1, pp. 19-25). Piscataway, NJ, USA. IEEE Educational Activities Department.

Wilson, A. D. and Benko, H. (2010). Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In Proceedings of the symposium on User interface software and technology – UIST'10 (pp.273-282). New York, NY, USA. ACM Press.

Welch, G., Fuchs, H., Raskar, R., Towles, H., and Brown, M., Projected Imagery in Your Office in the Future. IEEE Computer Graphics and Applications, Jul-Aug 2000, 20, 4, 62-67.

Wilson, A., and Benko, H., Combining multiple depth cameras and projectors for interactions on, above and between surfaces. Proc. UIST 2010, 273-282.

Yang, J. and Wigdor, D. (2014). Panelrama: enabling easy specification of cross-device web applications. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems – CHI'14 (pp.2783-2792). New York, NY, USA. ACM Press.

## **SURFACE APPLICATIONS**

Adeyemi, Taye. n.d. "Interact.js." <http://interactjs.io/> Accessed: 2015-21-08.

Akamatsu, M. et al. A comparison of tactile, auditory, and visual feedback in a pointing task using a mouse-type device. *Ergonomics*. 38, (1995), 816–827.

- Alberdi, E. et al. Effects of incorrect computer-aided detection (CAD) output on human decision-making in mammography. *Academic Radiology*. 11, (2004), 909–918.
- Alizadeh, H., Tang, R., Sharlin, E., Tang, A.: Haptics in remote collaborative exercise systems for seniors. In: *Ext. Abs. CHI 2014*, pp. 2401–2406 (2014) .
- Anderson, F., Grossman, T., Matejka, J., Fitzmaurice, G.W.: YouMove: enhancing movement training with an augmented reality mirror. In: *Proc UIST 2013*, pp. 311–320 (2013).
- Andersen, P.K. and Gill, R.D. Cox's Regression Model for Counting Processes: A Large Sample Study. *The Annals of Statistics*. 10, 4 (1982), 1100–1120.
- André, P., schraefel, m. c., Teevan, J., and Dumais, S. T. Discovery is never by chance: Designing for (un)serendipity. In *Proc. of Creativity and Cognition (2009)*, 305–314.
- G. Andrienko, N. Andrienko, S. Bremm, T. Schreck, T. Von Landesberger, P. Bak and D. Keim, "Space-in-Time and Time-in-Space Self-Organizing Maps for Exploring Spatiotemporal Patterns," *Computer Graphics Forum*, vol. 29, no. 3, p. 913–922, 2010.
- Andriole, K.P. et al. Optimizing analysis, visualization, and navigation of large image data sets: one 5000-section CT scan can ruin your whole day. *Radiology*. 259, 2 (2011), 346–362.
- Aseniero, B.A., Sharlin, E.: The looking glass: visually projecting yourself to the past. In: *Proc ICEC 2011*, pp. 282–287 (2011).
- Atkins, M.S. et al. Evaluating Interaction Techniques for Stack Mode Viewing. *Journal of digital imaging*. 22, (2009), 369–382.
- Baker, K., Greenberg, S., & Gutwin, C. (2001). Heuristic Evaluation of Groupware Based on the Mechanics of Collaboration. In *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction (EHCI '01)* (pp. 123–140). Springer-Verlag. Retrieved from <http://dl.acm.org/citation.cfm?id=645350.650731>.
- Balakrishnan, R. et al. The Rockin' Mouse : Integral 3D Manipulation on a Plane. In *Proceedings of CHI 97*. (1997), 311–318.
- Bang, J. The meaning of plot and narrative. In *The computer as medium*. Cambridge University Press, Cambridge, New York, (1993) 209–220.
- Bau, O. and Mackay, W. (2008) OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the ACM Symposium on User Interface Software and Technology (UIST 2008)*, 37–46.
- M. U. A. Behnisha, "Urban data-mining: Spatiotemporal exploration of multidimensional data," *Building Research & Information*, vol. 37, no. 5–6, pp. 520–532, 2010.
- M. Belfqih, "Vodafone Fixed Network-Visualization Tool for ArcView GIS," in *ESRI International User Conference 2003*, San Diego, 2003.
- Blackwell, Alan F., Carol Britton, Anna Louise Cox, Thomas R. G. Green, Corin A. Gurr, Gada F. Kadoda, Maria Kutar, et al. 2001. "Cognitive Dimensions of Notations: Design Tools for Cognitive Technology." In *4th International Conference on Cognitive Technology: Instruments of Mind*, 325–41. CT '01. London, UK, UK: Springer-Verlag.
- Blackwell, Alan, and Thomas Green. 2003. "Notational Systems—the Cognitive Dimensions of Notations Framework." *HCI Models, Theories, and Frameworks*:

Toward an Interdisciplinary Science. Morgan Kaufmann.

E. Blevins, Y.-k. Lim, D. Roedl and E. Stolterman, "Using Design Critique as Research to Link Sustainability and Interactive Technologies," *Online Communities and Social Computing - Lecture Notes in Computer Science*, vol. 4564, pp. 22-31, 2007.

Christophe Bortolaso, Matthew Oskamp, T.C. Nicholas Graham, and Doug Brown. OrMiS: A Tabletop Interface for Simulation-Based Training. *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces – ITS '13*, ACM Press, pp 145–154, 2013.

Christophe Bortolaso, Matthew Oskamp, Carl Gutwin, Greg Phillips and T.C. Nicholas Graham, The Effect of View Techniques on Collaboration and Awareness in Tabletop Map-Based Tasks, in *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS)*, 2014, pp 79-88.

Bostock, Michael, Vadim Ogievetsky, and Jeffrey Heer. 2011. "Data-Driven Documents." *Visualization and Computer Graphics*, *IEEE Transactions on* 17 (12). IEEE: 2301–9.

Bostock, Mike. n.d. "Parallel Coordinates." <http://bl.ocks.org/mbostock/1341021> Accessed: 2015-21-08.

Brooke, J. (1996). SUS - A quick and dirty usability scale. In A. Jordan, Patrick, W., Thomas, Bruce, Weerdmeester, Bernhard, A., McLelland, Ian (Ed.), *Usability Evaluation in Industry* (pp. 189–194). CRC Press.

Brown, Judith M., Jeff Wilson, and Robert Biddle. 2014. "A Study of an Intelligence Analysis Team and Their Collaborative Artifacts." *School of Computer Science Technical Report TR-14-04*, Carleton Uni.

Brown, Judith M., Jeff Wilson, Stevenson Gossage, Chris Hack, and Robert Biddle. 2013. *Surface Computing and Collaborative Analysis Work. Synthesis Lectures on Human-Centered Informatics*. Morgan & Claypool.

Brown, Judith, Jeff Wilson, Stevenson Gossage, Chris Hack, and Robert Biddle. 2013. *Surface Computing and Collaborative Analysis Work. Synthesis Lectures on Human-Centered Informatics 19*. Morgan & Claypool Publishers.

Burton, M. n.d. "ACH:A free, open source tool for complex research problems." <http://competinghypotheses.org/>.

Buxton, B.: *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann (2010).

Buxton, B.: *Mediaspace—meaningspace—meetingspace*. In: *Media space 20+ years of mediated life*, pp. 217-231. Springer, London (2009).

Canadian Institute for Health Information: *Physiotherapists in Canada, 2010: National and Jurisdictional Highlights and Profiles* (2011).

Carroll, E. A., Lottridge, D., Latulipe, C., Singh, V., Word, M.: Bodies in critique: a technological intervention in the dance production process. In: *Proc CSCW 2012*, pp. 705-714 (2012).

Cabana, F., Boissy, P., Tousignant, M., Moffet, H., Corriveau, H., Dumais, R.: Interrater agreement between telerehabilitation and face-to-face clinical outcome measurements for total knee arthroplasty. *Telemedicine and e-Health*, vol. 16, num. 3, 293-298 (2010).

Chang, Y., Chen, S., Huang, J.: A Kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities. *Research in developmental*



disabilities vol. 32, no. 6, 2566-2570 (2011).

Chatti, Mohamed Amine, Tim Sodhi, Marcus Specht, Ralf Klamma, and Roland Klemke. 2006. "U-Annotate: An Application for User-Driven Freeform Digital Ink Annotation of E-Learning Content." In *Advanced Learning Technologies, 2006. Sixth International Conference on*, 1039–43. IEEE.

A. Cockburn, *Crystal Clear: A Human-Powered Methodology for Small Teams: A Human-Powered Methodology for Small Teams*, Addison-Wesley, 2004.

Colley, R. C., Garriguet, D., Janssen, I., Craig, C. L., Clarke, J., & Tremblay, M. S. Physical activity of Canadian adults: Accelerometer results from the 2007 to 2009 Canadian Health Measures Survey. *Health Reports*, 22(1), 2011a, 1-8.

Colley, R. C., Garriguet, D., Janssen, I., Craig, C. L., Clarke, J., & Tremblay, M. S. Physical activity of Canadian children and youth: accelerometer results from the 2007 to 2009 Canadian Health Measures Survey. *Health Reports*, 22(1), 2011b, 15-23.

Conti, Greg. 2007. *Security Data Visualization: Graphical Techniques for Network Analysis*. No Starch Press.

Contributors, Multiple. 2011. "Enron Email Dataset." Available at: <https://www.cs.cmu.edu/~./enron/>.

Cox, D.R. Regression Models and Life-Tables. *Journal of the Royal Statistical Society*. 34, (1972), 187–220.

Denoue, Laurent, and Laurence Vignollet. 2002. "Annotations in the Wild." In *ECAI 2002 Workshop on Semantic Authoring, Annotation & Knowledge Markup*. Position Paper.

Dix, A. (2002). Beyond intention-pushing boundaries with incidental interaction. In *Proceedings of Building Bridges: Interdisciplinary Context-Sensitive Computing*, Glasgow University (Vol. 9).

Dix, A. et al. Human-Computer Interaction in Radiotherapy Target Volume Delineation: A Prospective, Multi-institutional Comparison of User Input Devices. *Journal of digital imaging*. 24, (2010), 794–803.

DocsLogic Healthcare Solutions, "Geo-Spatial Network Analysis," 2013. [Online]. Available: <http://www.docsllogic.com/32/de/solutions/Geospatial-Network-Analysis>. [Accessed June 2013].

Doi, K. Current status and future potential of computer-aided diagnosis in medical imaging. *The British journal of radiology*. 78 Spec No, (2005), S3–S19.

Doucette, A., Gutwin, C., Mandryk, R. L., Nacenta, M., & Sharma, S. (2013). Sometimes when we touch: how arm embodiments change reaching and collaboration on digital tables. In *Proceedings of the 2013 conference on Computer supported cooperative work - CSCW '13* (pp. 193–202). New York, New York, USA: ACM Press.

ECMAScript 2015 Language Specification." 2015. Standard ECMA-262, Sixth Edition. Sixth. Ecma International.

Efron, B. The Efficiency of Cox's Likelihood Function for Censored Data. *Journal of the American Statistical Association*. 72, 359 (1977), 557–565.

Egglin, T.K.P. Context Bias, A Problem in Diagnostic Radiology. *Journal of the American Medical Association*. 276, 21 (1996), 1752.

S. G. Eick, A. Eick, J. Fugitt, J. E. Heath and M. Ross, "Geotemporal Analysis," in *IEEE Aerospace Conference, Big Sky, MT, USA*, 2008.

- Engestrom, Yrjo. 1992. *Interactive Expertise: Studies in Distributed Working Intelligence*. Research Bulletin 83. ERIC.
- Engeström, Yrjö. 2000. "Activity Theory as a Framework for Analyzing and Redesigning Work." *Ergonomics* 43 (7): 960–74. ISI:000088268400012.
- Erdelez, S. Information encountering: It's more than just bumping into information. *Bulletin of the American Society for Information Science* 25(3) (1999), 25–29.
- Farah, Hanna, and Timothy C. Lethbridge. 2007. "Temporal Exploration of Software Models: A Tool Feature to Enhance Software Understanding." In *Proceedings of the 14th Working Conference on Reverse Engineering*, 41–49. WCRE '07. IEEE Computer Society. doi:10.1109/WCRE.2007.49.
- Forlines, C., & Shen, C. (2005). DTLens: multi-user tabletop spatial data exploration. In *Proceedings of the 18th annual ACM symposium on User interface software and technology - UIST '05* (pp. 119–122). New York, New York, USA: ACM Press. doi:10.1145/1095034.1095055.
- Foster, A., and Ford, N. Serendipity and information seeking: An empirical study. *Journal of Documentation* 59(3) (2003), 321–340.
- Freeman, D., et al. (2009) ShadowGuides: visualizations for in-situ learning of multi-touch and whole-hand gestures. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS 2009)*, 165-172.
- Gao, Y., & Mandryk, R. The acute cognitive benefits of casual exergame play. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI)*, 2012, 1863-1872.
- Gao, Y., Gerling, K.M., Mandryk, R.L., and Stanley, K.S. Decreasing sedentary behavior among pre-adolescents using casual exergames at school. In *Proceedings of ACM SIGCHI Conference on Interactive Play (CHI PLAY)*. Toronto, Canada, 2014, 97-106.
- Gardner, M. Mathematical Games – The fantastic combinations of John Conway's new solitaire game "life." *Scientific American*, 1970, 120–123.
- Gerling, K.M., Dergousoff, K.K., & Mandryk, R.L. Is Movement Better? Comparing Sedentary and Motion-Based Game Controls for Older Adults. In *Proceedings of Graphics Interface*, 2013, 133-140.
- Gerling, K.M., Mandryk, R.L. *Designing Video Games for Older Adults and Caregivers*. In *Meaningful Play*, 2014a, East Lansing, MI, USA. 24 pages.
- Gerling, K.M., Miller, M.K., Mandryk, R.L., Birk, M., and Smeddinck, J. Effects of balancing for physical abilities on player performance, experience and self-esteem in exergames. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI)*, 2014b, 2201–2210.
- Gerling, K.M., Mandryk, R.L., Linehan, C. Long-term use of motion-based video games in care home settings. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI)*, 2015, 1573-1582.
- Geyer, F., Pfeil, U., Höchtl, A., Budzinski, J., & Reiterer, H. (2011). *Designing Reality-Based Interfaces for Creative Group Work*. In *Proceedings of C&C '11* (pp. 165–174). New York, New York, USA: ACM Press. doi:10.1145/2069618.2069647.
- Globalytica. n.d. "Globalytica Software Tools: Te@mACH." <http://www.globalytica.com/thinksuite-html/>.

- Goffman, E. *The Presentation of Self in Everyday Life*. Doubleday Anchor Books, New York, 1959.
- Gonsalves, T., Frith, C., Critchley, H., Picard, R., and El Kaliouby, R. *Chameleon*. 2008. <http://www.tinagonsalves.com/chamselectframe.html> (accessed 7/11/2014).
- Goyal, N. et al. *Ergonomics in radiology*. *Clinical Radiology*. 64, 2 (2009), 119–126.
- Gutwin, C., & Greenberg, S. (1998). Design for individuals, design for groups: tradeoffs between power and workspace awareness. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work - CSCW '98* (pp. 207–216). New York, New York, USA: ACM Press.
- Gutwin, C., & Greenberg, S. (2002). A Descriptive Framework of Workspace Awareness for Real-Time Groupware. *Computer Supported Cooperative Work*, 11(3-4), 411–446. doi:10.1023/A:1021271517844.
- Hackman, J Richard. 2011. *Collaborative Intelligence: Using Teams to Solve Hard Problems*. Berrett-Koehler Publishers.
- Hancock, M. S., Carpendale, S., Vernier, F. D., & Wigdor, D. (2006). Rotation and Translation Mechanisms for Tabletop Interaction. In *First IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP '06)* (pp. 79–88). IEEE. doi:10.1109/TABLETOP.2006.26.
- Heer, Jeffrey, Michael Bostock, and Vadim Ogievetsky. 2010. "A Tour Through the Visualization Zoo." *Commun. Acm* 53 (6): 59–67.
- Heuer Jr., Richards J., and Randolph.H. Pherson. 2010. *Structured Analytic Techniques for Intelligence Analysis*. CQ Press. <http://books.google.ca/books?id=ruGUQQAACAAJ>.
- Hill, G. *Tall Ships*. 1992. <http://garyhill.com/left/work/tallships.html?q=569> (accessed 7/11/2014).
- Hinckley, K. et al. Quantitative analysis of scrolling techniques. *Proceedings of CHI '02*. 4 (2002), 65.
- Hinckley, K. and Sinclair, M. Touch-sensing input devices. *Proceedings of CHI '99*. pages, (1999), 223–230.
- Hinrichs, U., Carpendale, S., & Scott, S. D. (2005). Interface currents: supporting fluent face-to-face collaboration. In *ACM SIGGRAPH 2005 Sketches on - SIGGRAPH '05* (p. 142). New York, New York, USA: ACM Press.
- Uta Hinrichs and Sheelagh Carpendale. *Gestures in the Wild: Studying Multi-Touch Gesture Sequences on Interactive Tabletop Exhibits*. In *CHI '11: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. (2011) 3023-3032.
- O. Hoerber, G. Wilson, S. Harding, R. Enguehard and R. Devillers, "Exploring geotemporal differences using GTdiff," in *IEEE Pacific Visualization Symposium*, Hong Kong, 2011.
- Hypothes.is. n.d. "Hypothes.is." <https://hypothes.is/> Accessed: 2015-21-08.
- Inselberg, Alfred. 1997. "Multidimensional Detective." In *Information Visualization, 1997. Proceedings., IEEE Symposium on*, 100–107. IEEE.
- Inselberg, Alfred, and Bernard Dimsdale. 1990. "Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry." In *Proceedings of the 1st Conference on Visualization '90*, 361–78. VIS '90. Los Alamitos, CA, USA: IEEE Computer Society Press. <http://dl.acm.org/citation.cfm?id=949531.949588>.

- Ishii, H., Kobayashi, M., Grudin, J.: Integration of interpersonal space and shared workspace: ClearBoard design and experiments. *ACM Trans. Inf. Syst.* vol. 11, no. 4, 349-375 (1993).
- Ion, A., Chang, Y.-L. B., Haller, M., Hancock, M., Scott, S. D., & Chang, B. (2013). Canyon: Providing Location Awareness of Multiple Moving Objects in a Detail View on Large Displays. In *CHI 2013* (pp. 3149–3158). ACM.
- Isenberg, P., D. Fisher, M.R. Morris, K. Inkpen, and M. Czerwinski. 2010. "An Exploratory Study of Co-Located Collaborative Visual Analytics Around a Tabletop Display." In *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, 179–86.
- Isenberg, P., Tang, A., & Carpendale, S. (2008). An exploratory study of visual information analysis. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (pp. 1217–1226).
- S. Jänicke, C. Heine and G. Scheuermann, "Comparative visualization of geospatial-temporal data," *Computer Vision, Imaging and Computer Graphics. Theory and Application* , vol. 359, pp. 160-175 , 2013.
- Javed, W., Kim, K., Ghani, S., & Elmqvist, N. (2011). Evaluating physical/virtual occlusion management techniques for horizontal displays, 391–408. Retrieved from <http://dl.acm.org/citation.cfm?id=2042182.2042218>.
- Khalilbeigi, M., Steimle, J., Riemann, J., Dezfuli, N., Mühlhäuser, M., & Hollan, J. D. (2013). ObjecTop: occlusion awareness of physical objects on interactive tabletops. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces - ITS '13* (pp. 255–264). New York, New York, USA: ACM Press.
- Kingston, G.A., Williams, G., Gray, M.A., Judd, J.: Does a DVD improve compliance with home exercise programs for people who have sustained a traumatic hand injury? Results of a feasibility study. *Disability and Rehabilitation: Assistive Technology*, 1-7 (2013).
- Kirk, D., Stanton Fraser, D.: Comparing remote gesture technologies for supporting collaborative physical tasks. In: *Proc CHI 2006*, pp. 1191-1200 ACM (2006).
- Kirsh, David, and Paul Maglio. 1994. "On Distinguishing Epistemic from Pragmatic Action." *Cognitive Science* 18 (4). Lawrence Erlbaum Associates, Inc.: 513–49.
- Kobayashi, K., Narita, A., Hirano, M., Kase, I., Tsuchida, S., Omi, T., ... Hosokawa, T. (2006). Collaborative simulation interface for planning disaster measures. In *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06* (pp. 977–982). New York, New York, USA: ACM Press. doi:10.1145/1125451.1125639.
- Kongsberg Gallium. (2013). InterMAPhics. Retrieved May 31, 2013, from <http://www.kongsberg.com/en/kds/kongsberggallium/products/intermaphics/>.
- Krasner, Glenn E, Stephen T Pope, and others. 1988. "A Description of the Model-View-Controller User Interface Paradigm in the Smalltalk-80 System." *Journal of Object Oriented Programming* 1 (3): 26–49.
- Krueger, M.W. *Responsive Environments*. Proc. of the June 13-16, National Computer Conference, ACM (1977), 423–433.
- Kruger, R., Carpendale, S., Scott, S. D., & Greenberg, S. (2004). Roles of Orientation in Tabletop Collaboration: Comprehension, Coordination and Communication. *Computer Supported Cooperative Work (CSCW)*, 13(5-6), 501–537.
- Kundel, H.L. et al. Visual scanning, pattern recognition and decision-making in pulmonary nodule detection. *Investigative Radiology*. 13, (1978), 175–181.

- Lai, J.C., Woo, J., Hui, E., Chan, W.M.: Telerehabilitation—a new model for community-based stroke rehabilitation. *Journal of telemedicine and telecare*, vol. 10, num. 4, 199-205 (2004).
- Ledo, D., Aseniero, B.A., Greenberg, S., Boring, S., Tang, A.: OneSpace: Shared Depth-Corrected Video Interaction. In: *Ext Abs of CHI 2013*, pp. 997-1002 ACM (2013).
- Lee, Bongshin, P. Isenberg, N.H. Riche, and S. Carpendale. 2012. "Beyond Mouse and Keyboard: Expanding Design Considerations for Information Visualization Interactions." *Visualization and Computer Graphics, IEEE Transactions on* 18 (12): 2689–98.
- Lévesque, V. et al. Enhancing physicality in touch interaction with programmable friction. *Proceedings of CHI '11*. 31, (2011), 2481–2490.
- Liestman, D. Chance in the midst of design: approaches to library research serendipity. *RQ* 31(4) (1992), 524—536.
- Lindsay MacDonald, John Brosz, Miguel A. Nacenta and Sheelagh Carpendale. *Designing the Unexpected: Endlessly Fascinating Interaction for Interactive Installations*. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction*. (2015) 41-48.
- A. Malik, R. Maciejewski, E. Hodgess and D. S. Ebert, "Describing temporal correlation spatially in a visual analytics environment," in *Hawaii International Conference on System Sciences, Kauai, HI, USA, 2011*.
- Mandryk, R.L., Gerling, K.M., and Stanley, K.S. Designing games to discourage sedentary behavior. In *Playful User Interfaces, Gaming Media and Social Effects*. A. Nijholt, ed. Springer, 2014, 253–274.
- Manning, D.J. et al. Perception research in medical imaging. *The British journal of radiology*. 78, 932 (Aug. 2005), 683–5.
- Matejka, J. et al. Swifter: improved online video scrubbing. *Proceedings of CHI '13* (2013), 1159.
- Mathie, A.G. and Strickland, N.H. Interpretation of CT scans with PACS image display in stack mode. *Radiology*. 203, (1997), 207–209.
- Miller, J.H. and Page, S.E. *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton University press, 2009.
- Morikawa, O., Maesako, T.: HyperMirror: toward pleasant-to-use video mediated communication system. In: *Proc CSCW 98*, pp. 149-158. ACM (1998).
- Morris, M. R., Ryall, K., Shen, C., Forlines, C., & Vernier, F. (2004). Beyond "social protocols": multi-user coordination policies for co-located groupware. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work - CSCW '04* (p. 262). New York, New York, USA: ACM Press.
- Mueller, F., Vetere, F., Gibbs, M.R., Agamanolis, S., and Sheridan, J. Jogging over a Distance: The influence of design in parallel exertion games. *Proc. of the SIGGRAPH Sandbox*. 2010, 63–68.
- T. Nagel, E. Duval and F. Heidmann, "Visualizing Geospatial Co-authorship Data on a Multitouch Tabletop," *Smart Graphics - Lecture Notes in Computer Science*, vol. 6815, pp. 134-137, 2011.
- T. Nagel, E. Duval and A. V. Moere, "Interactive exploration of geospatial network visualization," in *Extended Abstracts on Human Factors in Computing Systems*,

New York, 2012.

Nass, C., Moon, Y., Fogg, B.J., Reeves, B., and Dryer, D.C. Can computer personalities be human personalities? *International Journal of Human-Computer Studies* 43(2) (1995), 223–239.

Nakatsu, R., Rauterberg, M., & Vorderer, P. (2005). A New Framework for Entertainment Computing: From Passive to Active Experience. In F. Kishino, Y. Kitamura, H. Kato, & N. Nagata (Eds.), *Entertainment Computing - ICEC 2005* (Vol. 3711, pp. 1–12). Berlin: Springer Berlin / Heidelberg.

Nayak, S., Zlatanova, S., Hofstra, H., Scholten, H., & Scotta, A. (2008). Multi-user tangible interfaces for effective decision-making in disaster management. In S. Nayak & S. Zlatanova (Eds.), *Remote Sensing and GIS Technologies for Monitoring and Prediction of Disasters* (Environmen., pp. 243–266–266). Berlin, Heidelberg: Springer Berlin Heidelberg.

Network, Mozilla Develop. n.d. "Proxy." [https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global\\_Objects/Proxy](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Proxy).

Network, Mozilla Developer. n.d. "ECMAScript 6 Support in Mozilla." [https://developer.mozilla.org/en-US/docs/Web/JavaScript/New\\_in\\_JavaScript/ECMAScript\\_6\\_support\\_in\\_Mozilla](https://developer.mozilla.org/en-US/docs/Web/JavaScript/New_in_JavaScript/ECMAScript_6_support_in_Mozilla).

Nóbrega, R., Sabino, A., Rodrigues, A., & Correia, N. (2008). Flood Emergency Interaction and Visualization System. In M. Sebillio, G. Vitiello, & G. Schaefer (Eds.), *Visual Information Systems. Web-Based Visual Information Search and Management* (Vol. 5188, pp. 68–79). Berlin, Heidelberg: Springer Berlin Heidelberg.

Matthew Oskamp, Christophe Bortolaso, Robin Harrap, and T.C. Nicholas Graham. TerraGuide: Design and Evaluation of a Multi-Surface Environment for Terrain Visibility Analysis. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*, May 2015, pp 3553-3562.

Oakley, I. et al. Tilt and Feel : Scrolling with Vibrotactile Display. *Eurohaptics*. (2004), 316-323.

Oram, L., MacLean, K., Kruchten, P., & Forster, B. (2014, June). Crafting diversity in radiology image stack scrolling: control and annotations. In *Proceedings of the 2014 conference on Designing interactive systems* (pp. 567-576).

Oxford English Dictionary. <http://www.oed.com/view/Entry/176387?redirectedFrom=serendipity#eid>. Website visited Sept. 2011.

Palo Alto Research Center. 2010. "Analysis of Competing Hypotheses Software Version ACH2.0.5."

K. Patroumpas and T. Sellis, "Event processing and real-time monitoring over streaming traffic data," *Web and Wireless Geographical Information Systems*, vol. 7236, pp. 116-133, 2012.

Peek-Asa, C., Zwerling, C., Stallones, L.: Acute traumatic injuries in rural populations. *American journal of public health*, vol. 94, num. 10, (2004).

Pirolli, Peter, and Stuart Card. 2005. "The Sensemaking Process and Leverage Points for Analyst Technology as Identified Through Cognitive Task Analysis." In *Proceedings of International Conference on Intelligence Analysis*, 5:2–4.

Plimmer, Beryl, Samuel Hsiao-Heng Chang, Meghavi Doshi, Laura Laycock, and Nilanthi Seneviratne. 2010. "Annotate: Exploring Multi-User Ink Annotation in Web Browsers." In *Proceedings of the Eleventh Australasian Conference on User Interface - Volume 106*, 52–60. AUIC '10. Darlinghurst, Australia, Australia: Australian

Computer Society, Inc.

Qin, Y., Liu, J., Wu, C., & Shi, Y. (2012). uEmergency: a collaborative system for emergency management on very large tabletop. In Proceedings of the 2012 ACM international conference on Interactive tabletops and surfaces - ITS '12 (p. 399). New York, New York, USA: ACM Press.

Remer, T. G. *Serendipity of the Three Princes, from the Peregrinaggio of 557*. Norman, OK: University of Oklahoma Press, 1965.

K. P. Roe, M. Murphy and J. Schmidt, "Geo-temporal Visualization of Information Collected from Large Databases Using the Time-Based COCOM Operational Picture (TIMECOP) Server," in DoD High Performance Computing Modernization Program Users Group Conference, San Diego, CA, 2009.

Rogante, M., Grigioni, M., Cordella, D., Giacomozzi, C.: Ten years of telerehabilitation: A literature overview of technologies and clinical applications. *NeuroRehabilitation*, vol. 27, num. 4, 287-304 (2010).

Roman, P. A., & Brown, D. (2008). Games, Just How Serious Are They? In Proceedings of Interservice/Industry Training, Simulation & Education Conference (p. 11 pages).

Rosenman, M. F. Serendipity and scientific discovery. *Journal of Creative Behavior* 22 (1988), 132–138.

Rubin, G.D. et al. Pulmonary nodules on multi-detector row CT scans: performance comparison of radiologists and computer-aided detection. *Radiology*. 234, (2005), 274–283.

Russell, T.G., Buttrum, P., Wootton, R., Jull, G.A.: Internet-Based Outpatient Telerehabilitation for Patients Following Total Knee Arthroplasty A Randomized Controlled Trial. *The Journal of Bone & Joint Surgery*, vol. 93, num. 2, 113-120 (2011).

Salisbury, C., et al.: Effectiveness of PhysioDirect telephone assessment and advice services for patients with musculoskeletal problems: pragmatic randomised controlled trial. *BMJ: British Medical Journal* 346, (2013).

Sanford, J. A., Jones, M., Daviou, P., Grogg, K., Butterfield, T.: Using telerehabilitation to identify home modification needs. *Assistive Technology*, vol. 16, num. 1, 43-53 (2004).

Savery, C., & Graham, T.C.N. (2012). Timelines: simplifying the programming of lag compensation for the next generation of networked games. *Multimedia Systems*, 1 – 17.

Schmit, J. M., et al.: Dynamic patterns of postural sway in ballet dancers and track athletes. *Experimental Brain Research* vol. 163, num. 3, 370-378 (2005).

Scott, S. D., Allavena, A., Cerar, K., Franck, G., Hazen, M., Shuter, T., ... Scott, S. D. S. D. S. D. (2010). Investigating Tabletop Interfaces to Support Collaborative Decision-Making in Maritime Operations. In Proceedings of ICCRTS 2010: International Command and Control Research and Technology Symposium. Santa Monica, CA, USA, June 22-24.

Scott, S. D., Sheelagh, C., & Inkpen, K. M. (2004). Territoriality in collaborative tabletop workspaces. In Proceedings of the 2004 ACM conference on Computer supported cooperative work - CSCW '04 (pp. 294 – 303). New York, New York, USA: ACM Press.

S. D. Scott, K. D. Grant and R. L. Mandryk, "System Guidelines for Co-located, Collaborative Work on a Tabletop Display," in European Conference on Computer-

Supported Cooperative Work , Helsinki, Finland, 2003.

Segal, L.: Designing team workstations: The choreography of teamwork. Local applications of the ecological approach to human-machine systems, 2 (1995).

Selim, E., & Maurer, F. (2010). EGrid: supporting the control room operation of a utility company with multi-touch tables. In ACM International Conference on Interactive Tabletops and Surfaces - ITS '10 (p. 289). New York, New York, USA: ACM Press.

Seyed, T., Costa Sousa, M., Maurer, F., & Tang, A. (2013). SkyHunter: A Multi-Surface Environment for Supporting Oil and Gas Exploration. In Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces - ITS '13 (pp. 15–22). New York, New York, USA: ACM Press.

Sherbondy, A.J. et al. 2005. Alternative Input Devices for Efficient Navigation of Large CT Angiography Data Sets. *Radiology*. 234, (2005), 391–398.

Shneiderman, Ben. 1981. "Direct Manipulation: A Step Beyond Programming Languages." *SIGSOC Bull.* 13 (2-3). New York, NY, USA: ACM: 143.

Simon, Herbert A. 1956. "Rational Choice and the Structure of the Environment." *Psychological Review* 63 (2). American Psychological Association: 129.

Singh, V., Latiulipe, C., Carroll, E., Lottridge, D.: The choreographer's notebook: a video annotation system for dancers and choreographers. In: Proc C&C 11, pp. 197-206. (2011).

Snibbe, S.S. et al. Haptic techniques for media control. Proceedings of UIST 01. 3, (2001), 199.

Sodhi, R., Benko, H., Wilson, A.: Lightguide: projected visualizations for hand movement guidance. In: Proc CHI 2012, pp. 179-188. ACM (2012).

Statistics Canada: Canada's rural population since 1851: Population and dwelling counts, 2011 Census. (2012).

Stillman, B.C.: Making sense of proprioception: the meaning of proprioception, kinaesthesia and related terms. *Physiotherapy* vol. 88, no. 11, pp. 667-676. (2002).

Swanson, J.O. et al. Optimizing peer review: A year of experience after instituting a real-time comment-enhanced program at a children's hospital. *AJR. American journal of roentgenology*. 198, 5 (2012), 1121–5.

Szymanski, R., Goldin, M., Palmer, N., Beckinger, R., Gilday, J., & Chase, T. (2008). Command and Control in a Multitouch Environment. 26th Army Science Conference. Orlando, Florida. Retrieved from <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA503423>.

Tang, A., Pahud, M., Inkpen, K., Benko, H., Tang, J.C., Buxton, B.: Three's company: understanding communication channels in three-way distributed collaboration. In: Proc CSCW 2010, pp. 271-280 ACM (2010).

Tang, A., Tory, M., Po, B., Neumann, P., & Carpendale, S. (2006). Collaborative coupling over tabletop displays. In Proceedings of the SIGCHI conference on Human Factors in computing systems CHI 06 (Vol. pp, pp. 1181–1190). ACM Press.

Tang, J.C., Minneman, S.L.: VideoDraw: a video interface for collaborative drawing. *ACM TOIS* vol. 9, no. 2, pp. 170-184 (1991).

Tang, R., Yang, X., Bateman, S., Jorge, J., Tang, A. Physio@Home: Exploring Visual Guidance and Feedback Techniques for Physiotherapy Exercises. In: Proc CHI 2015, pp. 4123-4132.



- Thomassen, B. The uses and meanings of liminality. *International Political Anthropology* 2, 1 (2009), 5–27.
- Alice Thudt, Uta Hinrichs and Sheelagh Carpendale. The Bohemian Bookshelf: Supporting Serendipitous Book Discoveries through Information Visualization. In *CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2012.
- Tousignant, M., et al.: Patients' satisfaction of healthcare services and perception with in-home telerehabilitation and physiotherapists' satisfaction toward technology for post-knee arthroplasty: an embedded study in a randomized trial. *Telemedicine and e-Health*, 17(5), 376-382 (2011).
- Tousignant, M., et al.: A randomized controlled trial of home telerehabilitation for post-knee arthroplasty. *Journal of Telemedicine and Telecare*, vol. 17, no. 4, 195-198 (2011).
- Tremblay, M.S., Colley, R., Saunders, T.J., Healy, G.N., and Owen, N. Physiological and health implications of a sedentary lifestyle. *Applied Physiology, Nutrition, and Metabolism* 35, 6 (2010), 725–740.
- van Andel, P. Anatomy of unsought finding. serendipity: Origin, history, domains, traditions, appearances, patterns, and programmability. *The British Journal for the Philosophy of Science* 45, 2 (1994), 631–648.
- Velloso, E., Bulling, A., Gellersen, H.: MotionMA: motion modelling and analysis by demonstration. In: *Proc CHI 2013*, pp. 1309-1318. ACM (2013).
- Villar, N. et al. 2009. Mouse 2.0: multi-touch meets the mouse. *Proceedings of UIST 09*. (2009), 33–42.
- Vogel, D. and Baudisch, P. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. *Proceedings of CHI '07* (2007).
- Waldrop, M.M. *Complexity: The emerging science at the edge of order and chaos*. Simon and Schuster, New York, 1992.
- Wallace, J. R., Scott, S. D., & MacGregor, C. G. (2013). Collaborative sensemaking on a digital tabletop and personal tablets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13* (p. 3345). New York, New York, USA: ACM Press.
- Wang, F. and Ren, X. Empirical evaluation for finger input properties in multi-touch interaction. *Proceedings of CHI 09*. (2009), 1063.
- Wharton, C., J. Bradford, R. Jeffries, and M. Franzke. 1992. "Applying Cognitive Walkthroughs to More Complex User Interfaces: Experiences, Issues, and Recommendations." In *ACM Conference on Human Factors in Computing Systems (CHI)*.
- Wilson, Jeff, Judith M. Brown, and Robert Biddle. 2014. "Interactive Parallel Coordinates for Collaborative Intelligence Analysis." *School of Computer Science Technical Report TR-14-05*, Carleton Uni.
- Xiao, X., Ishii, H.: MirrorFugue: communicating hand gesture in remote piano collaboration. In: *Proc TEI 2011*, pp. 13-20. ACM (2011).
- Yarosh, S., Tang, A., Mokashi, S., Abowd, G.D.: Almost Touching: Parent-child remote communication using the sharetable system. In: *Proc CSCW 2013*, pp. 181-192 ACM (2013).
- Zeeman, E.C. Catastrophe Theory. *Scientific American*, 1976, 65–83.

H. Zhang, M. Korayem, You, Erkang and D. J. Crandall, "Beyond co-occurrence: Discovering and visualizing tag relationships from geo-spatial and temporal similarities," in ACM International Conference on Web Search and Data Mining, New York, 2012.







## **SURFNET / Designing Digital Surface Applications**

is a compendium of research findings from a Canadian research network that integrated innovative research in two critical areas –software engineering (SE) and human-computer interaction (HCI)– to identify critical requirements, design new engineering processes, and build new tools for surface-based application development. Funded by the Natural Sciences and Engineering Research Council of Canada (NSERC) from 2009 to 2015, SurfNet’s research clustered around three themes: Humanizing the Digital Interface, Improving Software Time to Market and Building Infrastructure for Digital Surfaces. Research was driven by the needs of four application areas: Planning, Monitoring and Control Environments; Learning, Gaming, New Media and Digital Homes; Software Team Rooms; and Health Technologies.