University of
South Australia

School of Information Technology and Mathematical Sciences

Division of Information Technology, Engineering and the Environment

Research Thesis for the Degree of Doctor of Philosophy

# Physical-Virtual Tools for Interactive Spatial Augmented Reality

Michael Robert Marner

Principal Supervisor: Bruce Hunter Thomas

Associate Supervisor: Christian Sandor

# Abstract

This dissertation presents a framework for developing user interfaces for spatial augmented reality applications, *Physical-Virtual Tools*, and novel interaction techniques based on this framework. Spatial augmented reality uses digital projectors to add computer generated images and information to objects in the real world. This technology has the potential to greatly improve processes in areas such as industrial design, architecture, manufacturing, maintenance, and training. To realise this potential, new interaction techniques must be developed that fit in with existing processes.

Physical-Virtual Tools user interfaces are constructed from physical tools, which the user interacts with to perform tasks. The physical design of a tool is tailored for specific kinds of interaction. Tools are projected onto just like other objects in the environment. This allows tools to be overloaded with several functions, with state information projected onto the tool itself. This removes the need for separate information screens, which may not be suitable in large, immersive systems. Two continua to aid in developing, evaluating, and comparing these user interfaces are presented. The Within System Tool Virtualisation Continuum is used to assess the complexity of a single system, and can be used to decide the number and nature of tools required for a system. The Inter-System Tool Virtualisation Continuum is used to compare the user interface complexity between systems.

Three demonstration applications featuring novel Physical-Virtual Tools user in-

terfaces have been developed for the industrial design domain. Digital airbrushing uses a two handed technique for digitally airbrushing onto physical objects with a physical-virtual stencil. This technique provides similar, but more flexible, functionality to a traditional airbrush, and allows designers to quickly evaluate product finishes. Augmented Foam Sculpting combines the act of physical mockup creation with CAD modelling. Designers are able to generate 3D CAD models by sculpting in foam with a hand-held, hot wire foam cutter, taking advantage of their existing skills. New interaction techniques for interior architecture and design tasks are presented. These techniques are embodied in a demonstration application, BuildMyKitchen, which allows architects to design and customise joinery such as kitchen cabinets, and the ability for clients to easily customise and preview aesthetic concerns in the design. Throughout the course of this research, a reusable SAR software framework has been developed. This dissertation discusses the architecture and features of this framework, and how the framework greatly improves the ability for researchers to build and evaluate SAR applications. In addition, this dissertation presents a novel modification to computer vision based finger tracking algorithms that allows for greater performance in SAR environments. The finger tracking hardware is also used to provide feedback to users when interacting with projected controls.

# Declaration

I declare that:

- this thesis presents work carried out by myself and does not incorporate without acknowledgment any material previously submitted for a degree or diploma in any university;
- to the best of my knowledge it does not contain any materials previously published or written by another person except where due reference is made in the text; and all substantive contributions by others to the work presented, including jointly authored publications, is clearly acknowledged.

Michael Robert Marner

Adelaide, March 2013

# Author Publications

The work presented in this dissertation has previously appeared in a number of publications, listed below.

- Marner, M. R., Thomas, B. H., and Sandor, C. "Physical-Virtual Tools for Spatial Augmented Reality User Interfaces". In: *Proceedings of the International Symposium on Mixed and Augmented Reality*. Orlando, FL, USA, 2009

- Porter, S., Marner, M. R., Eck, U., Sandor, C., and Thomas, B. H. "Rundle Lantern in Miniature: Simulating Large Scale Non-Planar Displays". In: *Proceedings of the International Conference on Advances in Computer Entertainment Technology*. Athens, Greece, 2009

- Marner, M. R. and Thomas, B. H. "Augmented Foam Sculpting for Capturing 3D Models". In: *Proceedings of the IEEE Symposium on 3D User Interfaces*. Waltham, MA, USA, 2010

- Marner, M. R. and Thomas, B. H. "Tool Virtualization and Spatial Augmented Reality". In: *Proceedings of the 20th International Conference on Artificial Reality and Telexistence*. Adelaide, South Australia, 2010

- Porter, S. R., Marner, M. R., Smith, R. T., Zucco, J. E., Thomas, B. H., and Schumacher, P. "Spatial Augmented Reality for Interactive Rapid Prototyping". In: *Proceedings of the 20th International Conference on Artificial Reality and Telexistence*. Adelaide, South Australia, 2010

- Porter, S. R., Marner, M. R., Smith, R. T., Zucco, J. E., and Thomas, B. H. "Validating Spatial Augmented Reality for Interactive Rapid Prototyping". In: *Proceedings of the 9th IEEE International Symposium on Mixed and Augmented Reality*. Seoul, Korea, 2010

- Maas, E., Marner, M. R., Smith, R. T., and Thomas, B. H. "Quimo: A Deformable Material to Support Freeform Modeling in Spatial Augmented Reality Environments". In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Singapore, 2011

- Smith, R. T., Marner, M. R., and Thomas, B. "Adaptive Color Marker for SAR Environments". In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Singapore, 2011

- Thomas, B. H., Von Itzstein, G. S., Vernik, R., Porter, S. R., Marner, M. R., Smith, R. T., Broecker, M., and Close, B. "Spatial Augmented Reality Support for Design of Complex Physical Environments". In: *Workshop on Interdisciplinary Approaches to Pervasive Computing Design*. 2011

- Marner, M. R., Smith, R. T., Porter, S. R., Broecker, M., Close, B., and Thomas, B. H. "Large Scale Spatial Augmented Reality for Design and Prototyping". In: *Handbook of Augmented Reality*. Springer-Verlag, 2011

- Marner, M. R., Broecker, M., Close, B., and Thomas, B. "Spatial augmented reality (SAR) application development system". Provisional Patent 2012903729. 2012

- Marner, M. R., Haren, S., Gardiner, M., and Thomas, B. H. "Exploring Interactivity and Augmented Reality in Theater: A Case Study of Half Real". In: *Proceedings of the International Symposium on Mixed and Augmented Reality*. Atlanta, Georgia, USA, 2012

- Marner, M. R. and Thomas, B. H. “Spatial Augmented Reality User Interface Techniques for Room Size Modeling Tasks”. In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Orlando, FL, USA, 2013

# Acknowledgements

I feel a little guilty about this, but I started thinking about these acknowledgements before I even had a first draft of this dissertation. However, the truth is I have a lot of people to be thankful for, whether or not this PhD adventure ended in success or burning failure, so I don't feel so bad for thinking about acknowledgements before I had anything to show for myself.

I shall begin by thanking my supervisor, Professor Bruce Thomas, and my associate supervisor, Doctor Christian Sandor. Bruce is a fantastic supervisor who has worked tirelessly with me discussing and refining research ideas, writing papers, talking about music and life in general, and demonstrating how to act in academia. He also has an uncanny ability to know exactly what has to happen next in a research project, and what to say in order to keep things moving and on track, no matter how confused I may be at the start of a meeting. Christian has been an enormous help, particularly at the beginning and end of my PhD, by teaching me how to form and articulate ideas, as well as guiding me in figuring out what good research actually is. So thank you, Bruce and Christian. I would also like to thank my honours supervisor, Doctor Wayne Piekarski, whose example inspired me to begin a PhD in the first place. I would like to thank my reviewers, Professors Gregory Welch and Henry Gardner. Their efforts in reviewing this dissertation have made it a much better work than it was when it was sent to them, so I thank them for their expert advice and criticism.

I've had the benefit of working for the last four years in the Wearable Computer Lab. The lab has been lucky enough to have a continual stream of researchers who are not only highly motivated, clever, and creative, but also genuinely nice people to be around. Without the people in the lab, the last few years would not have been nearly as fun. So, I thank the current members of the WCL, Ross Smith, Jo Zucco, Stewart Von Itzstein, Markus Broecker, Thuong Hoang, James Walsh, Andrew Irlitti, Tim Simon, Daniel Jackson, and Ash Doshi, for making the lab what it is. I would also like to thank past members of the WCL who have since moved on to other things. Ben Avery, Peter Hutterer, and Aaron Stafford were all in the final stages of their PhDs when I was just beginning. Their example inspired me on how to work in the lab. Benjamin Close was an amazing help to me. I would like to thank Ben for his knowledge and willingness to debate software architecture and programming ideas with me. I think it's safe to say I would still be trying to get code to compile if it weren't for Ben's help. I would also like to thank Robert Kong, Shane Porter, Mark Rebane, Georg Grossmann, Fabian Butzow, and Andrew Cunningham for their help and support.

During the course of my PhD I was fortunate to be able to work on two theatrical works, taking tools and technology developed in the lab and applying them to something completely different. I worked with some incredibly skilled people on these projects, and it really opened my eyes to what is possible with SAR technology. So, I thank the *Half Real* and *BLACK* teams: Cameron Goodall, Duncan Graham, Geoff Cobham, Chris More, Chris Petridis, Tyson Hopprich, Ray Gardiner, Aaron Herczeg, Katherine Fyffe, David Heinrich, Alirio Zavarce, Amber McMahon, Caitlin Byrne, Steve Tilling, and Monica Hart, Caleb Lewis, Cindi Drennan, Gavin Clarke, Jamie Harding, Jessica Foster, Joel Panther, John Crouch, James Lloyd-Smith, Monica Hart, Marty Hopkins, and Rowan Lee.

Away from work I am surrounded by some incredibly awesome people. I would

like to thank my friends, including but not limited to Domenic, TJ, Courtney, and Daniella Trimboli, Craig and Darren Mumford, Ry Wilkin, Jess Curtis, Robyn Clasohm, Sim and Trevor Jones, Johnny McIntyre, Heidi and Scott Rydquist, Steve South, Naomi Gordon, Callum Berry, Mark Agnew, Lisa Hill, Travis Longbottom, Danielle Parise, Sally Vine, Alicia O'Donoghue, Gemma, John, and Tara Butler, Zarah Hage, and Josh and Lisa Reichstein. You are all legends (Dave), and I don't know what I would do without you.

I thank my parents, Robert and Joyce for their unending love and support, and for understanding that a "real job" was not the right path for me to take. I would like to thank my brothers, Jason, Darren, and Andrew for being, you know, brothers, and the rest of my family. I also thank the Charman clan for so enthusiastically welcoming me into their family over the last three years.

Finally, I would like to thank my wife, Renée. You are incredibly fun, loving and supportive, and you even read my thesis! I love you, and thank you for being in my life.

# Contents

# List of Figures

# Code Listings

# List of Abbreviations

| | |
|---|---|
| AR | Augmented Reality |
| BLACK | If There Was A Colour Darker Than Black I'd Wear It |
| CAD | Computer Aided Design |
| CAVE | Cave Automatic Virtual Environment |
| CGAL | Computational Geometry Algorithms Library |
| CSG | Constructive Solid Geometry |
| DAG | Directed, Acyclic Graph |
| DOF | Degrees Of Freedom |
| FBO | Frame Buffer Object |
| GLSL | OpenGL Shader Language |
| GPS | Global Positioning System |
| GPU | Graphics Processing Unit |
| GUI | Graphical User Interface |
| HMD | Head Mounted Display |
| HSV | Hue, Saturation, Brightness |
| IR | Infra-Red |
| IS-TVC | Inter-System Tool Virtualisation Continuum |

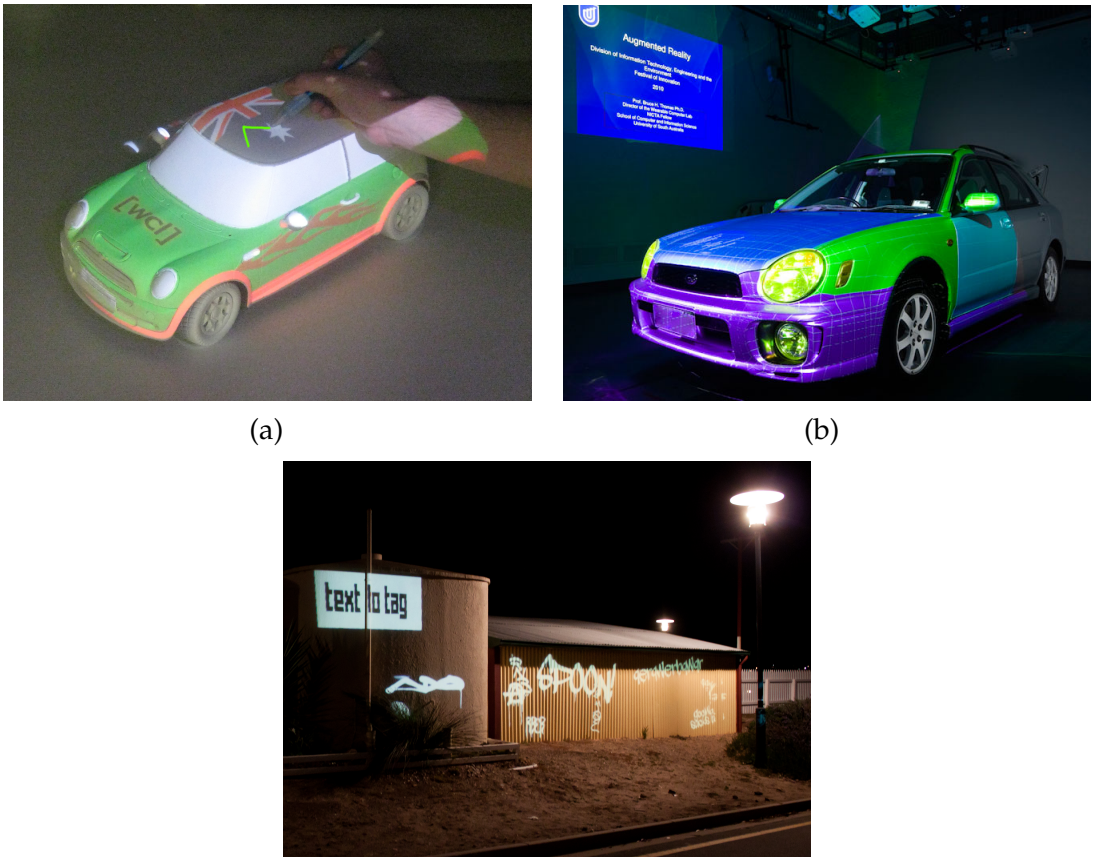
| | |
|---|---|
| LED | Light Emitting Diode |
| MR | Mixed Reality |
| PIP | Personal Interaction Panel |
| PVT | Physical-Virtual Tools |
| RGB | Red, Green, Blue |
| SAR | Spatial Augmented Reality |
| TUI | Tangible User Interface |
| TVC | Tool Virtualisation Continuum |
| VE | Virtual Environment |
| VR | Virtual Reality |
| WACL | Wearable Active Camera with Laser pointer |
| WARP | Workbench for Augmented Rapid Prototyping |
| WS-TVC | Within-System Tool Virtualisation Continuum |
| XML | Extensible Markup Language |
| YUV | Luminance (Y), Chrominance (UV) |

# 1

# Introduction

Augmented Reality (AR) enhances our experience of the world by adding computer generated information to it in real time [Azu97]. Head Mounted Displays (HMD) [Sut68; AB94] and hand-held devices [WS07] can be used to augment the user's view of the world. Alternatively, projectors can be used to project computer generated information onto objects in the physical world, a technique known as Spatial Augmented Reality (SAR) [RWF98]. SAR allows computer graphics to be projected onto physical surfaces in the environment. SAR systems can vary in scale, with sizes ranging from small table-top systems (Figure 1.1(a)), to the size of a car (Figure 1.1(b)), through to an entire building (Figure 1.1(c)). This dissertation presents new user interface and interaction techniques for SAR systems.

Using projectors, spatial augmented reality modifies the appearance of physical objects by projecting directly onto them [RWF98]. This is a fundamental difference between SAR and other AR display technologies, such as HMDs and hand-held devices. SAR systems actually change the visual appearance of the physical world with computer controlled illumination, where as other approaches augment the user's view of the world [Lan96]. The projective nature of SAR presents several advantages and limitations compared to other AR displays. First and foremost, the




(a) (b)

(c)

**Figure 1.1:** SAR systems developed at different scales. Small table-top systems allow for direct manipulation of both the physical and virtual objects (a). Larger scale systems provide one or more fixed physical objects, where users employ within arm's reach techniques to manipulate virtual information (b). Larger systems can support projections on entire buildings, and would typically employ at a distance interaction techniques (c).

user is not required to wear or carry the display. Compared to hand-held AR, users of SAR systems have both hands free for interacting with the system and objects in the world, raising the possibility of richer interaction techniques [RL01]. SAR is also better suited to collaborative tasks such as design review meetings; users are not required to wear cumbersome equipment over their faces, which would reduce their ability to communicate with each other [PNJ10]. Additionally, the number of people able to experience a SAR environment is limited by the physical size of the space, not by the amount of equipment available. SAR benefits from the natural

depth cues and passive haptic feedback provided by physical objects.

One of the downsides to SAR is the need for physical objects to project onto. Hand-held and head-mounted displays have an image plane at a known location on which to draw virtual objects [Azu+01]. This ability to render information at arbitrary locations in "mid air" is not yet possible with SAR technology. While devices such as the Fog Screen [Rak+05] provide semi-transparent projection surfaces using water vapour, there is no way to render at arbitrary locations. This forces SAR applications to focus on augmenting existing objects, rather than adding purely virtual objects to the world. Another potential downside to SAR is the difficulty in displaying view-dependent rendering effects, such as transparency and accurate reflections. This can be accomplished for a single user if their head position is tracked [Bim+05b]. As SAR changes the physical appearance of objects, rather than users' individual views, accomplishing these rendering effects for several users simultaneously requires additional technology such as active shutter glasses [Agr+97].

## 1.1 Problem Statement

Spatial augmented reality has the potential to greatly improve processes in fields such as industrial design [Ver+03], architecture, manufacturing[Sch+08], maintenance, and training [Sug+08]. While not an exhaustive list, these domains are particularly suited to SAR because they typically involve key physical objects, such as design mockups, which are the focus of the user's attention. One of the barriers to adopting SAR technology in these fields is the lack of suitable interaction techniques tailored to the peculiarities of SAR projection. Many existing AR interaction techniques are unsuitable for SAR. Often, they are concerned with manipulating purely virtual objects in an AR view of the world [Kat+00; SPS01]. As previously stated, SAR requires physical objects to project onto. Therefore, interaction will most likely

involve either virtual information located on the physical objects, or the physical objects themselves. The display itself is a core part of many interaction techniques for HMD [Pou+96; FHZ96; PS06] and handheld AR [OF09; San+10; Bar+12]. The interaction techniques take advantage of the fact the display is coupled to the user, either in front of their eyes or in their hand. Other interaction techniques developed for AR and VR take advantage of the image plane of the display to place virtual objects in mid air [Fei+93; SCP95; PSP99; PT02; Jun+04], an approach unsuitable for projection.

For these reasons, new interaction techniques need to be developed for SAR that take advantage of the display technology. In particular, physical surfaces are a fundamental component of any SAR system. Interaction techniques should make use of these to provide passive haptic feedback to the user. In addition to this, interaction techniques for SAR should minimise the effect of the medium's disadvantages. SAR user interfaces should not assume a specific location for the user, and should be usable from any viewpoint from which the user can see the visualisation. SAR user interfaces should support collaborative tasks, taking advantage of the fact that SAR displays are independent of the user.

### 1.1.1 Research Questions

This dissertation addresses a number of unsolved problems for SAR. The investigation presented in this dissertation began with the following two questions:

- How can a user intuitively control and interact with SAR systems of a variety of scales? SAR systems can be created ranging in scales from single projector table-top systems, through to immersive room sized environments, scaling up to an entire multi-story building. At each scale the user needs to be able to control the system and manipulate virtual information.

- How can SAR be put to use in the industrial design domain? Industrial design is a compelling application area for SAR. There is potential for SAR to greatly improve design processes and communication between designer and client. Industrial designers work with physical prototypes throughout the design process. The physical objects, so important for SAR systems, are readily available in industrial design.

In order to begin addressing these questions, this research has focussed on several sub-projects to develop interaction techniques for SAR systems. Specifically, this dissertation answers the following questions:

- How can SAR be used to aid a designer early in the design process to design the visual appearance and finish of the surfaces of a product?
- How can SAR be used to speed up iteration time early in the design process, particularly in the transition from hand made mock-ups to refined CAD designs?
- How can a designer specify dimensions in a SAR system, such as the width of an object or the distance between objects?
- How can preset designs be loaded, viewed, and saved in a SAR system without the use of a keyboard or mouse?
- How can a user be notified when activating projected controls such as virtual buttons and other widgets?

### 1.1.2 Research Goals

The core goal of this research has been to answer the questions defined above. In general, the goal has been to further the state of the art in user interfaces and interaction techniques for SAR. In order to do this, it has been an important goal of

this research to investigate the industrial design process, and assess where and how SAR technology could be utilised. This has been accomplished by working with industrial designers and students at the School of Art, Architecture and Design, University of South Australia, to find ways SAR can be put to use in the design process. Deploying SAR should fit in as best as possible with existing tasks and processes. Interaction should take advantage of the intrinsic physical nature of SAR, and of existing skills, so as to reduce the burden on users of learning the new systems. To this end, the demonstration applications presented in this dissertation have been developed for use by designers in the design process. Where possible, the user interface elements and interaction techniques have been generalised, in order to be useful in other application domains making use of SAR technology.

## 1.2 Contributions

This dissertation presents a number of research contributions in the field of augmented reality in general, and interaction techniques for SAR in particular. These contributions, while developed for SAR, could also be applied to other AR display types. These contributions are outlined below:

- Physical-Virtual Tools (PVT), a conceptual framework for creating, evaluating, and comparing SAR user interfaces. In these systems, the user interface (UI) is comprised of one or more tools. The tools are overloaded with different functions, with state information projected onto the tools themselves. Two continua are defined to aid in designing and comparing PVT user interfaces. The Within System Tool Virtualisation Continuum (WS-TVC) visualises the complexity of tools in a single system, and is used when deciding the number and nature of tools required for a user interface. The Inter-System Tool Virtualisation Continuum (IS-TVC) is used for comparing user interfaces of different

systems [MT10a].

- Digital Airbrushing, a two handed interaction technique for applying finishes and designing paint jobs for products, and the algorithms that enable this technique. The airbrushing technique uses a PVT user interface consisting of a physical airbrush and a physical, virtualised stencil, allowing the designer to paint using similar techniques to real airbrushing [MTS09; Maa+11].

- Augmented Foam Sculpting, an interaction technique that bridges two important parts of the industrial design process: hand made mock-up creation and CAD modelling, and the algorithms that make this technique possible. Designers are able to produce 3D digital models by sculpting with foam. In addition to the modelling technique, two SAR visualisations are provided to aid the designer in their work. [MT10b].

- BuildMyKitchen [MT13], an application for interior architecture and kitchen design. This application is intended for use by designers when specifying the design of cabinetry and for previewing and refining designs with clients. This application includes a two handed PVT technique for resizing components, and a PVT approach to loading and saving designs, and selecting materials.

- An improvement on existing colour marker based tracking approaches for use in SAR systems. An active marker is used, with the colour of the LED continuously updating to maintain robust tracking as the SAR projections change. The LED is also used to provide visual feedback to users as they interact with virtual objects [SMT11].

- A description of a software framework designed specifically for building SAR systems [Mar+12a]. This framework reduces the work required to build SAR applications, and the discussion includes several insights which are useful for other developers.

This dissertation focuses on integrating SAR into industrial design and architecture domains. In addition to the contributions listed above, I have also researched and deployed SAR systems into theatre productions. During the course of my PhD, I spent four months working on *Half Real*, a theatre experience featuring audience interactivity and SAR technology [Mar+12b]. This theatre show is the first of its kind to feature a 3D modelled stage, actor tracking, and a dynamic real-time projection system that reacts to audience and actor actions. Half Real successfully completed a three week sold-out season during the Melbourne Festival in 2011. Projects such as Half Real would not be possible to complete without the use of SAR, and the contributions listed above directly contributed to the development of Half Real. A detailed description of this work is included in Appendix A.

## 1.3 Dissertation Structure

The remainder of this dissertation is structured as follows: The following chapter presents an overview of background research in the areas of augmented and spatial augmented reality technology and applications, Tangible User Interfaces (TUI), and physical user interface prototyping. As later chapters describe SAR applications intended for industrial designers, Chapter 2 also provides an overview of the industrial design process.

Following this, Chapter 3 introduces Physical-Virtual Tools, a new methodology for creating SAR user interfaces. The two continua for classifying and comparing PVT based user interfaces are presented, and the SAR applications described in Chapter 2 are compared in the context of the continua.

The next three chapters present the SAR applications developed as part of this investigation. Each of these applications feature a PVT based user interface, and each chapter presents new tools and interaction techniques developed to support

the system. Chapter 4 describes Digital Airbrushing, a system that allows designers to paint with projected light and a physical-virtual stencil. Chapter 5 presents Augmented Foam Sculpting, a system which allows designers to create 3D models of design mockups by sculpting with foam. Chapter 6 presents a SAR system for helping architects design and evaluate kitchen joinery and other internal architecture with the client. Chapter 7 describes the software framework developed through the course of this research, and provides several insights useful for other developers and researchers building SAR applications. Finally, this dissertation concludes with a look towards future research directions.

# 2

# Background

This chapter provides an overview of previous work that has relevance to this dissertation. The chapter begins with an overview of augmented reality, comparing the different display technologies used to build AR systems. A detailed discussion of spatial augmented reality research follows this overview. In addition, this chapter provides an overview of related display technologies, but for various reasons should not be considered SAR technology. Tangible user interface based systems are investigated, as they share several characteristics with interactive SAR systems. A brief overview of tracking technologies is presented. While this dissertation is not focussed on tracking systems, tracking is an important enabling technology of interactive SAR systems. This chapter concludes with an overview of the industrial design process, with a focus on prototyping.

## 2.1 Augmented Reality

As previously mentioned, augmented reality enhances our perception of the world be adding computer generated information to our experiences. Augmented reality was first described by Ivan Sutherland as his Ultimate Display [Sut65]. This hy-

pothetical device could control the existence of matter to produce virtual objects indistinguishable from the real world. Sutherland was also the first to build an AR system, featuring the first head-mounted display. AR systems using HMDs share many properties with Virtual Reality (VR) [Bro99]. However, the goal of AR is to augment the physical with virtual, rather than immersing the user in an entirely virtual environment. As defined by Azuma [Azu97], an augmented reality system should:

- combine the real and virtual,
- be interactive in real time, and
- be registered in 3D.

Although not exclusively a visual medium [DHS02], most AR research has focused on augmenting the visual appearance of the environment. The three main AR display technologies are head mounted displays, hand-held devices such as mobile phones, and spatial displays such as projectors. HMD based AR visualisations often add menus and other information at a fixed location on the image plane of the display [Fei+93], as shown in Figure 2.1(a), or placing virtual objects so that they appear as part of the real world. Tinmith [PT02] is one of many examples of such a system, and is shown in Figure 2.1(b). Interaction techniques for HMD based AR benefit from the users wearing the display on their heads. This leaves the user's hands free to interact with both the system and objects in the physical world. However, communication with others is currently hindered by the display covering the user's face. Technology may improve in the future, and projects such as Google Glass[1], though not an AR display itself, show possible future directions for HMD devices.

[1] https://plus.google.com/+projectglass

(a)


(b)

**Figure 2.1:** HMD based AR supports placing menus and other items on the image plane in front of the user, such as in Windows On Your World (a), and into the user's view of the physical world (b). Images courtesy of Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon Computer Graphics and User Interfaces Lab Columbia University, and Thuong Hoang, University of South Australia.

As technology has advanced, it has become possible to implement AR systems using hand-held devices such as mobile phones [Wag07]. Today's mobile phones include GPS, orientation sensors, digital compasses, and powerful graphics capabilities. The core advantage hand-held AR has over HMD based systems is the ubiquity of smart-phones; users do not have to wear or carry additional equipment.

Hand-held systems present a *magic lens* [Bie+93] style AR view, with the user choosing when to view the augmented world. This is demonstrated in Figure 2.2, which shows Wikitude[2], a mobile AR browser. The interaction techniques possible with hand-held AR are limited by the fact one of the user's hands is required to hold the device. Hand-held AR typically employs touch-screen technology, with the display forming an important part of interaction techniques. The display device also needs to be held a comfortable distance from the user's eyes. This reduces the ability for interaction techniques to involve the user's other hand in the augmented view.


**Figure 2.2:** Handheld AR presents a window into the augmented world. This image demonstrates Wikitude, an AR browser for mobile phones, showing nearby restaurants.

Spatial Augmented Reality [RWF98] utilises digital projectors to augment the physical world with projected information. Projectors are calibrated so that the projected images are correctly aligned with physical objects [RWC99]. Where as HMD

[2]http://www.wikitude.com

based and hand-held AR augment the user's view, SAR places computer graphics onto actual surfaces in the physical environment. Therefore, SAR environments can be experienced by multiple users simultaneously, without the need for users to carry or wear equipment. Unlike HMD and hand-held AR, SAR requires physical objects to project onto. In addition, SAR systems require a more detailed geometric model of the environment. This is due to the fact that a SAR system needs an understanding of the physical surfaces of objects in order to project onto them, not just geometric locations of objects of interest, as is often the case in other AR systems [Fei+97; SK08]. SAR is described in more detail in Section 2.2.

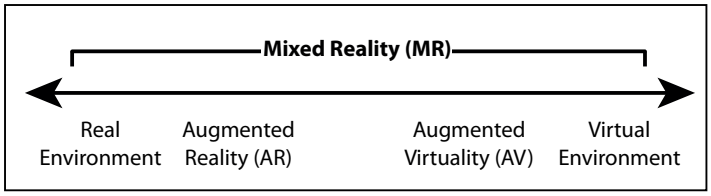



**Figure 2.3:** Milgram's Virtuality Continuum

Milgram's Virtuality Continuum [MK94] places AR in the context of other mixed reality displays. The continuum is depicted in Figure 2.3. On the left is the real environment, with no augmented information. On the right is an entirely virtual environment, with none of the real world present. Agumented reality is placed between these two extremes; the user experiences the real world with some virtual information added. As Milgram published his continuum in 1993, it unfortunately does not differentiate between different AR display technologies. However, each of the display technologies described above have different capabilities and limitations. HMD based systems are able to render virtual objects at arbitrary 3D locations into the user's field of view. Handheld devices place virtual objects at arbitrary locations in the camera's image which is rendered to its display. However, these virtual objects are not shown directly in the user's field of view. Instead, the handheld device is an augmented window, akin to a magic lens [Bie+93] which contains the virtual

information. The user can choose when to look at the augmented world. SAR again places virtual objects directly into the user's field of view. However, the nature of projection means SAR is unable to place virtual objects in arbitrary locations; SAR can not render objects in "mid air". In addition to arbitrary 3D locations, HMD and hand-held devices are also able to render information, such as menus, at a predictable location on the display. This is difficult to accomplish with SAR, because graphics can only be rendered on available surfaces in the environment.

## 2.2 Spatial Augmented Reality

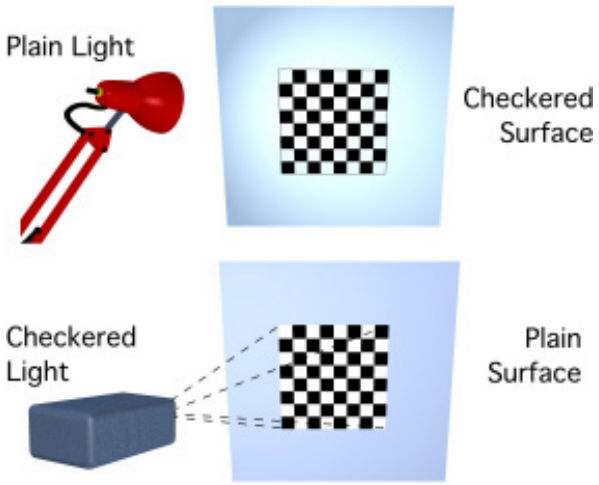



**Figure 2.4:** A patterned surface illuminated with a plain light is equivalent to a plain surface with a pattern projected onto it. Shader Lamps takes advantage of this insight to change the appearance of objects. Image courtesy of Greg Welch.

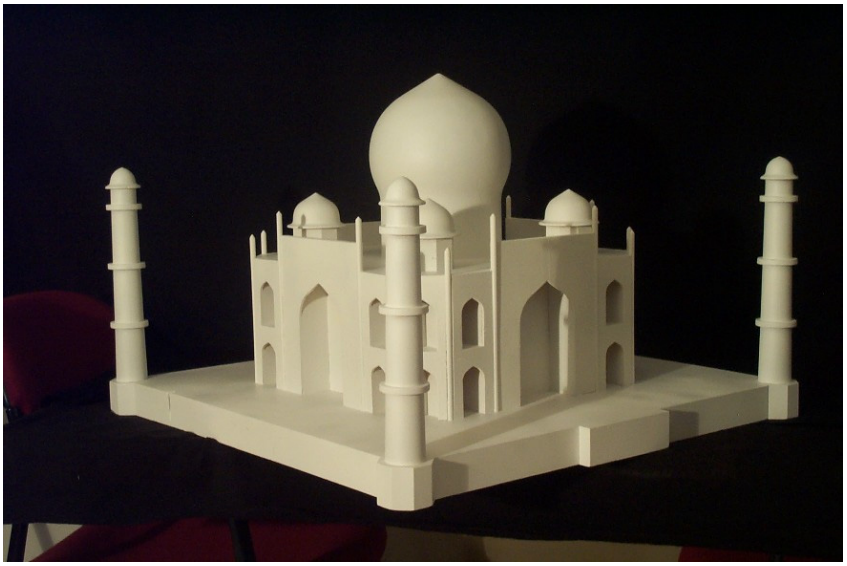
Spatial augmented reality has its roots in the vision outlined by The Office of the Future [Ras+98]. The Office of the Future describes an office environment where all surfaces, such as the walls, desks, and floor, are potential displays. Three dimensional models of people and objects in the room are captured in real time using cameras and structured light, allowing for 3D telepresence. Augmented Surfaces [RS99] allows users to seamlessly transfer digital information among computers, and other surfaces in the environment through the use of projectors. The DigitalDesk [Wel93]

is another forerunner to spatial augmented reality. The DigitalDesk is like a standard office desk, but with a graphical user interface projected onto the desk's surface, and cameras to capture user actions. Input for computer applications can be selected from printed documents. For example, selecting numbers on a printed sheet as input for a projected virtual calculator.

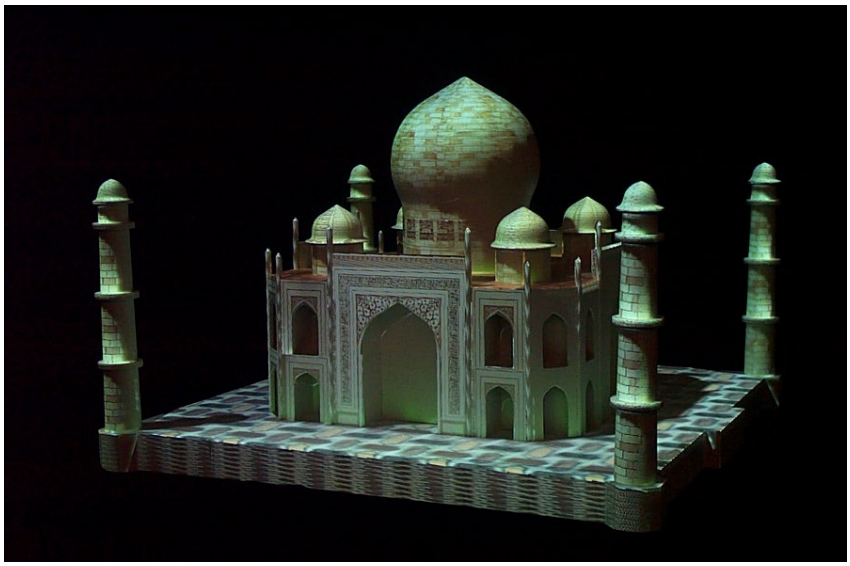
Shader Lamps [Ras+01] extends the SAR concept to not only augmenting surfaces with information, but changing their appearance entirely. A key insight is that a plain light illuminating a patterned surface is equivalent, from the user's viewpoint, to projecting a pattern onto a plain surface. This concept is illustrated in Figure 2.4. Shader Lamps takes advantage of this insight to give white painted objects entirely new appearances using projectors, as shown in Figure 2.5. Using these techniques, a variety of visual effects can be created. Multiple projectors can be used to illuminate objects from all sides. Most SAR research draws from the techniques introduced by Shader Lamps.

Dynamic Shader Lamps [BRF01] demonstrates how SAR systems can be created with movable objects. Using a tracked object and stylus, users are able to paint with projected light. A simple paint palette is projected onto the table. The paint colour is chosen by touching the palette with the stylus, with the current paint colour projected onto the stylus itself. The system requires a 3D model of the object in order to correctly project the paint, even as the object is moved. The virtual model is transformed according to data from the tracking system, ensuring the projected paint appears correctly on the surface.

An important consideration for many SAR systems is the realism or believability of the projected graphics. In virtual environment research, *presence* is "the experience of being in one place or environment, even when situated in another" [WS98]. This concept does not translate well to SAR, since the user is placed in an environment with physical objects, which can be augmented with computer graphics;



(a)



(b)

**Figure 2.5:** Shaderlamps. (a) A plain coloured model. (b) Geometrically aligned projected textures. Images courtesy of Greg Welch and Ramesh Raskar. Taj Mahal model by Linda Welch.

the user is not immersed in a virtual world. To address this, Stevens et al. [Ste+02] introduce the idea of *object presence*. Object presence is "the experience that a particular object exists in the user's environment, even when that object does not". Object presence was validated against SAR models, indicating that users accept the projected virtual objects. Bennett and Stevens [BS05] found that object presence is

reduced when objects are touched by the users. The authors argue reasons for this possibly include shadows cast by the user onto objects, projections falling on user's hands, and the lack of tactile feedback for specific material properties. However, other research suggests users can cope with shadows when multiple projectors are used [Sum+05]. Multiple projectors from different positions reduces blackout shadows on objects, since one projector can compensate for the other. Techniques for shadow removal in SAR also prevent projections falling on users' hands [But08].

### 2.2.1 SAR for Industry

Zaeh and Vogl [ZV06] show how a hybrid SAR/tablet system can be used for programming motion paths of industrial robots. A laser projector is used to project motion trajectories and target coordinates into the environment. These trajectories can be directly manipulated using a tracked stylus. The tablet interface is used to show 3D previews of the paths the robotic arms will take based on the trajectories entered by the user. An evaluation of this system indicated an 80% reduction in programming time compared to standard authoring techniques.

Schwerdtfeger et al. [Sch+08] present results of using laser projectors in industrial settings, such as indicating weld marks on a production line or for quality assurance tasks. Two projector configurations were considered, head mounted and tripod mounted. The head mounted projector increases the area that can receive augmentations, and reduces occlusions caused by the user. However, the equipment available was too heavy to be practical. In addition, the head mounted projector requires very fast, robust tracking. The highly specular surfaces, such as steel, common in industrial automotive settings are another problem. The highly reflective surfaces reduce the visual quality of the projected graphics.

### 2.2.2 SAR for Design

There is great potential for SAR to improve tasks in industrial design and prototyping. User interfaces, control panels, and material finishes can be previewed on physical mockups using projectors. Changes can quickly be made by modifying the projections.

Low et al. [Low+01; RL01] have successfully used SAR to create reconfigurable life-sized dioramas. Complex environments, such as interiors of buildings are constructed from inexpensive foam blocks. Projectors are used to create the visual appearance of the surfaces, with view-dependent rendering effects. Users are able to walk through and interact with the physical-virtual environment. This technique can be used to simulate real places, or to preview new designs.

Kurz et al. [Kur+07] demonstrates techniques for building interactive SAR systems for architectural environments. A custom pan-tilt-zoom camera system is fitted with an additional fish-eye camera and laser pointer. The system is placed in an environment, and the laser/camera pair automatically captures the geometry and visual appearance of the environment. Handheld laser pointers are then tracked using the camera system and used for interacting with the SAR projection system.

WARP [Ver+03; Ver+05], shown in Figure 2.6, is a system that allows designers to preview finishes on vehicles. A 3D printed model of the design is created, with projectors providing the details and material appearance. A graphical user interface is projected onto a wall next to the model, which the designer interacts with using a keyboard and mouse. This kind of SAR design system can be extended to record design review sessions in real time [VHN09]. In addition to video, data from the SAR system can be recorded, such as annotations. This provides a richer set of information than would be captured if designers made handwritten notes, because there is no possibility of ambiguity. The data captured is exactly what happened during the review.

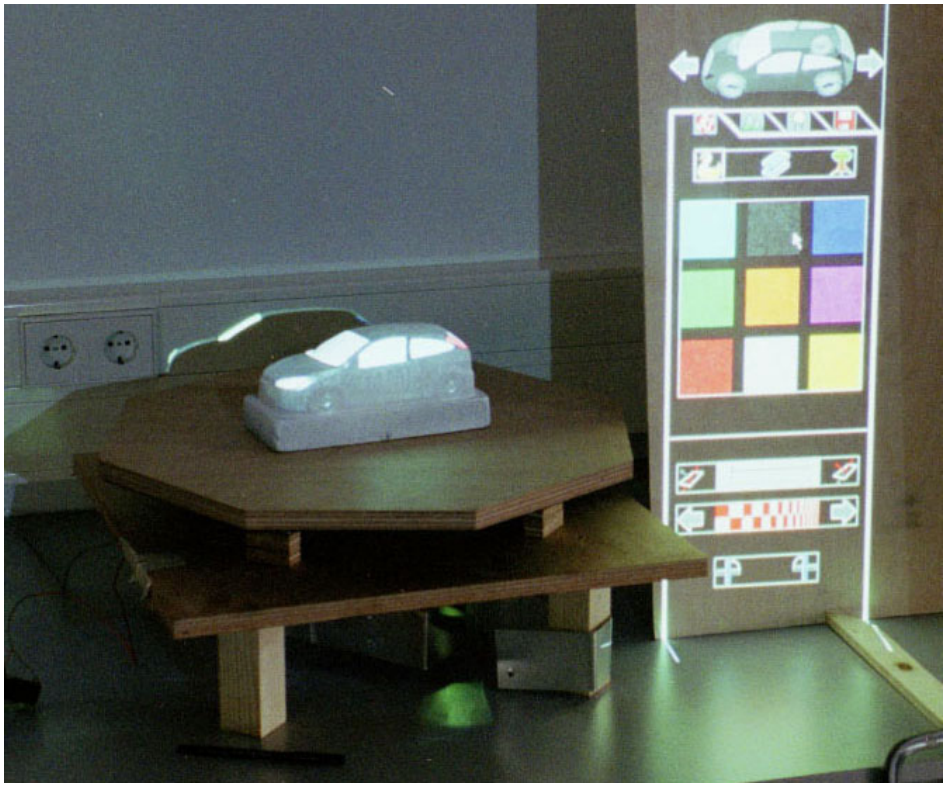
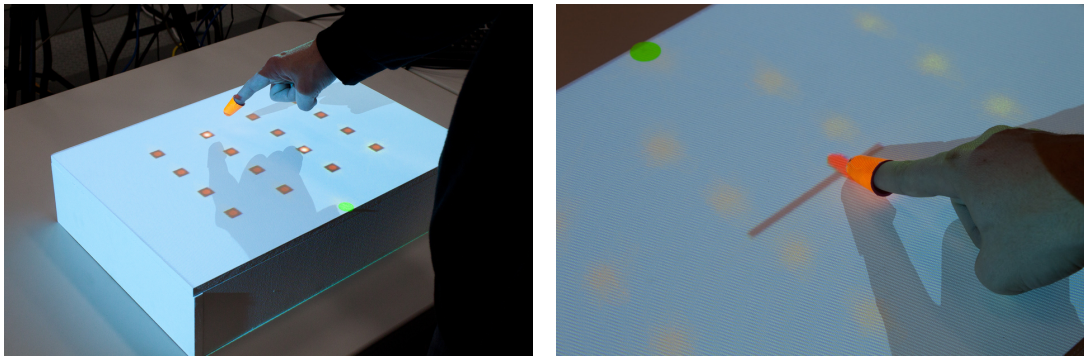


**Figure 2.6:** The WARP Prototyping system, with graphical user interface projected to the right of the augmented prototype. Image courtesy of Jouke Verlinden.

Porter et al. [Por+10a; Por+10b] investigated the use of SAR for virtual controls, such as buttons. By tracking the user's finger, the virtual controls can be made interactive. Figure 2.7 shows a user interacting with projected buttons and a projected slider. The advantage of virtual controls is the ease in which the control panel layout can be changed. The ability for users to interact with virtual buttons was compared to equivalent physical buttons. Results show virtual buttons were slower to interact with than physical ones. However, the performance of the virtual buttons was acceptable for design and prototyping scenarios. Reasons for the result include the performance of the tracking system, and the lack of tactile feedback when buttons were activated. Simon et al. [Sim+12] build on this work by adding "tangible buttons" to the virtual control panels. These buttons are simple plastic disks with RFID tags embedded inside. The buttons provide tactile feedback to the user, and can easily be reconfigured by changing their location. The user interacts with these buttons

using a glove with an embedded RFID reader. When touched, the glove communicates the ID of the pressed button to the host computer.



(a) (b)

**Figure 2.7:** SAR projected controls. (a) The user interacts with projected buttons. (b) The user interacts with a projected slider.

### 2.2.3 SAR for Instruction

SAR is a useful technology for in-situ instruction in a variety of different domains. The main advantage of SAR in these situations is that the user is not required to wear or carry equipment. This frees the user to focus on their task. CADcast [PI01] uses projectors to display assembly instructions onto a user's work area. A user assembles an object by lining parts up with the projected model of the finished product. Seo et al. [Byu+07] investigate the use of SAR for medical surgery applications, with the physician using a tracked stylus to mark locations and paths which are projected onto the patient. SAR has also been used for billiards instruction [Sug+08]. A projector/camera system is placed above the billiards table, and computer vision techniques are used to extract the positions of billiard balls from the camera images. Instructions for how the next shot should be made are then projected onto the surface of the table. OASIS [Zio+10] combines a depth camera with a projector to provide gesture based user input for table-top SAR applications. The depth camera is also used to detect objects in the environment [Zio+11]. *Kitchen*

*OASIS* demonstrates this framework being put to use as a cooking application, with recipes projected onto kitchen work surfaces.

GestureMan [Kuz+00] is a robotic SAR device for remote instruction. Consisting of a pan-tilt controllable laser pointer and camera mounted on a motorised base, GestureMan allows a remote expert to instruct a local worker. The main limitation of GestureMan is the single laser dot used for visual output. Animatronic Shader Lamps Avatars [Lin+09] are another approach to remote instruction and communication. An animatronic model human is augmented in real time with the appearance of a remote user. The appearance and geometry of the remote user is captured in real time using several cameras. The geometry is warped to match that of the animatronic model, so that the projections are correctly aligned to the model's surface. An alternative to capturing the appearance of the remote collaborator is to use a non-realistic cartoon appearance [Wel12]. The motion of the collaborator's face is captured, and this tracking information is applied to the cartoon model. The goal of this approach is to avoid the *uncanny valley* effect [Mor70] associated with using graphics that are more lifelike.

### 2.2.4 SAR For Information Presentation & Entertainment

SAR is a compelling candidate for information presentation on physical objects. Projected information can be seen simultaneously by many viewers. The display equipment can be permanently mounted out of reach, making SAR suitable for art installations and museums. SAR has been used to overlay computer generated information to paintings [Bim+05a]. Several effects are implemented, including overlaying earlier versions of works, colour correction based on ambient light, presenting information about the work, and changing the appearance to give the illusion of different materials. *Cartoon Dioramas in Motion* [RZW02] demonstrates how SAR can simulate motion and non-photorealistic rendering effects. The illusion of motion is created

by modifying textures and virtual lighting.

**Figure 2.8:** IncreTable is a tabletop mixed reality game system. Image courtesy of Maki Sugimoto.

IncreTable [Lei+08], shown in Figure 2.8, is a tabletop mixed reality gaming system. IncreTable allows interaction between physical objects, such as dominoes, and virtual objects projected onto the table. The table itself is a rear-projected display. Users interact with the system using a tracked pen, or by moving physical components. Physical robots are also supported by the system, which sense fiducial markers rendered on the table. These robots are either controlled by the system, or by players, using a gamepad. Jones et al. [Jon+10] describe another tabletop SAR gaming platform. Players can design an environment in which to play from foam blocks. The physical layout of the blocks is captured using structured light techniques. From there, players can insert game objects using a tracked stylus. Once the "level" is constructed, the game can be played.

### 2.2.5 Projector Calibration

Projecting graphics spatially aligned to physical objects requires a calibrated projector and a 3D understanding of the environment. Calibrating the projector involves calculating the *intrinsic* and *extrinsic* parameters of the projector. Intrinsic parameters include the projector's horizontal and vertical field of view, principal point, and focal length. The extrinsic parameters consist of the projector's position and orientation in the world [Zha00]. Calculating the intrinsic parameters requires finding correspondences between projector pixels and known 3D locations in the physical world. Raskar et al. [Ras+98] describe the algorithm for calculating the calibration parameters from a set of correspondences. These approaches are further described by Bimber and Raskar [BR05].

Several techniques have been developed for acquiring the projector-world correspondences needed for projector calibration. Shader Lamps uses a manual calibration process that involves finding known points on the physical geometry with a projected crosshair [Ras+01]. While this approach works well for simple geometry, it quickly becomes time consuming for more complex models or when several projectors are used. Alternatives to this basic approach attempt to automate the process and remove human errors. Raskar et al. describe how a stereo camera pair can be used both for capturing a 3D model of the physical geometry in the scene, and how projector correspondences can be found using projected structured light patterns [RWC99]. iLamps [Ras+03] combines a projector, camera, orientation sensor, and computing system into a single unit. This allows the iLamp projector to automatically calibrate its image based on the geometry detected by the camera system. Another approach is to embed light sensors into the physical model to be projected onto [Lee+04b]. This removes the need for calibrated cameras, since the structured light patterns can be sensed directly from the surface of the object. The projector can quickly be recalibrated when the object is moved.

In addition to geometric calibration, which ensures projected information appears correctly aligned with objects in the world, SAR systems also benefit from radiometric calibration, which improves the quality of projected images. Raskar et al. [Ras+99; Ras+01] describe algorithms for blending multiple projector images for static scenes. However, these algorithms are unsuitable for when objects and projection surfaces are moved at runtime, as new blend masks must be calculated each frame. An alternative to multiple projectors is the Everywhere Displays Projector [Pin01]. This device uses a rotating mirror to direct images from a single projector to different locations in a room. Bimber et al. [Bim+05b] introduce techniques for view-dependent stereo rendering on complex physical surfaces. Through stereo rendering, SAR systems can alleviate the limitation of only being able to project flat graphics onto physical surfaces. The stereo images can give the illusion of virtual objects and information appearing above or inside the physical surfaces. The visual appearance of SAR projections is best when projecting onto white diffuse objects. However, SAR systems are often required to project onto in objects in the real world, rather than specifically prepared objects. Grossberg et al. [Gro+04] introduce techniques for improving projection onto coloured and patterned surfaces. Using a calibrated projector-camera pair, their system calculates a compensation image which negates the pattern on the surface of the object. The results of this approach vary greatly depending on the surface of the object. For example, it is difficult to reduce the appearance of a very saturated colour on the surface. While this approach involved an offline calculation step, other approaches aim to perform radiometric calibration in real-time, either with an offline step that captures the properties of the surface [Gru+07], or by introducing structured light patterns into the projected images that are imperceptible to the users [ZB07].

Another issue that effects the visual quality of SAR systems is the limited focal depth of projectors. This is not a problem when using projectors for video walls,

since the entire surface of the screen is the same distance from the projector. However, SAR systems often need to project onto surfaces from a variety of angles and distances relative to the projector. An approach to solving this problem involves stacking several projectors in the same location, with their focal planes set at different distances [MW01; BE06]. The defocus for each projector pixel on the surface of an object can be estimated, and the final projected image composed for each pixel from the projector with the sharpest focus. Laser projectors, such as those available from MicroVision[3] do not suffer from this problem, since there are no lenses in the optical path.

### 2.2.6 Portable Projectors

As advances in technology have driven the size of digital projectors down, mobile SAR has become possible. Portable projectors reduce the infrastructure required for SAR because projectors can easily be moved to where they are needed. Users will not occlude the projections because the projectors are either held or worn by the user [Sch+08]. Movable projectors create additional challenges over SAR systems with static projectors. Firstly, the position and orientation of the projector must be recalculated automatically and in real time. A homography between projector and projection surface can be automatically calculated using a camera and orientation sensor attached to a projector [RB01] at interactive frame-rates [Ras+03]. Inami et al. [Ina+00] shows how correct occlusion of virtual projected objects can be performed with a head mounted projector.

Secondly, hand-held and wearable projection systems require adapted interaction techniques. Beardsley et al. [Bea+05] show how a hand-held projector can be used as both a movable spatial display and as a pointing device, imitating a laser pointer. However, the pointing precision is limited by jitter caused by the user's

[3] http://www.microvision.com/showwxplus

hand. Forlines et al. [For+05] address this with a *zoom-and-pick* technique. Here, the user controls a zooming window which magnifies the area around the curser, enabling more precise control. A two handed alternative to projector attached cursor techniques involves holding the projector in the non-dominant hand to control the location of the projection, and a tracked stylus in the dominant hand to interact with the system [CB06]. *RFIG Tags* [Ras+04] combines wireless ID tags with photosensors which are embedded into the environment to create a "self describing world". A handheld projector is used to detect and identify tags and direct the users to their locations.

(a) (b)

**Figure 2.9:** (a) OmniTouch incorporates a shoulder mounted projector and depth sensor. (b) The depth sensor is used to detect projection surfaces and allow interaction. Images courtesy of Chris Harrison.

A major disadvantage of hand-held projection, as with AR on hand-held mobile devices, is the difficulty in performing complex bi-manual interaction techniques, as the user needs to carry the display. An alternative to carrying the projector is to wear it on the body. This frees both hands for interaction. A forerunner to this approach is WACL [Sak+05]. WACL uses a shoulder worn camera and laser pointer mounted on a pan/tilt head. The laser pointer is used by a remote collaborator to direct the attention of the user to aid in maintenance and similar tasks. Wearable projector systems have been developed for industrial tasks. For example, Schwerdtfeger et

al. use a projector mounted on the user's head [Sch+08]. The projection will always be placed on the surface in front of the user, which may or may not be where the user is looking. SixthSense [MM09] takes another approach; the user wears the projector/camera system around the neck. In this system, the projector points in the direction of the body, which again may not be where the user is looking. The index fingers and thumbs are tracked, and the system provides a gesture based user interface. A limitation of wearable projector systems is that there may not be suitable surfaces on which to project graphics. OmniTouch [HBW11], shown in Figure 2.9, uses a shoulder mounted projector with a depth camera. The depth camera allows the system to detect flat surfaces for projecting onto and interacting with.

### 2.2.7 The Virtual Showcase

Where the systems in the previous sections have projected directly onto objects in the real world, the Virtual Showcase [Bim+01] illustrates another approach to spatial augmented reality. The Virtual Showcase is a glass case into which objects are placed, and is depicted in Figure 2.10. The case is composed of half-silvered mirrors which act as a projection surface, while still allowing the object inside the case to be visible. Viewer-dependent graphics projected onto the mirrored surfaces allow virtual objects and information to appear anywhere inside the case. In addition to adding virtual objects, the physical objects inside the case can be illuminated using projectors. By altering the projected light illuminating the physical object, shadows from the virtual objects can be cast onto the physical ones [BF02].

While still a spatial augmented reality display, there are several important differences between the Virtual Showcase and the direct-projection techniques described earlier. Perhaps most importantly, the object of interest sits inside the case, and cannot be touched by the user. This affects both the possible uses for the system and the types of interaction that is possible. Since the object cannot be touched,

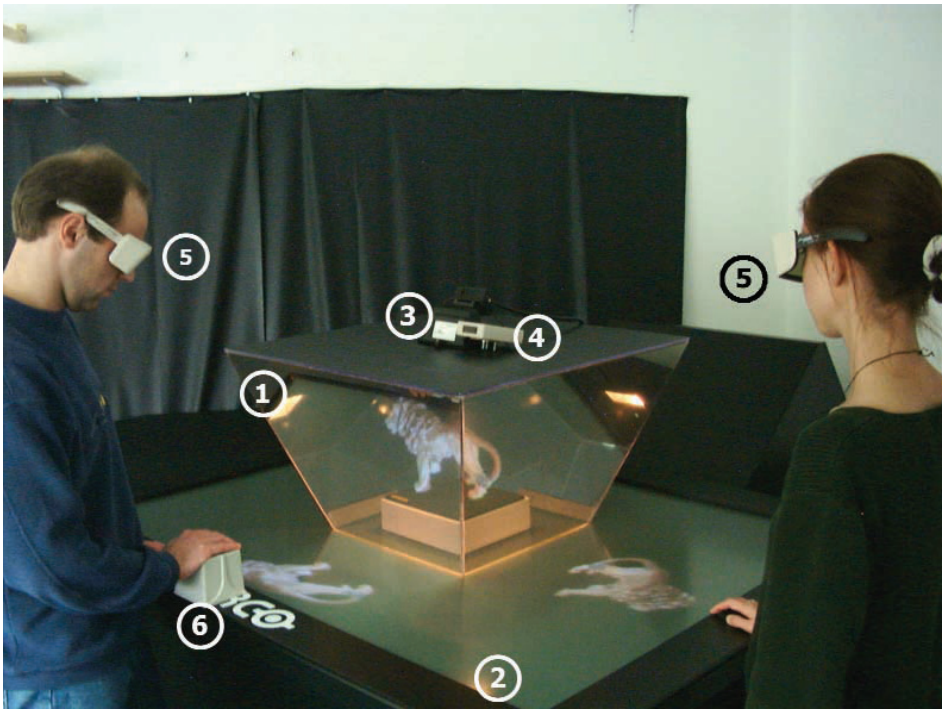



**Figure 2.10:** The Virtual Showcase. Image courtesy of Oliver Bimber.

interaction is inherently action-at-a-distance. Some VR techniques, such as virtual pointing [GM96], can be used. The Virtual Showcase must project view-dependent images for them to appear in the correct location on or around the object. Direct projected SAR systems only need to perform view-dependent rendering for certain visual effects. The number of users that can use the Virtual Showcase is limited because of the view-dependent rendering requirement. However, the Virtual Showcase is able to give the appearance of object's floating in mid air, which cannot be accomplished with direct projection systems. The Virtual Showcase is therefore suited to applications where the user should not touch the objects. The Virtual Showcase has been used as a museum exhibit for overlaying computer generated information onto fossils [Bim+02]. Another use of the Virtual Showcase is for digital story telling, with a physical object being the focus of the story [BES03].

## 2.3 Related Displays

This section discusses other display technologies that show similarities to spatial augmented reality, but should be differentiated from SAR. The list of requirements for AR as defined by Azuma [Azu97], listed previously, will be used as a guide to help categorise these displays.

### 2.3.1 Projection Mapping

Projection mapping is a technique for changing the appearance of objects often used by artists to create fantastic animations and visualisations onto buildings, sculptures, and other physical objects. Projection mapping, as it's name suggests, creates a 2D map of a particular projector's viewpoint. Using projection mapping typically involves the following steps:

1. Select suitable locations for each projector that will be used.

2. For each projector, create a map that shows what can be seen from that projector location. This is typically a manual process involving 'painting' over parts of the projection surface. Figure 2.11(a) shows a map being created for projecting onto a building.

3. Using the maps as templates, create the content for each projector using standard animation software.

4. When ready for viewing, set up projectors in their original locations, and playback the content, as shown in Figure 2.11(b).

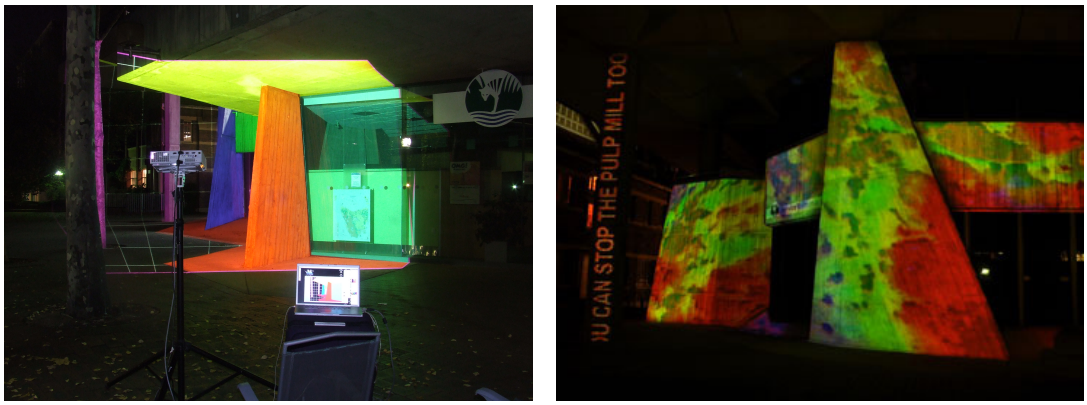
While 2D projection mapping is used by artists to create stunning visual effects, the approach suffers from two main drawbacks. Firstly, the projected content is created for a specific projector viewpoint. This makes it difficult to add additional

projectors later in development, as new content needs to be created specifically for the new projector location. Setup requires a tedious process of placing projectors in the correct locations, aligning them precisely, and performing keystone correction.

Early experiments with mapped projection involved recording a video sequence of a room with the camera rotating on a tripod. The video was then played back using a projector rotating at the same rate as the camera [Nai84]. This resulted in the appearance of the projected video matching the physical environment.

It is important to differentiate projection mapping from SAR for several reasons. Firstly, projection mapping is most often implemented as video playback. The videos have been "mapped" to the physical space by the author, but the system itself has no geometric understanding of the environment. Additionally, since the content is video playback, projection-mapped installations are typically not interactive, nor is the content generated in real time. Therefore, projection mapping should not be considered a form of AR.

SAR offers an alternative to 2D projection mapping. A 3D model of the projection surface can be created in CAD, and the projection content created for the 3D model. Later, SAR can be used to calculate the projector locations. This approach provides several advantages over projection mapping. Projected content needs to only be



(a) (b)

**Figure 2.11:** (a) Creating a map of a building from a projector location. (b) The Final artwork projected onto the building. Images courtesy of Cindi Drennan, Illuminart.

created once for the scene, regardless of how many projectors are used. This also makes the process of adding projectors trivial, since the system calculates what each projector needs to render based on the projector calibration.

### 2.3.2 The CAVE

The Cave Automatic Virtual Environment (CAVE) [CSD93] is an immersive virtual reality environment where perspectively correct graphics are projected onto the walls, floor, and in some cases the ceiling. The user stands in the room and looks out into the virtual world. The CAVE uses similar display technology as SAR. However, while SAR projects directly onto physical objects of interest, the projection screens comprising a CAVE are simply surfaces on which to view a virtual environment; windows into a virtual world. Caves project *view dependent* graphics, so the world appears correct from the point of view of the user. This limits the number of users able to use a CAVE simultaneously, although some systems time-multiplex multiple points of view using active shutter glasses. By contrast, SAR systems project *scene dependant* graphics. SAR functions independently from users, making collaboration with large groups possible. View dependent projections are only required for effects such as specular reflections and transparency. The virtual window metaphor of CAVEs also affects the type of interaction possible. Interaction in a CAVE is inherently action at a distance, since the objects of interest exist only in the virtual world beyond the projection screens. Physical handles and props can be used to aid interaction, but there is still a disconnect between user and virtual objects.

### 2.3.3 Displays in the Environment

The previous sections have looked at displays that change the appearance, or the user's perception of, the physical world. This section looks at displays that are embedded into the environment, but do not augment other objects.

Much research has been conducted into exploiting the ubiquitous presentation screen for interactive uses in scenarios such as meetings. Due to the size of these screens, interaction research has focussed on action-at-a-distance, such as through the use of laser pointers [ON01] for interacting with projected information. Shadow Reaching [STB07] takes advantage of shadows cast by the user for interacting with the display, rather than using pointing devices. Myers et al. [Mye+02] evaluated selection performance with several different styles of laser pointers, as well as tapping the projection screen, and using a mouse. The results showed that tapping the projection screen was the fastest and most accurate technique, followed by using a mouse, with the laser pointer the slowest. However, this experiment used a smartboard for the projection surface. The results may change as the screen gets larger and the user has to move in order to reach the entire area. Larger displays that require the user to move about the environment has been shown to increase user performance [BNB07]. However, the user will still need to be able to interact with the display from a distance. For these tasks, techniques such as Ninja Cursors [KI08] can be used. Ninja Cursors provide multiple cursors on screen, controlled simultaneously by the user. The user simply uses the closest cursor to their target.

The Fog Screen [Rak+05] is an interesting spatial display that has the ability to display computer generated information in "mid-air". This is accomplished by creating a thin curtain of water vapour, which can then be projected onto. The display can be made interactive by tracking the user's hands. The Sphere [BWB08] (Figure 2.12) is a touch-screen display device shaped as a sphere. It supports multi-touch interaction across the surface of the device. Compared to a standard touch-screen panel, it allows multiple users to interact with the device from any angle. This provides unique interaction properties. For example, two users can each work on a side of the display in a collaboration task, without disturbing one another.

**Figure 2.12:** The Sphere provides a touch-sensitive display on the surface of a sphere. Image courtesy of Hrvoje Benko.

## 2.4 Tangible User Interfaces

This section provides an overview of research into Tangible User Interfaces (TUI) and physical input devices. The key aspect of the systems described in this section is the fact that interacting with the system involves manipulating physical objects. This physicality makes TUI research particularly applicable to SAR, due to the fact SAR systems also feature physical objects to be manipulated by the user.

Graspable and tangible user interfaces were introduced by Fitzmaurice, Ishii, and Buxton [FIB95] as a user interface paradigm where the interaction with a computer system is performed through physical handles. These handles can be attached to virtual objects, and manipulating the physical handle affects the attached virtual object. For example, physical handles could be attached to the corners of a virtual object. Moving the handles would resize the virtual object. This concept is taken further in the implementation of the metaDESK [UI97]. The metaDESK brings the common Desktop Graphical User Interface (GUI) metaphor back to the physical

desktop, with a projector used for graphical output. Its user interface consists of physical analogues to common GUI widgets. These include:

- *Lenses* to replace GUI windows,
- *Phicons*, a tangible replacement for icons,
- *Trays*, which store several items, as a replacement for menus,
- *Phandle*, which act as handles for tasks such as resizing, and
- *Instruments*, which are special purpose devices to replace specific GUI controls.

The goal of tangible user interfaces is to create a more natural way of interacting with the system. Hinckley et al. [Hin+94] demonstrate this in their research into a prop-based user interface for neurosurgical visualisation. Their system allows neurosurgeons to manipulate medical images through the use of tracked props held in their hands. A doll's head is used to orient the 3D scan of the patient's brain, and a "cutting plane" tool is used to select a plane on which to slice the model and form an image of the scan. The Luminous Room [UUI99] provides another demonstration of how TUI systems can present more natural interaction techniques. The Luminous Room supports several TUI applications. Illuminating Light is a rapid prototyping tool for optical engineers. Physical handles representing optical components such as lenses, mirrors, lasers, etc. are placed on a desk and tracked with a camera system. Virtual light rays are projected onto the table, along with additional information such as path lengths and angles of intersection. Users modify the optical path by moving the physical handles. The authors note the benefit to users of this direct manipulation approach and its similarity to working with actual optical components. Another application, *Seep*, presents a TUI enabled fluid simulation projected onto a table. Users can interrupt the fluid flow simply by placing objects onto the table. A camera detects their presence and modifies the simulation accordingly.

*Urp* [UI99] is a TUI based application for urban planning. Users can preview urban designs by positioning model buildings in the work area. Urp extends the TUI paradigm by exploring special purpose physical tools. A ruler is adapted to take measurements between objects. The user selects the objects to be measured by pointing at them consecutively. An interesting aspect of this technique is the difference between the visual metaphor (a ruler), and the physical interaction (pointing). Shadows cast by buildings can be previewed at different times of day. The time is set using a physical analogue clock. Here, the visual metaphor matches the interaction technique. Finally, Urp allows reflections to be simulated. A transparent wand is used to point at buildings, which causes the projections to change as if the building was glass, and reflections are cast onto surrounding objects. Again, the physical appearance of the tool gives hints to the user of its function. *Illuminating Clay* [PRI02; Ish+04] is a TUI application for landscape analysis. Where previous TUI applications have allowed the user to work with physical tools on a desktop, Illuminating Clay allows designers to sculpt the landscape with their hands. A laser scanner captures the surface geometry, which is used to drive simulations such as water flow and erosion.

HandSCAPE [Lee+00] facilitates measuring tasks in a similar way to how measuring is accomplished in the real world. A digital tape measure is fitted with an orientation sensor, allowing the system to know which direction the tape measure is pointing. Measurements can then be made relative to a known location in 3D. uses for HandSCAPE include collecting measurements during excavations, or to aid in modelling interior environments.

The Tango is a unique hand-held, spherical input device which has pressure sensors embedded in its surface and accelerometers inside [Pai+05]. The Tango was designed for interaction with 3D environments presented on a desktop PC. The pressure sensors are used to recognise the user's grip [KP08], which can be used

to manipulate virtual objects or perform gesture based input. A 3D modelling system with tactile feedback has been created by attaching the Tengo to a Phantom[4] haptic device [PKC08].

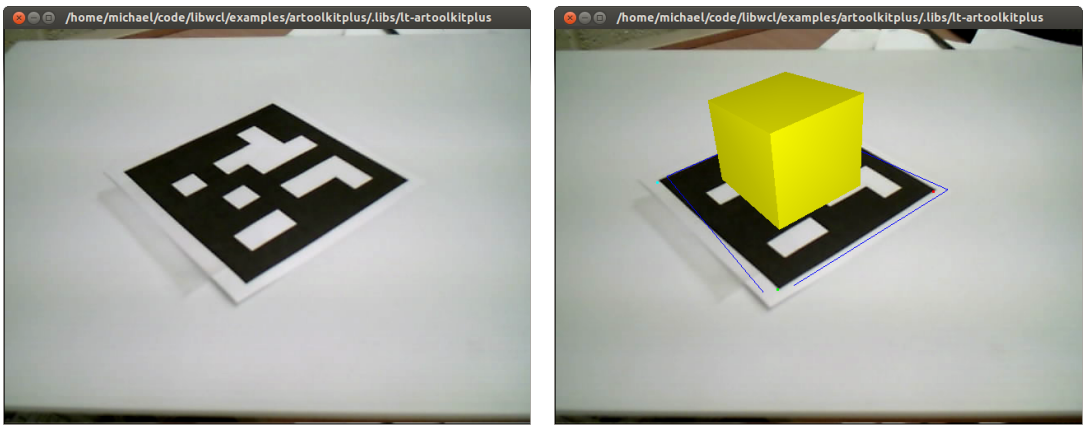
## 2.5 Tracking

This section gives a brief overview of approaches for tracking the location of people and other objects in the environment. Most interactive SAR systems will need to track the position and orientation of objects in order for the projections to appear correctly on their surfaces. In addition, tracking users makes it possible to produce view-dependent projection effects. This section focuses on tracking approaches that are most related to the work presented in Section 7.2, and is not an exhaustive overview of tracking technologies. Several surveys have explored tracking technologies in detail, including work by Meyer, Applewhite, and Biocca [MAB92], Bhatnagar, who focuses on systems suited for use with HMDs [Bha93], and Welch and Foxlin [WF02].

Several approaches to tracking objects involve the use of cameras, such as webcams. Rasmussen et al. present blob tracking algorithms which use colour alone to find the orientation, scale and position of markers in a variety of lighting conditions [RTH96]. Brusey et al. [BP00] further explored using colour alone for tracking and found that there are certain objects and colour combinations that are indistinguishable when combined. ARToolkit [KB99] allows tracking of printed fiducial markers, such as the marker shown in Figure 2.13 with six degrees of freedom. ARToolkit has become ubiquitous in AR systems, with several projects improving upon the techniques used, such as ARToolkitPlus [WS07]. The downside to approaches such as ARToolkit is the need for the entire square marker to be visible. If part of the marker is occluded by the user or another object, the tracking fails. Random Dot

[4] http://www.sensable.com/haptic-phantom-omni.htm

Markers [US11] uses fiducial markers comprised of randomly placed dots. The advantage of this approach is several of the dots can be occluded and the object can still be tracked. An improvement of this approach [UM11] allows the surface of the fiducial marker to be bended and folded. Not only will the object continue to be tracked, but the shape of the deformed surface can be reconstructed. Mohan et al. use a camera to detect small optical tags from a distance using Bokeh-code [Moh+09] to find active markers with unique identification.




(a) (b)

**Figure 2.13:** (a) ARToolkit and ARToolkitPlus detect square fiducial markers in camera images (ARToolkitPlus marker shown). (b) Computer graphics rendered on top of a detected marker.

One of the difficulties in using cameras for tracking in a SAR environment is that the projections are constantly changing the appearance of the environment. The changes in brightness cause difficulties in selecting appropriate camera exposure settings, making it difficult to track fiducial markers. Chapter 7 of this dissertation describes an improvement of blob tracking techniques using an adaptive active marker.

An alternative to visual-spectrum markers is to use markers that rely on Infra-Red (IR) light. IR markers have been created using invisible tapes [NKY08; NKY05], paints and inks [PP04], and light emitting diodes[Ras+07]. Commercial systems

such as those from Vicon[5], IOTracker[6] and OptiTrack[7] all leverage IR light with retro-reflective markers, as shown in Figure 2.14. These systems incorporate several calibrated cameras to triangulate the 3D position and orientation of rigid marker sets. IR light is more suitable for SAR systems as the tracking performance is not affected by the projected graphics. The downside is the need for specialised equipment, such as IR cameras.

**Figure 2.14:** Tracking systems such as those from Optitrack use retro-reflective markers to track the position and orientation of objects.

## 2.6 Industrial Design

Industrial design is a multidisciplinary process which involves all aspects of bringing a physical product from initial concepts, through to market. Several methodologies formalise the industrial design process, including Pugh's *Total Design Model* [Pug91]

[5]http://www.vicon.com
[6]http://iotracker.com
[7]http://www.naturalpoint.com/optitrack

and Pahl and Beitz's model of design [PBW84]. Both of these models are iterative processes, which typically include the following broad, iterative steps:

1. *Market*. This step includes market analysis to determine whether there is a need for the product.

2. *Specification*. This step involves developing the requirements for the product.

3. *Concept Design*. This step involves generating ideas to meet the requirements, creating concepts that embody these ideas, evaluating, and selecting the most promising of these.

4. *Detail Design*. In this step, the concepts that were selected from the previous step are developed further to finally create a finished product design.

5. *Manufacture*. This step involves actually manufacturing the product for sale.

6. *Sell*. This step involves marketing and selling the final product.

This dissertation is most concerned with improving the concept design phase of the industrial design process. Using CAD and design applications to express conceptual ideas is common place, as is creating physical mock-ups to assess the viability of a concept. It is most important for the designer to quickly evaluate many concepts, as this will lead to a better final product.

The appearance of material finishes is an important aspect of the physical mock-up. Designers use paint and inks to colour and annotate their mock-ups. A disadvantage with this approach is when changing a design, either a separate mock-up needs to be constructed or the initial mock-up must be re-painted. Although clay and polymer plasticine are able to change shape continuously, this is not possible once painted. The painted surfaces become less malleable.

SAR can greatly improve this process, by projecting material finishes directly onto design prototypes. SAR can also make use any preliminary concept sketches,

textures and design ideas that are created in the early concept stage. SAR provides the ability not only to generate a new appearance, but also present the initial sketches by projecting them directly onto the physical models. This functionality is provided by the SAR system and intended to maximize the visual information and presentation to the designer.

### 2.6.1 Prototyping Interactive Components

In addition to prototyping the physical form of products, creating and testing interactive controls is an important part of the design process. Hare et al. [Har+09] investigated physical prototyping with varying degrees of flexibility early in the design process. They found that for initial exploration of an idea, simple paper based prototypes are suitable. However, higher fidelity prototypes are necessary if specific feedback is required, especially for interactive components.

Several toolkits have been created to aid in developing interactive controls, with varying degrees of fidelity and flexibility. Phidgets [GF01] combines a set of physical devices, including switches, rotary dials, servos, etc. which can be connected as needed to a controller board. A selection of these components is shown in Figure 2.15. A software interface abstracts the hardware, so that programs using Phidgets can be written in a similar way to standard GUI programs. iStuff [Bal+03] is another toolkit for building physical user interfaces. The unique feature of iStuff is the concept of a *Patch Panel*. Applications using iStuff have a set of inputs and outputs, and a particular hardware configuration will have a set of input and output devices. The Patch Panel allows applications to be configured to specific hardware configurations without having to modify the application logic. This is analogous to the patch panels in early telephone exchanges, where signals were routed in different ways as required. The Calder Toolkit [Lee+04a] also provides a set of components for integrating with early design prototypes, in order to test usability early in the design

process. Components include analogue dials, button sets, joysticks, tilt sensors, and LED arrays. These components communicate with the host computer using wireless or wired connections, making them suitable for handheld and portable designs.

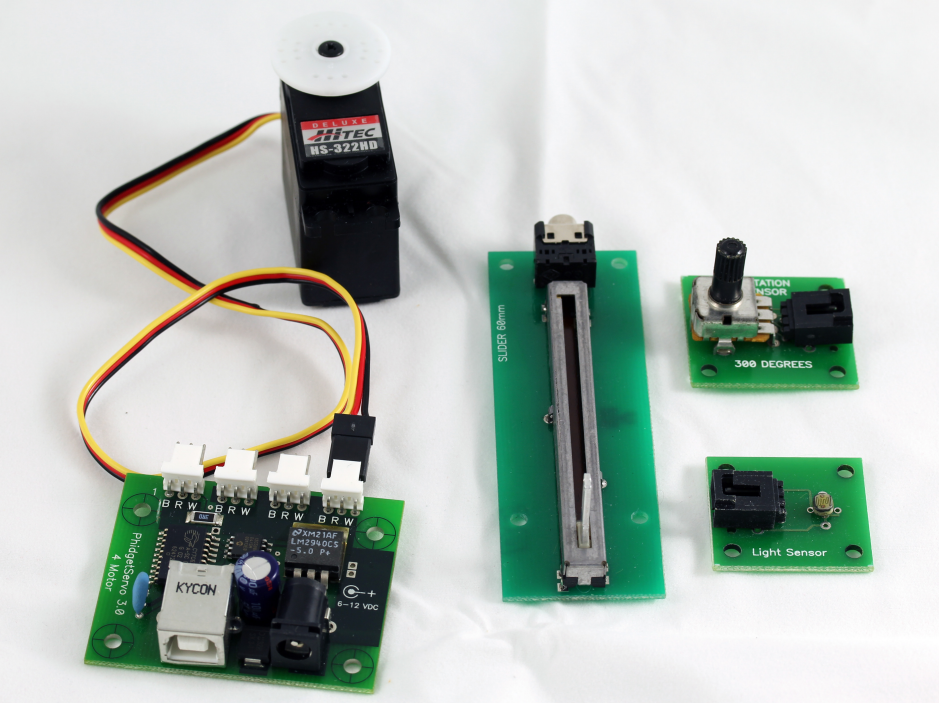
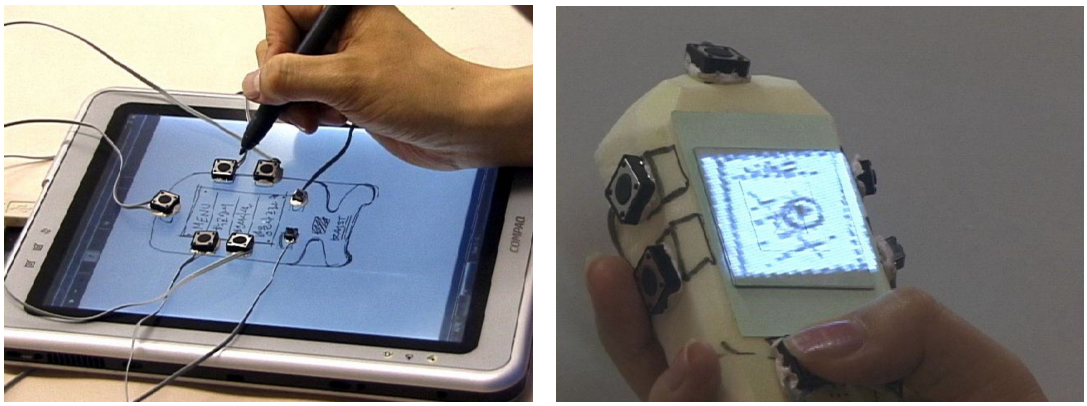



**Figure 2.15:** A selection of Phidget components. From left to right, a servo and controller, analogue slider, rotation sensor, and light sensor.

While both Phidgets and the Calder Toolkit simplify the process of constructing physical interfaces, there is still some wiring required, and the components still need to be assembled into the physical prototype or enclosure. This reduces the ability for designers to quickly modify the layout of controls. Pushpin computing [LBP05] addresses this issue. The system consists of a foam substrate that provides power and ground connections via two insulated back-planes, and a set of modular interactive nodes. The nodes are attached to the substrate and receive power using two pin that connect to the power and ground planes. The nodes communicate wirelessly using infra-red light. The main benefit of this system is that the layout of the nodes can be changed rapidly, simply by moving them around as needed. However, the considerable downside to Pushpin Computing is the need for the purpose built

substrate. This makes it difficult to place the nodes onto existing design prototypes. The peer-to-peer IR communication mode also inhibits the construction of complex curved surfaces.

Nam [Nam05] combines a sketch based concept exploration program, a set of physical user interface widgets, and a projected tabletop SAR system. The goal of the system is to improve physical user interface design by taking advantage of sketching, a skill already widely used by designers. Designers can sketch their design, and then attach physical buttons to the sketch to test interactivity, as shown in Figure 2.16(a). In addition to testing directly on the sketch, the buttons can be fitted to physical prototypes and evaluated using the SAR system, such as the prototype shown in Figure 2.16(b). Again, the main drawback of this system is the difficulty in reconfiguring the controls.



(a) (b)

**Figure 2.16:** (a) Attaching physical widgets to a user interface sketch, (b) a physical prototype with attached buttons and projected display. Images courtesy of Tek-Jin Nam.

DisplayObjects [AV09] does away with physical controls entirely. Instead, the interface is projected onto design prototypes. The user's finger is tracked, allowing them to adjust the layout of the controls with their hands. The main drawback to DisplayObjects is the use of 2D projection, limiting the complexity of interfaces that can be constructed. In addition, the controls themselves are not interactive, reducing

the ability to test the usability of control layouts.

# 3

# Tool Virtualisation and Physical-Virtual Tools

This chapter introduces Tool Virtualisation with Physical-Virtual Tools (PVT), a new user interface paradigm for spatial augmented reality systems. PVT user interfaces address the characteristics of SAR, described in the previous chapters, by constructing user interfaces from physical tools. These tools are augmented with extra information by the projection system. This is invaluable in a large scale SAR system, where there is no fixed location suitable for a traditional user interface. As part of the paradigm, two continua have been developed for assessing the complexity of PVT user interfaces.

This chapter begins by describing the Physical-Virtual Tools concept, and discussing how it can be applied when developing interactive SAR systems. Following this, the *Within System Tool Virtualisation Continuum* (WS-TVC) and *Inter-System Tool Virtualisation Continuum* (IS-TVC) are introduced, with a discussion of how they can be used when building applications. Existing SAR user interfaces are evaluated and placed on the continua for comparison. Chapters 4, 5, and 6 present SAR applications featuring PVT user interfaces.

## 3.1 Physical-Virtual Tools

As previously mentioned, SAR requires physical objects to project onto. Unlike HMD and hand-held devices, it is not possible to draw an object in 'mid air,' or display menus and other information at a predictable location on the image plane of the display. Users of SAR systems benefit from an unencumbered view of the physical world, and are free from having to wear or hold the display device. SAR systems require new user interface techniques that make the most of the physical nature of the projection system. With these concerns in mind, SAR user interfaces should be based around physical tools manipulated by the user. The user interacts with objects in the system through these tools.

Physical-virtual tools build on both tangible user interface research and the use of props in virtual reality systems. Like props, PVT take advantage of the manual dexterity of humans, and of proprioception cues to enhance virtual interaction [MFS97]. Physical interaction techniques are a compelling choice for SAR user interfaces. Due to the nature of projection technology, SAR is best applied in domains where physical objects are important. While the keyboard and mouse is ubiquitous for desktop PCs, these input devices are not suited to large, immersive SAR systems, where users are free to move around and experience the augmentations from different angles and positions. Except for very large, building scale SAR systems, users are able to touch and interact directly with physical objects and the augmented information projected onto them. Virtual reality research has demonstrated many benefits of being able to touch objects in a virtual environment. Hoffman et al. [Hof+98] has shown this enhances the realism of virtual experiences. Experiments by Ware and Rose [WR99] have shown that using physical handles for virtual objects improves the ability of the user to perform manipulations such as rotations to virtual objects. Other experiments have shown that interaction techniques that make use of users' proprioception lead to better performance [MFS97].

Several studies have also demonstrated the benefits of two handed interaction techniques [BM86; KBS94; LZB98]. While these experiments focussed on VR applications, the results support the decision to investigate physical interaction techniques for SAR. SAR does not inhibit users' sense of proprioception, since they are still able to see the physical world. SAR also frees the user's hands from having to wear equipment or hold the display, making them available for more complex bi-manual interaction techniques.

PVT user interfaces differ from props and previous TUI research in several important ways. Most importantly, PVTs are specifically designed tools for performing certain interaction tasks with a system. Props, on the other hand, are typically physical stand-ins for virtual objects [SPS01]. They may be metaphors for the virtual objects, such as using a doll's head to control a medical imaging application [Hin+94]. In the same vein, the physical icons [UI97] of TUI systems serve as physical analogues to standard GUI controls [UI97], or are physical handles to other virtual constructs [FIB95]. The physical design of props and TUI handles are not always specifically designed for particular interactions in mind. Two exceptions are the time and distance tools of Urp [UI99], which uses a clock and ruler to control these aspects of the system.

In both TUI and prop based systems, additional state information and GUI controls are shown at a fixed location on a display, or projected on to a table-top. This approach works well for VR systems and table-top TUI applications. However, this approach is unsuitable for large immersive SAR systems, as the user may need to interact with the system from many different locations, or multiple users could interact simultaneously. For these reasons, information that would usually be placed in a traditional GUI is instead projected onto the tools in a PVT system. This places information close to where the user is currently working, avoids the need for additional screens, and supports multiple users without any extra effort.

### 3.1.1 Virtualising Tools

Projecting state information onto tools allows the possibility for tool virtualisation. Tool virtualisation is the overloading of one tool with a number of functions and attributes. For example a pen can be overloaded with a number of different colours and tips. In the case of a physical pen the ink cartridge and the tip of the pen may be replaced. A virtualised tool has information, such as the current state of the tool, projected onto it. For example, the tip of a stylus can have the current colour projected onto it. Or, the active mode of a multi-purpose tool can be conveyed to the user by changing the tool's appearance. This is similar to the approach taken with the Virtual Tricorder [WG95], where the entire visual appearance, including shape, is modified depending on the task. The Virtual Tricorder aims to be the sole interaction device for the VR system. By contrast, a PVT only has its surface appearance changed, not the shape. Rather than using a single tool for all interaction, a set of tools is provided based on the tasks the user will need to accomplish.

### 3.1.2 Tool Design and Selection

When designing user interfaces, the designer needs to make compromises between having many single purpose tools and a single tool that is used for all interaction.

An example from the 'real world' are the tools used by a mechanic. A standard toolbox would include a set of wrenches, one for each size bolt. The toolbox would also include a shifting spanner; a single tool that can be used on any bolt. These tools are shown in Figure 3.1. Choosing to use the shifting spanner over a fixed wrench is a trade-off. On one hand, the shifting spanner is more flexible, and the mechanic would only need to carry one tool no matter what the task. The shifting spanner can be used when the bolt size is unknown. On the other hand, a shifting spanner has poorer performance overall. The large adjustable head prevents the tool being used in tight spaces which would not affect a fixed ring spanner. The mechanism


**Figure 3.1:** A mechanic can choose between several single sized wrenches, or a shifting spanner which works with any size.

could move during use, causing the spanner to slip off the bolt or damage the part. Additionally, if the mechanic is working with many different sized bolts, the time taken to adjust the spanner would hinder their efficiency. The mechanic owns both types of tool so the best one can be chosen for each situation.

The same trade-off exists in the virtual domain, such as a virtual drawing application. The developer could provide separate pens for each colour. The user can choose a colour by picking up a particular pen. SMART Technologies[1] provides this functionality in their smart-board products. Alternatively, a single pen could be provided, with the user selecting the colour through the user interface. This is typically how an artist interacts with drawing applications such as Adobe Photoshop[2] with a

[1] http://smarttech.com
[2] http://adobe.com

tablet input device. Again, this is a trade-off between many specialised tools and a single, multi-purpose tool.

SAR systems project information and graphics onto physical objects. Interactive SAR systems are suited to direct manipulation of this information using physical interaction techniques. Physical-virtual tools need to be designed for the specific types of interactions required for each application. Like the wrenches scenario, there is a trade-off between having few multi-purpose tools and many specialised tools. SAR systems require several tools with different physical designs, because different form factors are better suited to certain interactions. For example, for out of arms reach operations such as pointing, a laser pointer style grip is most suitable, but a pen is better for annotation directly on an object.

The approach proposed in this dissertation is to combine tasks that use similar interactions onto a single tool, with each tool designed to physically support a specific type of interaction. SAR can then be used to convey tool state information to the user by changing the appearance of the tool. Consolidating the tools in this manner reduces the cognitive load associated with having a large, complicated tool set, and the time taken to physically change tools. However, the benefits of having tools tailored to certain tasks is preserved. SAR improves on this, as state information is available where the user is currently working, directly on the surface of the tool.

## 3.2 The Tool Virtualisation Continua

As part of the PVT paradigm, two continua have been developed to aid application designers when developing PVT user interfaces. These provide a mechanism for meaningfully assessing and comparing PVT systems.

The *Within System Tool Virtualisation Continuum* compares the level of task overloading for tools in a single application. Application designers can use this contin-

uum to aid in the development of the tools required for a system's user interface, by comparing the different complexities of the overloaded tools. Suppose a selection of different physical tools are available for the application architect to choose from or design themselves. Based on the complexity of tasks and the application, the number and styles of physical tools with appropriate attribute overloading will be determined. In the case of designing a SAR-based PVT, the continuum provides a means of determining which tools would have the most complex virtualisation. When designing the physical shape of the tool, the ability to display virtual information on the tool itself needs to be taken into account.

The *Inter-System Tool Virtualisation Continuum* compares the entire sets of tools between systems. This helps the designer to understand to what extent a toolkit of a system virtualises its tools compared to other systems, and to gauge the relative complexities of their user interfaces.

Each Tool Virtualisation Continuum (TVC) has been developed specifically for SAR user interface designers. The two continua aid designers in:

1. developing user interface toolkits for SAR applications; in particular, assessing application requirements in terms of the TVC will help designers to decide on the number and nature of tools required,

2. quantifying and ranking the level of virtualisation of the set of tools in one application, and

3. comparing the complexity of user interfaces between applications.

The following sections describe the two continua for classifying and comparing individual tools used in SAR systems, for comparing the systems that make use of these tools, and how application designers can use the TVC to help when designing SAR applications. In particular, the TVC help to determine the number of tools required, and the nature (physical shape) of these tools.

### 3.2.1 Within System TVC

The Within System TVC (WS-TVC) helps designers of SAR applications to make decisions on the number of tools required for an application, and the physical nature of these tools. The continuum considers two factors: tasks and attributes. A *task* is the physical interaction required to achieve a goal. For example, the act of drawing a predefined shape on an object. Each task can have zero or more user changeable *attributes*. Selection is an example of a task requiring no attributes. In the shape drawing example, attributes may include fill colour, stroke colour and style, and shape geometry.

Designers can use the WS-TVC when deciding on the number and type of tools required for a system. The designer must first define the tasks and attributes for each task the system will support. Tasks that contain similar types of interaction are suitable for consideration for overloading on a single tool. For example, virtual spray painting and a laser pointer both involve pointing at an object, and therefore would require a tool with a similar form factor. These two tasks are ideal for consolidation onto a single tool. When deciding whether to have a separate task, or add an attribute to an existing task, the designer should look at the interaction required to complete the task. If the two potential tasks require identical interactions, then a single task should be used, and an attribute added. For example, 'drawing red' requires identical interaction from the user as 'drawing blue', so a single 'draw' task should be added with a colour attribute.

The WS-TVC is shown in Figure 3.2. It contains two axes. A tool's position on the x-axis is defined by the number of tasks a tool has been overloaded with, and its position on the y-axis is defined by the number of attributes for the task with the most attributes. The x-axis tells the designer the relative burden of the tool. There is more cognitive load required in swapping the active task, and recalling the active task as the number of tasks a tool supports increases. In addition, the

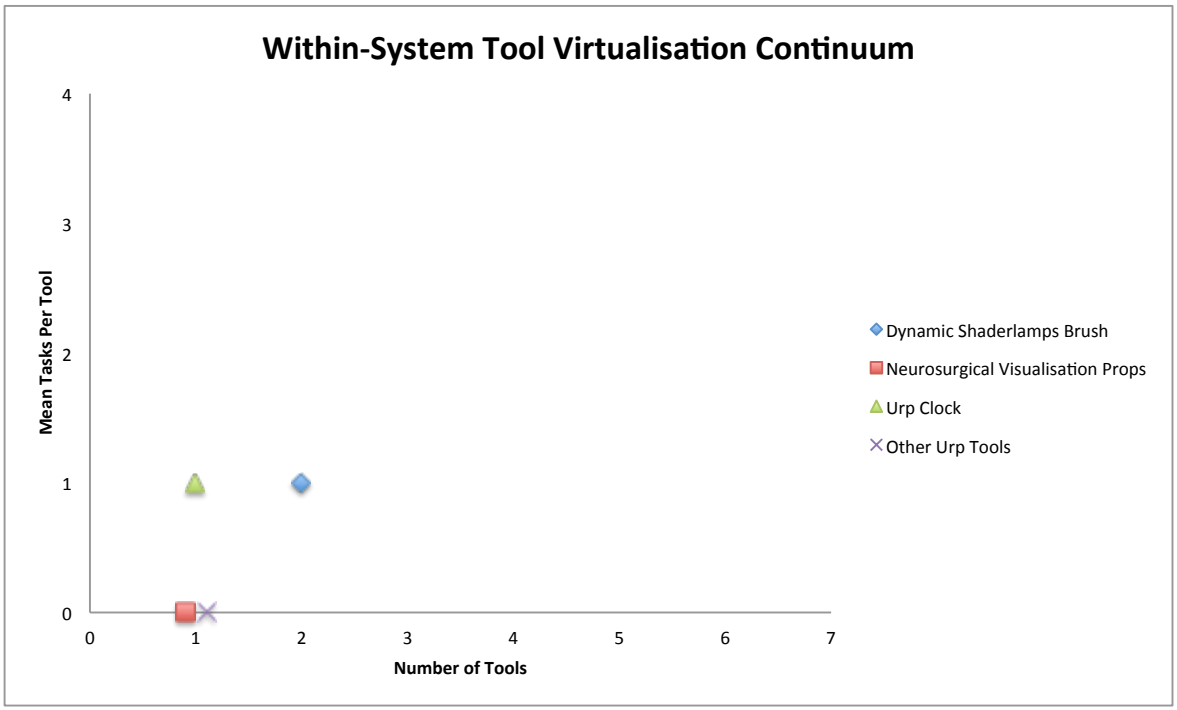



**Figure 3.2:** The Within System Tool Virtualisation Continuum, showing the positions of several tools.

physical design of the tool will be less ideal for any particular task as the number of supported tasks increases, as the design must accommodate different kinds of interaction. The y-axis places design constraints on the physical tool, since as the number of attributes increases, more information must be shown simultaneously, requiring a larger projection area.

This leads to the two competing factors on placing a system on the continuum, number of *physical tools* verses the number of *attributes* and *tasks* overloaded on each *physical tool*. In the case of the pen, there are a number of standard attributes that can be overloaded including colour, line thickness, and tip shape. The tip shape is good example of how the continuum allows the developer to decide between two factors. On the one hand, virtually overloading a single physical pen with different tool tips requires fewer physical tools to accommodate all tasks, while on the other hand drawing with different tip shapes on the pen *feels* different to the user. The developer must make a design decision on the best outcome for the system based

on the requirements given to them. Virtual pens can do different actions to physical pens, such as line patterns, gradient line colours, automatic straightening, and automatic arrowheads.

The above pen example describes different attributes for a single physical tool that had the same basic functionality, a hand-held drawing instrument. However, the tool could also be overloaded with completely different tasks. For example the pen device could also perform selection tasks. There are many different modes of performing selections, such as: select, copy, paste, duplicate, cut, or resize.

#### 3.2.1.1 Using the WS-TVC

Once a designer has decided on the base number of tools, they should be placed on the WS-TVC and evaluated. As a tool's position in the x-axis increases, the designer should consider adding another tool with a *different* physical design. The new tool should be designed for a subset of the original tool's supported tasks. As a tool's position on the y-axis increases, the designer should consider adding another tool with an *identical* physical design to remove the need for some of the attributes. For example, changing from a 'predefined shape' tool with geometry and fill colour attributes, to separate tools for each geometric shape, so each tool only requires the colour attribute.

### 3.2.2 Inter-System TVC

Where the WS-TVC compares the individual tools that make up a system, the Inter-System TVC (IS-TVC) is used to compare the relative complexity of user interfaces among different systems. The IS-TVC, shown in Figure 3.3, illustrates the transition in the x-axis from a single virtualised tool responsible for all user input to having dedicated tools for each task, and on the y-axis the transition of mean number of tasks per tool from a single task to every task mapped onto each tool.

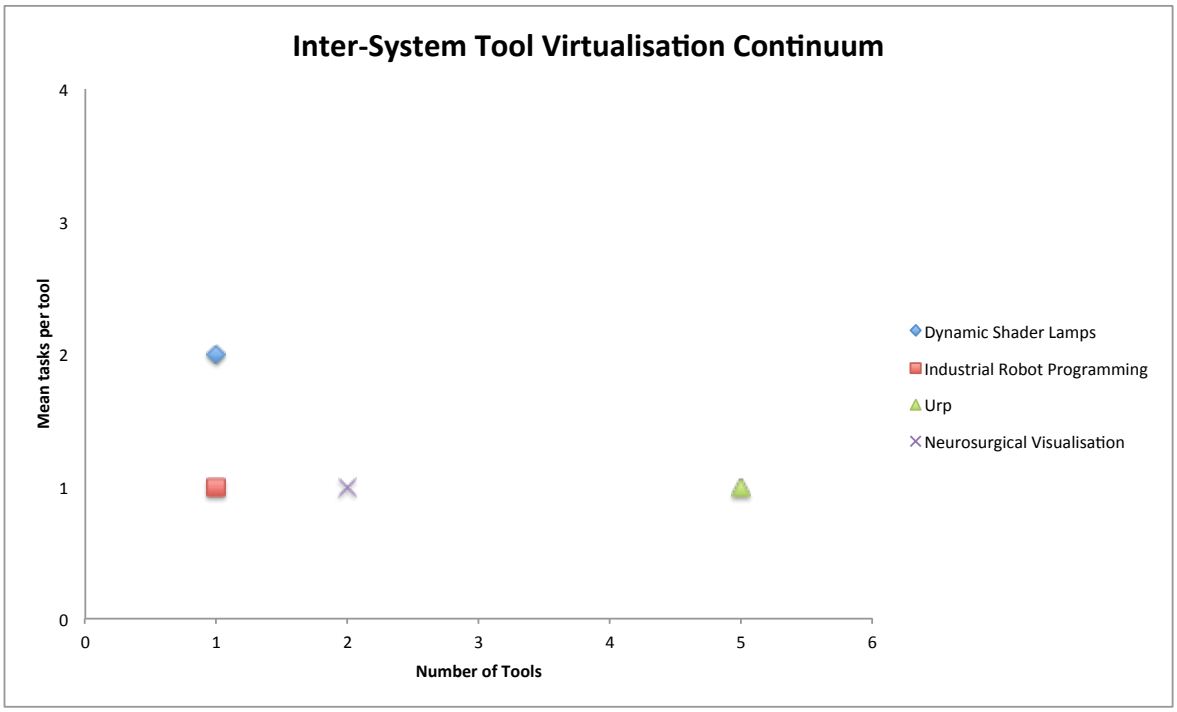



**Figure 3.3:** The Inter-System Tool Virtualisation Continuum. Several applications have been placed on the graph.

The increase in the y-axis is an increase of cognitive load of a total system. On average, each tool is required to support more tasks, and as such requires a more complicated mental mapping when changing tasks. While an increase in the x-axis increases the complexity of physical workspace to support more physical tools. This increase in complexity would impact the size and organisation of the workspace to accommodate the tool set and the time required to physically change tools.

#### 3.2.2.1 Using the IS-TVC

IS-TVC allows one to compare the complexity of the user interface between systems and visualise their relative cognitive load and complexity of physical workspace. The complexity of a user interface increases as the system moves away from the origin of the graph. Systems in the lower left of the IS-TVC would have less complex user interfaces. While moving into the upper right of the continuum would have the most complex user interfaces. Currently most systems would be towards the

left hand side of the continuum but spread vertically. TUI research is exploring the regions more to the right of the continuum.

## 3.3 Assessing Existing SAR User Interfaces

This section revisits some of the interactive SAR systems outlined in Chapter 2, and assesses their user interfaces in the context of the TVC. Many of the applications discussed previously have no tool based user interface. Often there is no user interface at all, with the system responding to the user's actions or other events in the environment [Sch+08; PI01; Sug+08]. Other systems allow interaction with virtual objects, such as projected buttons using the user's hand or finger [Por+10a; Por+10b; Zio+10; Sim+12]. SixthSense [MM09] also utilises the user's hands for gesture based control. It is difficult to assess these applications in the context of the TVC, since they either have no user interfaces, or do not use tool based input. The number of systems described in this section is therefore limited. However, the selection illustrates how the PVT concept can be applied to existing systems.

Several SAR systems utilise a tracked stylus for user input without augmenting the stylus with extra information [ZV06; Byu+07]. Since there is no information projected onto the stylus, and in these applications the stylus is used for a single task, the stylus would be placed at (1, 0) on the WS-TVC. Since each of these applications have a single tool for interaction, they would each be placed at (1, 1) on the IS-TVC. Dynamic Shader Lamps [BRF01] provides a tracked paintbrush for digitally paint onto physical objects. This tool has two tasks: painting and colour selection, with a single attribute: the active colour. Therefore the paintbrush can be placed at (2, 1) on the WS-TVC. Dynamic Shader Lamps as a whole would be placed at (1, 2) on the IS-TVC.

Tangible user interface based systems are more suited to comparisons against

the TVC, since their user interfaces are comprised of physical objects. However, the TUI systems shown in Section 2.4 do not project onto the tools themselves. Urp [UI99] is a TUI system for evaluating and designing urban environments. Urp's user interface is comprised of five tools: clock, distance, reflection, wind, and camera. This system is a good example of having many single purpose tools. As such, it is placed at (5, 1) on the IS-TVC, with each tool receiving a score of (1, 0) on the WS-TVC. Similarly, each of the tools in The Luminous Room [UUI99] would be placed at (1, 0) on the WS-TVC. The position on the IS-TVC depends on how many objects are in use at a given time. The prop-based neurosurgical visualisation presented Hinckley et al. [Hin+94] features two single purpose tools with no visualisations. Therefore, this system would be placed at (2, 1) on the IS-TVC, and each tool placed at (1, 0) on the WS-TVC.

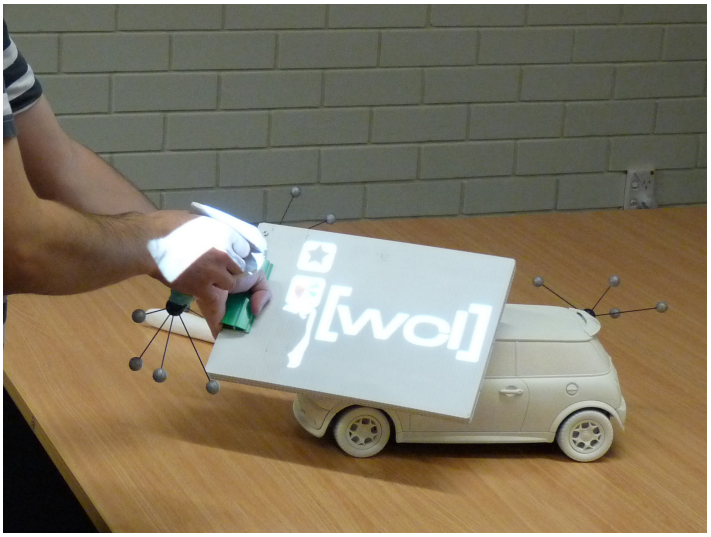
As this section demonstrates, existing SAR and TUI systems do not consider the tools that comprise their user interfaces as projection surfaces. This forces additional information to be placed elsewhere on other surfaces in the environment. The following chapters present richer PVT user interfaces, exploring the higher regions of the TVC.

# 4

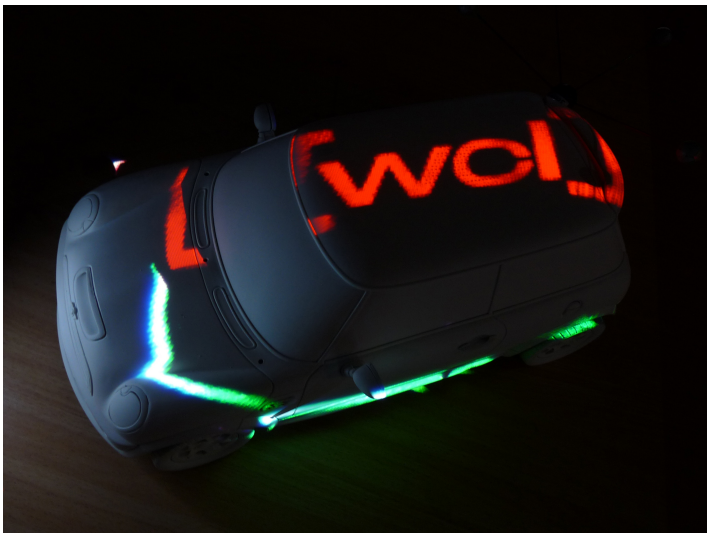
# Digital Airbrushing

The previous chapter introduced Physical-Virtual Tools as a technique for building user interfaces for SAR. This chapter introduces the first example PVT system, Digital Airbrushing.

Digital Airbrushing, shown in Figure 4.1, is a SAR system that allows a designer to freely create and preview different designs for the finish of physical prototypes. Designing the surface properties has been the focus of previous SAR applications; Dynamic Shaderlamps [BRF01] provides a paint brush for painting physical objects with projected light. SixthSense [MM09] implements an action-at-a-distance finger painting application. WARP [Ver+03] provides a desktop user interface for selecting different materials. The motivation for digital airbrushing is to develop a tool that fits in with existing design processes, using natural interaction techniques that take advantage of existing skills. To that end, Digital Airbrushing uses a two handed interaction technique that allows the designer to paint onto physical objects, using a stencil in a similar way to how an artist uses a real airbrush. The designer can create digital textures and annotations for design artefacts directly onto matching physical models. Objects and tools are tracked in 3D space, but not the users themselves. Status information is projected onto the tools, allowing the user to move around




(a)




(b)

**Figure 4.1:** (a) The user stencils paint onto the car using the airbrush tool. (b) The result of the paint operation.

freely while still being able to interact with the system.

## 4.1 Airbrushing in the Real World

Airbrushing allows artists to create impressive artworks on any number of objects, from canvas, to the side of a car. Digital Airbrushing aims to replicate the techniques artists use in the real world. Airbrush artists require precise control of paint flow

(a) (b)

**Figure 4.2:** Airbrushing in the real world. (a) Example French Curve stencils used by designers. (b) An example artwork created with airbrushing and stencils. Image courtesy of Ry Wilkin, www.are-why.com.

in order to mix colours and create a variety of effects. This is often accomplished through the use of stencils, which serve to mask areas of the work from receiving paint. In the physical world, designers rely on a variety of different shaped stencils, such as French Curves, depending on the task at hand (Figure 4.2(a)). Stencil artists take this approach to the extreme, creating highly specific, single use stencils, such as the cat stencil used in Figure 4.2(b). In addition to simple masking, the use of stencils gives an artist fine grained control of the paint, allowing them to create gradients and other effects. This is illustrated with the range of colours in the arrow at the top left of Figure 4.2(b).

## 4.2 Airbrushing with Physical-Virtual Tools

Digital Airbrushing aims to bring the techniques used by airbrush artists into the digital domain using spatial augmented reality. The user interacts with the system using a PVT based user interface, consisting of three tools: The *Stencil*, *Airbrush*, and

*Stylus*. The physical appearance of the Airbrush and Stylus tools were designed by industrial design students, Coby Costi, Ahmed Mehio, Benjamin Hill, and Timothy Grisbrook. This section discusses the tools developed for this system, the tasks they are used for, and how the system relates to the TVC introduced in Chapter 3.

### 4.2.1 Stencil Tool

The *Stencil Tool*, as shown in Figure 4.3, is a physical board the user holds in their non-dominant hand. The stencil tool serves three main functions:

- Stencilling for the airbrush. This task has two attributes: the stencil shape and whether the stencil is normal or inverted. A French curve projected onto the tool is shown in Figure 4.4(a).
- The physical base on which stencils are drawn, as illustrated in Figure 4.4(d). Here we can consider the stencil shape being drawn as an attribute.
- Command entry. The tablet is used to change attributes for other tasks through buttons and controls projected onto it. The number of attributes can therefore be considered as the sum of attributes for the other tasks. Figure 4.4(b) shows the colour wheel used for paint colour selection. Figure 4.4(c) shows the stencil selection menu.

The default function for the Stencil tool is the paint stencil. A stencil shape is projected onto the board, masking areas from the paint. It operates in two modes: either the stencil shape masks the paint, or the entire tool is the mask, with the stencil shape acting as a hole allowing paint to pass through. The appearance of the tool changes depending on the stencil mode.

One of the advantages of using a PVT with SAR[1] is that a single physical tool can be used for any number of stencil shapes. A set of geometric shapes is provided, as

[1]This would also be possible with video see-through HMD based AR using props.

**Figure 4.3:** The Stencil tool

well as straight edge and French curve shapes. Using the tool in this way allows the user to interact with the system in a similar way to how an artist would use a real airbrush. For example, painting while slowly dragging the stencil over the object will vary the amount of paint areas of the surface receive over time. The result of this is a gradient effect from no paint to dense coverage. This effect can be used to blend different paint colours, in the same way gradients are created with physical paint (see Figure 4.2(b)).

In addition to the shapes provided, the user is able to create custom stencil shapes while the system is running, as shown in Figure 4.4(d). New stencils can be created by drawing onto the tool with the airbrush. An outline of the shape is projected onto the tool as it is being drawn. When a closed loop is detected, the previous stencil is replaced with the custom one, with the appearance of the tool

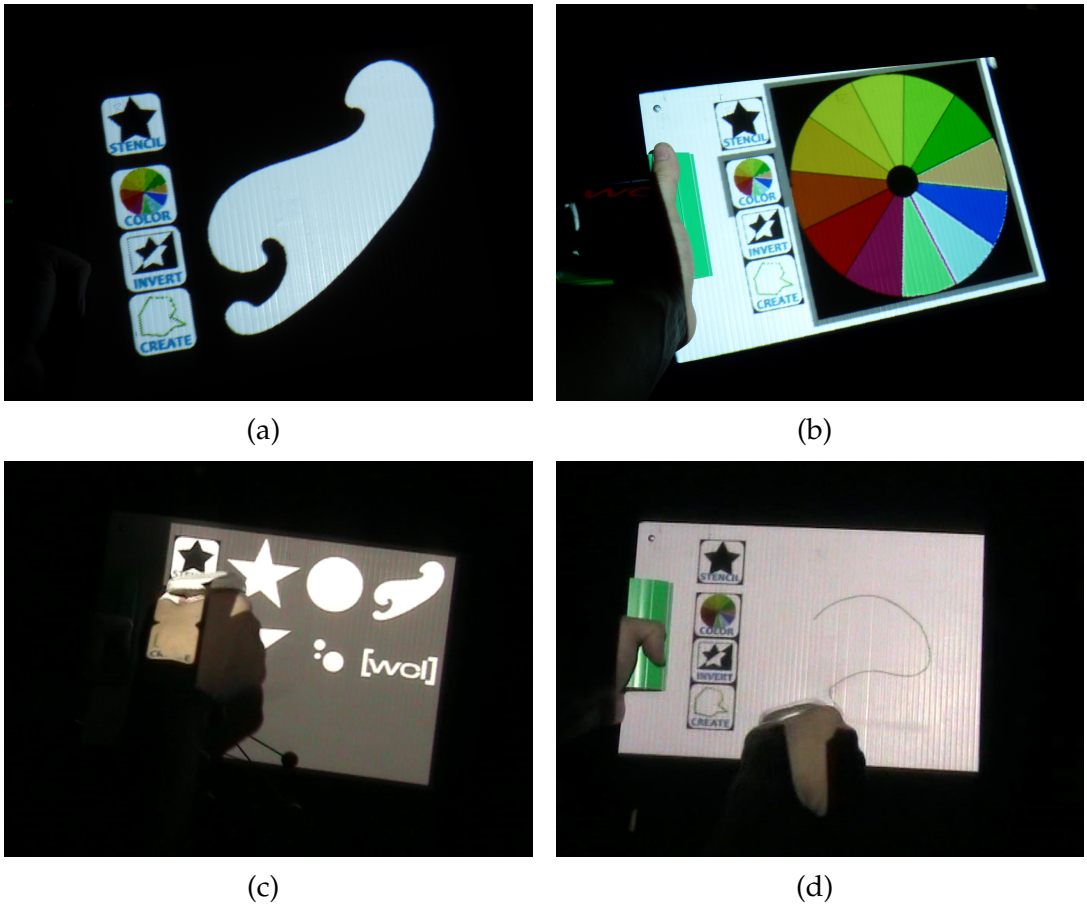



**Figure 4.4:** The stencil tool in use. (a) Stencil mode with stencil shape masking paint, (b) choosing paint colour, (c) stencil selection menu, (d) drawing a custom stencil.

reverting to stencil mode. This offers the user more flexibility, as they can create a stencil shape for a specific situation.

A physical stencil tool gives the user passive haptic feedback against the object being painted and when drawing stencil shapes, providing a more natural experience. An alternate approach is to use dedicated tools for each stencil shape. The shape of each tool would precisely match the stencil. However, a single physical tool with projected virtual shapes allows greater flexibility. This is a trade off between multiple physical tools that precisely match the task, and a single tool that can approximate all stencils. Virtualised stencils allow additional functionality to be provided. For example, it would difficult to allow the user to draw custom stencils at runtime without virtualisation of the stencil shape.

The stencil tool also acts as a Personal Interaction Panel (PIP) [SG97] user interface for controlling other aspects of the system. The stencil tool is virtualised by placing the user interface controls onto the same physical board the user is already holding, rather than providing another device for the interface. The user can quickly change the state of the system without having to turn away from the work area or put the tools down. The menu is used by pressing four buttons, which are always visible on the side of the tool, with either the airbrush or stylus. The menu allows the user to change the paint colour, select from a set of predefined stencil shapes, create their own stencil shape, or invert the stencil. This menu can be seen in Figure 4.4.

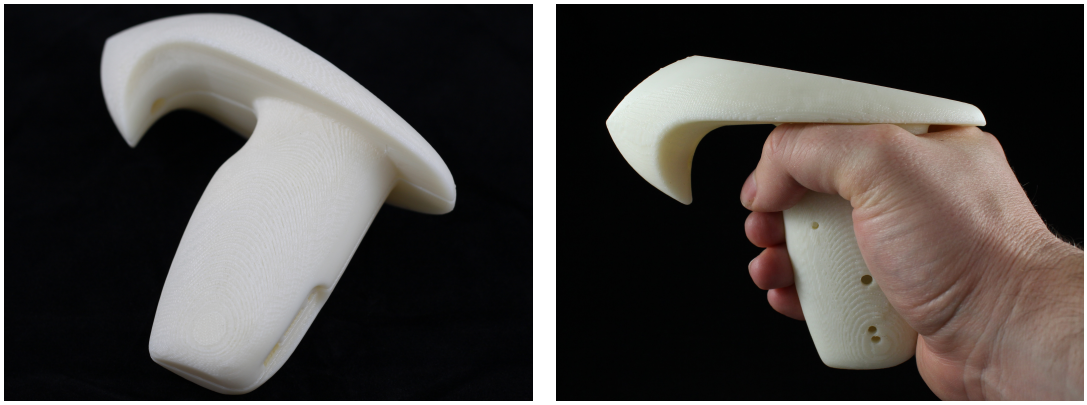
In general, this command entry task allocated to this tool makes it difficult to place in the continuum, as the position depends on a specific application. However, if the classification is restricted to the airbrushing application, the tablet would receive a rank of (3, 6). In this context, the tablet is closest to a single virtual tool, as it supports the greatest number of tasks. In theory a PIP could support any user interface controls that appear on a traditional desktop.

### 4.2.2 Airbrush Tool

The *Airbrush*, shown in Figure 4.5, is a ray-gun shaped device held in the user's dominant hand, suited for at-a-distance interaction techniques. The tool was intentionally designed with a large flat area for projection, allowing for more overloaded tasks and more complex attributes to be shown. Projecting information onto the tool allows the user to know the state of the tool without having to look away from the current task.

The Airbrush is used for the following tasks:

- airbrushing onto objects. This task has several user changeable attributes: paint colour, spray angle, brush hardness (shown in Figure 4.6), and paint



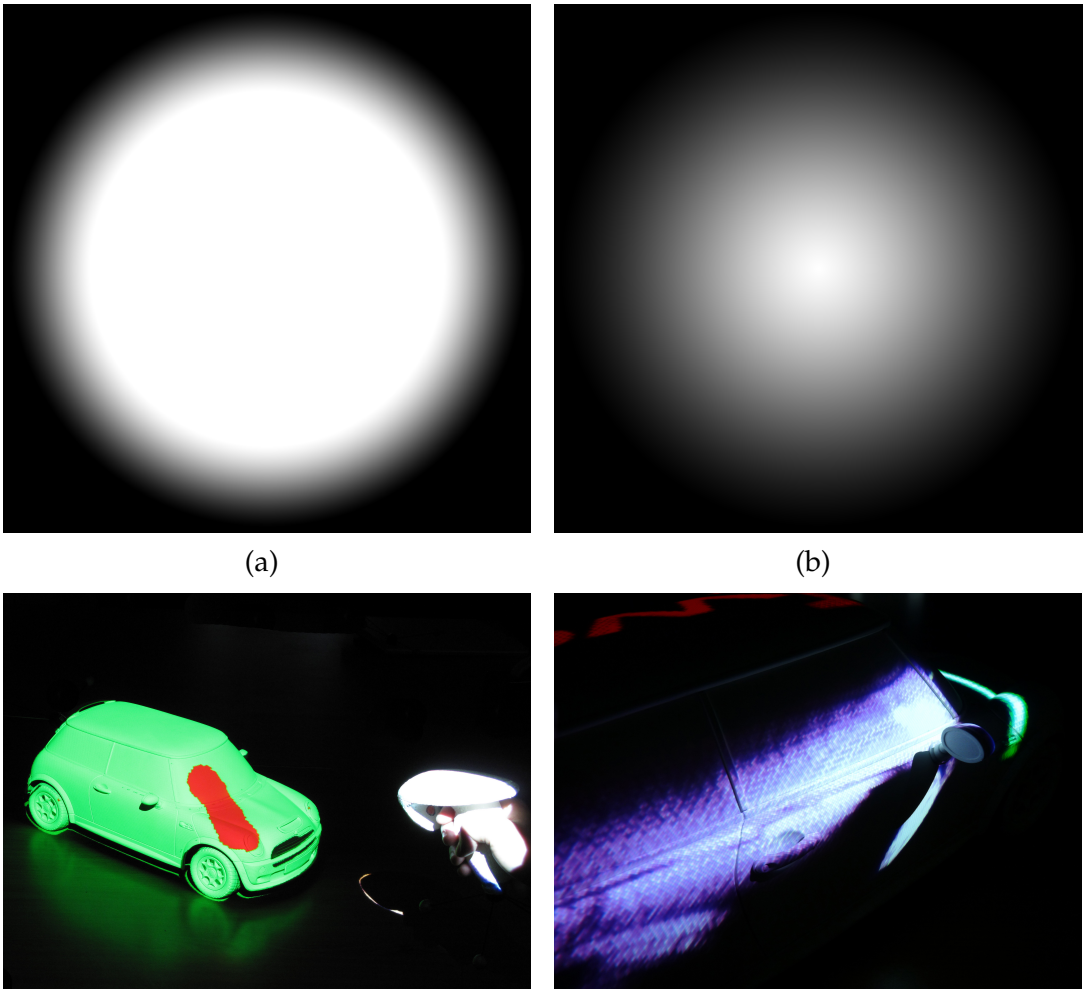
(a) (b)

**Figure 4.5:** The Airbrush Tool.

flow,

- virtual laser pointer, this task has a single user changeable attribute: The colour of the laser dot,
- drawing custom stencils on the tablet, this task has no user changeable attributes, and
- command entry on the PIP, this task also has no user changeable attributes.

When placed on the WS-TVC, this tool receives a rank of (4, 4). As this tool has a high task rank, the appearance of the tool is modified depending on the active task. In airbrush mode, an arc is projected onto the tool filled with the current paint colour. As with a physical airbrush, the further one holds the airbrush away from the spray surface, the wider the painted area. The angle of the arc represents the spray angle of the brush when painting. This representation allows the user to quickly see the brush mode. When painting, the top of the tool lights up with the paint colour, indicating a paint operation is in progress. This gives the user feedback that they are painting, even if the user has the device pointed away from any objects. In laser pointer mode, the projection onto the device changes to an arrow pointing towards the tip of the tool, indicating laser pointer mode. Laser pointer



(a) (b)

(c) (d)

**Figure 4.6:** (a) Hard edge airbrush mask. (b) Soft edge airbrush mask. (c) Airbrush result using a hard edge mask. (d) Airbrush result using a softer edge mask.

mode is activated through the PIP, but this could be changed to a button on the tool. Separate tools could have been provided for each of these functions; however as the form factors of the tools would be similar, the functions are combined onto a single virtualised tool. Command entry can be considered a special case, as it is always active. There is no need to change the appearance of the Airbrush during command entry, because this feedback is already shown on the Stencil tool.

### 4.2.3 Stylus

Annotation directly onto objects is supported through the use of the *Stylus* tool, as shown in Figure 4.7. The Stylus is an easy to hold pen-like device that requires minimal pressure of the user's grip to hold and manipulate. While it is possible to use this system with only the airbrush, the stylus is provided for tasks where the pistol grip is less suitable, such as annotation directly onto an object. The Stylus is used for the following tasks:

- Annotation onto design artefacts. This task requires a single attribute: the colour of the pen.
- Drawing custom stencils on the tablet.
- Command entry on the PIP. Again, this task also has no user changeable attributes.

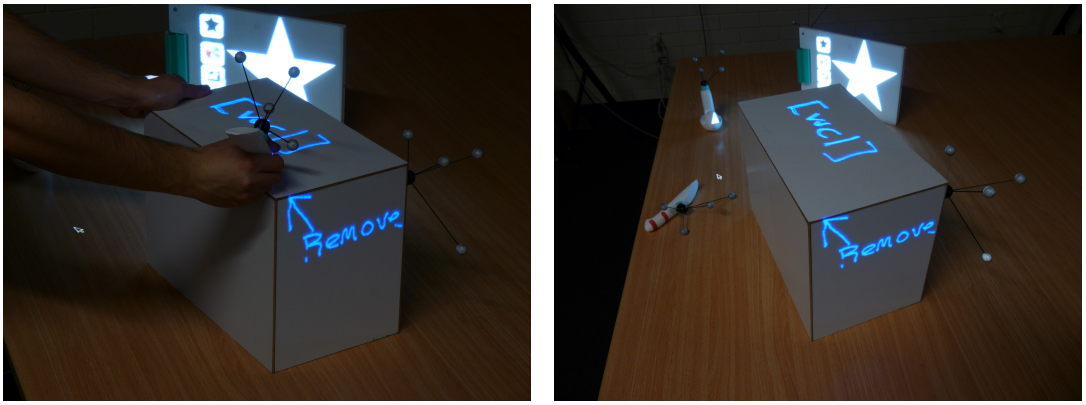
As the stylus is meant to be held with a pen-like grip, it has a much smaller area for projection at one end of the tool. The pen colour is projected onto this area. While the information projected onto the airbrush changes when painting is in progress, this is unnecessary for the stylus. The user activates annotation mode by touching the stylus to the object. No changes to the projected information are required, since the user is made aware of the change in state through passive haptic feedback; the fact that they can feel the stylus touching the surface of the object is enough. Examples of annotation are shown in Figure 4.8.

Note that two of the tasks supported by the Stylus are also supported by the Airbrush. This 'doubling up' was chosen so the user could work with whatever tool they are currently holding, and so two users can work at the same time with different tools.

The stylus tool can be placed on the WS-TVC at (3, 1). The low attribute rank is important for this tool. It is designed to be held in the hand like a pen, and therefore

(a) (b)

**Figure 4.7:** The Stylus Tool.




(a) (b)

**Figure 4.8:** (a) User annotating an object using the Stylus tool. (b) The annotation result.

can only provide a small area for projection. As the system grows more complex, more attributes would be added, such as line style and thickness, increasing the attribute rank. Eventually there will be too many attributes and another tool would need to be added, taking over some of the functionality.

## 4.3 Modelling Paint

The airbrushing algorithm has been developed to produce a similar paint effect to a physical airbrush. Conceptually, the paint leaves the airbrush in a cone shaped

volume. The hardness of the paint edges can be controlled to produce hard or soft, feathered edges. In addition, the intensity of the paint hitting the surface can be controlled by varying the distance between the brush and the surface. Up close, a smaller intense circle with hard edges is produced. From further away, the paint is less intense, with softer edges over a larger area.

Airbrushing is implemented in two steps. The first step calculates which parts of the object are hit by the paint. The second step then uses this information to update the texture image that is projected onto the object. The first step is implemented on the Graphics Processing Unit (GPU) using an OpenGL Shader Language (GLSL) shader. The scene is rendered to a Framebuffer Object (FBO) from the point of view of the airbrush. Rather than rendering colour information, the shader writes object IDs and UV texture coordinates at each pixel. The code for the fragment shader is shown in Listing 4.1.

```
uniform float AirbrushAngle;
uniform float objectID;
uniform float textureSize;

void main() {
    // Red and green values store UV coordinates
    gl_FragColor.r = gl_TexCoord[0].x;
    gl_FragColor.g = gl_TexCoord[0].y;

    // Calculate the intensity of paint and store in B value
    float screenDistance = length(gl_FragCoord.xy - vec2(textureSize/2,
       textureSize/2));
    float a = 1.0 - (screenDistance/AirbrushAngle);

    if (a > 0.0)
        gl_FragColor.b = a;
    // Pixel out of bounds, clear the values
    else
        gl_FragColor.rgba = vec4(0.0);

    // Alpha value stores the Object ID
    gl_FragColor.a = objectID;
}
```

**Listing 4.1:** Encoding paint data into FBO

Once this data has been written to the FBO, it is read back into main memory.

The program then iterates over this data, updating the textures for each object hit by paint. Pseudocode for this process is shown in Listing 4.2.

```
glBindTexture(GL_TEXTURE_2D, FBO->getColorbuffer());
glGetTexImage(GL_TEXTURE_2D, 0, GL_RGBA, GL_FLOAT, FBOData);
textures[0]->use();
for (unsigned int i=0;i<FBO->getWidth()*FBO->getHeight()*4; i+=4) {
    float u = FBOData[i];
    float v = 1 - FBOData[i+1];
    float intensity = FBOData[i+2];
    int objectID = (int) mFBOData[i+3]);

    if (intensity > 0.001) {
        if (objectID != previousID) {
            textures[previousID]->done();
            previousID = objectID;
            textures[objectID]->use();
        }
        //update the texture
        textures[objectID]->paintCircle(3, u, v, brushColour, intensity);
    }
}
```

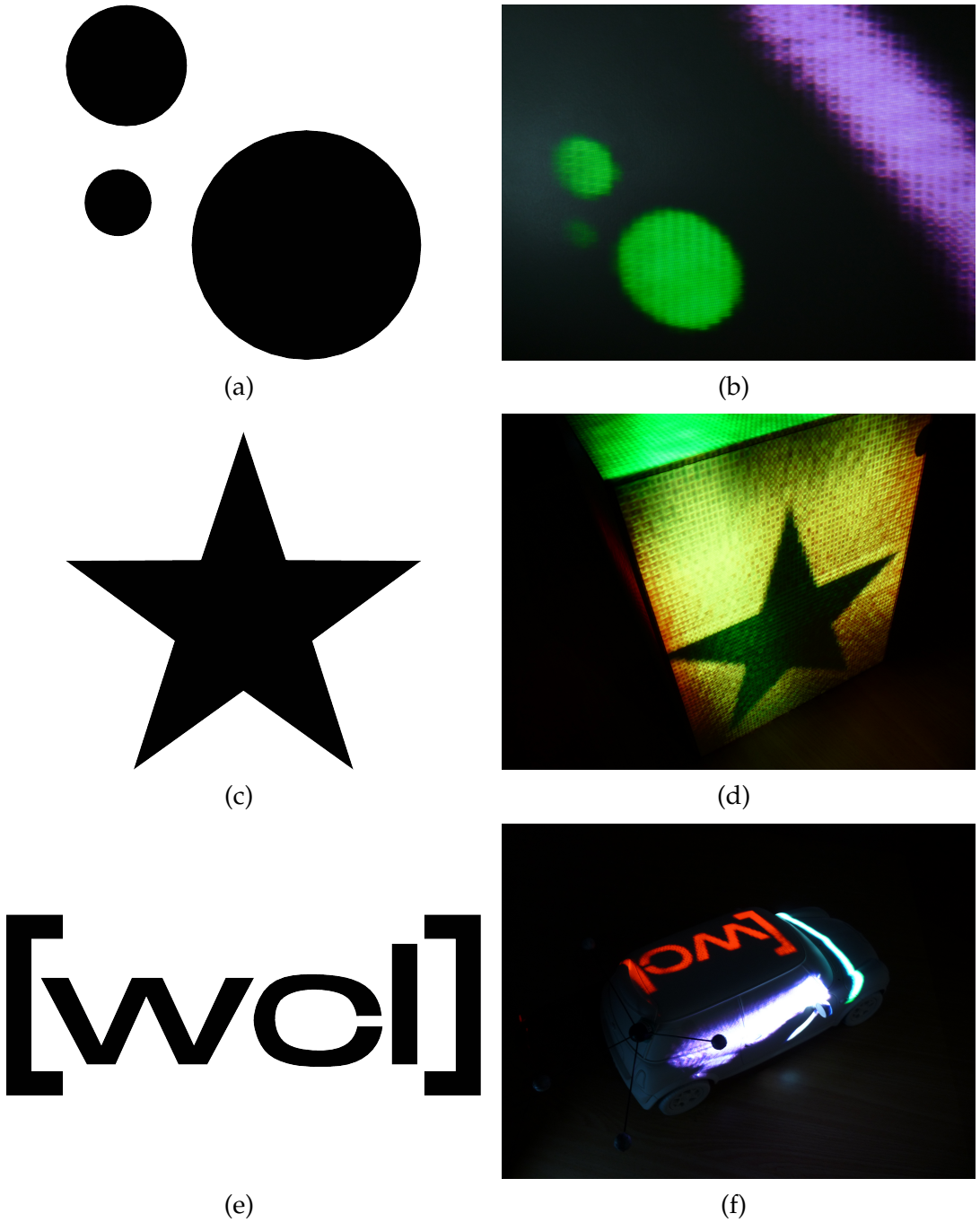
**Listing 4.2:** Processing FBO and updating textures

The sizes of the FBO and texture images effect the quality of the paint result, and also has a great effect on the performance of the system. Reading back and iterating over the data in the FBO is a slow process. In practice, a 64x64 pixel FBO, with 1024x1024 pixel texture images results in acceptible image quality without adversely affecting the framerate of the system.

### 4.3.1 Implementing the Stencil

Stencil shapes are represented as 3D geometry. This allows the system to take advantage of OpenGL's stencil buffering functionality. During the first phase of the paint algorithm, the stencil shape is rendered to the stencil buffer. This creates a mask in the shape of the stencil, correctly transformed based on the tracking data of the airbrush and stencil tools. The scene can then be rendered as described, with the stencil having the effect of masking data being written to the FBO. The mode of the

stencil can be changed simply by changing the operation of the depth test. Figure 4.9 shows three examples of painting with different stencil masks.




(a) (b) (c) (d) (e) (f)

**Figure 4.9:** Three different stencil shapes, and results of airbrushing using them. Subfigures (b) and (f) show the stencil acting as a hole for paint to pass through, where as (d) shows the result of the stencil shape acting as the mask.

## 4.4 Painting With Quimo

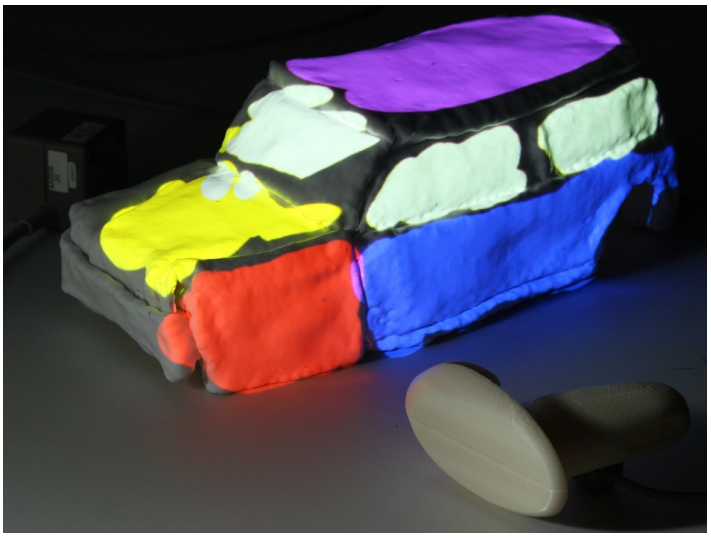
Quimo [Maa+12] is a white malleable material, produced as sheets, that can be moulded with the hands to produce low-fidelity physical prototypes of design artefacts, such as the car shown in Figure 4.10(a). The utility of Quimo can be enhanced by using SAR to project images onto the design mock-ups. The goal of Quimo was to provide the ability of using SAR technology early in the design process. The Quimo project was a group collaboration between myself, Ewald Maas, and Ross Smith. This section describes my contributions to the project, which focussed on integrating Quimo into the digital airbrushing system, allowing designers to use the system earlier in the design process.

### 4.4.1 Painting on Quimo Prototypes Without a Virtual Model

Digital Airbrushing, as described in the previous sections, requires a 3D geometric model of the object being painted. It is important for the virtual model to closely match the physical version to produce a compelling paint effect. This requirement limits how early in the design process the system can be used, as 3D virtual models are not created early on in the design process. However, with some modifications, Digital Airbrushing can be used with Quimo, without an underlying virtual model.

Rather than using precisely matching 3D geometry, a bounding box texture map that encompasses the Quimo model is used. The physical object is placed inside this volume, and the designer paints as normal. The nature of projected light ensures the virtual paint is placed in the correct location on the object, even without a corresponding virtual model. Figure 4.11 shows how the textured bounding box, when drawn from the point of view of the projector, correctly maps digital paint onto the mock-up. This works well when the bounding box closely fits the sculpted model, but projections do not align as well if the bounding box is significantly larger

(a)



(b)

**Figure 4.10:** Creating a Quimo prototype. (a) A sheet of Quimo sculpted into the shape of a car. (c) The result of airbrushing onto the Quimo prototype.

than the model. The Quimo model also cannot be rotated without a 3D model, as the projected light would appear to move on the surface. Figure 4.10(b) shows the result of airbrushing on the car prototype. Using the same texture map, the paint could also be applied to a CAD model of the design at a later stage of development. However, the goal of this technique is to preview different material properties early in the design process, before CAD models have been created.

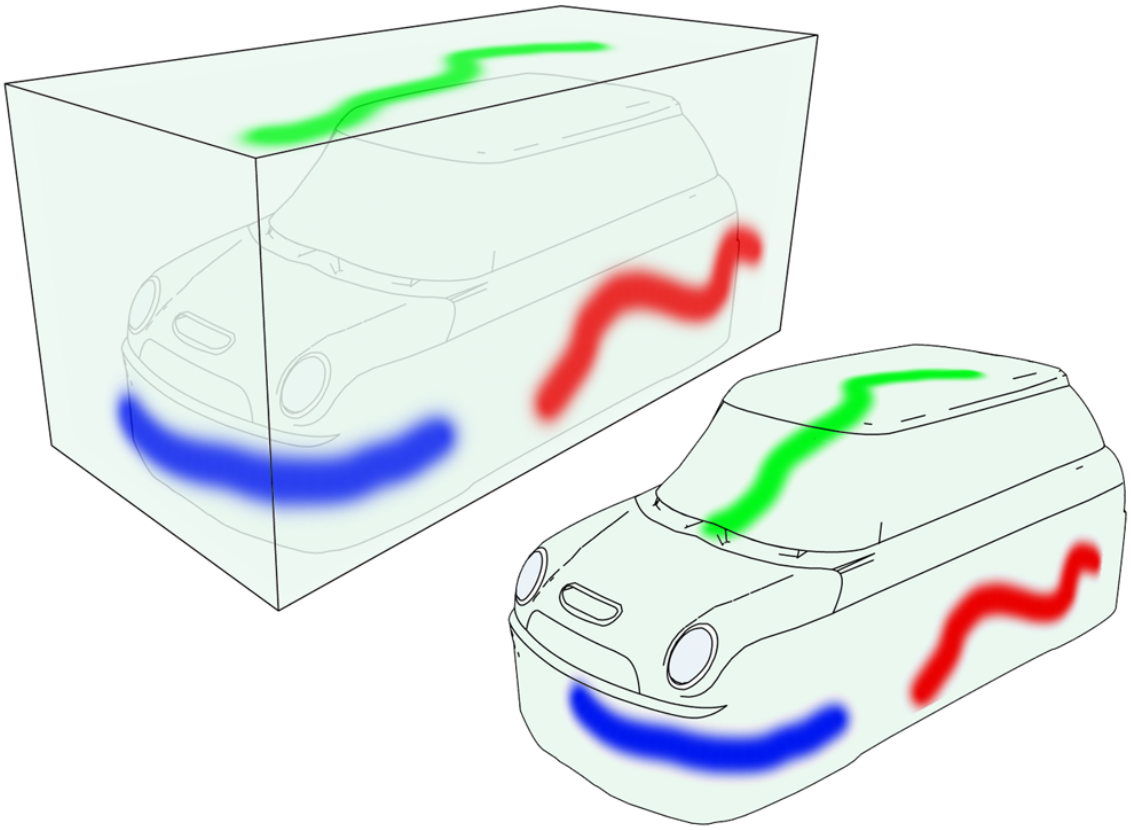
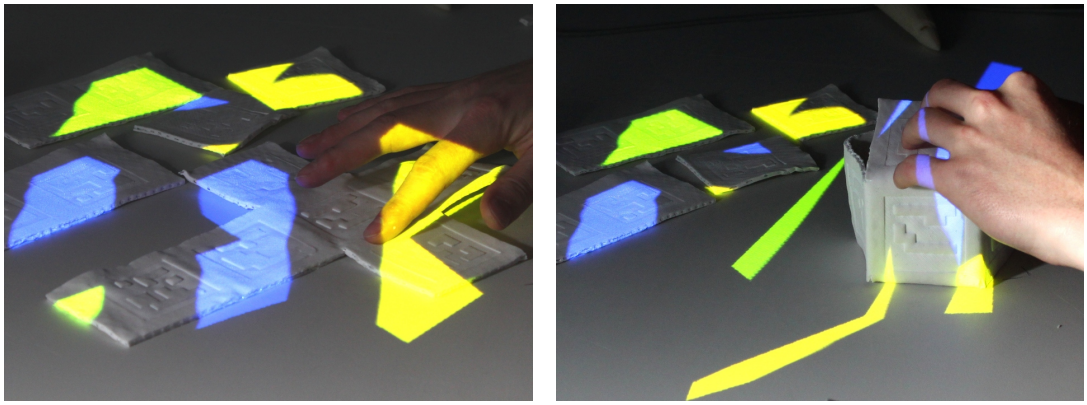


**Figure 4.11:** Left: The virtual bounding box, drawn from the point of view of the projector. The location of the physical mock-up is shown inside the bounding box. Right: The projected textures map correctly onto the physical mock-up.

### 4.4.2 Simultaneous Physical and Virtual Modelling

The previous section described how a Quimo model can be used with Digital Airbrushing without a 3D model. However, Quimo sheets can be constructed with embedded fiducial markers [Maa+11]. This allows a matching 3D model of the material to be constructed in real-time as the designer sculpts. The system also supports cutting the material into pieces. By utilising this information, Digital Airbrushing can be used in a more natural design setting, as the user can move, bend, and cut the material as needed, with the SAR projections "sticking" to the surface as it is deformed. An example of this technique is shown in Figure 4.12. Here, an approximately A4 sized Quimo sheet is painted with the airbrush system. The sheet is then

cut into several pieces and moved about, as shown in Figure 4.12(a). The projected paint is correctly projected onto the pieces in the same way as if a sheet of paper was painted with physical paint. Figure 4.12(b) shows how the paint is projected even as the Quimo sheet is folded.



(a) (b)

**Figure 4.12:** Deforming the Quimo material with projected images attached to markers. (a) Sheet of Quimo with projected images. (b) Folding material into a cube shape.

#### 4.4.2.1 Deformable Surface Implementation

A naïve approach to implementing the deformable surface would be to simply treat each fiducial marker as a tile that can be painted. However, the reliability of the tracking is reduced by the low contrast in the images obtained from the camera using the embedded markers. This is exacerbated by the user obscuring the camera's view of the markers with their hands or tools. This results in a projected image that flickers and is unusable to the designer.

To overcome this, the entire sheet is treated as a deformable polygon mesh, with vertices placed between each fiducial marker. The position of each vertex is determined by its neighbouring markers. During tracker updates, each visible marker contributes a position for each vertex it affects. These positions are then averaged for each vertex to obtain its final location. This approach means there are four markers that can contribute to each of the inner vertices, two markers for each edge vertex,

with the corners the only vertices without this redundancy. If the types of deformation supported is constrained, redundancy can be added to these markers by using more distant neighbours.

## 4.5 Summary

Building on the methodology presented in Chapter 3, this chapter has demonstrated new user interaction techniques for SAR, exploring the middle to mid-high region of the WS-TVC. Three physical tools have been developed, overloaded to provide four logical tools. The prototype application, Digital Airbrushing, can be used by industrial designers to aid them in the design process. A two handed technique allows a designer to digitally airbrush onto an object using of a semi-virtual stencil for masking areas from the paint. The designer is able to annotate 3D models by drawing on a physical mockup with a stylus.

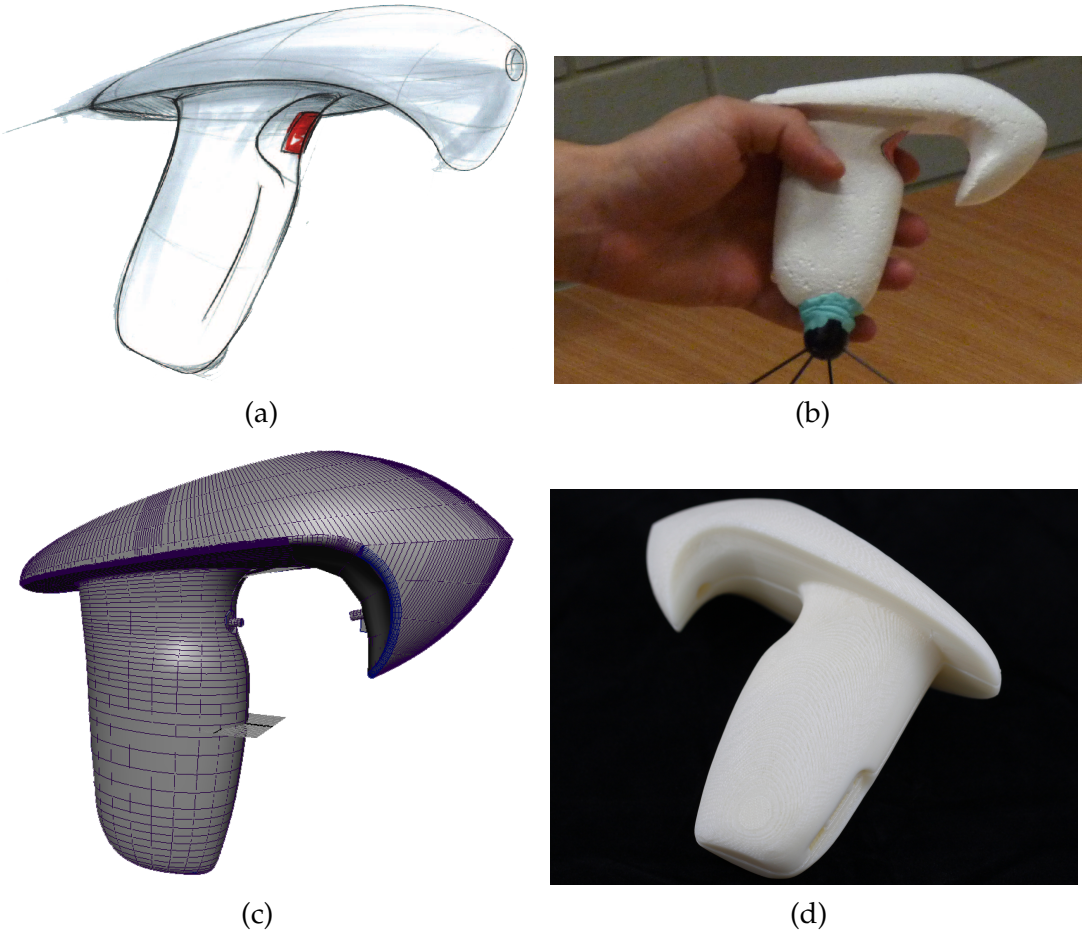
# 5

# Augmented Foam Sculpting

This chapter describes Augmented Foam Sculpting, a PVT based application allowing industrial designers to create 3D CAD models of design prototypes by sculpting foam. SAR is used to project visualisations onto the foam to aid the designer during the sculpting process. The chapter begins by outlining the motivation for applying SAR in this way, and how this new modelling technique fits into existing design practices. The functionality of the system is described in detail along with the visualisations developed to enhance the modelling process. The system's current limitations are discussed, and potential solutions to these limitations are proposed. Finally, results from both a system performance analysis and expert review are presented.

## 5.1 Physical Models in the Design Process

As described in Section 2.6, industrial design typically involves an iterative process where the designer progresses from hand drawn sketches (Figure 5.1(a)), through to handmade physical mock-ups (Figure 5.1(b)), to CAD models (Figure 5.1(c)) and high fidelity prototypes, before arriving at the final design (Figure 5.1(d). The aim



(a) (b)

(c) (d)

**Figure 5.1:** As the design process progresses, designers move from working with (a) hand drawn sketches, to (b) hand crafted mock-ups, through to (c) CAD models, and (d) the final product.

of Augmented Foam Sculpting is to combine physical mock-up creation with CAD modelling into a single process. The CAD model is created in real time as the designer sculpts the physical version. The designer physically sculpts a design prototype from foam using a hand-held hot wire foam cutter. This technique was inspired by working with industrial design students on the tools presented in Chapter 4. Both the foam and cutting tool are tracked, allowing the system to digitally replicate the sculpting process to produce a matching 3D virtual model. The system is able to playback the actions of the designer actions for additional tasks, such as training.

Augmented foam sculpting allows designers to create 3D models of designs with traditional techniques already in use in the design process, and can be integrated with little change in existing workflows. Constructing mock-ups and prototypes by carving light weight Styrofoam with a hot wire foam cutter is a common technique for industrial designers. Physical prototypes are used to aid the designer in developing ideas and design concepts. Designers are also able to find problems with designs that are difficult to identify without a physical mock-up. Once these mock-ups have been evaluated, and key design decisions made, they are used as a basis for CAD models. Previous work has investigated reverse engineering the CAD models from physical mock-ups [HC03]. This process involves scanning the object and processing the resulting point cloud to produce a usable model. Augmented foam sculpting completely removes the need for this step, as the 3D model is generated while the sculpting is taking place. Virtual models created with the foam cutter are often less complex than the scanned model compared to commercial laser scanners, as vertices are only added where a cut occurs. This results in edges that are much better defined. This is beneficial if the design features movable parts such as hinged joints, because the designer does not have to spend time "cleaning up" the 3D model in CAD software.

Augmented foam sculpting also provides new abilities and more flexibility during the design phase. The designer can save versions of the CAD model at different stages of sculpting, allowing them to explore several ideas in the digital realm, based on a single physical mock-up. The foam model is enhanced by projecting information onto it. Using the digital airbrushing system presented in Chapter 4, the designer is immediately able to paint onto their newly sculpted mock-up, or to annotate on the object indicating changes for future versions, or clarifying design features. The paint textures created can be saved with the model for future reference. Furthermore, SAR can be used to instruct the user how to produce a specific model.

Two visualisations have been developed to support this instructive process: *Cut Animation*, which projects cutting paths onto the object, and *Target Visualisation*, which colours the foam to illustrate areas to be removed. SAR can also be employed to test different materials by lighting the foam mock-up with projected textures. Many more visualisations could potentially be incorporated into this system. Therefore, the visualisations presented in this chapter provide a foundation of visualisations which could be extended in the future.

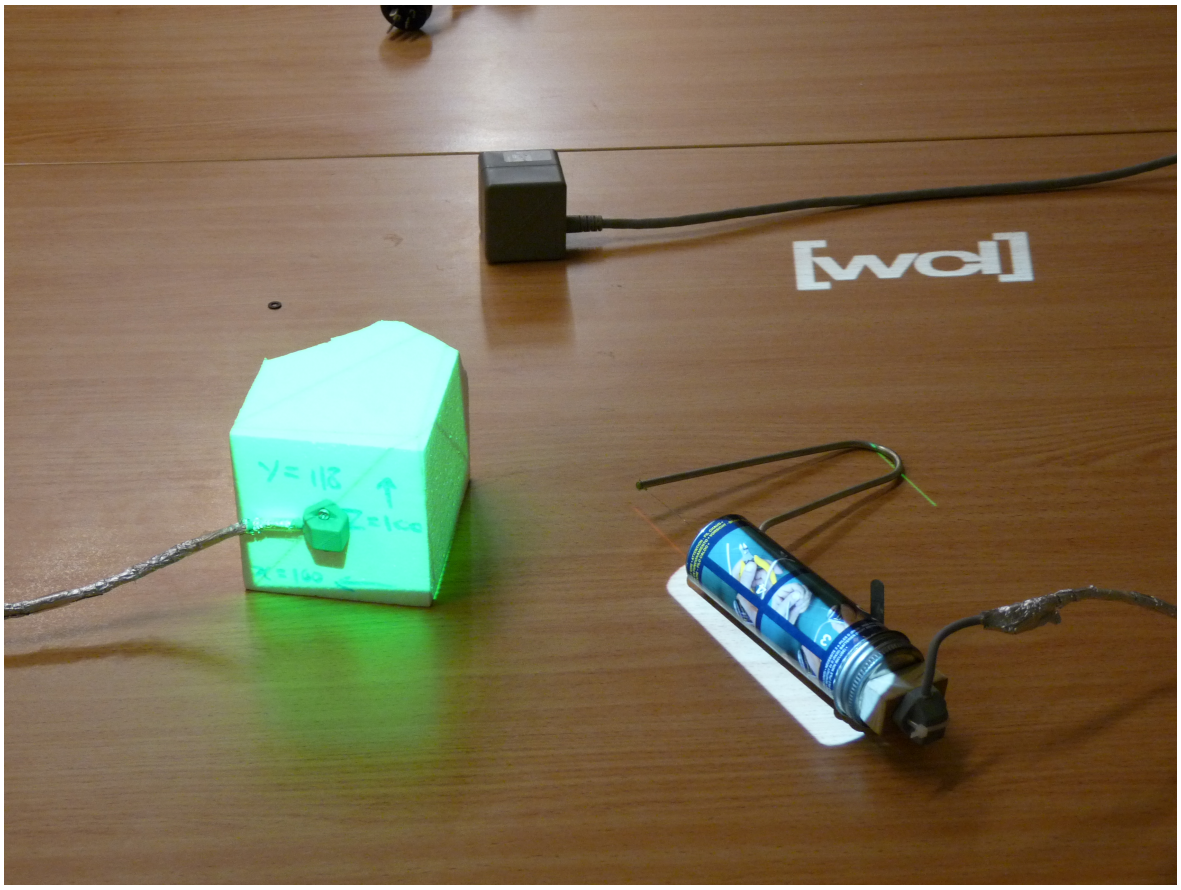


**Figure 5.2:** The Augmented Foam Sculpting System

## 5.2 Existing Approaches

3D modelling has been a common focus of AR and VR research. Spray Modelling [Jun+04] offers an intuitive way of developing 3D models. While traditional appli-

cations require the user to create a model through explicitly adding and modifying polygons, Spray Modelling allows the user to build up 3D models by spraying virtual material onto a base mesh, in a similar way to how an artist sculpts with clay. SKETCH [ZHH06] uses a stylus based interface. A user builds a 3D model by successively drawing and refining an object from different angles. ModelCraft [Son+06] also uses natural interaction for annotating and changing 3D models. Paper or rapid prototype models are printed with special patterns. A user annotates changes for the design using special pens that track their position on the object's surface. A gesture based command system allows the user to specify changes for the 3D model. This is a simple, intuitive interface, but the user cannot directly modify the 3D model. Augmented Foam Sculpting builds on the idea of mimicking physical tools for 3D modelling.

Surface Draw [SPS01] allows a user to sculpt 3D models in a VR environment. The user defines a path for a surface through the motion of their hand. The motivation for developing Augmented Foam Sculpting is similar to that of Surface Draw. While their process is additive, Augmented Foam Sculpting uses a subtractive process where geometry is removed from an existing object. In addition, Augmented Foam Sculpting system benefits from a physical object, where as Surface Draw is a completely virtual system.

Digital Foam [STP08] is a unique input device using conductive foam that is manipulated by the user's hands. As the foam is compressed, the resistance through the foam changes. This change is measured by an array of sensors. The user manipulates 3D models by gesturing into the foam, such as pinching. The main difference between digital foam and Augmented Foam Sculpting, is Augmented Foam Sculpting has the user sculpt a physical piece of foam to create a 3D model, where as Digital Foam is a computer input device allowing gestures as a means of modelling in a traditional CAD environment. Illuminating Clay [PRI02] is a system that allows

the user to sculpt a landscape with clay. A ceiling mounted laser scanner captures the height of the clay to produce a depth map. Visualisations are then projected onto the clay. The main limitation of this system as a modelling tool is it can only create height maps of the surface. Augmented Foam Sculpting can be used to create arbitrary 3D objects.

## 5.3 The Modelling Process

As previously mentioned, Augmented Foam Sculpting (see Figure 5.2 for the physical setup) allows a designer to create 3D models using traditional sculpting techniques and tools already used in the industrial design process. The user begins with a block of Styrofoam of known dimensions, and uses a hot wire foam cutter to remove pieces of foam to produce the design artefact. Both the foam and the cutting tool are tracked with six degree of freedom (6DOF) sensors. The system digitally replicates the sculpting as a series of Constructive Solid Geometry (CSG) [LTH86] Boolean difference operations. The geometry removed from the digital model is the same as what is removed from the physical foam. The end result is a physical foam design artefact and a matching 3D virtual model that can then be the basis for further work in a traditional CAD environment. This ties in with existing industrial design processes, where a CAD model is often created based on a physical prototype. The foam mock-up can immediately be used within a SAR system.

The Augmented Foam Sculpting approach fits in well with current design techniques. Previously, designers would sculpt mock-ups to test design ideas, then model these in a CAD package, or use a 3D scanner to obtain geometry from the mock-up. In the system presented here, 3D modelling is combined with sculpting. The user does not have to work within the confines of a CAD user interface, and can more freely articulate their design ideas.

To create a model, the user first enters the dimensions of the foam block so an initial virtual model can be created. This model is then projected at a known location on the workbench. The user places a tracking sensor anywhere on the foam, and places the foam over the projection. The system records the offset from the tracker to the foam. This allows the user to quickly change pieces of foam without having to perform a time consuming calibration process at startup. This calibration step is described in more detail in Chapter 7. The user can then create the virtual model by sculpting the foam.

### 5.3.1 Performing CSG Operations

Sculpting is modelled as a series of CSG Boolean difference operations on a starting triangle mesh. The system begins recording the wire path through the foam when a line segment representing the cutting wire intersects with a bounding box of the foam. The position of the line segment is recorded at regular intervals as it passes through the foam. The interval chosen limits the resolution of the resulting geometry. However, in practice, the minimum possible value is dictated by the performance of the tracking system. 3mm interval was been used for the prototype system. This value was chosen after experimentation, and is limited by the accuracy of the tracking system. When the line segment leaves the bounding box, the cut is complete, and the recorded points are used to modify the 3D geometry.

Once a cut has been made, a volume suitable for the CSG operation must be generated from the cut path. The actual path of the wire forms the main surface of the volume (Figure 5.3(b)). Four new vertices need to be added at positions outside the foam. The position of these additional surfaces is not important, as long as they fall outside the foam volume. The first and last recorded line segment are used to calculate the location of the new vertices. A direction vector for the position of the new vertices is calculated from the centre of the foam to the average between

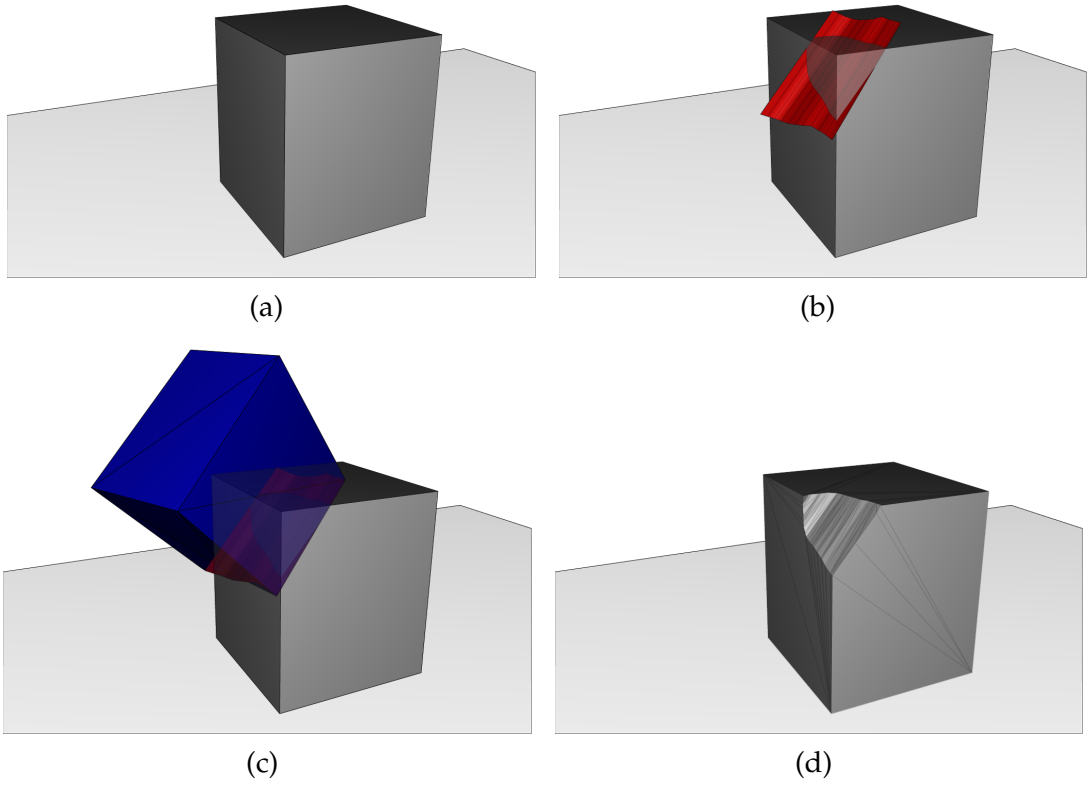



**Figure 5.3:** Generating cut geometry and performing CSG operations. The foam is represented as a box in the system (a), the path the foam cutter takes as it moves through the foam is recorded (b). From this path, a volume is generated suitable for the CSG operation (c). The CSG operation is performed, removing a section of foam from the virtual model (d).

the four points of the line segments. The direction vector is used to place the new vertices away from the original line segments and outside the foam geometry, using the algorithm described in Listing 5.1.

In the pseudocode depicted in Listing 5.1, Offset is simply a number large enough to ensure the new vertices will be outside the foam. This value does not need to be calculated for each cut, as the physical dimensions of the cutting tool limit the maximum size of cuts. Based on the size of the cutting tool, the prototype developed uses a value of 100mm. Once these vertices have been calculated, they are used to create the remaining walls of the cut volume. Figure 5.4 shows a 2D cross section of how

```
void calculatePoints(Vertex s1v2, Vertex s1v2, Vertex s2v1, Vertex s2v2)
{
    Vertex Pavg = (s1v1 + s1v2 + s2v1 + s2v2) / 4
    Vector Direction = Pavg - FoamCentroid

    Vertex NP0 = s1v1 + (Direction*Offset)
    Vertex NP1 = s1v2 + (Direction*Offset)
    Vertex NP2 = s2v1 + (Direction*Offset)
    Vertex NP3 = s2v2 + (Direction*Offset)
}
```

**Listing 5.1:** Calculating new vertices for the cut volume

the new vertices are used to create a 3D triangle mesh, particularly the side walls. The final product of this construction is an enclosed 3D volume suitable for CSG operations (Figure 5.3(c)). A Boolean difference operation is performed, simulating the volume of foam that was removed by the cutting tool (Figure 5.3(d)). The new foam geometry matches the foam within the limitations of the tracking system, and is used for further sculpting operations.

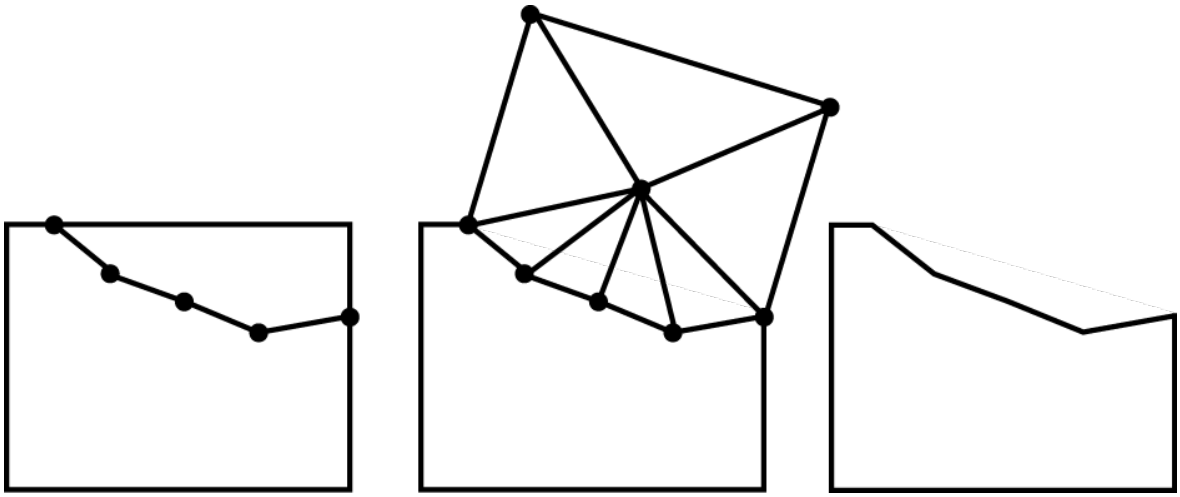


**Figure 5.4:** Cross section of generated cut geometry

The system makes use of the CGAL [CGA11] library to perform CSG calculations and to generate 2D texture coordinates for models. During testing, the library often raised errors when performing CSG operations. These errors were found to be caused by jitter in the tracking system causing adjacent vertices to overlap. To cope with this, an adaptive resampling technique was implemented. A CSG operation is

attempted, and if it fails the cut points are resampled at a higher minimum distance. The distance is incremented at 1mm intervals until the cut succeeds.

### 5.3.2 Handling Difficult Cuts

Augmented Foam Sculpting functions correctly with concave cuts with several curves. The algorithm will generate correct cut geometry even if the cut path passes through the centroid of the bounding box. The degenerate case is if the user attempts to remove more than half the foam in a single cut. Here, *Direction* must be reversed so the correct section of geometry is removed. By assuming the designer wishes to keep the portion of foam holding the tracking sensor, this condition can easily be tested for. If the location of *Pavg* is closer to the tracking sensor than *Centroid*, then the direction vector must be reversed. In practice, this rarely occurs, since the physical size of the foam cutter usually prevents such drastic cuts from being made.

The CSG operation may fail for extremely concave shapes due to the way side walls of the cut geometry are generated. The centroid calculated for the side walls can be outside of the cut path, causing an invalid surface to be constructed. A simple modification to the algorithm using a more advanced triangulation technique [KKT90] would resolve this problem. This scenario is illustrated in Figure 5.5. In testing, this has not been an issue, as users generally make small, rather than drastic, complex cuts into the foam.

As the entire cut through the bounding box is recorded, cuts into small, concave areas will be modelled correctly. This is because the user must remove the foam cutter from the bounding box without touching the physical foam. However, the system will generate invalid cut geometry in cases where the path through the foam crosses itself. A future iteration of this software would contain a modified algorithm to swap vertices that cause the path to cross itself before performing the Boolean operation. This would result in the correct cut being performed.

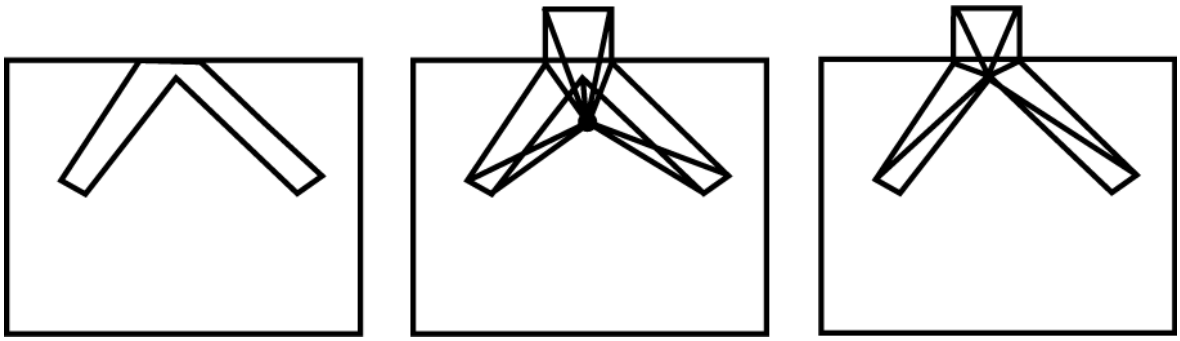


**Figure 5.5:** Highly concave cut paths (left) cause invalid geometry to be constructed (centre). In this scenario, side walls should be triangulated as shown (right)

## 5.4 SAR Visualisations

While the modelling system itself can function without any video output, the usability of the system can be greatly increased through the use of SAR. The following section describes the visualisations that have been implemented to enhance the system: Cut Animation, Target, and volumetric texturing. In addition to these visualisations, sculpted objects can be used in the digital airbrushing system described in Chapter 4. All of these visualisations are projected onto the foam and updated in real time as the user sculpts.

### 5.4.1 Cut Animation

Cut Animation (Figure 5.6) displays the cuts required to reproduce a model, one after the other. This visualisation can be used for instruction. For example, an expert can use the system to produce an object, with the system recording the cuts made. The cuts can then be played back to a novice user, instructing them how to produce the object. The Cut Animation technique can guide the designer to reproduce any of the previously saved versions of a design.

This visualisation projects lines representing the path taken by the cutting tool through the object. The cut to be made is animated, illustrating the technique used to make the cut. This aids the user in understanding how to produce the model. The

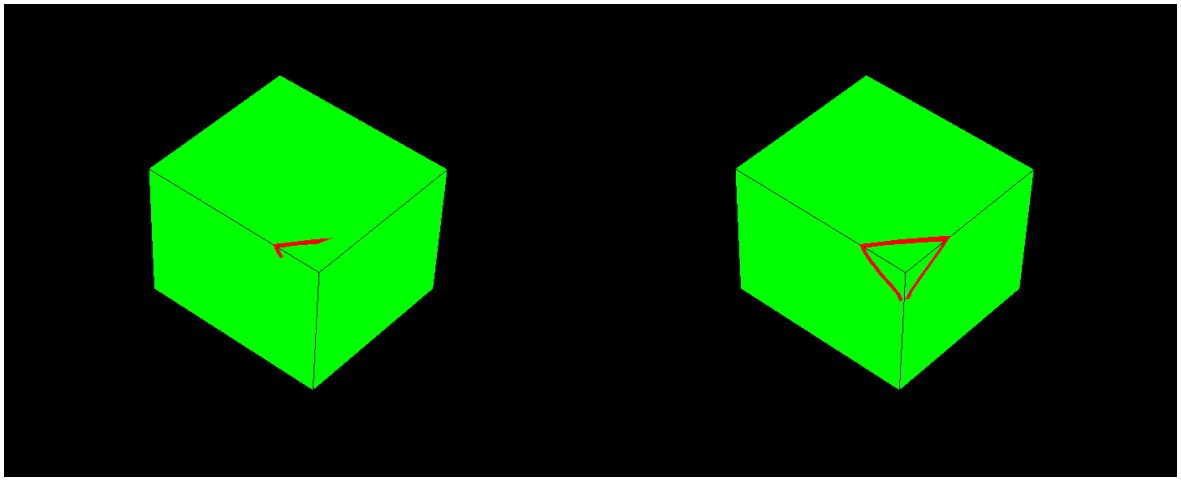


**Figure 5.6:** Animating the path for a cut by projecting onto the foam

line is animated such that the cut appears at the start and moves through the object. The user follows the lines to make the cut, and the system then moves to the next cut. This process continues until the model is complete.

The visualisation is implemented using two framebuffer objects and GLSL shaders. The rendering process consists of three stages:

1. The foam is rendered as normal to a FBO. This captures both the colour and depth information of the foam.

2. Geometry representing the cut through the foam is drawn to another FBO. However, the FBO is only written to if the current fragment depth is similar to the depth recorded for the foam at the same pixel location. This ensures that cuts are only drawn on the surface of the foam, and prevents self occlusion of the cut geometry for complex paths.

3. The two FBO are combined in a final shader. Here, the foam is drawn to the projector. However, for fragments containing the cut, the cut is drawn instead. This has the effect of superimposing the cut path on top of the foam geometry.

By increasing the number of cut points drawn over time, the path the cutting tool took through the foam is animated, illustrating to the user how the cut should

be made. This approach is quite flexible, and much less computationally intensive than if intersection points between the foam and cut path were calculated. As the visualisation only involves drawing the path as geometry, it can handle complex cut paths with many curves.

### 5.4.2 Target

Target (Figure 5.7) visualises the sections of foam that must be removed from the foam to match a target 3D model. This visualisation would also be useful to create a modified version of an earlier design mock-up.

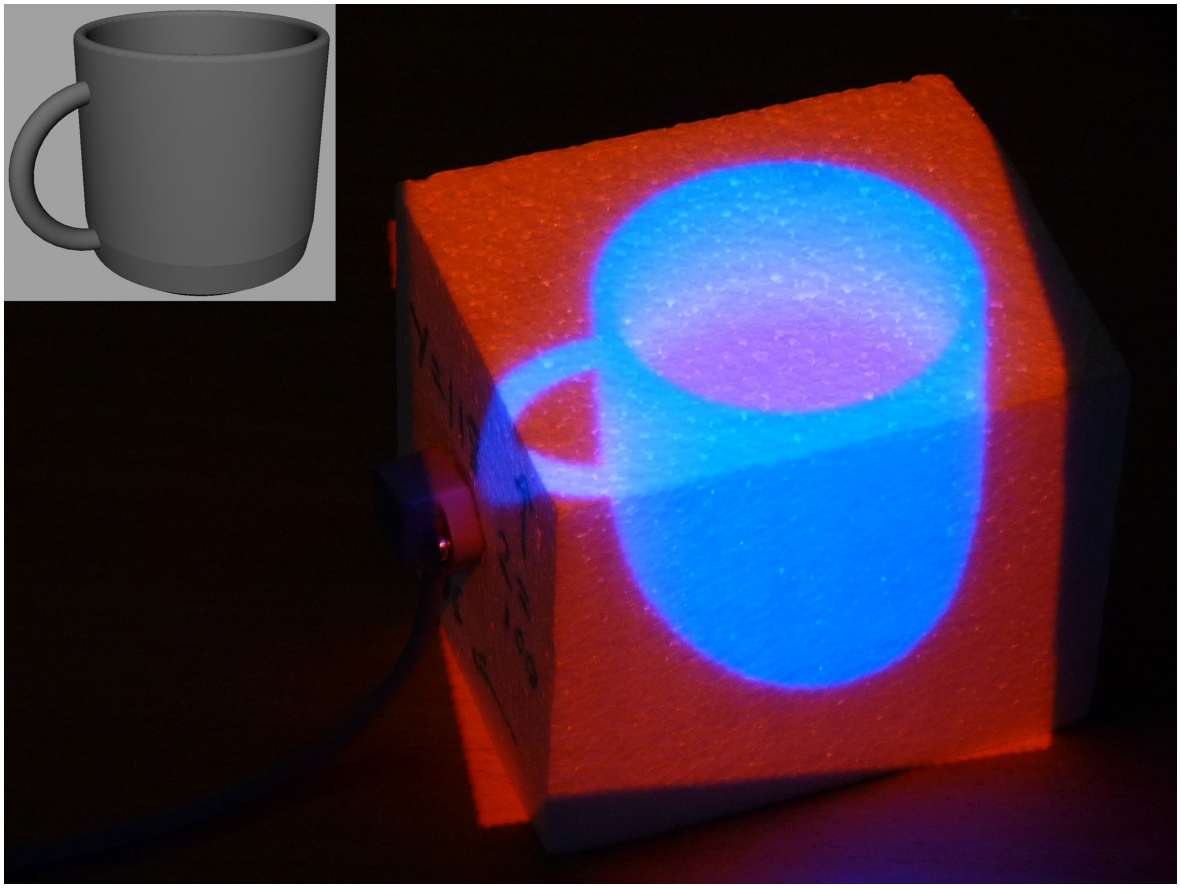


**Figure 5.7:** Target Visualisation onto foam block, inset: target model. The area of the foam encompassing the target cup is visualised as blue, indicating less foam needs to be removed. The outer areas of the foam are red, indicating larger amounts of foam need to be removed to match the target.

This visualisation represents sections to be removed as a colour gradient pro-

jected onto the foam. Green areas on the foam indicate that no more cutting is required. The remainder is coloured using a gradient from red to blue. Red indicates a large area should be removed, and blue indicates that only a small area needs to be removed. There are potential problems in using red, green, and blue as the sole indicators for this visualisation in the case of users who are colour-blind. In these cases, the visualisation could be modified to use a gradient from black to white, or use a different combination of colours.

As with Cut Animation, this visualisation also utilises depth information to generate the colour gradient. The scene is organised with the target 3D model placed inside the bounds of the foam. Two FBO are used to individually record the depth information of the foam and target model. The two depth textures are then utilised in a fragment shader. For each fragment, the two depths are compared. If the foam depth is greater or equal to the target no more foam should be removed. Otherwise the colour is mixed between red and blue, depending on the difference in depths. Due to the non-linear nature of the depth buffer, the differences in recorded depths are very small. To achieve a reasonable colour blend, the mix value is multiplied by a constant. The value of this constant varies depending on the distance from the projector. In testing, a value of 0.005 was found to be an appropriate constant. Pseudocode for the fragment shader is shown in Listing 5.2.

```
void main()
{
  /* discard empty fragments */
  if (foam == 0)
    discard;
  if (foam >= target)
    output GREEN;
  else
  {
    mixValue = (target - foam) * dConst;
    output mix (BLUE, RED, mixValue)
  }
}
```

**Listing 5.2:** Fragment Shader for Target Visualisation

**Figure 5.8:** The demonstration projector setup. The projector is mounted on a tripod, projecting over the top the the chair and onto the desk. The computer and tracking system is placed on the left of the desk.

All calculations for this visualisation are performed from the point of view of the projector. This limits the use of this visualisation to single projector systems, as the colour calculated for a single point on the foam will be different depending on the location of the projector. In the prototype, the projector was placed low and positioned over the user's shoulder, as shown in Figure 5.8. This positioning made the point of view of the user similar to that of the projector. This limitation can be resolved by performing calculations from the point of view of the user. The resulting gradient texture can be re-projected from the point of view of the projectors, such that each projector displays the same colour for each point on the foam. However, the current system does not track the user.

### 5.4.3 Volumetric Texturing

The SAR projection system is also used to project different textures and information onto the foam while the system is in use. The wireframe (Figure 5.9) of the 3D model is projected onto the foam during the sculpting session. This allows the designer to easily verify that the generated 3D model matches the physical version.

**Figure 5.9:** A user sculpts into the foam, with the wireframe projected onto the physical model.

While the wireframe is useful for verifying the correctness of a generated model, it can be distracting to have projected all the time. Augmented Foam Sculpting supports projecting 3D textures onto the surface of the foam, with the designer having the ability to switch to the wireframe as needed. 3D volumetric textures are ideal for Augmented Foam Scupting. The position of the fragment in 3D space can be used directly, without needing to generate a 2D texture map. Texture map generation is

a computationally expensive process, and would need to be recalculated each time a cut is made in the foam. In addition, it would also be difficult to generate a map in such a way that 2D textures could be reused on multiple models, as they would have drastically different geometry. 3D textures will continue to look correct even as many cuts are made in the foam. In addition, compelling 3D textures simulating complex materials can be generated procedurally. Procedural textures reduce the system's video memory requirements, as texture images are not required.

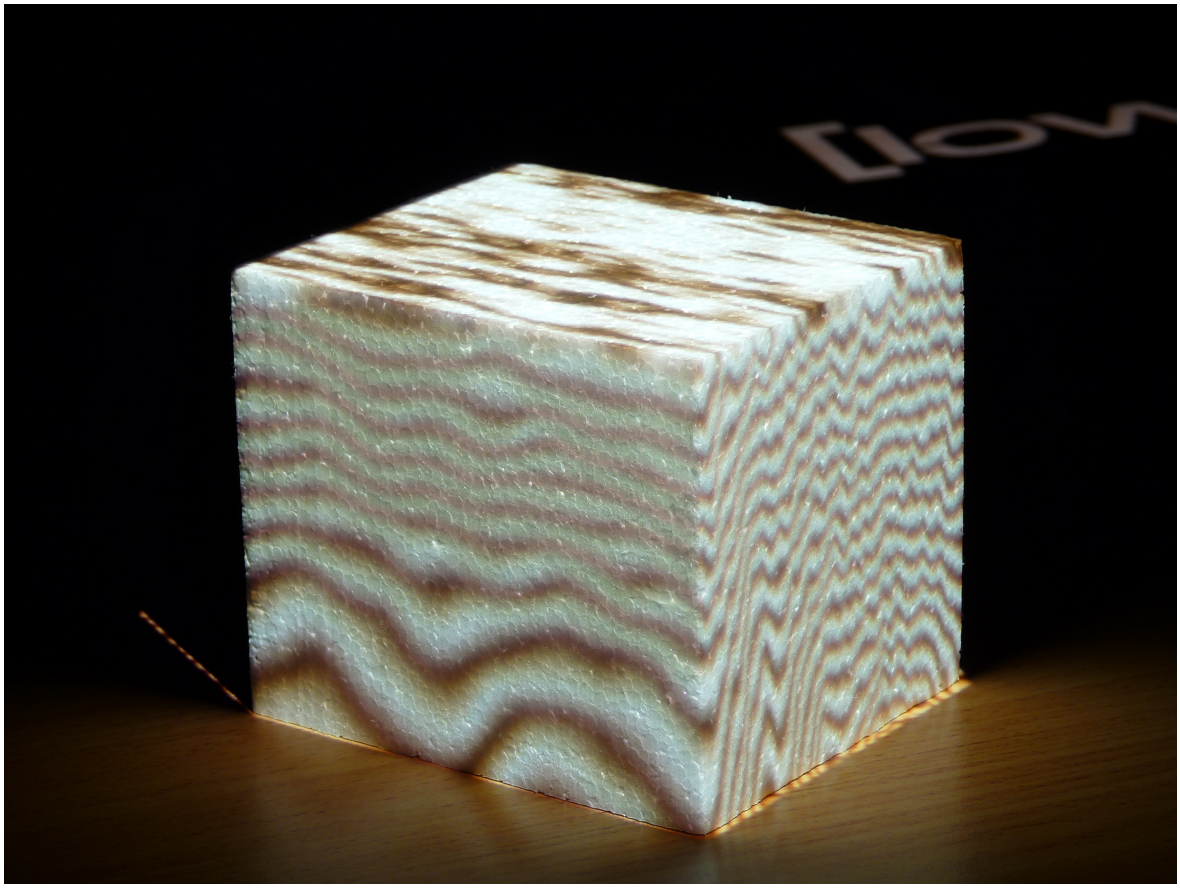


**Figure 5.10:** Projecting a 3D procedural wood texture onto the foam

An example is re-texturing the foam as wood using a 3D procedural texture (Figure 5.10). The texture is implemented as a GLSL shader, using the approach described by Rost [Ros06]. Rost's approach includes a view-dependent component for more realistic specular highlights. Specular highlights were removed from the shader as the user's position is not tracked in the system. The wood appearance

also remains consistent as cuts are made, giving the illusion that pieces of wood are actually being removed. New textures and materials are easily added through the pluggable GLSL shader system.

### 5.4.4 Painting the Prototype

An important requirement for this system is that once the designer has finished sculpting a design artefact it can be used for later stages of the design process, including use in standard CAD packages. However, by extending the Digital Airbrushing system presented in Chapter 4, a system allowing iterative design entirely in the SAR domain has been created.

One major limitation of Digital Airbrushing is the need for 3D models of the design artefact before the designer is able to paint. Previously, models were created using standard 3D modelling software, which was a time consuming process. Using the cube-map texturing technique partially overcomes this, but the objects cannot be moved while the system is running. Augmented Foam Sculpting addresses this by allowing the designer to sculpt a design, and have the 3D model immediately available for further work. The designer is immediately able to airbrush onto and annotate their design.

First, 2D texture coordinates are generated for the object. Texture map generation is only performed when requested by the designer, as when changing modes, rather than continuously as cuts are made. Multi-texturing is used to provide both a paint and annotation layer. These textures can be saved to disk independently, along with the 3D model using common file formats for later use. A designer can then use these to produce a high quality ray traced render of their design using existing software.

## 5.5 Evaluation

This section describes the evaluation of the Augmented Foam Sculpting system. The 3D geometry generated by Augmented Foam Sculpting is compared to that of a commercially available laser scanner. The performance of the system is discussed in terms of both model accuracy and computational complexity. In addition, a discussion of an expert review is provided.

### 5.5.1 Quality of 3D Models

A common workflow for industrial designers is to use a 3D scanner to produce a digital model of a design mock-up for use in CAD software. Industrial designers have many physical tools and materials to work with when producing models, and the Augmented Foam Sculpting system replicates only one of these tools. The system is not yet a complete replacement for traditional workflows. However, comparing the quality of 3D models generated with the system to those generated by a commercial 3D scanner is important. To compare, a simple foam sculpture with many small, concave cuts was created. This model was then scanned with a Polhemus FastSCAN[1] handheld 3D laser scanner. The results of the sculpting and scan are shown in Figure 5.11.

Augmented Foam Sculpting has the advantage that faces are only added to the model where a cut is made. Depending on the sculpture, this can produce a virtual model with far fewer polygons compared to the output from the FastSCAN, which produces a high polygon model with uniform vertex density. In the example in Figure 5.11, the Augmented Foam Sculpting system generated a mesh containing 684 triangles, where as the laser scanner's output contained 81,925 triangles. Augmented Foam Sculpting also captured the sharp edges of the cuts more accurately than the soft edges from the scanner.

[1] http://www.polhemus.com/?page=Scanning_Fastscan

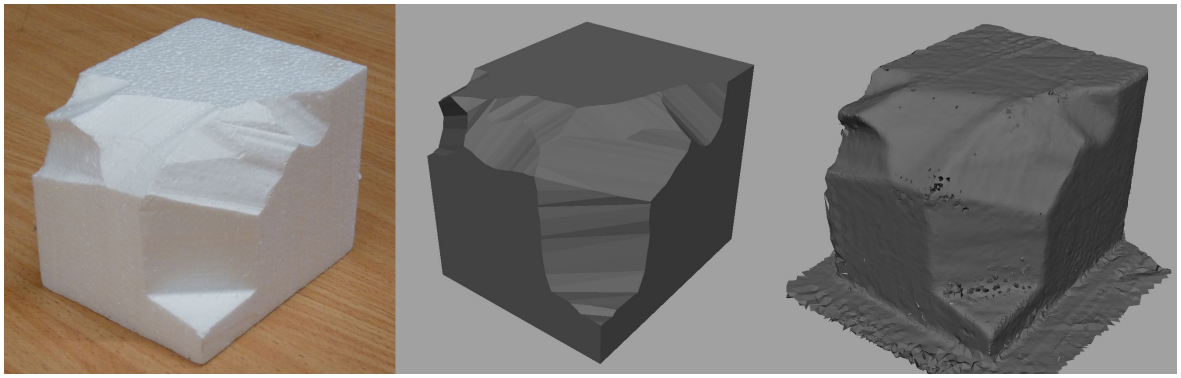


**Figure 5.11:** Comparison of physical foam model (left), Augmented Foam Sculpting (centre), laser scanned model (right)

The quality of the digital models is limited to the quality of the tracking system used. The tools and foam are tracked with six degrees of freedom, using a Polhemus Patriot magnetic tracking system. In practice, the magnetic tracker suffered from interference introduced by the metal casing of the foam cutting tool. In addition, the foam sculpting tool uses a battery for heating the cutting wire. Electric current flows through the body of the foam cutter, which causes additional interference. This reduced the quality of the generated 3D models, as they do not always match the physical version. The spatial resolution of the projection system has no effect on the model resolution. It does however affect the quality of the visualisations, and therefore the ability for a designer to precisely match a target model. At the time this work was developed, a magnetic tracker was all that was available. An optical tracking system would greatly improve the performance of the system.

It is worth noting that both the FastSCAN and the Augmented Foam Sculpting prototype implementation use a magnetic tracking system. Therefore, both systems introduce some inaccuracies due to tracker drift and interference. The current prototype also introduces a small amount of error during the calibration phase.

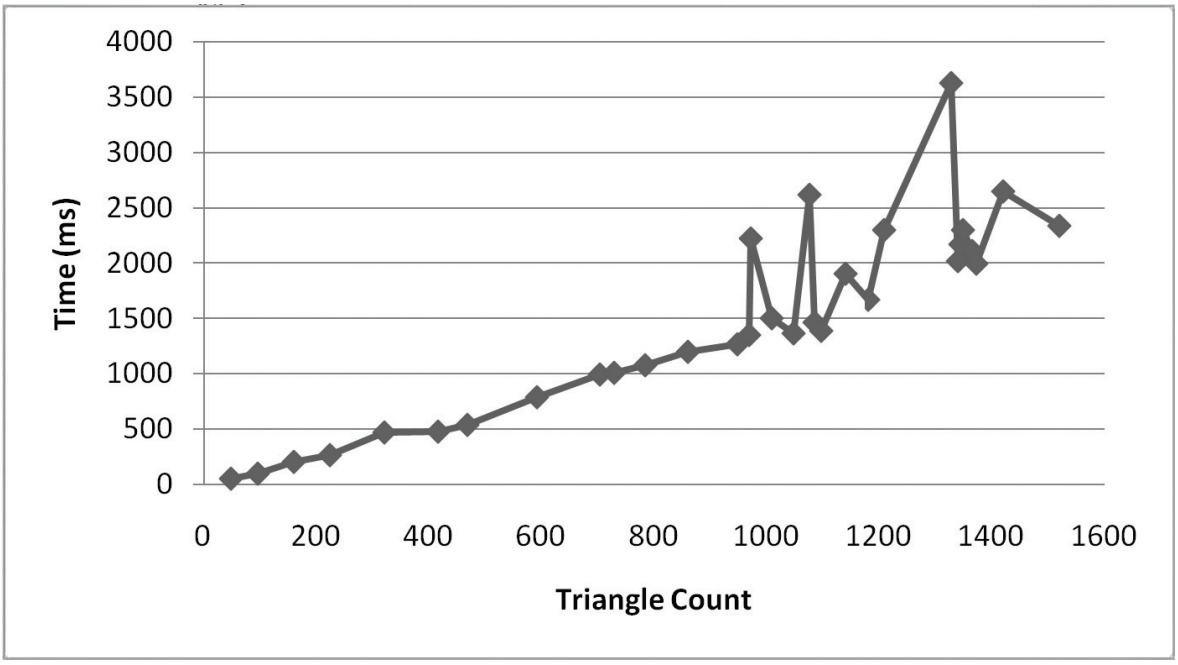



**Figure 5.12:** Calculation time for CSG operations as geometry size increases

### 5.5.2 System Performance

Augmented Foam Sculpting requires many CSG operations to be performed during the sculpting process. The complexity of the foam geometry generally increases as more cuts are made. The CSG operations must be completed quickly so the user is not left waiting for an operation to complete before continuing to sculpt. The system was tested using blocks of foam approximately 100x100x118mm in size, with random cuts made into the foam. With blocks of this size the mean cut size was 55 triangles, and mean CSG calculation time was 1466ms. However, this time varies greatly depending on the complexity of both the cut and foam geometry. The graph in Figure 5.12 illustrates the increase in time as the geometry complexity grows. The spikes in the graph can be attributed to the system resampling and reattempting to perform the CSG operation in the event of CGAL raising an error. In general, users only experience a short delay between cuts. The system frame rate is 60Hz. However, the prototype is a single threaded application, causing the video updates to pause while CSG operations are calculated. Augmented Foam Sculpting uses

CGAL, a popular computational geometry library. The CSG algorithm, as described by [LTH86], has aspects that can be executed in parallel, as most of the time is spent intersection testing each triangle with every other triangle. Overcoming the current performance issues would require writing a new multithreaded CSG algorithm. It may also be possible to implement the algorithm with CUDA[2], taking advantage of the massively parallel nature of current GPUs. These solutions, while desirable for a production system, are beyond the scope of this research prototype.

### 5.5.3 Expert Review

The system was demonstrated to a group of final year undergraduate and honors students from the Louis Laybourne Smith School of Architecture and Design at the University of South Australia. The system and visualisations were demonstrated, with students able to make cuts into the foam. This was followed by an open ended discussion to gain feedback on the system.

The student response to the system was quite positive. The most compelling aspects of the system from their point of view included the real time update of the virtual model during the carving process and the ability for the system to help in planning foam cutting. One student noted "the ease in which a CAD model would be produced, through manual labor and the freedom to possibly carve foam accurately without the need for templates", another noted the system "will help with any planning of foam cutting, making errors more unlikely".

While this chapter has focused on a workflow that moves from foam sculpting into CAD, students commented that they also work in the other direction. Sometimes, they use foam sculpting as an inexpensive alternative to 3D printing. In this scenario a lot of their time is spent in planning and producing stencils to cut against to produce a physical version of the CAD model. The system supports this work-

[2]http://www.nvidia.com/object/cuda_home.html

flow through the Cut Animation and Target visualisations. The addition of an algorithm to automatically generate a list of cuts needed to produce a specific 3D model, rather than playing back a previously recorded session would allow the system to completely replace the current process of producing stencils manually.

Several students stated they would like to see new tools integrated into the system. In particular, students would like to see a rasp or the ability to sand the model, as these tools would allow much finer modifications to be made than what is possible with the cutting tool. Another tool suggested was a soldering iron, which would provide the ability to put holes and other patterns onto the surface of the foam. One student suggested a more realistic physical simulation, taking into account the properties of the foam. For example, if the cutting wire is held at a constant position during a cut, the area of foam around the wire would begin to melt away. The students also stated they would like to have the ability to have high resolution textures projected onto the models. The quality of the images projected onto objects is limited by the resolution of the projector, and the distance between the object and projector. This limitation can be reduced by using more projectors closer to the object, which would provide a higher spatial resolution.

## 5.6 Summary

This chapter has presented a new PVT technique, Augmented Foam Sculpting for generating 3D models using traditional tools for foam sculpting, already in use by industrial designers. By using the tools that designers currently use, designers do not have to change their workflow to begin using the system. By using foam blocks and a physical cutter, users benefit from the passive haptic feedback the objects provide, and the users can create 3D models with an intuitive bimanual interaction technique. The foam cutting tool is an example of a highly specialised tool, used

for a single task: cutting through a piece of foam while simultaneously simulating the cut operation on a virtual model. No customisable attributes are available to the user. Therefore, this tool would receive a rank of (1, 0). Using SAR in this way allows extra information to be projected onto the objects as they are sculpted. Augmented Foam Sculpting takes advantage of this ability to visualise how to produce target models from foam, either by changing the colour of the surface to represent areas that should be cut away, or by 'playing back' cuts from a previous session. The projector is also used to display the wireframe of the 3D model, or 3D volumetric textures such as wood grain onto the foam, as it is being sculpted. Although industrial design was the focus when developing the system, augmented foam cutting could be used in other areas. The system would support any domain requiring physical objects with matching 3D models, or where 3D models are created from hand sculpted designs, for example the computer animation and visual effects industry.

# 6

# SAR User Interface Techniques for Room Size Modelling Tasks

This chapter presents a set of SAR tools to aid in interior architecture work such as kitchen design. An example SAR design application, *BuildMyKitchen*[1] has been developed in consultation with architect Steve Kelly from the University of South Australia's School of Architecture and Design. In developing BuildMyKitchen, new user interface tools and techniques have been created. BuildMyKitchen's user interface is comprised of physical-virtual tools that support the designer in both the design process and designer/client meetings. The tools that comprise the user interface of BuildMyKitchen are shown in Figure 6.3. The tasks supported by the tools are: *Cabinet Layout*, *Design Presets*, and *Modifying Finishes*. While these techniques have been developed in the context of kitchen design, they can easily be generalised for use in other application domains and address important interaction issues for SAR systems, such as interaction techniques for room size environments and the lack of virtual projection surfaces.

[1] An earlier version of this work [MT13] used the name "PimpMyKitchen", a name inspired by the television programme "Pimp My Ride", which involves customising and improving cars.

This chapter makes the following contributions to SAR research:

- Cabinet Layout. Physical-virtual tools that allow designers to specify dimensions and distances in a room size SAR environment. These techniques have been optimised for a range of physical movements from floor level to overhead structures, as most Virtual Environment (VE) interaction techniques are designed for the user to be in a single posture, standing or sitting.

- Design Presets. PVT user interface techniques based on a "colour swatch" metaphor for saving and loading preset designs in a SAR environment. These have been developed to allow both the designer and client to quickly access previous developed designs without the need for external technology.

- Modifying finishes using a "magic touch" metaphor based user interface technique for modifying attributes of the final design.

- BuildMyKitchen. This application represents an example exploration of how SAR can be used as part of the design process for interior architecture projects, and serves as a concept demonstrator encapsulating the ideas of the new PVT's.

This chapter is structured as follows. Firstly, the motivation for exploring the use of SAR for interior architecture is described. The remainder of the chapter describes the functionality of BuildMyKitchen, and shows how the system improves the design process for both the architect and client. This application provides the motivation for the specialised PVT's presented in this paper. The chapter describes the new user interface tools and techniques developed to support room size modelling tasks, and how they can be generalised for use in other application areas.

## 6.1 Motivation

The motivation for investigating the use of SAR in interior architecture stems from the difficulty and complexity of this design process. SAR allows full scale white replicas (walls, cabinets, and large appliances) to provide a flexible and understandable environment to convey large scale designs. In the case of a kitchen design process, clients currently go through several steps with the designer. Firstly, the client will meet with the architect to discuss their requirements. Different lifestyles are suited to different kitchen designs, and the architect takes these requirements into consideration during the design process. Following this discussion, measurements of the available space are taken, and an initial design is created. This design is presented to the client through 2D plans and 3D rendered images on a computer. Based on these images, the client and architect discuss changes to the design. An additional task for the client is to decide on material finishes, appliances, door handles, etc. To aid in this process, kitchen builders have large showrooms with several finished kitchens. Even still, the client usually will not be able to truly visualise their new kitchen until the final installation in their home.

A key motivating goal of BuildMyKitchen is to enhance the understanding and decision making ability of the client. While architects are accustomed to visualising the end result from plans and CAD drawings, the client does not usually possess these skills. Even with high quality 3D renders, it can be difficult for the client to envision the scale of the final product. BuildMyKitchen makes it possible to preview designs at 1:1 scale early in the design process. Furthermore, using physical blanks for the kitchen cabinets provides a more intuitive environment than what is possible with a standard computer. This helps not only in deciding on the visual design of the kitchen, but also the functionality of the layout. For example, Figure 6.1 shows a design flaw in a completed kitchen. Here, the dishwasher door is obstructed by the door to the pantry. To access the dishwasher, the pantry door must

**Figure 6.1:** A design flaw in a completed kitchen, where the dishwasher door and pantry door collide. Damage can bee seen on the pantry door.

be completely closed. Even a slight opening is enough to knock the dishwasher, resulting in damage to both the appliance and the door. These kinds of problems are difficult to predict when looking at a 3D render. Previewing the kitchen at 1:1 scale, with physical mock-ups can help in detecting these kinds of problems. Using SAR allows the projected content of the physical mock-ups to be changed quickly, making iterating the design easier. This ability becomes even more important given the rise in popularity of self-install flat-pack kitchens, where there is no trained architect and limited opportunities to preview the design before construction commences.

Another motivating goal is to provide designers and architects with tools to manipulate and visualise designs on a 1:1 scale. While these professionals are quite capable of visualising designs from drawings in full scale, SAR provides a new

medium to explore designs. Discussions with professional architects [Tho+11] indicate SAR provides an extra dimension to the design process. To facilitate the manipulation of designs in-situ of the SAR environment and not force the designer to make all their changes in a desktop CAD system, new tools and techniques are required beyond the current state of the art for SAR environments.

## 6.2 BuildMyKitchen

As previously mentioned, BuildMyKitchen was developed to support the early design process of kitchens and designer/client meetings. To support an architect when developing the initial kitchen design, *blanks* (white simple shaped light weight 3D projection substrates) representing the cabinets are placed in an environment of the shape and size of the target kitchen. One blank is shown in Figure 6.2. The architect decides on the basic layout of the kitchen by moving the cabinet blanks into the desired locations. The prototype workflow involves the following tasks. Architects first block out key positioning decisions, such as doors and major appliances, to optimise the efficiency of the kitchen, and then work on the next level of detail. Once the blanks are in position, the architect can focus on the layout of the cupboards, drawers, and other components. Some of these components will need to be resizable, depending on the vision of the architect; some, such as appliances, have set dimensions.

The second intended use case for BuildMyKitchen is during meetings where both the architect and client are present. The goal of these meetings is to come to an agreement on the layout of the kitchen, and for decisions regarding material finishes and component selection to be made. There are an overwhelming number of possible combinations of appliances, materials, handles, etc. Rather than have the client select from such a large number of choices, BuildMyKitchen allows the

**Figure 6.2:** An example Cabinet Blank.

client to select from and customise a set of "preset" designs. These presets are created beforehand by the architect using their experience and design sense. The client chooses a preset on which to base their design and is then able to customise it using a smaller set of options. This set of options is also put together before hand by the architect.

## 6.3 Cabinet Layout

A key task for the architect is to decide on the layout of components placed into the cabinets, such as cupboards, drawers, and appliances. As previously mentioned, some of these components, such as appliances, have fixed dimensions that need to be accommodated. Others will be set by the architect based on their design expertise. Some components do not require a specific size, and the component can simply make use of whatever space is available. For example, evenly sizing the height of a set of drawers to make use of the entire height of the cabinet.

**Figure 6.3:** The tools that comprise the BuildMyKitchen user interface.

In addition to components in the cabinets, other design features also need to have dimensions set by the architect. These include:

- Bench thickness. Different benchtop materials are manufactured to different thicknesses. The architect may also choose a specific thickness for the benchtop.
- Kickboard height. The kickboard is the inset material at the base of the cabinet. Choosing the height of the kickboard requires a tradeoff between a functional kickboard and maximum cupboard space.
- Kickboard inset. The inset is the amount the kickboard is set in from the face of the cabinet. Commercial kitchens often have deep insets so workers do not kick cabinets with their feet.
- Bench overhang. The overhang is how much the benchtop overhangs the rest of the cabinet.

- Component spacing. The component spacing is the horizontal and vertical space between components, such as adjacent cupboard doors. Older kitchen designs often have large spaces between components, resulting in smaller doors, where as modern designs tend to maximise the size of the doors, making it easier to see into the cupboards when the door is open.

To accommodate these requirements, BuildMyKitchen automatically places and sizes components according to their properties. This is done using the following basic component types:

- Containers. A container contains zero or more other components, and is responsible for arranging them either vertically or horizontally according to the space available.
- Doors and drawers. A door is placed into the cabinet with the handle either to the left or right. Drawers have the handle placed horizontally, centred at the top of the component.
- Appliances, such as dishwashers and ovens.
- Blank spaces, used for filling an area in the layout.

Drawers, doors, and spaces are able to have preferred dimension set. When layout out a cabinet, containers will try to give components their preferred size. Appliances have dimensions that cannot be altered by the architect, so they are given priority over other components. As containers can be nested, any layout can be produced using the right combination of containers and components. Producing the final layout is a recursive process, with each container allocating space to its children. The algorithm shown in Listing 6.1 produces a horizontal layout. Vertical layouts are produced in the same way, adjusting component heights instead of widths.

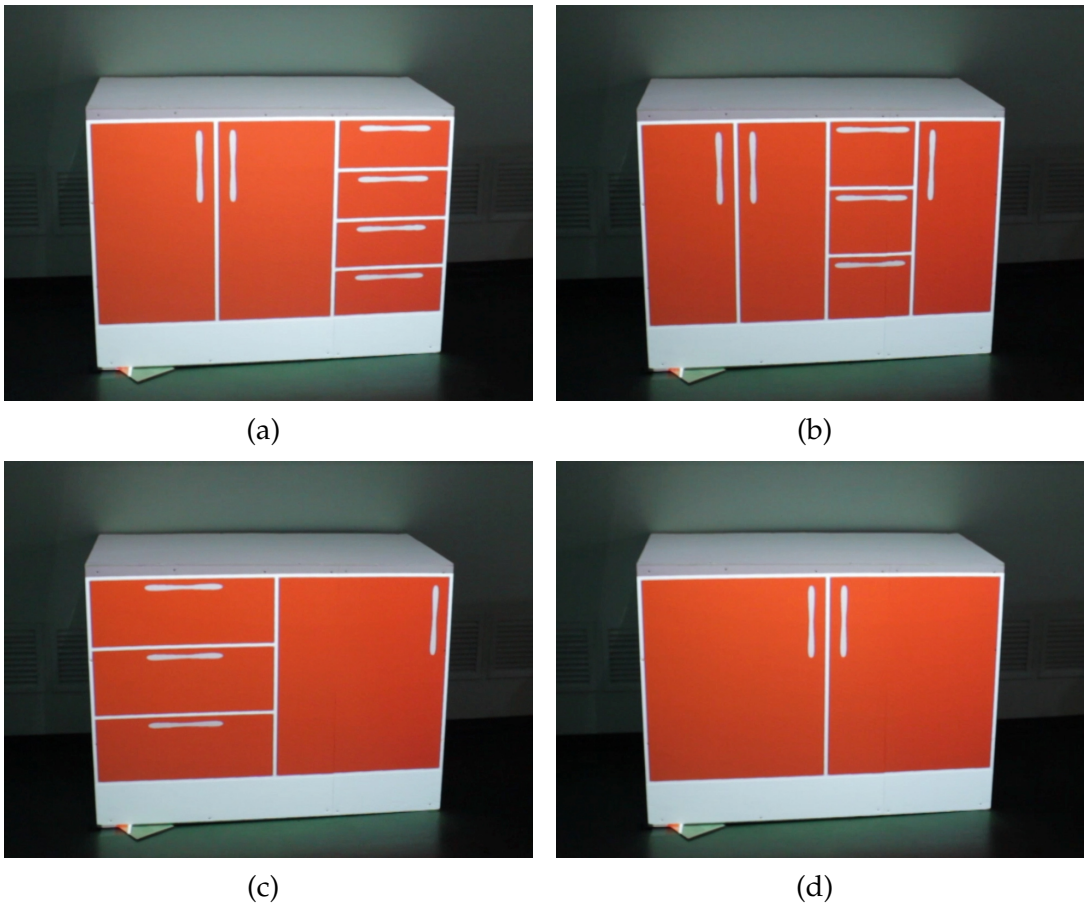
```
void Container::resize(float w, float h, float padding, Vector topLeft) {
    int elasticCount = 0;
    float availableSize = w;
    foreach (PanelComponent* component, mContents) {
        if (component->hasPreferredWidth())
            availableSize -= component->getPreferredWidth();
        else
            elasticCount++;
    }
    availableSize -= (mContents.size()-1) * padding;
    wcl::Vector currentTopLeft = topLeft;
    foreach (PanelComponent* component, mContents) {
        if (component->hasPreferredWidth()) {
            if (elasticCount > 0) {
                component->resize(component->getPreferredWidth(), h,
                    padding, topLeft);
                topLeft.x += component->getPreferredWidth() + padding;
            }
            else {
                component->resize(w / mContents.size(), h, padding,
                    topLeft);
                topLeft.x += w / mContents.size() + padding;
            }
        }
        else {
            component->resize(availableSize / elasticCount, h, padding,
                topLeft);
            topLeft.x += availableSize / elasticCount + padding;
        }
    }
}
```

**Listing 6.1:** Algorithm for producing a horizontal cabinet layout

### 6.3.1 Creating a Cabinet Layout

The architect starts with a cabinet blank with no components added to it. They can then add components to the layout using a simple keyboard interface. As components are added, the system automatically arranges the components to fit the available space. Figure 6.4 shows four different layouts on the same cabinet. Once the necessary components have been added, the architect can move on to specifying dimensions to the components, as shown in Figure 6.5. For example, drawers used to store saucepans will need to be wider and deeper than drawers used to store cutlery. At a higher level, the architect can experiment with the overall kitchen layout

by moving the cabinet blanks into different positions in the room.



(a) (b)

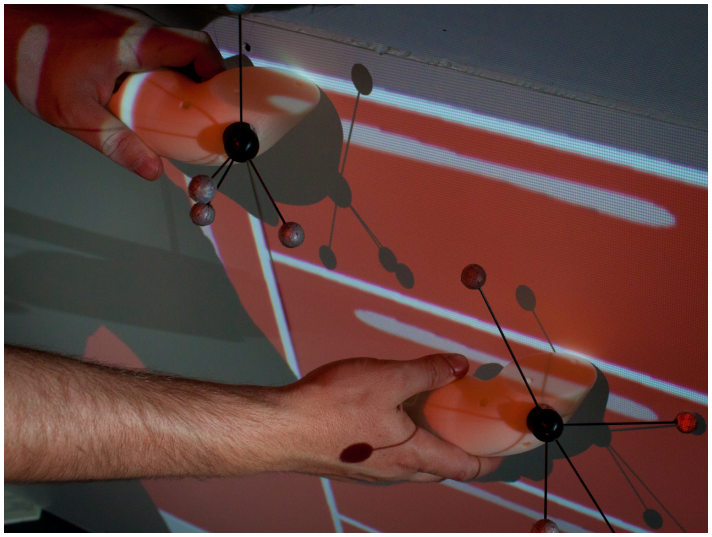
(c) (d)

**Figure 6.4:** Example cabinet layouts.

### 6.3.2 Resizing Using The Two-Handed Resizing Tool

The two-handed manipulation tool has been developed for performing operations such as resizing components of the kitchen. The tools consists of two identical halves which are held in each hand. To resize a component, the user places each half of the tool against opposite edges of the component to be resized, and adjusts the distance between their hands until the desired size is obtained. This is an intuitive gesture, and is a similar motion to how people estimate and communicate distances in the real world.

This tool is used exclusively for resizing. Therefore, it is placed at (1) on the X axis



(a)

(b)

**Figure 6.5:** (a) A user resizes the height of a drawer, and (b) the thickness of the bench top.

of the WS-TVC. Resizing in BuildMyKitchen has a single user definable attribute: the snapping mode. The user is able to choose between snapping to 1mm, 5mm, and 10mm intervals when resizing. In addition, when a set of common dimensions are used in kitchen design, the user can choose to snap to the nearest preset dimension. The snapping mode is conveyed to the user by projecting this information onto both halves of the tool. The current dimension is also projected onto the tool, allowing

the architect to precisely set dimensions with little effort. With these two items in mind, the tool is placed at (2) on the Y axis of the WS-TVC.

Resizing components on the surface of the blank is accomplished by placing each half of the tool on opposite edges of the component to be resized. The architect resizes the component simply by changing the distance between the two halves. Resizing the kickboard inset and the benchtop overhang is a more difficult problem, because the physical cabinet blank does not match the virtual geometry. Resizing these components is accomplished by placing one half of the tool on the component to be resized, such as the front of the bench overhang, and the other on the top of the cabinet. A line is projected onto the cabinet blank indicating the amount of overhang. The architect changes this dimension by dragging the tool along the top of the cabinet.

#### 6.3.2.1 Resizing Wand

In addition to the resizing tool, a *Resizing Wand*, as shown in Figure 6.6, is also provided. This was added after experimenting with resizing kickboard components. As the kickboard is near the floor, using the resizing tool requires kneeling down, which quickly becomes uncomfortable. The resizing wand can be used instead of one half of the resizing tool, allowing resizing kickboard properties from a more comfortable stance. This type of tool could be extended to specifying a range of dimensions outside the user's reach, such as cabinets over benchtops and lighting fixtures on the ceiling, but these techniques require further evaluation against current out of arm reach VE interaction techniques.

**Figure 6.6:** Resizing the Kickboard Inset using the Resizing Wand and one half of the Resizing Tool.

## 6.4 Design Presets

While the cabinet layout process would be conducted exclusively by the architect, previewing and modifying kitchen designs is a process that involves the client. Rather than giving the client a 'blank canvas', they would instead be working with designs created earlier by the designer. These *Design Presets* can be a starting point from which the client can decide on their own design.

### 6.4.1 SAR Swatches

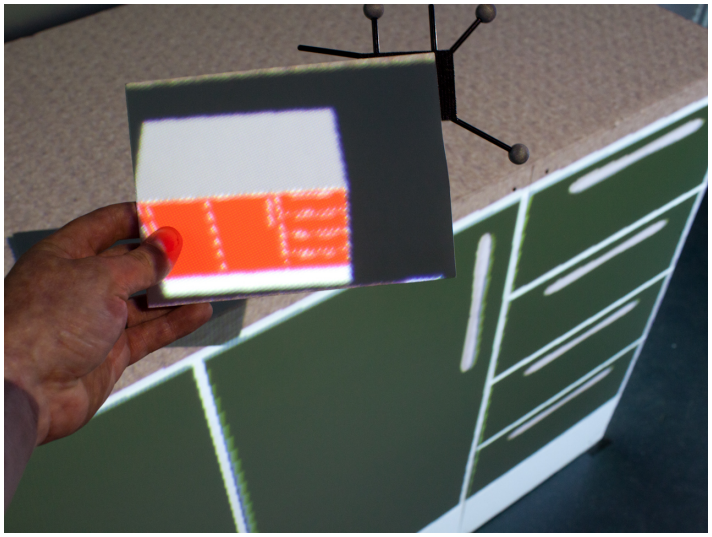
To facilitate loading and saving kitchen designs, BuildMyKitchen makes use of *SAR Swatches*. SAR Swatches are based on a metaphor adapted from the colour swatches

used to choose paint colours. A SAR Swatch is a small board, and a 3D render of a kitchen design is projected onto the board. Clients can quickly evaluate different designs by holding several in their hands at once. SAR Swatches are also related to Tangible Bits [Ish08], in that the digital information is represented as a physical object.

The client can preview one of the designs simply by placing a swatch onto the cabinet. Doing this causes the kitchen design shown on the swatch to be loaded onto the cabinet blanks, and the client can experience the design at 1:1 scale on the physical surfaces. The other side of the swatch is used for saving designs. A new design can be saved to a swatch by placing it upside-down on one of the cabinets. The projected visualisation swatch will change, indicating the system is asking for confirmation (Figure 6.7(b)). To confirm the save operation, the user simply slides the swatch to the right. This motion will save the current kitchen design to the swatch, with the visualisation projected onto the other side of the swatch updating accordingly. The number of swatches is only limited by the number of physical swatches constructed for use; theoretically we could have thousand of swatches. They are uniquely identified by the tracking system.

SAR Swatches provide a good example of the tradeoff between several physical tools and a single tool. Rather than individual swatches, designs could instead be previewed one at a time on a PIP style device. The PIP could then be placed in the load area and the kitchen selected would be loaded. Alternatively, a tool for previewing could be removed entirely, and instead the user would only view designs on the full sized mock-up.

SAR Swatches were chosen because they have several advantages compared to the alternatives described above. Firstly, swatches allow the user to compare different designs simultaneously. This can improve their ability to choose a design. For example, when comparing two designs, the client may decide they prefer one de-



(a)


(b)

**Figure 6.7:** (a) Comparing one design with another using a SAR Swatch, (b) saving a new design to a SAR Swatch by placing the swatch on the cabinet, and sliding right.

sign overall, but with the door handles from another design. This realisation would be difficult to come to if only one design could be seen at a time. Using several physical tools also makes it easier to quickly flick through several different designs in a more natural way than using a menu. Finally, SAR Swatches adapt a device already in use in interior design: the paint colour swatch. Kitchen swatches extend this to storing several aspects of the design on a single swatch.

SAR Swatches are interesting to consider because these tools are the first presented in this dissertation where the primary role is to hold information, not manipulate elements of the design. However, previewing a design on the swatch can be considered as one task, loading a design on the prototype as another, and saving the current design to the swatch as a third task. Each swatch has a single parameter: the kitchen design it is storing. Therefore, each SAR Swatch can be placed at (3, 1) on the WS-TVC, indicating that each tool is used for a single, simple task.

#### 6.4.1.1 Generalising Swatches

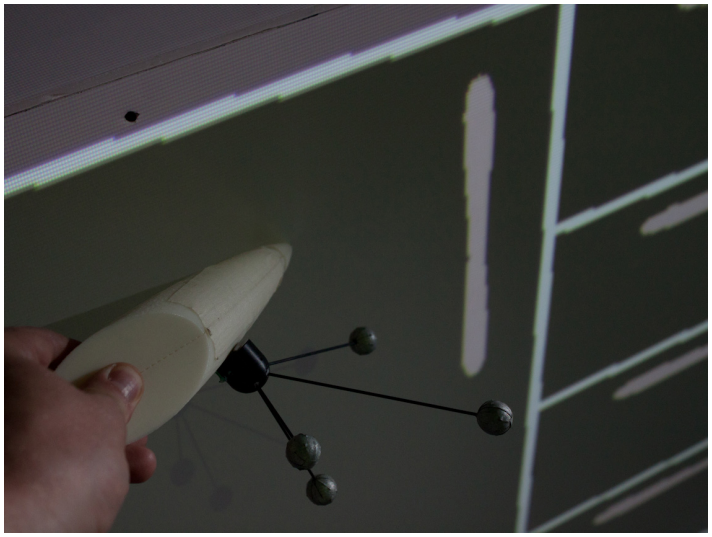
Swatches solve the problem of loading and saving data in a SAR environment in a way that is specifically suited to the nature of SAR. Text input is difficult in large scale SAR environments because there is no logical location for a keyboard. However, for the kinds of industrial design applications discussed in this dissertation, the data to be saved is often visual in nature. Therefore, there is no need for the user to enter filenames. Instead, they can access the data in a visual manner. This is accomplished through the use of PVT. The physical SAR Swatches are treated like any other object in the SAR environment, and have virtual information projected onto them. The user is able to refer to and access data in a visual manner, and does not have to concern themselves with text entry. This simplifies the user interface by removing the keyboard, which may not have a logical location in a SAR environment.

## 6.5 Modifying Finishes

The final use for BuildMyKitchen is to allow the client and architect to modify one of the presets in order to arrive at the final design. The client is able to select from several options to customise the following aspects of the design: materials (for surfaces such as benchtops, doors, kickboard, and splashback) and fixture designs (such as

handles, power points, and taps).

There are of course infinite possible combinations of features. However, at this stage of the design, the number of choices is deliberately limited by the architect. Just like the kitchen presets, the designer makes available a set of options that will work together, based on their design experience. The client is able to customise their kitchen based on a smaller number of choices, rather than allowing every possible combination.



(a)

(b)

**Figure 6.8:** (a) A user changes the colour of the cabinet doors using the stylus, (b) a finished kitchen design.

A *Magic Touch* metaphor is used for the modifying finishes tool, as this greatly simplifies the user interface. To select different options, the user taps the surface of the cabinet blank with the *Stylus*. The system cycles through the choices one at a time. The process is exactly the same for door handles and such. For these small items a selectable region is defined around the projected texture. In the case of a finish being replicated in more than one location as with a handle, the system changes all occurrences to match the selected style. For example there might be different sizes of handles for different sized doors; therefore, the change would modify the style of all the handles but with the correct sized handles for each occurrence. This mode of selection is acceptable because the number of choices is small. In the demonstration system between five and ten options are loaded for each of the changeable components. In addition, this task is fundamentally exploratory in nature. The client is interested in experimenting with different designs, and does not need to load specific finishes at this stage. However, if there were many choices, an alternate system, such as a pie menu displayed with a PIP could be employed.

Reusing the Stylus for BuildMyKitchen demonstrates how the complexity of a physical-virtual tool depends greatly on the application where it is used. Here, the tool is used for a single task: changing a property of the kitchen. This task has no attributes. While the single tool is used to change all the properties of the kitchen design, the active property is not an attribute of the tool. The property being changed simply depends on what the user touches with the Stylus. Therefore, the Stylus is placed at (1, 0) on the WS-TVC for the BuildMyKitchen application.

## 6.6 Summary

BuildMyKitchen is an example of many specialised or single purpose tools. The demonstration system developed provides four kitchen swatches, a Stylus, and a

two handed resizing tool. Additional swatches could easily be added, but this would quickly become unmanageable, since the user would not be able to hold all of them at the same time. Therefore, BuildMyKitchen would be placed at position 6 on the X axis of the IS-TVC, indicating the system uses many tools. The mean number of tasks for each tool is 2.33, indicating that the tools are not heavily virtualised. The final position for BuildMyKitchen on the IS-TVC is (6, 2.33), showing a user interface that is comprised of many simple tools.

This chapter has described an investigation into using SAR for kitchen design and interior architecture. New user interface techniques for SAR are presented. A demonstration SAR design application, BuildMyKitchen, embodies these techniques. This chapter has discussed PVT-based tools and techniques developed for the application, and how they can be generalised for use in other application domains.

# 7

# Software Support for Spatial Augmented Reality

This chapter describes several aspects of the software and support infrastructure that have been developed through the course of this research. A new software framework, *LibSAR* has been developed specifically for SAR research in collaboration with fellow researchers Markus Broecker and Benjamin Close. This work also forms part of a provisional patent [Mar+12a]. This chapter describes my contributions to this framework. In addition to LibSAR, this chapter describes approaches for tracking the user's fingers using an active red/green/blue (RGB) Light Emitting Diode (LED), and how this hardware can be used to provide visual feedback to the user when interacting with virtual projected controls. I developed the software aspects of this system, with fellow researcher Ross Smith developing the hardware. Therefore, the details of the marker hardware are omitted from this chapter.

## 7.1 A Reusable, Modular Software Framework for SAR

Throughout the last four years of SAR research in the Wearable Computer Lab at the University of South Australia, a new software framework has been written specifically for SAR research applications. Initial investigations used pre-existing software libraries such as OpenSceneGraph[1]. OpenScenegraph provides a great deal of built-in functionality, including support for a variety of different image and geometry formats and a high performance scene-graph data structure. However, it became apparent that this approach was not suitable for a SAR research environment for the following reasons:

- Libraries such as OpenSceneGraph place constraints on software and algorithm design, such as abstracting the graphics library and requiring the use of a scene-graph structure. This limits the ability to quickly prototype proof of concept applications, as the researcher has to make the new algorithm "fit" into the scene-graph.

- The SAR research applications under development typically only required a small number of virtual objects to be drawn. This reduced the need for complex scene-graph architectures, which are best suited to dealing with large numbers of objects.

- Although libraries such as OpenSceneGraph provide comprehensive libraries of functionality to the researcher, there is still a significant amount of standard code that needs to be written for each SAR application. This includes configuring projector outputs, building the scene-graph structure, and dealing with hardware such as cameras and other peripherals.

A new software framework, *LibSAR* was developed to overcome these limitations. LibSAR has been written in C++ and runs on Linux operating systems. The

[1] http://www.openscene-graph.org

systems presented in this dissertation have all been developed using different versions of LibSAR. In addition to addressing the problems listed above, the library was designed to meet the following requirements:

1. Support multi-projector SAR environments.
2. Abstract the hardware setup so that SAR applications can be developed without having to account for certain hardware configurations. For example, while the system supports multiple projectors, this should be transparent to the SAR application. Researchers should be able to write SAR applications that work with any projectors, and any particular hardware configuration.
3. Provide direct access to the GPU and graphics library if needed.
4. Allow larger applications to be built up from smaller components.
5. Provide support for cameras and other hardware.
6. Provide class abstractions for common resources for SAR applications, including images, 3D geometry, OpenGL textures, Framebuffer Objects, videos, etc.

### 7.1.1 Runtime Architecture

To meet these requirements, LibSAR separates application logic from common tasks that need to be performed for any SAR application. The application flow of a simple SAR system is shown in Figure 7.1. This kind of application begins with initialisation steps, which load resources, configure the hardware and graphics system, and set the projection and view matrices for a calibrated projector. At this point, the application enters the "main loop", which involves repeatedly updating the program state and rendering to the projector.

LibSAR substantially changes this application model, as shown in Figure 7.2. Application logic is separated out into *Modules*, which will be described in detail in

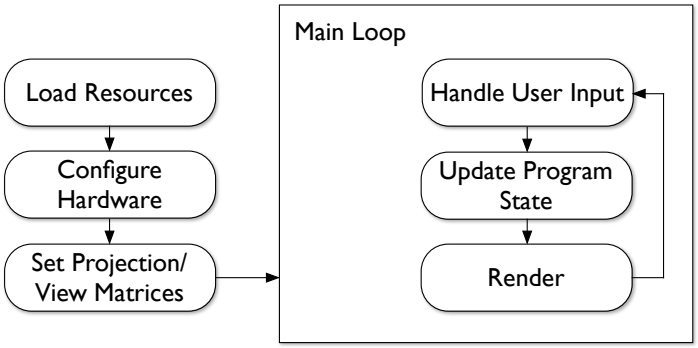



**Figure 7.1:** A typical system flow for a graphical application

Section 7.1.2. This separation makes running several modules at the same time on a single system possible, allowing complex systems to be built up of several simple modules. The initialisation part of the LibSAR's flow is similar to the basic application described above, with the addition of a step to initialise each module. The main loop of LibSAR allows each module to process all user input and to update their internal state. The render step is executed for each projector and module. The system configures the projector's calibration parameters before each module renders. This allows modules to be written without being concerned with particular projector configurations, and allows a module to work with any number of projectors supported by the computer hardware[2]. For example, Figure 7.3 shows the LibSAR system running, configured as a six projector video wall. The module displaying the content is simply drawing a textured rectangle for each projector. The calibration matrices ensure each projector renders the correct region on the rectangle. This idea is further illustrated in Section 7.1.5.

### 7.1.2 Application Modules

The LibSAR runtime is responsible for handling tasks that are common to all SAR applications, such as configuring projectors and other hardware. Actual application

[2]The actual number of projectors that can be used is limited by both the graphics hardware and the X Server configuration. Due to limitations with available hardware, the software has only been tested with up to eight projectors.

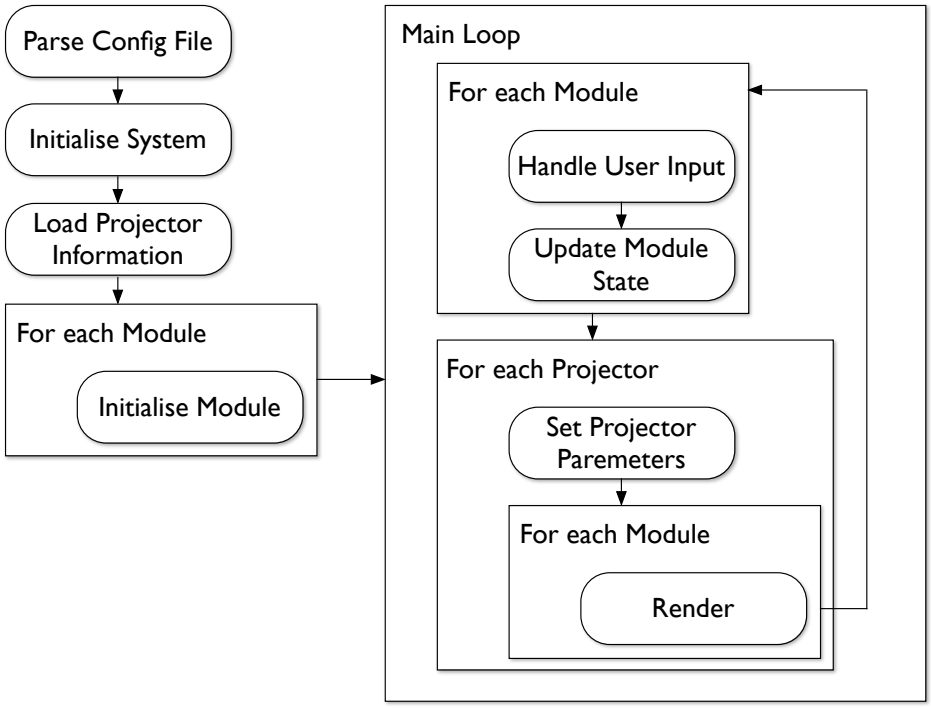



**Figure 7.2:** The system flow of LibSAR, allowing for multiple applications running simultaneously with multiple projectors.

logic is contained inside Modules. When executing, a module has full control of the system, including the OpenGL state. As LibSAR does not know about the possible functionality of a module (a module represents a current research problem), as few restrictions are placed on modules as possible. The task of writing SAR applications is simplified, as the researcher does not have to concern themselves with complex scene-graph structures and libraries.

In order to provide this functionality, LibSAR requires that modules implement the following interface:

```
class Module
{
    public:
        virtual void init() = 0;
        virtual void update(unsigned int timestamp) = 0;
        virtual void draw(const Projector* p)=0;
        virtual void handleInput(const SDL_Event& event) = 0;
};
```

**Listing 7.1:** LibSAR's Module interface

The *init* function is called once during system startup, after the graphics system has been configured. This function allows modules to initialise any resources required, such as textures. The *update* and *handleInput* functions are called once each pass through the LibSAR's main loop. The handleInput function provides access to user input from the mouse and keyboard. The update function allows modules to update their internal state. The *draw* function is called once, per projector, each pass through the main loop. Any rendering a module requires is placed into this function. Additions to this initial Module interface by Markus Broecker allow for inter-module communication.

### 7.1.3 Post Processes

In addition to application modules, LibSAR provides *Post Processes*. Post Processes manipulate a projector's back-buffer before it is projected, which makes a variety of effects possible, including projector blending, colour correction, etc. Like Modules, LibSAR provides a framework for post processes to be loaded and configured at runtime. To accomplish this, Post Processes implement the following interface:

```
class PostProcess {
    public:
        PostProcess(Projector& p, OptionList options);
        virtual void init() = 0;
        virtual void process() = 0;
        virtual ~PostProcess();
};
```

**Listing 7.2:** LibSAR's PostProcess interface

A Post Process object processes a single projector's frame buffer. Multiple objects are created if the same process needs to be applied to several projectors. This approach simplifies the interface substantially, as a Post Process object only needs to be concerned with a single projector. Several objects can be configured with different parameters, such as different colour profiles for each projector. The system

configures the projector's viewport parameters before the *process* function is called, allowing Post Processes to focus on their algorithm, without having to handle projectors.

Post Process objects are loaded during system initialisation. Each projector maintains a list of Post Process objects that are associated with it. During the main loop, the *process* function for each object is called sequentially. This allows Post Processes to be "stacked" to combine effects.

#### 7.1.3.1 Post Process Example - Projector Blending

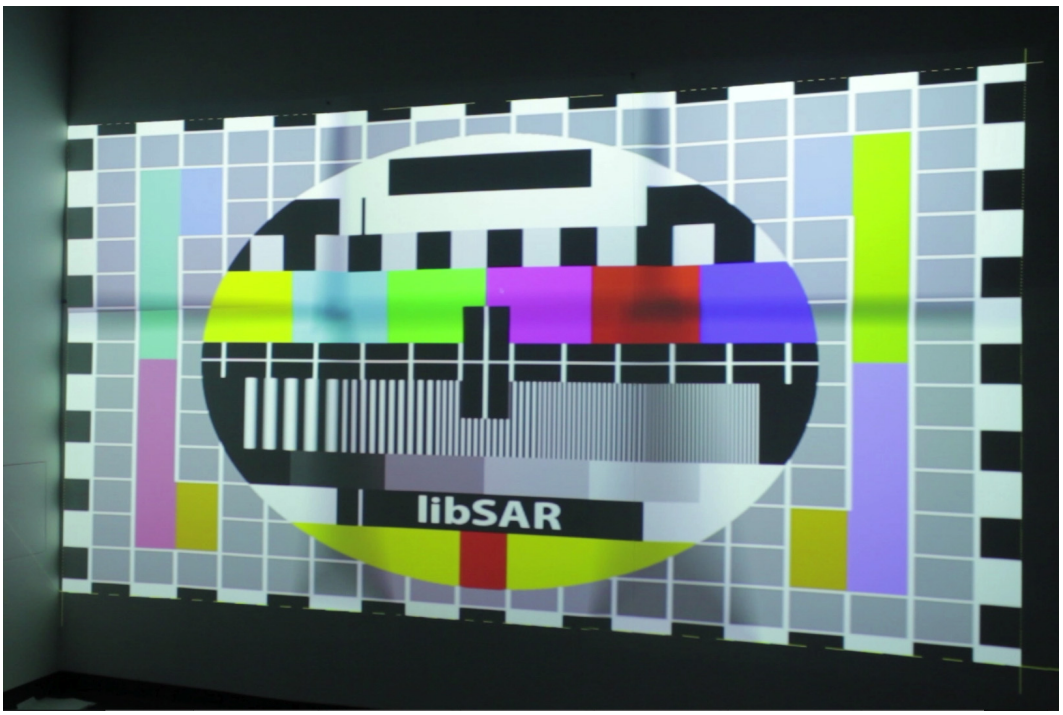



**Figure 7.3:** LibSAR configured as a six projector video wall. The StaticMask post process is used to blend the edges of each projector's image.

This section shows how the Post Process framework can be used to implement static projector blending. Projector blending algorithms, such as those presented by Stuerzlinger, and Bimber and Raskar [Stu99; BR05] calculate alpha masks for each projector. Alpha masks such as these only need to be generated once, and can be used to blend projected images, regardless of what is displayed. A *StaticMask*

class can be written to post process a projector's image with an alpha mask after all modules have rendered to the projector. The class is defined below:

```
class StaticMask : public PostProcess {
    public:
        void StaticMask::init() {
            string filename = mOptions.find("Mask")->second;
            AbstractImage* image = resourceManager.getImage(filename);
            mTexture = new graphics::Texture(image);
        }

        void StaticMask::process() {
            graphics::ScreenQuad sq;
            glEnable(GL_BLEND);
            glBlendEquationSeparate(GL_FUNC_ADD, GL_FUNC_ADD);
            glBlendFuncSeparate(GL_ZERO, GL_SRC_COLOR, GL_ONE, GL_ONE);
            mTexture->bind();
            sq.draw();
            mTexture->disable();
            glDisable(GL_BLEND);
        }

        StaticMask::~StaticMask() {
            delete mTexture;
        }
};
```

**Listing 7.3:** The StaticMask class

The *init* function looks for the name of the file to load from the list of configuration options for the post process. It then loads this file to an OpenGL texture. The *process* function uses OpenGL's blending functionality to render the alpha mask to the projector's viewport. Blending can be accomplished using a StaticMask object for each projector, independently of what has been rendered by modules. Figure 7.3 shows the result of applying the StaticMask Post Process to configure a six projector video wall.

### 7.1.4 Cameras

Cameras are important for many SAR applications, allowing information about the world to be captured. SAR applications need to be able to access images from cam-

eras, without having to be aware of any particular hardware configuration. Camera support is integrated into LibSAR in two layers. The first provides hardware abstraction for a variety of camera types, and the second provides higher level access of cameras to Modules. This section describes LibSAR's camera support.

#### 7.1.4.1 Hardware Abstraction

LibSAR implements a camera abstraction to provide a consistent interface for several types of cameras, including USB, Firewire, and Point Grey GigE cameras[3]. The portion of the Camera interface is shown in Listing 7.4. Subclasses of Camera implement this interface for specific camera types.

```
class Camera {
    public:
        virtual void startup() = 0;
        virtual void shutdown() = 0;
        virtual void update() = 0;

        virtual CameraID getID();

        virtual void setConfiguration(const Configuration &c) = 0;
        Configuration getActiveConfiguration();
        std::vector<Configuration> getSupportedConfigurations();
        Configuration findConfiguration(Configuration partialConfig);

        virtual void getCurrentFrame(unsigned char* buffer, const
           ImageFormat& format) const;

        virtual bool hasParameters() const;
        virtual CameraParameters getParameters() const;
        virtual void setParameters(const CameraParameters& p);
};
```

**Listing 7.4:** The Camera interface

This class provides access to images from a camera in a variety of formats. Different cameras use different image formats natively, such as RGB, or YUV. Image conversion occurs if necessary, so Camera can return images in the format requested. Camera calibration parameters are also provided.

[3] http://www.ptgrey.com/products/grasshopper2/grasshopper2_gige_camera.asp

Camera objects are created from a factory class. So that LibSAR applications can be hardware agnostic, the Camera Factory automatically detects all cameras available to the system. Cameras can then be acquired based on required configuration parameters, such as resolution or framerate. The factory class will then search available cameras for one that meets the requirements, and return one if found. Alternatively, cameras can be loaded by an ID. However, this requires LibSAR systems to have more knowledge of the hardware configuration.

#### 7.1.4.2 Camera Access for Modules

The LibSAR runtime provides access of cameras to modules. The reason for this is because many modules may need to access the same camera, and therefore should receive exactly the same camera image during the update phase. Different modules may need images in different formats. LibSAR needs to function correctly with many different cameras, and modules should not be concerned with particular cameras; they should simply access whatever cameras are available.

Cameras are loaded at runtime based on an XML configuration file, such as that shown in Listing 7.5. All of the configuration parameters are optional, including the device ID.

```
<Camera name="logitech">
    <Config key="DeviceID">/dev/video0</Config>
    <Config key="Width">640</Config>
    <Config key="Height">480</Config>
    <Config key="Framerate">30</Config>
</Camera>
```

**Listing 7.5:** Describing cameras in an XML file

Once cameras are loaded, modules can access them by name. For example, a module can obtain access to the camera named “logitech” using the code shown in Listing 7.6.

```
void init(const OptionList& options) {
    camera = mSystemManager.getCamManager().getCamera("logitech");
}
```

**Listing 7.6:** Obtaining a camera in a module's init function

### 7.1.5 Example Application

This section demonstrates how LibSAR can greatly simplify the task of writing a SAR application. The example application will project a television onto a physical box, with a video stream from a camera shown on the box. The result of running this module is shown in Figure 7.4. The module is implemented as a class that extends from Module. The class definition is shown in Listing 7.7.

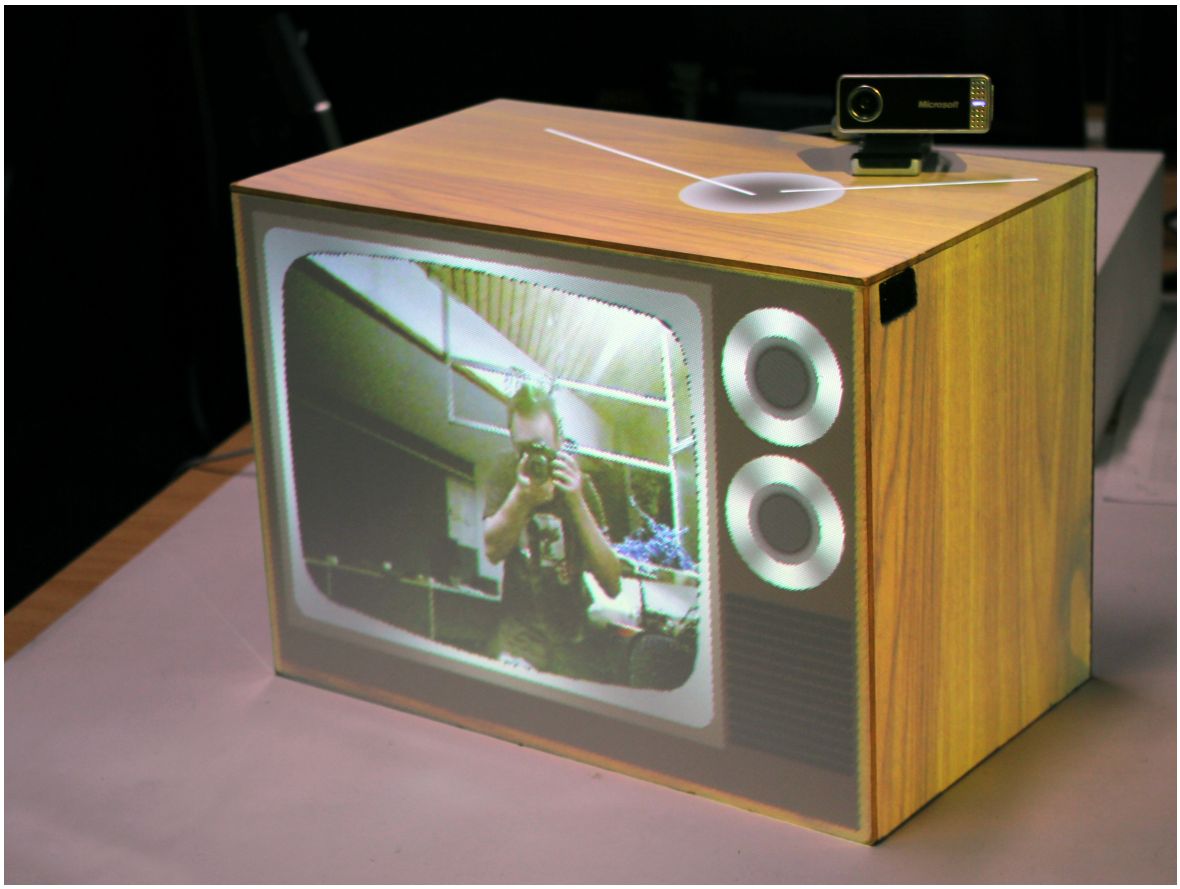


**Figure 7.4:** Result of running the TV module. A textured 3D model is projected onto a blank white box. The video stream is captured from the attached webcam and displayed on the TV.

```
class TV : public Module {
    public:
        TV(const std::string& name, SystemManager& sysMgr);
        virtual ~TV();
        virtual void update(unsigned int timestamp);
        virtual void draw(const Projector* p);
        virtual void init(const OptionList& options);
    private:
        wcl::Camera* camera;
        unsigned char* buffer;
        graphics::Texture* tvTexture;
        graphics::Texture* videoTexture;
        geometry::Geometry* tv;
};
```

**Listing 7.7:** Example Module: The TV class

#### 7.1.5.1 Module Initialisation

The *init* function, shown in Listing 7.8, loads all resources the module will need. The Camera Manager is used to obtain a handle for a camera that will be used for the video stream. A Texture object, which abstracts OpenGL textures, is created for the camera's video stream. The 3D model is loaded by the Resource Manager, as is the texture used for the TV's appearance.

```
void TV::init(const OptionList& options) {
    camera = mSystemManager.getCamManager().getCamera();
    buffer = new unsigned char[camera->getFormatBufferSize()];
    Configuration c = camera->getActiveConfiguration();
    image = new image::BufferedImage(c.width,
                                     c.height,
                                     image::AbstractImage::RGB8, buffer);

    videoTexture = new graphics::Texture(image);

    ResourceManager& resMgr = mSystemManager.getResMgr();
    tv = resMgr.getGeometry("tv.obj");
    tvTexture = new graphics::Texture(resMgr().getImage("tv.bmp"));
}
```

**Listing 7.8:** Initialising the TV Module

#### 7.1.5.2 Updating

The *update* function is responsible for updating the video texture with the latest frame from the camera. The utility libraries provided by LibSAR allow this to be accomplished with just two lines of code, as shown in Listing 7.9.

```
void TV::update(unsigned int timestamp) {
    camera->getCurrentFrame(buffer, Camera::RGB8);
    videoTexture->update(image);
}
```

**Listing 7.9:** Updating the TV Module

#### 7.1.5.3 Rendering

The *draw* function renders the TV to the configured projectors, and is shown in Listing 7.10. First, the TV geometry is rendered using the texture image. Following this, an additional polygon is rendered to display the video feed from the camera. This illustrates one of the benefits of LibSAR. Developers can simplify their work by making use of the functionality provided by the library and runtime. However, direct access to the graphics library is available if required.

#### 7.1.5.4 Discussion

While it is difficult to formally evaluate a large software framework such as LibSAR, the TV module illustrates the benefits of LibSAR. The module is free to focus on application logic, and does not have to handle any specific hardware configuration. All that the module requires is an available camera. The library greatly simplified the task of loading 3D geometry and textures.

Rendering was simplified by using the Geometry class provided by the library. However, direct access to OpenGL was also available for rendering the video stream. In addition, the *draw* function did not have to deal with specific projector configurations. As the projector parameters are set before *draw* is called, the module can focus

```
void TV::update(unsigned int timestamp) {
    tvTexture->bind();
    tv->draw();

    videoTexture->bind();
    glBegin(GL_QUADS);
    glTexCoord2f(GL_TEXTURE1, 0,0);
    glVertex3f(161.8058, 240.9763, 102.5 );

    glTexCoord2f(GL_TEXTURE1, 1,0);
    glVertex3f(-101.9348, 240.9763, 102.5 );

    glTexCoord2f(GL_TEXTURE1, 1,1);
    glVertex3f(-101.9348, 27.8879, 102.5);

    glTexCoord2f(GL_TEXTURE1, 0,1);
    glVertex3f(161.8058, 27.8879, 102.5);
    glEnd();
}
```

**Listing 7.10:** Rendering the TV

on rendering what it needs to, and can assume the projector is already configured. This also allows modules to function correctly with any number of projectors.

## 7.2 Finger Tracking with an Active Marker

This section explores the idea of using an adaptive Red, Green, Blue light-emitting diode (RGB LED) in place of a passive colour marker for a vision based tracking solution specifically for SAR environments. While this dissertation is primarily concerned with interaction using physical tools, finger tracking is important for tasks such as interacting with virtual controls.

Our fingertips are one of the most sensitive parts of the human body, the distribution of the primary motor cortex is pictorially represented on the Cortical Homunculus diagram [MH07] . We use our fingers to feel textures, test the temperature and perform fined grained manipulation tasks. The dexterous nature of our fingers make them an obvious choice for interactions in SAR environments. Attaching a passive coloured marker to the index finger allows the finger's position to be

tracked and used as a gateway between the physical and virtual worlds. This section advances the marker's functionality further by making it possible to actively change colour. This provides more robust tracking and visual feedback directly on top of a user's fingertip. The colour light sensor is also positioned directly above the fingertip to capture the environmental light of a SAR system which is used to adapt the marker to changing conditions in real-time.

The novel active marker allows its own colour appearance to be altered in real-time to avoid conflicting environmental lighting conditions. By incorporating a colour light sensor that can capture the surrounding environmental lighting conditions, an optimal marker colour can be chosen and applied to maximise the performance of the tracking system. This approach is particularly useful for SAR environments where the projected light interferes with the perceived colour of passive markers. The coloured light of the marker is also used to present visual feedback on the top of the user's fingertip to provide a combined input/output device for a virtual-physical interface.

Previous research used a blob-tracking system [Tay07] with a passive coloured marker to allow the designer to use their fingers to interact with a physical-virtual dashboard prototype [Por+10a]. This approach was successful but suffered from a few limitations. Firstly, the orange marker would not track well when the projected imagery was also orange. A workaround required selecting orange for the marker colour and avoiding using orange in the projected images. Although this was successful, the approach restricts the flexibility of the projection environment. The performance of tracking a passive coloured marker also deteriorates with changing ambient lighting; recalibration is required when the room lights are turned on and off.

The active marker described in this section overcomes these limitations. Two styles of active marker were considered: Infra-Red (IR) and visible light. IR markers are particularly useful for creating invisible markers that the human eye can not see.

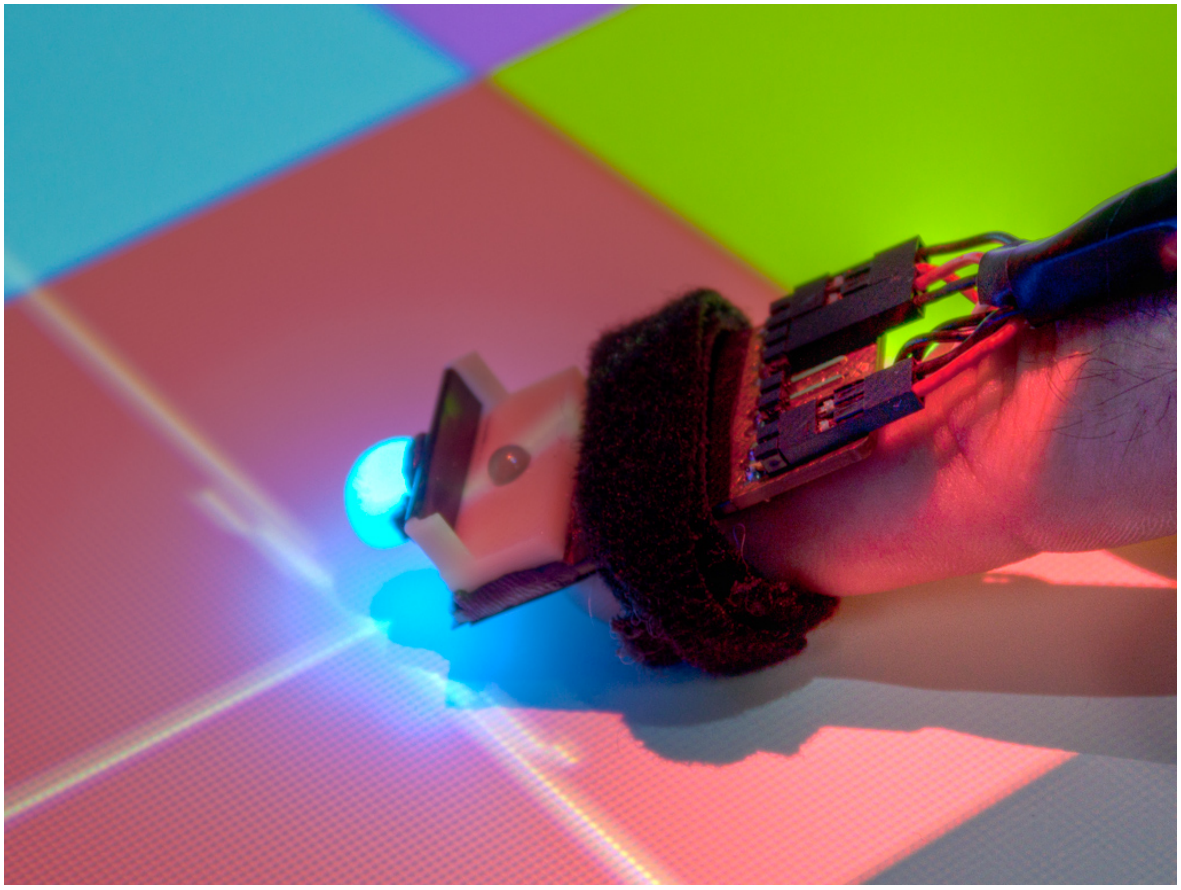


**Figure 7.5:** Adaptive marker being tracked. The colour of the RGB LED is automatically selected based on the colour of the projected light that is detected on the integrated colour sensor.

Alternatively, light in the visible spectrum can be used to provide a tracked marker. An advantage of visible light is the marker can be used for two functions. Firstly, it is used to capture a 3D position allowing interactions with the system without specialised cameras. Secondly, the active coloured marker can be used to provide visual feedback when operating interactive physical-virtual controls. Figure 7.5 shows the adaptive marker prototype capturing the colour of the red projected light and selecting blue as an optimal tracking colour.

### 7.2.1 Colour Selection and Tracking Strategy

The tracking software previously employed [Tay07] was modified to support an active, adaptive marker. The integrated colour sensor is used to sense the colour being projected onto the marker. The goal is to select the *opposite* of this colour for the LED. To achieve this, the detected colour is converted to the Hue/Saturation/Value (HSV) colour space.

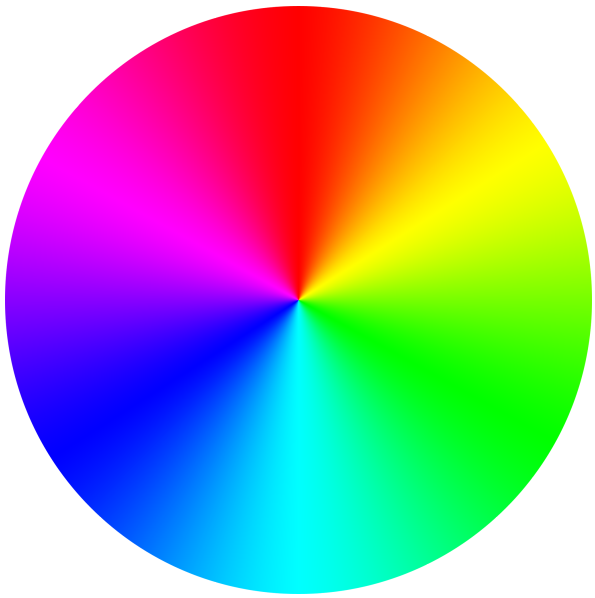


**Figure 7.6:** The HSV colour space places hues on a colour wheel.

HSV places hues at specific locations on a virtual colour wheel, as shown in Figure 7.6. The *opposite* colour will be the hue 180° from the detected colour. The LED on the marker is set to this colour. For example, if the light falling on the sensor is mostly green, then magenta will be chosen for the tracking colour. The cameras capture the finger tip marker, and colour segmentation (thresholding) is performed to separate the marker from the background. Using these images, the tracking software calculates the 3D position of the marker.

### 7.2.2 Visual Feedback Technique and Applications

The primary application for the active visible light marker is to support interactions in a SAR environment. Traditional buttons commonly use an internal light to illuminate the top surface when depressed. This provides an intuitive form of visual feedback to indicate the operation has been successful. When operating the combined physical-virtual buttons in a SAR environment (Figure 7.7(a)) the illuminated button press can be difficult to display when the user's finger is on the projection surface (Figure 7.7(b)). For this scenario, the visible light of the active marker can be leveraged to provide interactive feedback that is incorporated with the function of virtual buttons. For example, Figure 7.7(c) shows a physical-virtual button in its inactive state where the button is red. When the system registers a button press the colour of the active marker can be altered to indicate a successful button press with clear visual feedback (LED changes to red as shown in Figure 7.7(d)).

There are also other applications that can leverage the finger mounted active markers functionality. Tabletop systems such as the Diamondtouch [DL01] or Microsoft PixelSense[4] allow complex multi-user interactions on their surface. Using a fingertip active colour marker the same feedback technique can be used to provide additional information to the user. A slider could also leverage a similar functionality where the intensity of the LED is used to represent the current value of the slider.

### 7.2.3 Performance Evaluation

Another function of the adaptive marker is to improve the performance of tracking compared to a static colour marker. This section compares the performance of both static and dynamic markers in a SAR environment and identifies a number of scenarios where the static colour markers fail in comparison to the adaptive marker.

[4] http://www.microsoft.com/en-us/pixelsense/default.aspx

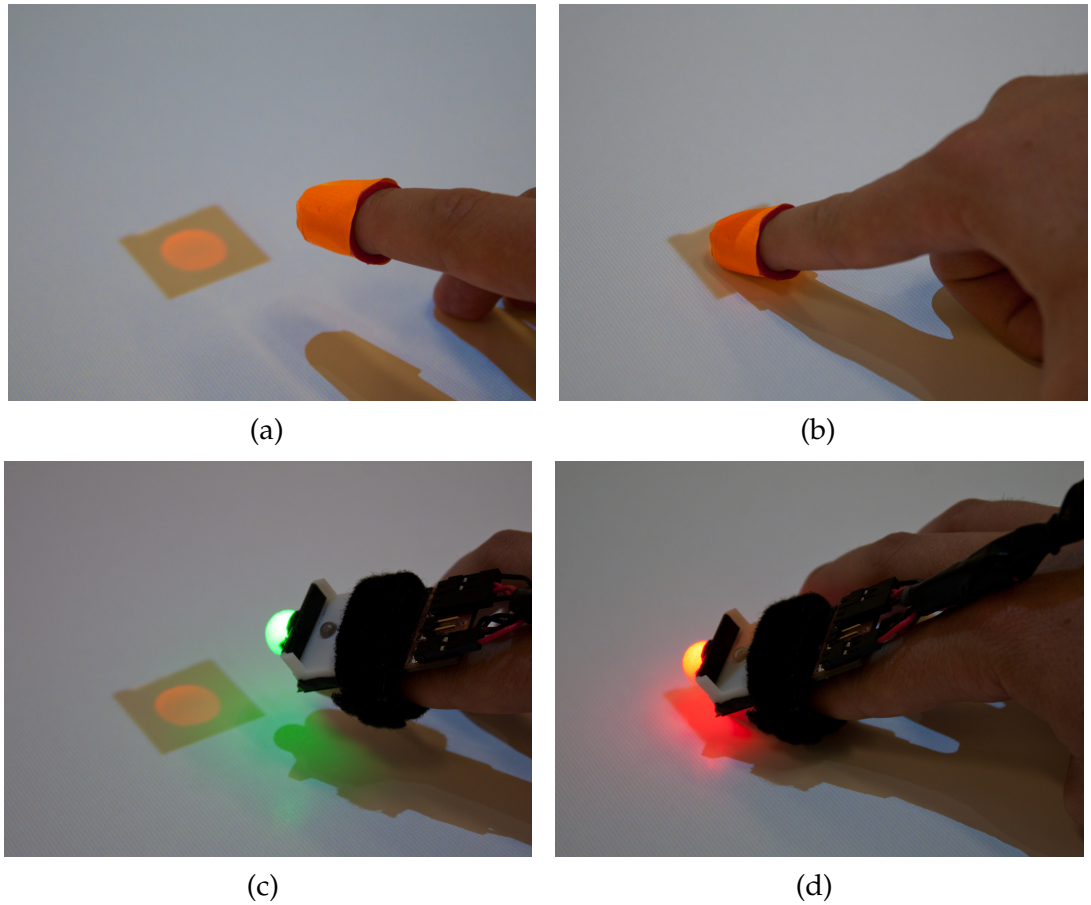
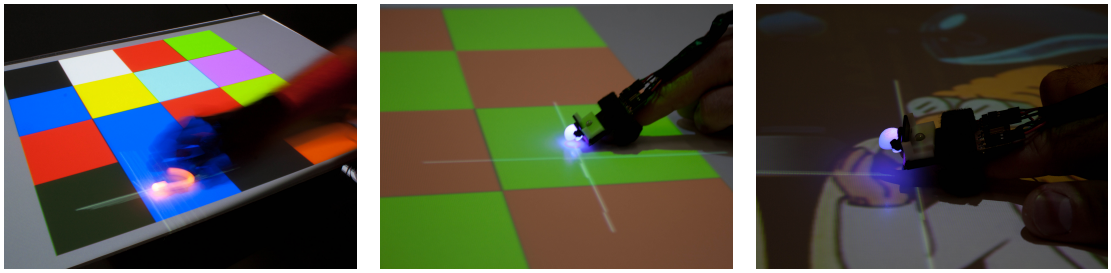



**Figure 7.7:** Interacting with a passive marker obscures feedback provided by the projection (a, b). The tracking colour can be used to replace this feedback (c, d).

The passive marker is shown in Figure 7.9 and the active marker used for evaluation is shown in Figure 7.5. A summary of the results of this evaluation is depicted in Figure 7.10.

#### 7.2.3.1 Experimental Setup

A tabletop SAR environment was created with two projectors, two cameras, and a white projection surface on the table. One projector was used to project a crosshair on the marker, and another to project the SAR information onto the table. The two cameras were positioned above the table surface and used for tracking the marker at 15 frames per second. The tracking software was calibrated to maximise the per-



(a) (b) (c)

**Figure 7.8:** The projected images used for the colour palette and changing ambient light tests (a), the orange-green test (b), and the dynamic projection test (c).

formance for each marker. The passive marker performed best with more ambient light (room lights on), and the active marker operated best with less ambient light (room lights off). Both the adaptive marker and orange passive marker were subjected to four test conditions: colour palette, changing ambient light, orange-green, and dynamic projection. Each condition was performed five times for each marker type. During each test, the system ran for two minutes and the marker was moved at a speed ranging from approximately 20-400cm per minute. The total number of frames processed and the number of frames where accurate tracking was obtained were recorded. The performance of each marker type was measured using the percentage of tracked frames per test condition. For the sake of the experiment, "accurate" tracking simply means the crosshair was positioned over the marker, and not elsewhere on the projection surface. As the tracking software used was developed elsewhere [Tay07], a robust performance evaluation was not conducted. Instead, the differences in performance between the active marker and passive marker were measured. For this task, the simplified accuracy definition was sufficient.

#### 7.2.3.2 Colour Palette Test

In this condition, a colour palette consisting of 16 colour tiles was projected onto the table. During each test, the marker was moved around the palette, covering all tiles. The goal of this condition was to evaluate how well both markers tracked with

different coloured backgrounds and to test the performance cost of the adaptive marker changing colour as it moved across the palette.

**Results:** The adaptive marker was accurately tracked 87.97% of the total frames, where as the passive marker was tracked 46.52% of the time. This gives a good baseline of the relative performance of the two marker types. The adaptive marker is able to modify its colour based on what is being projected onto it. In testing, many of the missed frames for the adaptive marker were during the transition from one colour to the next. The reason for this is the latency in the camera system; the tracker would look for the new marker colour in camera images before the change occurred.

#### 7.2.3.3 Changing Ambient Light Test

This condition used the same colour palette as the previous condition, but with the ambient light changing throughout the test. The room lighting was cycled on and off every twenty seconds. The goal of this condition was to measure how well each marker type tracks in situations with changing ambient lighting beyond their optimal configurations.

**Results:** The adaptive marker also performed much better in this condition than the passive marker, achieving tracking 92.05% of the time compared to 42.38%. The tracking of the adaptive marker was actually better for this condition. This is because when the room lights were turned on, the projected background became washed out, resulting in the marker changing colour less often. The passive marker performed worse. This is because as a passive device, good ambient lighting is needed to segment the image. Tracking failed when the ambient light was off. An active marker is less susceptible to changes in ambient light.

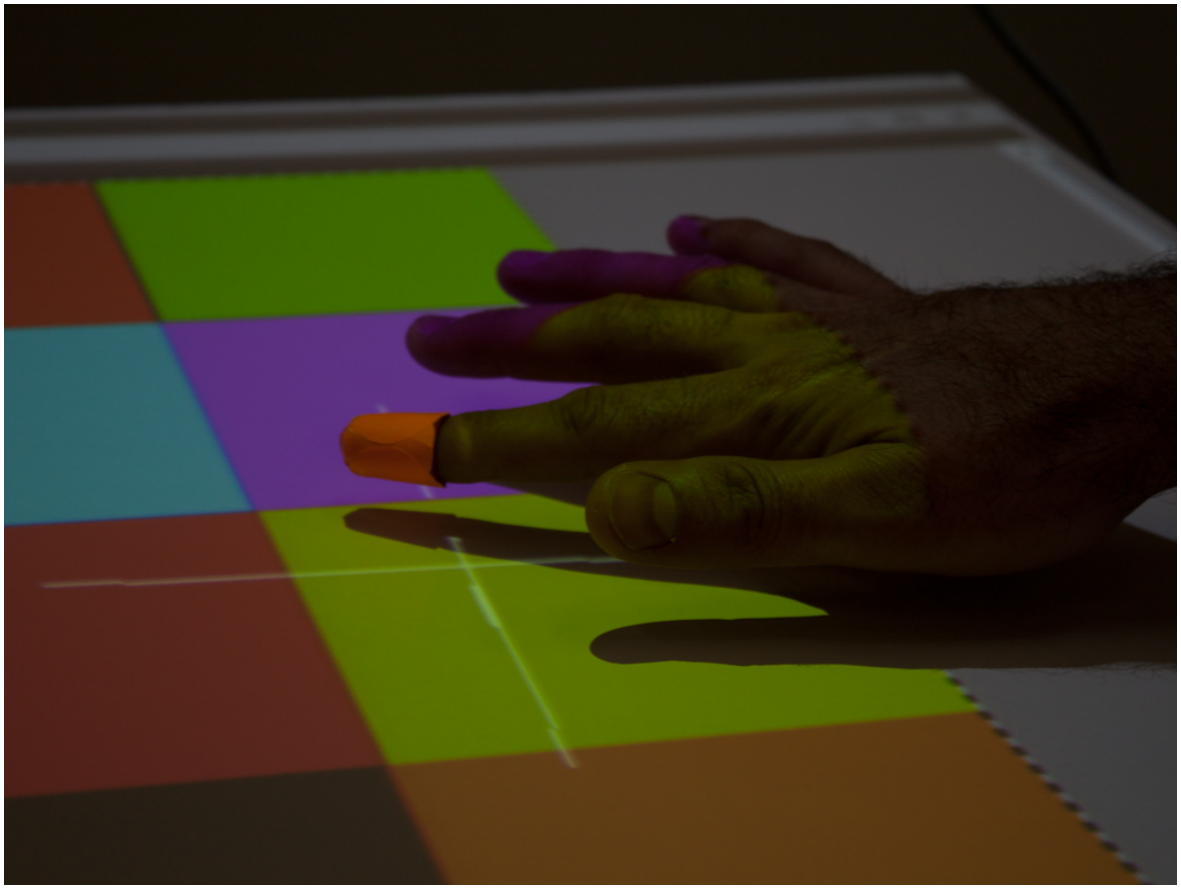


**Figure 7.9:** Passive marker tracked in a projected SAR environment.

#### 7.2.3.4 Orange-Green Test

This test compared the passive fingertip marker (orange in colour) and the adaptive marker when the projected background has a high percentage of orange. The purpose of this is to point out a specific case where the passive marker does not work and the adaptive marker is advantageous.

**Results:** The adaptive marker tracked 90.67% of the time in this condition; the passive marker only tracked 33.16% of the time. As expected, this condition resulted in the lowest performance for the passive marker, as the background contained a lot of orange, the same colour as the marker. This shows the flexibility of the active marker, as it works in a larger variety of projected light conditions.

#### 7.2.3.5 Dynamic Projection Test

This condition used a streaming video clip as the background to provide large variations in the colour of the projected light. The goal of the test was to measure the performance of the both markers under constantly changing conditions. Both marker types received the same section of video.

**Results:** Both the adaptive and passive markers obtained their highest performance in this condition; tracking 96.32% and 47.60% of the time, respectively. The adaptive marker once again performed much better than the passive marker. The higher scores overall can be attributed to the video clip containing many more darker shades of colours, making it easier to segment the markers from the background.

### 7.2.4 Discussion

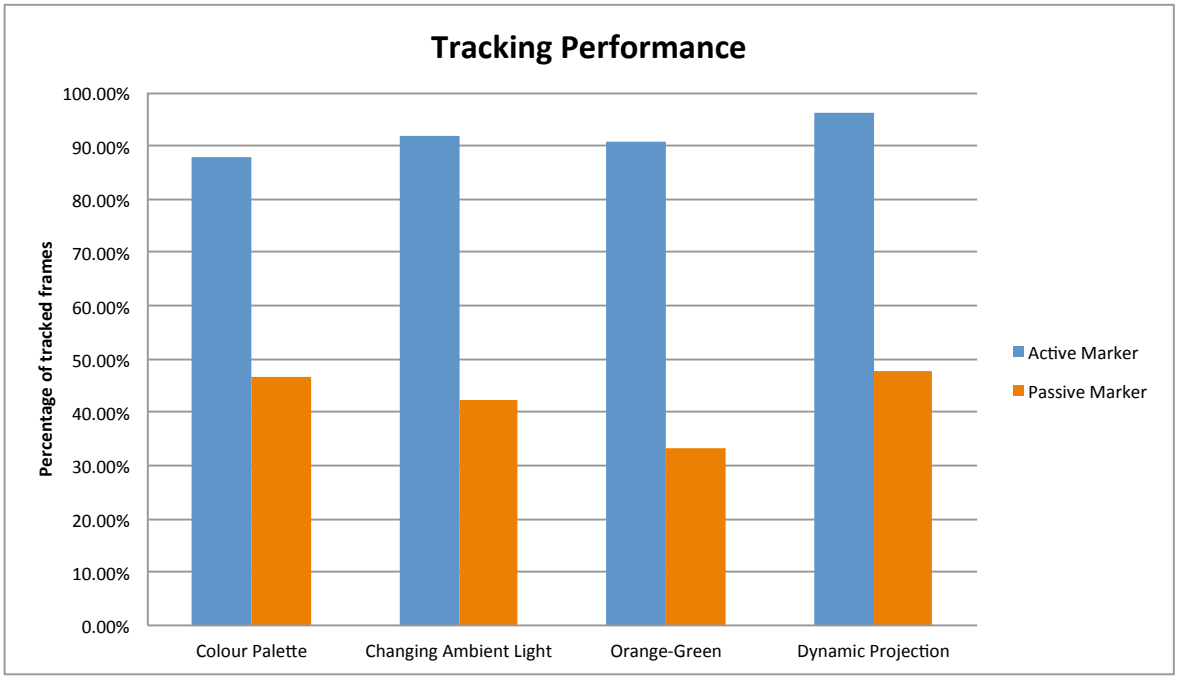



**Figure 7.10:** Comparison of Tracking Performance between Adaptive Marker and Passive Marker.

The results of the evaluation reveal much higher performance from the adaptive

marker compared to a passive coloured marker. One reason for this is the adaptive marker emits light. This makes it less susceptible to poor or changing ambient lighting conditions. This can be seen from the ambient light test where the performance of the passive marker suffered while the adaptive marker continued to operate correctly. Secondly, the marker is able to change its colour, making it much easier to segment from the background. This can be seen in all the tests, in particular the "Orange-Green Test" that has a large percentage of orange projected light where the passive marker had its lowest performance. For both the active and passive markers the cameras captured at 15 frames per second, this could be increased with more sophisticated camera hardware.

#### 7.2.4.1 Limitations

Another limitation of the current implementation is the colour light sensor and the LED are close to each other but not in identical locations. Given the separation it is possible that when moving through a projected light area, the colour light sensor and LED are exposed to different environmental light conditions. Our current implementation has a 3mm gap between the two lenses. This limitation has not affected operations, but would be a consideration for fine grained manipulations that have dynamic projected images.

#### 7.2.4.2 Alternate Colour Selection Strategies

The colour selection strategy developed and tested uses an integrated colour sensor on the marker. However, other techniques are possible. This section describes two additional strategies that could be implemented on an adaptive colour marker.

**Least Common Colour:** The cameras' images could be processed and the least commonly occurring colour found. The least common colour is a good candidate for the marker colour. As with the light sensor approach, the colour could be updated

every frame, accounting for lighting changes or moving objects in the environment. The downside to this approach is the colour chosen for tracking may match the background near the sensor.

**Random Selection:** This strategy cycles through all possible LED colours. At each iteration, both the LED and tracking software are configured to a new colour, with the colour changing each frame. This technique would work most of the time, as the colour is constantly changing. However, eventually a similar colour to the background would be chosen, and tracking would fail until the colour changed.

# 8

# Conclusion

This dissertation presents several significant and original contributions to the field of spatial augmented reality interaction. The interaction techniques advance the state-of-the-art for SAR user interface design. The research has explored how and where SAR can be applied to the industrial design process to improve the workflow for designers with three demonstration applications, which embody the tools and techniques developed.

The contributions presented in this dissertation can be summarised as the following:

- Physical-Virtual Tools, a methodology for creating SAR user interfaces, and two continua to aid in designing and comparing PVT user interfaces.
- Digital Airbrushing, a two handed PVT technique for applying finishes and designing paint jobs for products using similar techniques to real airbrushing, and a description of algorithms to implement this technique.
- Augmented Foam Sculpting, a technique that allows designers to produce 3D digital models by sculpting with foam, taking advantage of tools and techniques already in use by industrial designers.

- PVT techniques for architecture and interior design tasks, and a demonstration application, BuildMyKitchen, which embodies these techniques.
- A description of a model software framework designed specifically for building SAR systems.
- Improvements to existing techniques for tracking the user's finger with visible marker, and techniques for using the tracking marker to provide user feedback when interacting with virtual controls.

This final chapter summarises each of these contributions, describes limitations with the work developed, and ends with a look towards future directions for SAR research and applications.

## 8.1 Physical-Virtual Tools

Physical-Virtual Tools is a novel framework for constructing interactive SAR systems. User interfaces are comprised of physical tools the user interacts with to accomplish their tasks. The physical design of each tool should be tailored to specific kinds of interaction. A core benefit of a PVT user interface is that the tools are also surfaces for projecting information. Rather than requiring an additional computer monitor or projection area, all of the information required by the user is projected onto the tools the user works with. Projecting onto tools also allows for a meaningful overloading of the tasks a tool supports. The active task and relevant settings are conveyed to the user through the tool's appearance.

Two continua for measuring virtualisation and task overloading of tools in virtual environments were presented. The WS-TVC compares individual tools used in a single system. It features two axes: task rank and attribute rank. Application designers can use this continuum to aid in making decisions regarding the number

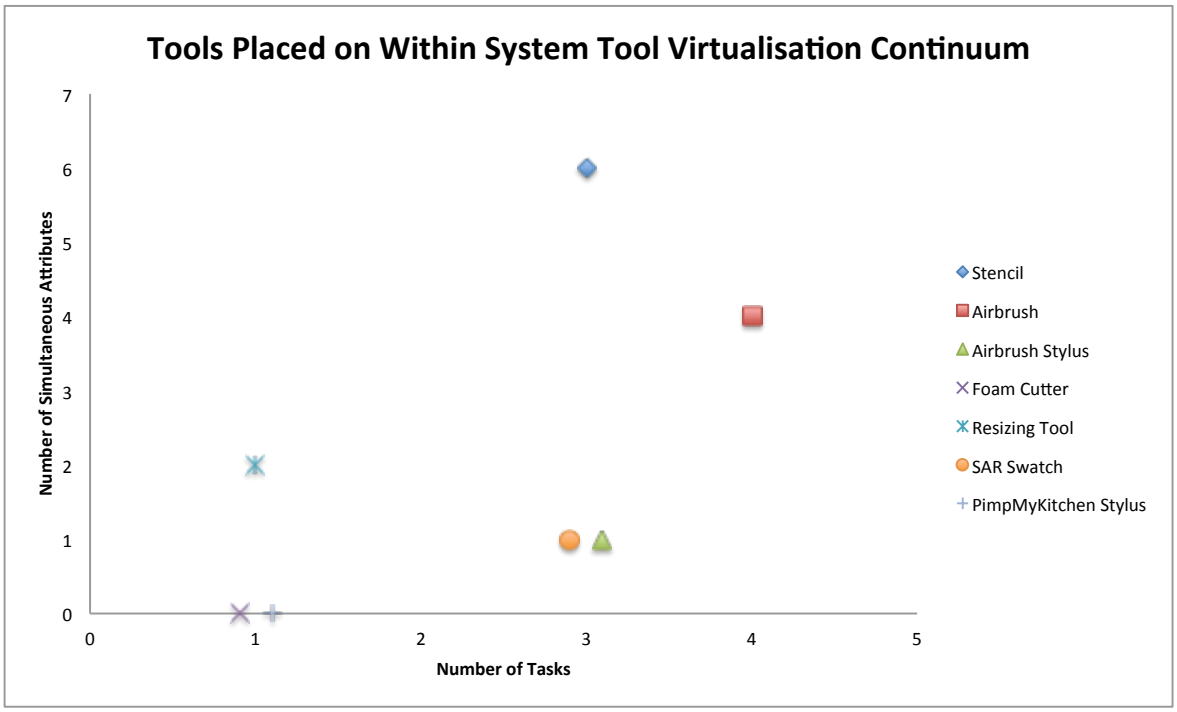



**Figure 8.1:** The tools presented in this dissertation placed onto the WS-TVC.

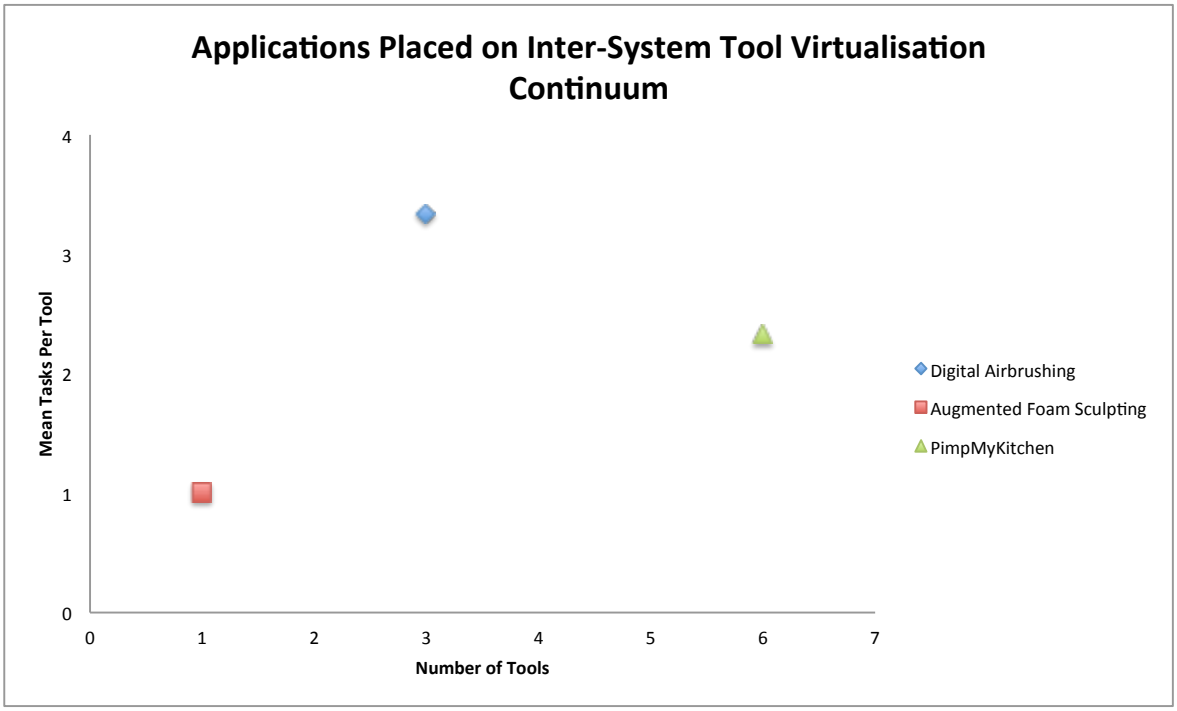



**Figure 8.2:** The applications presented in this dissertation placed onto the IS-TVC.

and physical design of the tools required for a system. The tools presented in this dissertation have been placed on the WS-TVC, illustrated in Figure 8.1. The IS-TVC

is used to compare the relative complexity of user interfaces among multiple systems. This two axis continuum considers the number of tools in a system, and the mean number of tasks per tool. This dissertation has described how the WS-TVC can aid designers when designing systems. The applications presented in this thesis are placed on the IS-TVC, illustrated in Figure 8.2.

In the future I would like to investigate whether general optimal numbers of tools and levels of task overloading exist. The TVC allows the complexity of tools and systems, but it remains to be seen whether there is, in general, an optimal level of task overloading. This would further aid application designers, as the continua would be divided into "acceptable" and "unacceptable" regions.

## 8.2 Digital Airbrushing

Digital Airbrushing is an interactive SAR system built using the PVT paradigm, which improves common tasks for designers at early stages of the design process. This system allows the designer to annotate 3D models by drawing on a physical mock-up. The designer can also paint with an airbrush, using a stencil to mask out areas in a similar way to how an artist would use a real airbrush. Three physical tools have been developed for this application. Each tool has a form factor suitable for different tasks. Future enhancements to Digital Airbrushing would include a more accurate paint simulation, and to extend the system with different tools.

## 8.3 Augmented Foam Sculpting

Chapter 5 has presented a new technique, Augmented Foam Sculpting, for generating 3D models using traditional tools for foam sculpting, already in use by industrial designers. By using the tools that designers currently use, designers can more easily adapt their workflow and begin using the system. By using foam blocks and a phys-

ical cutter, users benefit from the passive haptic feedback the objects provide, and the users can create 3D models with an intuitive bimanual interaction technique.

The use of SAR enables the projection of extra information onto the objects as they are being sculpted. This ability has been used to visualise how to produce target models from foam, either by changing the colour of the surface to represent areas that should be cut away, or by 'playing back' cuts from a previous session. The projector is also used to display the wireframe of the 3D model, or a wood grain texture onto the foam, as it is being sculpted. Although the target application domain for Augmented Foam Sculpting is industrial design, the technology could be used in other areas. The system would support any domain requiring physical objects with matching 3D models, or where 3D models are created from hand sculpted designs, for example the computer animation and visual effects industry.

In the future I would like to extend this system to support extra tools. For example, it is not possible to add small grooves or indents into the foam using only the hot wire cutter. However, other tools such as a hot tip iron could be modelled using CSG, in a similar way to the cutter. I believe there is much more potential for SAR visualisations to improve our system, and will look towards new functionality and visualisations. I would also like to modify the algorithms that generate the cut geometry to address the limitations described in the paper, making the system more robust.

## 8.4 SAR for Interior Architecture

Chapter 6 presented investigations into using SAR for kitchen design and interior architecture. The prototype SAR design application, BuildMyKitchen, was presented, and how it improves the design process for both architects and their clients was discussed. BuildMyKitchen features a PVT based user interface. The chapter

discussed PVT-based tools and techniques developed for the application, and how they can be generalised for use in other application domains.

Future extensions of BuildMyKitchen could allow functional analysis of kitchen layouts. For example, showing the available storage space once appliances, plumbing, and other components are in place. Designers could optimise the functional layout by having the system highlight the predicted use of areas of the design, based on data on the client's actual usage patterns. In addition to these analysis tasks, the graphics system utilised could also be improved. This would allow accurate previews of lighting configurations, based on photometric models of actual light fittings.

## 8.5 Software Support for SAR Systems

The software framework developed though the course of this research greatly improves the ability of researchers to develop and assess SAR applications and research. The architecture described in Chapter 7 can be applied to other platforms and programming languages as required. Perhaps the best indication of the benefits of the techniques described in this dissertation is the fact that the software is now in use by several researchers inside the Wearable Computer Lab and in larger collaborations with other groups, such as Intel [CMT10].

While the software implementation is ready for use now, there is still work to be completed. One of the most pressing issues is the need for better integration with existing CAD systems. For SAR to truly be integrated into industrial design processes, the data interchange pipeline must be further developed and refined. Another area to be addressed is running a single SAR application across several computers. The hardware currently in use in the Wearable Computer Lab allows for eight projector outputs per computer. However, the requirements for large room or building sized

SAR environments demand greater numbers of projectors. Efficiently distributing workloads between several hosts is still to be investigated.

## 8.6 Active Marker for SAR Tracking

The tracking system presented in Chapter 7 incorporates a colour light sensor and RGB LED. The sensor is used capture the environmental light, so that a contrasting colour can be used for LED marker. This optimises the contrast between the marker and environment, improving the performance of the tracking algorithm. To compare the performance of this system, it was tested and compared to a passive coloured marker in four conditions. All four test conditions highlight the benefits of the active marker. Techniques for how the visible spectrum active marker can be further leveraged to provide visual feedback when operating virtual buttons are described. This is advantageous since the "click" operation of a virtual button can be seen directly on top of the user's finger when operating occluded controls.

## 8.7 Future Directions & Final Comments

This dissertation has focussed on bringing SAR to industrial design and architecture domains. However, the technology presented can be put to use in many other areas. Manufacturing and maintenance would benefit from augmented tools. For example, a technician could be directed to a specific bolt to be removed from a piece of machinery using arrows and a distance measure projected onto the tool. Additional information could be given to the user such as nut sizes, tension, etc. Training scenarios could also benefit from SAR. Directions for performing a task could be given by projecting onto both the work area and tools.

SAR technology also has the potential to revolutionise theatre and interactive art. Through the course of my research I have worked on two interactive theatre

projects. *Half Real* uses SAR technology to track actors as they move about on stage. Actors interact with a virtual world that is projected onto the set. Audience members control the flow of the story. A show such as Half Real would simply not be able to be produced without SAR technology. Half Real is described in detail in Appendix A. The second show, entitled *If There Was A Colour Darker Than Black I'd Wear It* (BLACK), utilised SAR technology alongside more traditional visual effects, such as lighting, silk flames, and smoke machines. One of the effects created for BLACK is shown in Figure 8.3. Where Half Real pushed SAR towards its limits, BLACK used the technology sparingly, alongside projection-mapped art and other media. This work hints at perhaps a more realistic future for SAR in theatre: as another tool available to the creative team. The biggest challenge facing SAR in theatre is the need for user interfaces and authoring toolkits that can be operated by non-experts.

**Figure 8.3:** BLACK uses a combination of SAR, smoke machines, and traditional theatre lighting effects to create the illusion of a burning car. Image courtesy of Lachlan Tetlow-Stuart.

The algorithms and hardware technology that make SAR possible have reached a level of maturity that means SAR can be deployed today. The key challenge is identifying niches where SAR can be applied and be truly useful, and continuing to research new ways of interacting with these systems. This dissertation has successfully identified areas of the industrial design domain where SAR can be put to use. However, many more possibilities still remain. I look forward to continuing to both push the state of the art in the lab, and take proven technologies and put them to use in interesting, unique ways into the future.

# References


[AB94] Azuma, R. and Bishop, G. "Improving static and dynamic registration in an optical see-through HMD". In: *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. SIGGRAPH '94. New York, NY, USA: ACM, 1994, 197–204.

[Agr+97] Agrawala, M., Beers, A., McDowall, I., Frohlich, B., Bolas, M., and Hanrahan, P. "The two-user Responsive Workbench: support for collaboration through individual views of a shared space". In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, pp. 327–332.

[AV09] Akaoka, E. and Vertegaal, R. "DisplayObjects: prototyping functional physical interfaces on 3D styrofoam, paper or cardboard models". In: *CHI 2009*. Boston, MA, 2009.

[Azu+01] Azuma, R., Baillot, Y., Behringer, R., Feiner, S., Julier, S., and MacIntyre, B. "Recent Advances in Augmented Reality". In: *IEEE Computer Graphics Applications* 21.6 (2001). 618862, pp. 34–47.

[Azu97] Azuma, R. T. "A Survey of Augmented Reality". In: *Presence: Teleoperators and Virtual Environments* 6.4 (1997), pp. 355–385.

[Bal+03] Ballagas, R., Ringel, M., Stone, M., and Borchers, J. "iStuff: a physical user interface toolkit for ubiquitous computing environments". In: *Pro-*

*ceedings of the SIGCHI conference on Human factors in computing systems*. CHI '03. New York, NY, USA: ACM, 2003, 537–544.

[Bar+12] Baricevic, D., Lee, C., Turk, M., Hollerer, T., and Bowman, D. "A hand-held AR magic lens with user-perspective rendering". In: *Proceedings of the 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Nov. 2012, pp. 197–206.

[BE06] Bimber, O. and Emmerling, A. "Multifocal projection: a multiprojector technique for increasing focal depth". In: *Visualization and Computer Graphics, IEEE Transactions on* 12.4 (2006), pp. 658–667.

[Bea+05] Beardsley, P., Baar, J. van, Raskar, R., and Forlines, C. "Interaction using a handheld projector". In: *Computer Graphics and Applications, IEEE* 25.1 (2005), pp. 39–43.

[BES03] Bimber, O., Encarna, L. M., and Schmalstieg, D. "The virtual showcase as a new platform for augmented reality digital storytelling". In: *Workshop on Virtual environments 2003*. Zurich, Switzerland: ACM, 2003, pp. 87–95.

[BF02] Bimber, O. and Frohlich, B. "Occlusion shadows: using projected light to generate realistic occlusion effects for view-dependent optical see-through displays". In: *International Symposium on Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings*. 2002, pp. 186–319.

[Bha93] Bhatnagar, D. K. *Position Trackers for Head Mounted Display Systems: A Survey*. Tech. rep. 1993.

[Bie+93] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. "Toolglass and Magic Lenses: The See-Through Interface". In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. Anaheim, CA, USA: ACM, 1993, pp. 73–80.

[Bim+01] Bimber, O., Frohlich, B., Schmalsteig, D., and Encarnacao, L. M. "The Virtual Showcase". In: *Computer Graphics and Applications, IEEE* 21.6 (2001), pp. 48–55.

[Bim+02] Bimber, O., Gatesy, S. M., Witmer, L. M., Raskar, R., and Encarnacao, L. M. "Merging Fossil Specimens with Computer-Generated Information". In: *IEEE Computer, September* (2002). article, pp. 45–50.

[Bim+05a] Bimber, O., Coriand, F., Kleppe, A., Bruns, E., Zollmann, S., and Langlotz, T. "Superimposing pictorial artwork with projected imagery". In: *Multimedia, IEEE* 12.1 (2005), pp. 16–26.

[Bim+05b] Bimber, O., Wetzstein, G., Emmerling, A., and Nitschke, C. A. "Enabling view-dependent stereoscopic projection in real environments". In: *Proceedings of the Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2005, pp. 14–23.

[BM86] Buxton, W. and Myers, B. "A study in two-handed input". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Boston, MA, USA: ACM, 1986, pp. 321–326.

[BNB07] Ball, R., North, C., and Bowman, D. A. "Move to improve: promoting physical navigation to increase user performance with large displays". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '07. New York, NY, USA: ACM, 2007, 191–200.

[BP00] Brusey, J. and Padgham, L. "Techniques for Obtaining Robust, Real-Ti Colour-Based Vision for Robotics". In: *RoboCup-99: Robot Soccer World Cup III*. London, UK: Springer-Verlag, 2000, pp. 243–253.

[BR05] Bimber, O. and Raskar, R. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. Wellesley: A K Peters, 2005.

[BRF01] Bandyopadhyay, D., Raskar, R., and Fuchs, H. "Dynamic Shader Lamps: Painting on Movable Objects". In: *IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2001, pp. 207–216.

[Bro99] Brooks, F. P. "What's real about virtual reality?" In: *Computer Graphics and Applications, IEEE* 19.6 (1999), pp. 16–27.

[BS05] Bennett, E. and Stevens, B. "The effect that touching a Projection Augmented model has on Object-Presence". In: *Ninth International Conference on Information Visualisation*. 2005.

[But08] Butzow, F. "Development of a Shadow Removal Techique for Projector Illuminated 3D Objects in Spatial Augmented Reality". Honours Thesis. University of South Australia, 2008.

[BWB08] Benko, H., Wilson, A. D., and Balakrishnan, R. "Sphere: multi-touch interactions on a spherical display". In: *Proceedings of the 21st annual ACM symposium on User interface software and technology*. Monterey, CA, USA: ACM, 2008, pp. 77–86.

[Byu+07] Byung-Kuk Seo, Moon-Hyun Lee, Hanhoon Park, Jong-Il Park, and Young Soo Kim. "Direct-Projected AR Based Interactive User Interface for Medical Surgery". In: *Artificial Reality and Telexistence, 17th International Conference on*. 2007, pp. 105–112.

[CB06] Cao, X. and Balakrishnan, R. "Interacting with dynamically defined information spaces using a handheld projector and a pen". In: *Proceedings of the 19th annual ACM symposium on User interface software and technology*. Montreux, Switzerland: ACM, 2006, pp. 225–234.

[CGA11] CGAL. *Computational Geometry Algorithms Library*. http://www.cgal.org. 2011.

[Chu06] Chunky Move. *Glow*. http://chunkymove.com.au/Our-Works/Current-Productions/Glow.aspx. 2006.

[Chu08] Chunky Move. *Mortal Engine*. http://chunkymove.com.au/Our-Works/Current-Productions/Mortal-Engine.aspx. 2008.

[CMT10] Close, B., McCulley, D., and Thomas, B. H. "ARPipes: Aligning Virtual Models to their Physical Counterparts with Spatial Augmented Reality". In: *Proceedings of the 20th International Conference on Artificial Reality and Telexistence*. Adelaide, South Australia, 2010.

[CSD93] Cruz-Neira, C., Sandin, D. J., and DeFanti, T. A. "Surround-screen projection-based virtual reality: the design and implementation of the CAVE". In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*. Anaheim, CA, USA: ACM, 1993, pp. 135–142.

[DHS02] Dobler, D., Haller, M., and Stampfl, P. "ASR: augmented sound reality". In: *ACM SIGGRAPH 2002 conference abstracts and applications*. San Antonio, Texas: ACM, 2002, pp. 148–148.

[DL01] Dietz, P. and Leigh, D. "DiamondTouch: a multi-user touch technology". In: *Proceedings of the 14th annual ACM symposium on User interface software and technology*. Orlando, FL, 2001, pp. 219–226.

[Fei+93] Feiner, S., MacIntyre, B., Haupt, M., and Solomon, E. "Windows on the world: 2D windows for 3D augmented reality". In: *Proceedings of the 6th annual ACM symposium on User interface software and technology*. Atlanta, GA, USA: ACM, 1993, pp. 145–155.

[Fei+97] Feiner, S., MacIntyre, B., Hollerer, T., and Webster, A. "A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment". In: *Proceedings of the 1st IEEE International*

*Symposium on Wearable Computers*. ISWC '97. Washington, DC, USA: IEEE Computer Society, 1997, pp. 74–.

[FHZ96] Forsberg, A., Herndon, K., and Zeleznik, R. "Aperture based selection for immersive virtual environments". In: *Proceedings of the 9th annual ACM symposium on User interface software and technology*. Seattle, Washington, USA: ACM, 1996, pp. 95–96.

[FIB95] Fitzmaurice, G. W., Ishii, H., and Buxton, W. A. S. "Bricks: Laying The Foundations for Graspable User interfaces". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Denver, Colorado, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 442–449.

[For+05] Forlines, C., Balakrishnan, R., Beardsley, P., Baar, J. van, and Raskar, R. "Zoom-and-pick: facilitating visual zooming and precision pointing with interactive handheld projectors". In: *Proceedings of the 18th annual ACM symposium on User interface software and technology*. UIST '05. New York, NY, USA: ACM, 2005, 73–82.

[GF01] Greenberg, S. and Fitchett, C. "Phidgets: easy development of physical interfaces through physical widgets". In: *Proceedings of the 14th annual ACM symposium on User interface software and technology*. Orlando, FL, USA: ACM, 2001, pp. 209–218.

[GM96] Graham, E. D. and MacKenzie, C. L. "Physical versus virtual pointing". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '96. New York, NY, USA: ACM, 1996, 292–299.

[Gro+04] Grossberg, M. D., Peri, H., Nayar, S. K., and Belhumeur, P. N. "Making One Object Look Like Another: Controlling Appearance Using a Projector-Camera System". In: *Computer Vision and Pattern Recognition,*

*2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*. 2004, pp. 452–459.

[Gru+07] Grundhofer, A., Seeger, M., Hantsch, F., and Bimber, O. "Dynamic Adaptation of Projected Imperceptible Codes". In: *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR'07)*. Nara, Japan, 2007, pp. 181–190.

[Har+09] Hare, J., Gill, S., Loudon, G., Ramduny-Ellis, D., and Dix, A. "Physical Fidelity: Exploring the Importance of Physicality on Physical-Digital Conceptual Prototyping". In: *Human-Computer Interaction – INTERACT 2009*. 2009, pp. 217–230.

[HBW11] Harrison, C., Benko, H., and Wilson, A. D. "OmniTouch: wearable multitouch interaction everywhere". In: *Proceedings of the 24th annual ACM symposium on User interface software and technology*. UIST '11. New York, NY, USA: ACM, 2011, pp. 441–450.

[HC03] Hsiao, S. and Chuang, J. "A reverse engineering based approach for product form design". In: *Design Studies* 24 (Mar. 2003), pp. 155–171.

[Hin+94] Hinckley, K., Pausch, R., Goble, J. C., and Kassell, N. F. "Passive real-world interface props for neurosurgical visualization". In: *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*. Boston, MA, USA: ACM, 1994, pp. 452–458.

[Hof+98] Hoffman, H., Hollander, A., Schroder, K., Rousseau, S., and Furness, T. "Physically touching and tasting virtual objects enhances the realism of virtual experiences". In: *Virtual Reality* 3.4 (1998), pp. 226–234.

[Ina+00] Inami, M., Kawakami, N., Sekiguchi, D., Yanagida, Y., Maeda, T., and Tachi, S. "Visuo-haptic display using head-mounted projector". In: *Virtual Reality, 2000. Proceedings. IEEE*. 2000, pp. 233–240.

[Ish+04] Ishii, H., Ratti, C., Piper, B., Wang, Y., Biderman, A., and Ben-Joseph, E. "Bringing Clay and Sand into Digital Design — Continuous Tangible user Interfaces". In: *BT Technology Journal* 22.4 (Oct. 2004), pp. 287–299.

[Ish08] Ishii, H. "Tangible bits: beyond pixels". In: *Proceedings of the 2nd international conference on Tangible and embedded interaction*. TEI '08. New York, NY, USA: ACM, 2008, xv–xxv.

[Jon+10] Jones, B. R., Sodhi, R., Campbell, R. H., Garnett, G., and Bailey, B. P. "Build Your World and Play In It: Interacting with Surface Particles on Complex Objects". In: *International Symposium on Mixed and Augmented Reality*. Seoul, Korea, 2010.

[Jun+04] Jung, H.-K., Nam, T.-J., Lee, H.-S., and Han, S.-Y. "Spray Modeling: Augmented Reality Based 3D Modeling Interface for Intuitive and Evolutionary Form Development". In: *International Conference on Artificial Reality and Telexistence*. 2004.

[Kat+00] Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., and Tachibana, K. "Virtual object manipulation on a table-top AR environment". In: *Augmented Reality, 2000. (ISAR 2000). Proceedings. IEEE and ACM International Symposium on*. 2000, pp. 111–119.

[KB99] Kato, H. and Billinghurst, M. "Marker tracking and HMD calibration for a video-based augmented reality conferencing system". In: *2nd IEEE and ACM International Workshop on Augmented Reality, 1999. (IWAR '99) Proceedings*. 1999, pp. 85–94.

[KBS94] Kabbash, P., Buxton, W., and Sellen, A. "Two-handed input in a compound task". In: *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*. Boston, MA, USA: ACM, 1994, pp. 417–423.

[KI08] Kobayashi, M. and Igarashi, T. “Ninja cursors: using multiple cursors to assist target acquisition on large screens”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’08. New York, NY, USA: ACM, 2008, 949–958.

[KKT90] Kirkpatrick, D. G., Klawe, M. M., and Tarjan, R. E. “Polygon triangulation in O(n log log n) time with simple data-structures”. In: *Proceedings of the sixth annual symposium on Computational geometry*. Berkley, California, USA: ACM, 1990, pp. 34–43.

[KP08] Kry, P. G. and Pai, D. K. “Grasp Recognition and Manipulation with the Tango”. In: *Experimental Robotics*. Ed. by Khatib, O., Kumar, V., and Rus, D. Springer Tracts in Advanced Robotics 39. Springer Berlin Heidelberg, Jan. 2008, pp. 551–559.

[Kur+07] Kurz, D., Hantsch, F., Große, M., Schiewe, A., and Bimber, O. “Laser Pointer Tracking in Projector-Augmented Architectural Environments”. In: *Proceedings of the International Symposium on Mixed and Augmented Reality*. 2007, pp. 19–26.

[Kuz+00] Kuzuoka, H., Oyama, S., Yamazaki, K., Suzuki, K., and Mitsuishi, M. “GestureMan: a mobile robot that embodies a remote instructor’s actions”. In: *Proceedings of the 2000 ACM conference on Computer supported cooperative work*. CSCW ’00. New York, NY, USA: ACM, 2000, 155–162.

[Lan96] Lantz, E. “The future of virtual reality: head mounted displays versus spatially immersive displays (panel)”. In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. SIGGRAPH ’96. New York, NY, USA: ACM, 1996, 485–486.

[LBP05] Lifton, J., Broxton, M., and Paradiso, J. A. “Experiences and directions in pushpin computing”. In: *Proceedings of the 4th international sympo-*

*sium on Information processing in sensor networks*. Los Angeles, California: IEEE Press, 2005.

[Lee+00] Lee, J., Su, V., Ren, S., and Ishii, H. "HandSCAPE: a vectorizing tape measure for on-site measuring applications". In: *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. CHI '00. New York, NY, USA: ACM, 2000, 137–144.

[Lee+04a] Lee, J. C., Avrahami, D., Hudson, S. E., Forlizzi, J., Dietz, P. H., and Leigh, D. "The calder toolkit: wired and wireless components for rapidly prototyping interactive devices". In: *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques*. Cambridge, MA, USA: ACM, 2004, pp. 167–175.

[Lee+04b] Lee, J. C., Dietz, P. H., Maynes-Aminzade, D., Raskar, R., and Hudson, S. E. "Automatic projector calibration with embedded light sensors". In: *Proceedings of the 17th annual ACM symposium on User interface software and technology*. Santa Fe, NM, USA: ACM, 2004, pp. 123–126.

[Lei+08] Leitner, J., Haller, M., Yun, K., Woo, W., Sugimoto, M., and Inami, M. "IncreTable, a mixed reality tabletop game experience". In: *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*. Yokohama, Japan: ACM, 2008, pp. 9–16.

[Lin+09] Lincoln, P., Welch, G., Nashel, A., Ilie, A., State, A., and Fuchs, H. "Animatronic Shader Lamps Avatars". In: *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality, 2009. ISMAR 2009*. Oct. 2009, pp. 27–33.

[Low+01] Low, K.-L., Welch, G., Lastra, A., and Fuchs, H. "Life-sized projector-based dioramas". In: *VRST '01: Proceedings of the ACM symposium on*

*Virtual reality software and technology*. inproceedings. New York, NY, USA: ACM, 2001, pp. 93–101.

[LTH86] Laidlaw, D. H., Trumbore, W. B., and Hughes, J. F. "Constructive solid geometry for polyhedral objects". In: *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*. ACM, 1986, pp. 161–170.

[LZB98] Leganchuk, A., Zhai, S., and Buxton, W. "Manual and cognitive benefits of two-handed input: an experimental study". In: *ACM Trans. Comput.-Hum. Interact.* 5.4 (1998), pp. 326–359.

[Maa+11] Maas, E., Marner, M. R., Smith, R. T., and Thomas, B. H. "Quimo: A Deformable Material to Support Freeform Modeling in Spatial Augmented Reality Environments". In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Singapore, 2011.

[Maa+12] Maas, E. T., Marner, M. R., Smith, R. T., and Thomas, B. H. "Supporting Freeform Modelling in Spatial Augmented Reality Environments with a New Deformable Material". In: *Proceedings of the 13th Australasian User Interface Conference*. Melbourne, Victoria, Australia, 2012.

[MAB92] Meyer, K., Applewhite, H. L., and Biocca, F. A. "A Survey of Position Trackers". In: *Presence* 1.2 (1992), pp. 173–200.

[Mar+11] Marner, M. R., Smith, R. T., Porter, S. R., Broecker, M., Close, B., and Thomas, B. H. "Large Scale Spatial Augmented Reality for Design and Prototyping". In: *Handbook of Augmented Reality*. Springer-Verlag, 2011.

[Mar+12a] Marner, M. R., Broecker, M., Close, B., and Thomas, B. "Spatial augmented reality (SAR) application development system". Provisional Patent 2012903729. 2012.

[Mar+12b] Marner, M. R., Haren, S., Gardiner, M., and Thomas, B. H. "Exploring Interactivity and Augmented Reality in Theater: A Case Study of Half Real". In: *Proceedings of the International Symposium on Mixed and Augmented Reality*. Atlanta, Georgia, USA, 2012.

[MFS97] Mine, M. R., Frederick P. Brooks, J., and Sequin, C. H. "Moving objects in space: exploiting proprioception in virtual-environment interaction". In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1997, pp. 19–26.

[MH07] Marieb, E. and Hoehn, K. *Anatomy and Physiology*. 7th. Pearson Benjamin Cummings: San Francisco, 2007.

[MK94] Milgram, P. and Kishino, F. "A Taxonomy of Mixed Reality Visual Displays". In: *IEICE Transactions on Information Systems* E77-D.12 (1994).

[MM09] Mistry, P. and Maes, P. "SixthSense: a wearable gestural interface". In: *ACM SIGGRAPH ASIA 2009 Sketches*. Yokohama, Japan: ACM, 2009.

[Moh+09] Mohan, A., Woo, G., Hiura, S., Smithwick, Q., and Raskar, R. "Bokode: imperceptible visual tags for camera based interaction from a distance". In: *ACM SIGGRAPH 2009 papers*. SIGGRAPH '09. New Orleans, Louisiana, USA: ACM, 2009, 98:1–98:8.

[Mor70] Mori, M. "The Uncanny Valley". In: *Energy* 7.4 (1970), pp. 33–35.

[MPS02] Maynes-Aminzade, D., Pausch, R., and Seitz, S. "Techniques for Interactive Audience Participation". In: *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*. ICMI '02. Washington, DC, USA: IEEE Computer Society, 2002.

[MT10a] Marner, M. R. and Thomas, B. H. "Tool Virtualization and Spatial Augmented Reality". In: *Proceedings of the 20th International Conference on Artificial Reality and Telexistence*. Adelaide, South Australia, 2010.

[MT10b] Marner, M. R. and Thomas, B. H. "Augmented Foam Sculpting for Capturing 3D Models". In: *Proceedings of the IEEE Symposium on 3D User Interfaces*. Waltham, MA, USA, 2010.

[MT13] Marner, M. R. and Thomas, B. H. "Spatial Augmented Reality User Interface Techniques for Room Size Modeling Tasks". In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Orlando, FL, USA, 2013.

[MTS09] Marner, M. R., Thomas, B. H., and Sandor, C. "Physical-Virtual Tools for Spatial Augmented Reality User Interfaces". In: *Proceedings of the International Symposium on Mixed and Augmented Reality*. Orlando, FL, USA, 2009.

[MW01] Majumder, A. and Welch, G. "Computer graphics optique optical superposition of projected computer graphics". In: *Proceedings of the 7th Eurographics conference on Virtual Environments & 5th Immersive Projection Technology*. EGVE'01. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2001, 209–218.

[Mye+02] Myers, B. A., Bhatnagar, R., Nichols, J., Peck, C. H., Kong, D., Miller, R., and Long, A. C. "Interacting at a distance: measuring the performance of laser pointers and other devices". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '02. New York, NY, USA: ACM, 2002, 33–40.

[Nai05] Naimark, M. "Two Unusual Projection Spaces". In: *Presence: Teleoperators and Virtual Environments* 14.5 (2005), pp. 597–605.

[Nai84] Naimark, M. “Spatial Correspondence in Motion Picture Display”. In: *Proceedings of Optics in Entertainment II*. Vol. 462. 1984.

[Nam05] Nam, T.-J. “Sketch-based rapid prototyping platform for hardware-software integrated interactive products”. In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '05. New York, NY, USA: ACM, 2005, 1689–1692.

[Ngu+06] Nguyen, Q., Novakowski, S., Boyd, J. E., Jacob, C., and Hushlak, G. “Motion swarms: video interaction for art in complex environments”. In: *Proceedings of the 14th annual ACM international conference on Multimedia*. MULTIMEDIA '06. New York, NY, USA: ACM, 2006, 461–469.

[NKY05] Nakazato, Y., Kanbara, M., and Yokoya, N. “Localization of wearable users using invisible retro-reflective markers and an IR camera”. In: *Proc. SPIE Electronic Imaging*. Vol. 5664. 2005, pp. 1234–1242.

[NKY08] Nakazato, M., Kanbara, M., and Yokoya, N. “Localization System for large indoor environments using invisible markers”. In: *ACM Symposium on Virtual Reality Software and Technology*. 2008, pp. 295–296.

[OF09] Oda, O. and Feiner, S. “Interference avoidance in multi-user hand-held augmented reality”. In: *Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality, 2009. ISMAR 2009*. Oct. 2009, pp. 13–22.

[ON01] Olsen, D. R. J. and Nielsen, T. “Laser pointer interaction”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Seattle, Washington, USA: ACM, 2001, pp. 17–22.

[Pai+05] Pai, D., VanDerLoo, E., Sadhukhan, S., and Kry, P. “The Tango: a tangible tangoreceptive whole-hand human interface”. In: *Eurohaptics Conference, 2005 and Symposium on Haptic Interfaces for Virtual Environment*

*and Teleoperator Systems, 2005. World Haptics 2005. First Joint*. Mar. 2005, pp. 141–147.

[PBW84] Pahl, G., Beitz, W., and Wallace, K. *Engineering design: A systematic approach*. Springer, 1984.

[PI01] Piper, B. and Ishii, H. *CADcast: a Method for Projecting Spatially Referenced Procedural Instructions*. Tech. rep. MIT Media Lab, 2001.

[Pin01] Pinhanez, C. "The Everywhere Displays Projector: A Device to Create Ubiquitous Graphical Interfaces". In: *Ubicomp 2001: Ubiquitous Computing*. 2001, pp. 315–331.

[PKC08] Pihuit, A., Kryt, P., and Cani, M.-P. "Hands on virtual clay". In: *IEEE International Conference on Shape Modeling and Applications, 2008. SMI 2008*. June 2008, pp. 267–268.

[PNJ10] Prytz, E., Nilsson, S., and Jönsson, A. "The importance of eye-contact for collaboration in AR systems". In: *Proceedings of the 2010 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Oct. 2010, pp. 119–126.

[Por+09] Porter, S., Marner, M. R., Eck, U., Sandor, C., and Thomas, B. H. "Rundle Lantern in Miniature: Simulating Large Scale Non-Planar Displays". In: *Proceedings of the International Conference on Advances in Computer Entertainment Technology*. Athens, Greece, 2009.

[Por+10a] Porter, S. R., Marner, M. R., Smith, R. T., Zucco, J. E., and Thomas, B. H. "Validating Spatial Augmented Reality for Interactive Rapid Prototyping". In: *Proceedings of the 9th IEEE International Symposium on Mixed and Augmented Reality*. Seoul, Korea, 2010.

[Por+10b] Porter, S. R., Marner, M. R., Smith, R. T., Zucco, J. E., Thomas, B. H., and Schumacher, P. "Spatial Augmented Reality for Interactive Rapid Prototyping". In: *Proceedings of the 20th International Conference on Artificial Reality and Telexistence*. Adelaide, South Australia, 2010.

[Pou+96] Poupyrev, I., Billinghurst, M., Weghorst, S., and Ichikawa, T. "The go-go interaction technique: non-linear mapping for direct manipulation in VR". In: *Proceedings of the 9th annual ACM symposium on User interface software and technology*. Seattle, Washington, USA: ACM, 1996, pp. 79–80.

[PP04] Park, H. and Park, J. "Invisible marker tracking for AR". In: *International symposium on mixed and augmented reality*. 2004, pp. 272–273.

[PRI02] Piper, B., Ratti, C., and Ishii, H. "Illuminating clay: a 3-D tangible interface for landscape analysis". In: *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*. Minneapolis, Minnesota, USA: ACM, 2002, pp. 355–362.

[PS06] Piekarski, W. and Smith, R. "Robust gloves for 3D interaction in mobile outdoor AR environments". In: *Proceedings of the IEEE/ACM International Symposium on Mixed and Augmented Reality, 2006. ISMAR 2006*. Oct. 2006, pp. 251–252.

[PSP99] Pierce, J. S., Stearns, B. C., and Pausch, R. "Voodoo dolls: seamless interaction at multiple scales in virtual environments". In: *Proceedings of the 1999 symposium on Interactive 3D graphics*. Atlanta, Georgia, USA: ACM, 1999, pp. 141–145.

[PT02] Piekarski, W. and Thomas, B. H. "The Tinmith system: demonstrating new techniques for mobile augmented reality modelling". In: *Proceedings of the Third Australasian conference on User interfaces - Volume 7*. Mel-

bourne, Victoria, Australia: Australian Computer Society, Inc., 2002, pp. 61–70.

[Pug91] Pugh, S. *Total Design: integrated methods for successful product engineering*. Addison-Wesley, 1991.

[Rak+05] Rakkolainen, I., DiVerdi, S., Olwal, A., Candussi, N., Hüllerer, T., Laitinen, M., Piirto, M., and Palovuori, K. "The interactive FogScreen". In: *ACM SIGGRAPH 2005 Emerging technologies*. SIGGRAPH '05. New York, NY, USA: ACM, 2005.

[Ras+01] Raskar, R., Welch, G., Low, K.-L., and Bandyopadhyay, D. "Shader Lamps: Animating Real Objects With Image-Based Illumination". In: *Rendering Techniques 2001: Proceedings of the Eurographics*. 2001, pp. 89–102.

[Ras+03] Raskar, R., Baar, J. v., Beardsley, P., Willwacher, T., Rao, S., and Forlines, C. "iLamps: geometrically aware and self-configuring projectors". In: *ACM SIGGRAPH 2005 Courses*. Los Angeles, California: ACM, 2003.

[Ras+04] Raskar, R., Beardsley, P., Baar, J. v., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T. "RFIG lamps: interacting with a self-describing world via photosensing wireless tags and projectors". In: *ACM Trans. Graph.* 23.3 (2004), pp. 406–415.

[Ras+07] Raskar, R., Nii, H., deDecker, B., Hashimoto, Y., Summet, J., Moore, D., Zhao, Y., Westhues, J., Dietz, P., Barnwell, J., Nayar, S., Inami, M., Bekaert, P., Noland, M., Branzoi, V., and Bruns, E. "Prakash: lighting aware motion capture using photosensing markers and multiplexed illuminators". In: *ACM Trans. Graph.* 26 (3 July 2007).

[Ras+98] Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., and Fuchs, H. "The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays". In: *SIGGRAPH '98*. 1998.

[Ras+99] Raskar, R., Brown, M. S., Yang, R., Chen, W.-C., Welch, G., Towles, H., Seales, B., and Fuchs, H. "Multi-projector displays using camera-based registration". In: *Proceedings of the conference on Visualization '99: celebrating ten years*. San Francisco, CA, USA: IEEE Computer Society Press, 1999, pp. 161–168.

[RB01] Raskar, R. and Beardsley, P. "A self-correcting projector". In: *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*. Vol. 2. 2001, pages.

[RL01] Raskar, R. and Low, K.-L. "Interacting with spatially augmented reality". In: *Proceedings of the 1st international conference on Computer graphics, virtual reality and visualisation*. Camps Bay, Cape Town, South Africa: ACM, 2001, pp. 101–108.

[Ros06] Rost, R. J. *OpenGL(R) Shading Language*. 2nd ed. Addison-Wesley Professional, 2006.

[RS99] Rekimoto, J. and Saitoh, M. "Augmented surfaces: a spatially continuous work space for hybrid computing environments". In: *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 378–385.

[RTH96] Rasmussen, C., Toyama, K., and Hager, G. D. *Tracking Objects By Color Alone*. Tech. rep. Yale University, 1996.

[RWC99] Raskar, R., Welch, G., and Chen, W.-C. "Table-top spatially-augmented realty: bringing physical models to life with projected imagery". In: *2nd IEEE and ACM International Workshop on Augmented Reality, 1999. (IWAR '99) Proceedings*. 1999, pp. 64–71.

[RWF98] Raskar, R., Welch, G., and Fuchs, H. "Spatially Augmented Reality". In: *In First IEEE Workshop on Augmented Reality (IWAR'98)*. 1998, 11–20.

[RZW02] Raskar, R., Ziegler, R., and Willwacher, T. “Cartoon dioramas in motion”. In: *NPAR '02: Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering*. inproceedings. New York, NY, USA: ACM, 2002.

[Sak+05] Sakata, N., Kurata, T., Kato, T., Kourogi, M., and Kuzuoka, H. “WACL: supporting telecommunications using - wearable active camera with laser pointer”. English. In: *Seventh IEEE International Symposium on Wearable Computers, 2003. Proceedings*. IEEE, Oct. 2005, pp. 53–56.

[San+10] Sandor, C., Cunningham, A., Eck, U., Urquhart, D., Jarvis, G., Dey, A., Barbier, S., Marner, M. R., and Rhee, S. “Egocentric Space-Distorting Visualizations for Rapid Environment Exploration in Mobile Mixed Reality”. In: *IEEE Symposium on Virtual Reality*. Waltham, MA, USA, 2010.

[Sch+08] Schwerdtfeger, B., Pustka, D., Hofhauser, A., and Klinker, G. “Using laser projectors for augmented reality”. In: *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*. Bordeaux, France: ACM, 2008, pp. 134–137.

[SCP95] Stoakley, R., Conway, M. J., and Pausch, R. “Virtual reality on a WIM: interactive worlds in miniature”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Denver, Colorado, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 265–272.

[SG97] Szalavári, Z. and Gervautz, M. “The personal interaction panel - a Two handed interface for augmented reality”. In: *EUROGRAPHICS 97*. Budapest, Hungary, 1997, pp. 335–346.

[Sim+12] Simon, T. M., Smith, R. T., Thomas, B. H., Von Itzstein, G. S., Smith, M., Park, J., and Park, J. “Merging Tangible Buttons and Spatial Augmented Reality to Support Ubiquitous Prototype Designs”. In: *Proceed-*

*ings of the 13th Australasian User Interface Conference*. Melbourne, Victoria, Australia, 2012.

[SK08] Schwerdtfeger, B. and Klinker, G. "Supporting order picking with Augmented Reality". In: *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*. 2008, pp. 91–94.

[SMT11] Smith, R. T., Marner, M. R., and Thomas, B. "Adaptive Color Marker for SAR Environments". In: *Poster Sessions: Proceedings of the IEEE Symposium on 3D User Interfaces*. Singapore, 2011.

[Son+06] Song, H., Guimbretière, F., Hu, C., and Lipson, H. "ModelCraft: capturing freehand annotations and edits on physical 3D models". In: *Proceedings of the 19th annual ACM symposium on User interface software and technology*. Montreux, Switzerland: ACM, 2006, pp. 13–22.

[SPS01] Schkolne, S., Pruett, M., and Schröder, P. "Surface drawing: creating organic 3D shapes with the hand and tangible tools". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. Seattle, Washington, USA: ACM, 2001, pp. 261–268.

[STB07] Shoemaker, G., Tang, A., and Booth, K. S. "Shadow reaching: a new perspective on interaction for large displays". In: *Proceedings of the 20th annual ACM symposium on User interface software and technology*. UIST '07. New York, NY, USA: ACM, 2007, 53–56.

[Ste+02] Stevens, B., Jerrams-Smith, J., Heathcote, D., and Callear, D. "Putting the virtual into reality: assessing object-presence with projection-augmented models". In: *Presence: Teleoper. Virtual Environ.* 11.1 (2002), pp. 79–92.

[STP08] Smith, R. T., Thomas, B. H., and Piekarski, W. "Digital foam interaction techniques for 3D modeling". In: *Proceedings of the 2008 ACM symposium*

*on Virtual reality software and technology*. Bordeaux, France: ACM, 2008, pp. 61–68.

[Stu99] Stuerzlinger, W. "Imaging all visible surfaces". In: *Proceedings of the 1999 conference on Graphics interface '99*. Kingston, Ontario, Canada: Morgan Kaufmann Publishers Inc., 1999, pp. 115–122.

[Sug+08] Suganuma, A., Ogata, Y., Shimada, A., Arita, D., and Taniguchi, R.-i. "Billiard instruction system for beginners with a projector-camera system". In: *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*. Yokohama, Japan: ACM, 2008, pp. 3–8.

[Sum+05] Summet, J., Abowd, G. D., Corso, G. M., and Rehg, J. M. "Virtual rear projection: do shadows matter?" In: *CHI '05 extended abstracts on Human factors in computing systems*. Portland, OR, USA: ACM, 2005, pp. 1997–2000.

[Sut65] Sutherland, I. "The Ultimate Display". In: *IFIP Congress*. New York, NY, 1965, pp. 506–508.

[Sut68] Sutherland, I. "A Head-Mounted Three-Dimensional Display". In: *AFIPS Fall Joint Computer Conference*. Washington, DC, 1968, pp. 757–764.

[Tay07] Taylor, S. *Natural Interaction for Table-Top Environments*. Tech. rep. University of Cambridge, 2007.

[Tho+11] Thomas, B. H., Von Itzstein, G. S., Vernik, R., Porter, S. R., Marner, M. R., Smith, R. T., Broecker, M., and Close, B. "Spatial Augmented Reality Support for Design of Complex Physical Environments". In: *Workshop on Interdisciplinary Approaches to Pervasive Computing Design*. 2011.

[UI97] Ullmer, B. and Ishii, H. “The metaDESK: models and prototypes for tangible user interfaces”. In: *Proceedings of the 10th annual ACM symposium on User interface software and technology*. Banff, Alberta, Canada: ACM, 1997, pp. 223–232.

[UI99] Underkoffler, J. and Ishii, H. “Urp: a luminous-tangible workbench for urban planning and design”. In: *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*. Pittsburgh, Pennsylvania, USA: ACM, 1999, pp. 386–393.

[UM11] Uchiyama, H. and Marchand, E. “Deformable Random Dot Markers”. In: *Poster Sessions: Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality*. Switzerland, 2011.

[US11] Uchiyama, H. and Saito, H. “Random dot markers”. In: *Proceedings of the 2011 IEEE Virtual Reality Conference (VR)*. IEEE, Mar. 2011, pp. 35–38.

[UUI99] Underkoffler, J., Ullmer, B., and Ishii, H. “Emancipated pixels: real-world graphics in the luminous room”. In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press / Addison-Wesley Publishing Co., 1999, pp. 385–392.

[Ver+03] Verlinden, J., Smit, A. d., Peeters, A., and Gelderen, M. v. “Development of a flexible augmented prototyping system”. In: *Journal of WSCG* (2003).

[Ver+05] Verlinden, J., Kooijman, A., Edelenbos, E., and Go, C. “Investigation on the use of illuminated clay in automotive styling”. In: *Proceedings of the 6th International Conference on Computer-Aided Industrial Design & Conceptual Design*. 2005.

[VHN09] Verlinden, J., Horváth, I., and Nam, T.-J. “Recording augmented reality experiences to capture design reviews”. en. In: *International Journal on*

*Interactive Design and Manufacturing (IJIDeM)* 3.3 (Aug. 2009), pp. 189–200.

[Wag07] Wagner, D. "Handheld Augmented Reality". PhD thesis. Graz University of Technology, 2007.

[Wel12] Welch, G. "Physical-Virtual Humans: Challenges and Opportunities". In: *2012 International Symposium on Ubiquitous Virtual Reality (ISUVR)*. Aug. 2012, pp. 10–13.

[Wel93] Wellner, P. "Interacting with paper on the DigitalDesk". In: *Commun. ACM* 36.7 (1993). 159630, pp. 87–96.

[WF02] Welch, G. and Foxlin, E. "Motion tracking: no silver bullet, but a respectable arsenal". In: *IEEE Computer Graphics and Applications* 22.6 (2002), pp. 24–38.

[WG95] Wloka, M. M. and Greenfield, E. "The virtual tricorder: a uniform interface for virtual reality". In: *Proceedings of the 8th annual ACM symposium on User interface and software technology*. Pittsburgh, Pennsylvania, USA: ACM, 1995, pp. 39–40.

[WR99] Ware, C. and Rose, J. "Rotating virtual objects with real handles". In: *ACM Trans. Comput.-Hum. Interact.* 6.2 (1999), pp. 162–180.

[WS07] Wagner, D. and Schmalsteig, D. "ARToolKitPlus for Pose Tracking on Mobile Devices". In: *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*. 2007.

[WS98] Witmer, B. G. and Singer, M. J. "Measuring Presence in Virtual Environments: A Presence Questionnaire". In: *Presence: Teleoperators and Virtual Environments* 7.3 (June 1998), pp. 225–240.

[ZB07] Zollmann, S. and Bimber, O. "Imperceptible Calibration for Radiometric Compensation". In: *Eurographics*. 2007, pp. 61–64.

[Zha00] Zhang, Z. "A flexible new technique for camera calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (Nov. 2000), pp. 1330–1334.

[ZHH06] Zeleznik, R. C., Herndon, K. P., and Hughes, J. F. "SKETCH: an interface for sketching 3D scenes". In: *ACM SIGGRAPH 2006 Courses*. Boston, MA: ACM, 2006, p. 9.

[Zio+10] Ziola, R., Grampurohit, S., Landes, N., Fogarty, J., and Harrison, B. *OASIS: Examining a Framework for Interacting with General-Purpose Object Recognition*. Tech. rep. Intel Labs Seattle, 2010.

[Zio+11] Ziola, R., Grampurohit, S., Landes, N., Fogarty, J., and Harrison, B. "Examining interaction with general-purpose object recognition in LEGO OASIS". In: *Proceedings of the 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. Sept. 2011, pp. 65–68.

[ZV06] Zaeh, M. and Vogl, W. "Interactive laser-projection for programming industrial robots". In: *Proceedings of the Mixed and Augmented Reality, 2006. ISMAR 2006. IEEE/ACM International Symposium on*. 2006, pp. 125–128.

# A

# Spatial Augmented Reality in Theatre: A Case Study of Half Real

This appendix presents a case study of the development of *Half Real*, an interactive theatre experience which features extensive use of SAR technology. I was responsible for developing the SAR projection system and integrating it with the ZigZag voting system. This work has previously been published at the International Symposium of Mixed and Augmented Reality in 2012 [Mar+12b].

## A.1 Introduction

*Who killed Violet Vario?* That is the question posed to the audience of Half Real, an interactive theatre show where the audience votes on how a murder investigation proceeds. Half Real utilises SAR [BR05] technology to immerse the actors in a projected virtual world (Figure A.1). This use of SAR represents a departure from the simple video projection that has been used in performance art in the past. Half Real's projection system represents the set as a 3D environment. Actors are tracked in real time, allowing the projected content to react to actors' movements, and for

virtual projected items to appear attached to actors. Half Real also takes advantage of this technology to support interactivity. The show puts the audience in control, voting on the path the story will take using wireless controllers.

This appendix provides a description of the technology used and how this technology made the production possible. It also discusses the lessons learned in developing a SAR system for performance art. The remainder of the appendix is as follows: first, a brief discussion of previous work relating to audience interactivity, real-time procedural visual content for performance art, and spatial augmented reality is presented. Section A.3 describes the SAR system developed for the show, how the stage is mapped as a 3D environment, and how projected content is created. Next, Section A.4 discusses the interactive aspects of Half Real, and how the technology was used to support this. Section A.5 describes how actors were tracked on stage throughout the show, and how this tracking data was used to drive the projected content. This appendix ends with an overview of the implementation details, discuss lessons learned from the production, and conclude with a look to the future of using SAR in performance art.

### A.1.1 Creative Goals and Intentions

Half Real is the third theatre work created by The Border Project which employed the audience interacting via the ZigZag controller system[1]. The Zigzags allow an audience to collectively make decisions to affect how the performance unfolds. The previous works (Trouble on Planet Earth and Escape from Peligro Island) were both inspired by the 'choose-your-own-adventure' genre of books. In these works, the audience was able to choose the decisions of the protagonist, allowing them to navigate a pre-rehearsed narrative tree of possibilities.

Half-Real is a departure from the audience controlling narrative. Instead, the

[1]http://matthewgardiner.net/art/ZigZag

**Figure A.1:** Half Real merges live action with a virtual world

performance is an experience where the audience "investigates" a narrative world with the goal of determining which of three suspects murdered character Violet Vario. The aesthetic and mode of the work referenced contemporary gaming within the thriller genre, such as Heavy Rain[2]. The audience had three 'levels' to inves-

[2]http://www.heavyrainps3.com

tigate. The first was to investigate three pieces of evidence and their associated events; the second was to investigate two of the three suspects further (eliminating one); and the final was to interpret one of the characters' recent nightmares.

Half Real had some clear objectives for its video system design:

1. To integrate a GUI into the represented place on stage, embedded within the dramatic action of the scene. In previous interactive works, the narrative world was paused, and a vote sequence would occur where the options were posed to the audience, a screen would display the GUI options for the audience's choice and the vote would occur. For Half Real, the goal was to integrate the GUI within the visual world occupied by the performers, and for the GUI options to appear at the point an investigative pathway emerged dramatically in the scene, and remain present until the point of the vote.

2. As the narrative investigations continually jumped location, time, and sometimes into a characters dream or fantasy, a video system was required that could easily represent a vast multitudes of spaces with minimal changes to the physical space.

3. To assist in representing a diverse range of characters. As Half-Real had a cast of three performers (mirroring the three suspects to be investigated), the production sought a different theatrical language to represent the many minor characters from the key suspects.

4. To create a design that was sophisticated and intricate, but which was also highly tourable and easy to set up.

## A.2 Background

Half Real was inspired by previous performances and art installations that employ projection. The work by Naimark [Nai05] reprojects a captured physical environment onto a white painted version. *Be Now Here* used a similar technique to recreate an outdoor public plaza. Dance productions such as Glow [Chu06] and Mortal Engine [Chu08], use camera based tracking to generate projected animated content in real time based on the dancers' movements and the soundtrack. Half Real builds upon this concept by using 3D tracking and a 3D projection environment. Half Real also combines procedurally generated content with pre-made video to enhance the performance and support interactivity. Audience interactivity has previously been implemented using camera based techniques. Maynes-Aminzade et al. [MPS02] demonstrate how computer vision can detect audience movements to control a virtual paddle in a game of pong, and how laser pointer tracking on a projection screen can be used for large scale interactive systems. Motion Swarms [Ngu+06] uses image processing techniques to produce a particle system based on audience movements. This particle system is then used to control a virtual beach ball. One of the main limitations of these systems is that it is difficult to identify and respond to the actions of an individual audience member. Maynes-Aminzade et al. however note that this is not necessary for many types of interaction. The illusion of interaction is what is important from the audience's perspective. Rather than use computer vision techniques for broad audience participation, Half Real gives each audience member a physical controller. This allows much more fine grained control of the interactivity than other techniques.

Half Real uses a spatial augmented reality system based on Shader Lamps [Ras+01] to project perspectively correct virtual content onto the physical set. Previously Shader Lamps has been used to create physical cartoon dioramas [RZW02] and life size immersive environments [Low+01] which can be modified in real time by the

user [RL01]. Superficially, the Half Real set is similar to the Cave Automatic Virtual Environment (CAVE) [CSD93]. However, where a CAVE provides an immersive virtual environment to the users inside the CAVE, the Half Real set provides a virtual environment for the audience. The actors placed in the virtual environment are part of the illusion.

## A.3 The Stage as a 3D Environment

Projectors are becoming more and more popular in performance art. In the past, projectors have been used for image projection and video playback. Whether they are used to project onto flat screens, or more complex objects on stage, 2D content has been created and used. This approach suffers from two main drawbacks:

1. The projected content is created for a specific projector. This makes it difficult to add additional projectors later in development, as new content needs to be created specifically for the new projector location.

2. Setup requires a tedious process of placing projectors in the correct locations, aligning them precisely, and performing keystone correction.

The approach taken in Half Real is fundamentally different. The entire set is modelled as a 3D scene (Figure A.2(a)), and SAR technology is used to illuminate the set. Rather than creating unique content for each projector, the content is created for the scene. This approach requires a 3D model of the set to be created. However, 3D models are becoming more common to aid in tasks such as pre-production lighting design.

The 3D scene based approach taken for Half Real provides several advantages over 2D:


(a) (b)

**Figure A.2:** The 3D model of the Half Real set (a), and physical version with projected textures (b).

1. Projected content needs to only be created once for the scene, regardless of how many projectors are used.

2. Projectors can easily be added as required.

3. Scenes can be previewed in 3D pre-production, before the physical sets have been built. This is similar to visualisations made in lighting design software.

4. Pre-show setup is much simpler. The projectors only need to be roughly placed, and the SAR calibration algorithm will calculate what content will be projected from each projector.

The use of a real-time spatial augmented reality system also brings new possibilities to the creative team. Tracking systems, cameras, and other sensors can be used to track actors' movements and other actions. Projected content generated procedurally, rather than pre-rendered video and animation, can be created to react to the data coming in from these sources. This enables much more dynamic projection effects to be created. In addition, because the projected content is controlled by a computer system, rather than using simple video playback, the performance does not need to follow a linear path. Half Real takes advantage of these possibilities by

tracking actors, using procedurally generated projected content, and an interactive performance where the audience votes on the direction the story takes.

### A.3.1 Creating Content

Once a 3D model has been created for the set, content can be authored. Texture maps for the 3D objects are used as guides, and the content to be projected is created in standard animation software.

Treating the entire set as a projected virtual world creates challenges for both content creators and live actors. As the walls are projected onto from the front, the actors inevitably cast shadows onto the walls. Ceiling height in venues meant high projector locations could not be used. Previous research [Sum+05] has shown that users are able to cope with shadows introduced in projected environments, although users prefer systems utilising dual projectors to eliminate total occlusion. However, instead of trying to reduce shadows, Half Real embraces the interplay between the live action and the virtual world. Virtual characters are rendered as silhouettes rather than realistic people. Actors' movements on stage were developed to enhance the effect given by their shadows.

Projecting content onto the walls and floor caused problems with content creation. For example, a bold, bright colour on the floor reflects onto the walls. Scenes that look good on a standard computer monitor could look dark and bland when projected, due to the limited colour gamut and brightness of the projectors, and ambient light. The interplay between projected light and stage lighting also caused problems. Stage lighting is necessary for the live actors, but drowns out the virtual environment. Actor blocking, projected content, and lighting design were iterated simultaneously to achieve the best balance. In some instances the best approach was to modify the projected content so that it, rather than stage lighting, could illuminate the actors. This is shown in Figure A.3.


(a) (b)

**Figure A.3:** Stage lighting could drown out the projected images, particularly when actors needed to be close to the walls. In some cases, using the projectors to light the actors was necessary, and the projected content was modified to allow this. These images show the original content (a), and the modifications made to light the actor (b).

## A.4 Interactivity in Half Real

Half Real is a live murder mystery theatre production, with the investigation conducted by the audience. At key points during the show, the audience votes on how the investigation proceeds. This results in each performance being unique, with the audience seeing only a part of the story. The whole story contains approximately 4x the material as is seen in any single performance.

Half Real's interactivity created a unique context for the audience, where they shifted from being spectators to participants. The inclusion of the dynamic GUI within the dramatic action, coupled with the ZigZag voting sequences, created a more informal mode than traditionally experienced during a performance. The audience was extremely aware of its collective decision making during the work. At the point of voting, the audience were given a permission to "interact", be it through conversing with nearby members, observing what other people choose on their ZigZag, or vocally respond when their preferred choice was lost by one percentage point.

Choices that were extremely close or extremely divergent were often the most

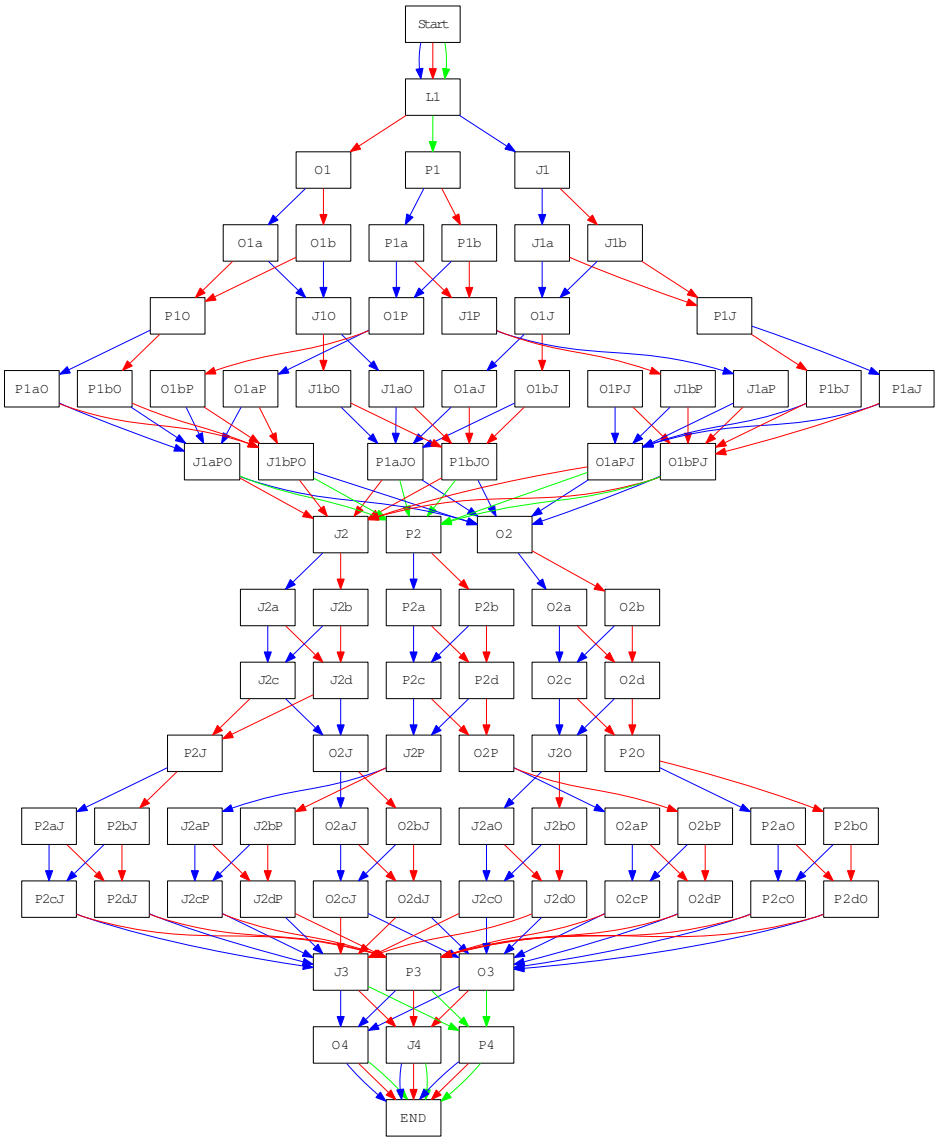



**Figure A.4:** The voting graph of Half Real. Vertices represent scenes, and edges represent audience votes.

fascinating, as the audience often became vocal in the manner they might be at a sporting match or playing a board game with friends. This participatory aspect of the performance altered the audience behaviour traditionally associated with modern theatre – where an audience quietly receives the creative authorship of playwright or director. Half-Real allowed the audience to playfully engage with the meta-theatrical context of a group democratically deciding what they'd like to see happen next, and also activated the audience to define what kind of theatrical expe-

rience they would have through their collective decision-making.

The interactivity also provided the audience with a "portrait" of their assumptions (i.e. who the killer was), with each node was a "litmus test" of the audiences suspicions, and each subsequent choice sought to build upon the audiences previous selection.

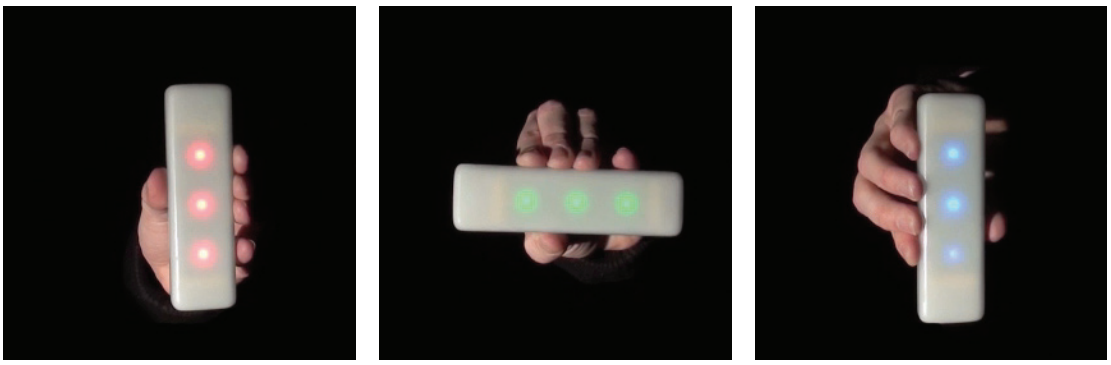
### A.4.1 Show Structure

To accommodate this interactivity and variation, the computer system represents the show as a Directed, Acyclic Graph (DAG), shown in Figure A.4. Each vertex is a scene, with audience vote results represented as edges. From the audience and actors' perspective, some scenes in the graph are duplicates. This means that some scenes can be seen in different orders. However, as the vote sequences into and out of particular instances of scenes are different, this looping is "unrolled" into a DAG.

The show is roughly structured into four acts. During each act, the story branches out as the audience decides which paths to take, before converging back to the start of the next act. The first act allows the audience to discover the origin of three pieces of evidence. Here, the suspects are introduced. In act two, the audience is able to investigate two of the three suspects in more detail. In act three a single character is investigated. Act four requires the audience to vote on who they believe was the murderer, based on the evidence uncovered in their investigation. During each scene, key pieces of evidence are uncovered. When this occurs, a vote option appears in the environment, attached to either the evidence or the character associated with the evidence. At the end of each scene, a voting sequence occurs to decide how the investigation proceeds.

### A.4.2 Voting

Each audience member votes using a ZigZag. The ZigZag is a remote control style device without buttons. Instead, the device contains a three axis accelerometer, three red/green/blue LEDs, batteries, and an Xbee[3] wireless module that handles the wireless communication to and from the computer system. The computer system coordinates the tallying of votes, and initiation of voting sequences. At the beginning of a voting sequence, the LEDs in the ZigZag begin flashing. The audience then has ten seconds to cast a vote. As the ZigZag is rotated, the colour of the LEDs changes to match one of the vote options, as shown in Figure A.5. After ten seconds have passed, the current colour is locked in and the vote is cast. A random seed minimises communication timing collisions between the 200 ZigZags. The vote results are then tallied and shown to the audience through the projection system. At the start of the show, a short tutorial on the devices was presented, before a "lesson" vote, that did not affect the path through the story.



(a) (b) (c)

**Figure A.5:** Audience members vote by orienting the ZigZag to select a colour, either red (a), green (b), or blue). Colours correspond to vote options that change colour at the beginning of a vote sequence.

[3]http://www.digi.com/xbee/

## A.5 Actor Tracking

A major advantage of representing the set as a 3D scene, rather than simply treating the walls and floor as 2D projection surfaces, is that virtual objects can be placed relative to any 3D location. This ability was used in Half Real to pin virtual information, such as vote options, to actors during the show (Figure A.6).

Actor tracking was implemented using a Microsoft Kinect[4]. The Kinect was chosen for the following reasons:

1. The Kinect operates in the infra-red spectrum. Therefore, tracking is not affected by changes in the projected images or stage lighting, which would affect visible light cameras.

2. Actors were not required to wear or carry sensors or fiducial markers, which would be unsuitable for a theatre performance.

3. The Kinect is quite a robust tracking system, making it particularly suitable to theatre.

4. The set of Half Real was small enough for the Kinect to be able to track almost the entire area.

5. The Kinect is inexpensive and readily available, so the hardware could be replaced if necessary while on tour.

The main downside to the Kinect is that it could not be used to track props on the set. It was not possible to project explicitly onto objects the actors carried or moved during the show. Skeletal tracking was not performed, as this would require a calibration step that would interrupt the performance.

[4]http://www.xbox.com/en-US/Kinect


**Figure A.6:** A vote option attached to an actor

### A.5.1 Attaching Information to Actors

The Kinect and the 3D scene representation provided accurate position information of actors as they moved about the space. However, like any SAR environment, virtual information had to be projected onto the available physical surfaces, such

as the walls of the set. The 3D actor positions needed to be translated to a suitable location on the walls behind them. This position needed to give the illusion of the virtual information being attached to the actors.

Half Real uses a ray-casting algorithm to calculate a suitable location for attached content. A ray is cast from the audience location, through the actor's location, and onto the wall behind them. This position was then adjusted to accommodate smooth transitions between the left and right walls. In addition, the location information from the Kinect was averaged over the most recent 30 frames. This gives the moving virtual information a visually appealing slow-in, slow-out animation effect. It also greatly reduced jitter when actors moved close to the range of the Kinect's tracking distance.

Like any projection system, view dependent rendering effects only appear correct from a single viewpoint. For Half Real, an ideal audience viewport was chosen at centre-stage, approximately 1/3 back from the front row. This location was chosen to give a good visual effect for the majority of the audience.

### A.5.2 Identifying Actors

While the Kinect is quite good at tracking people, it is not able to reliably identify them. Actors enter and exit the set many times throughout each performance of Half Real. Having actors tracked without interrupting the performance was an important goal when developing the projection system. Therefore, an explicit identification process could not be used. Instead, through the course of blocking during rehearsals, catch areas were identified in each scene. These catch areas are regions on the set that an actor would always walk through during the scene. Once the tracking system registered an actor passing through a catch area, that actor was associated with the correct virtual information.

In addition to catch areas, dead areas that never associated tracked objects in the

system were needed. The set of Half Real was not simply a static scene, it contained a door and window that actors could use and walk or climb through, and a chair that was moved about on stage. The detection algorithm in the tracker would sometimes incorrectly register these objects as actors. By marking these areas as "dead areas", the system would ignore these objects when associating virtual information. As with catch areas, the dead areas were specifically defined for each scene, as in some scenes actors moved into the range of the window or door when the virtual information needed to appear.

## A.6 Implementation Details

The show control system consisted of two subsystems. The Voting System, which managed the ZigZag devices and voting sequences, and the Projection System, which managed the virtual environment and actor tracking. This section describes the implementation details of these systems.

### A.6.1 Projection System

The projection system ran on a single computer containing an Intel Core i7-2600 CPU with 8GB of RAM. Two Nvidia Geforce GTX560 graphics cards drove the projectors and the operator screen. The projection software was implemented using our SAR software framework, written in C++ using OpenGL. The software ran on Ubuntu 11.04 Linux.

#### A.6.1.1 Resource Management

Half Real's projected content consisted of images, video files, and procedurally generated content, such as the vote options. In all, 38GB of assets, mostly video, were used during the show. A "level loading" approach was used to manage these as-

sets. Each scene was described in an XML file, which listed the assets required for the scene. A content manager was responsible for freeing data no longer required and loading new assets if required. Assets that were needed in consecutive scenes were reused, rather than reloaded. This approach was chosen due to the interactive nature of the show. The next scene, and therefore the assets required, would not be known until the audience voted. This meant that loading and unloading assets at predefined times was not possible.

### A.6.2 Pre-show Calibration

A pre-show calibration process is required for both the projectors and the Kinect. The projectors are calibrated using the algorithm described by Raskar et al. [Ras+01], which finds the intrinsic and extrinsic parameters of the projectors. This process involves finding landmarks on the set with a projected crosshair (Figure A.7). This process takes approximately 2-3 minutes per projector, and only needs to be performed when the projector locations change, such as moving to a new venue.

Once the projectors are calibrated, the Kinect must also be calibrated. By default, the Kinect returns 3D locations as a distance from the Kinect. A coordinate space transform must therefore be calculated to bring the Kinect's tracking data into the world coordinate space of the projection system. This calibration process involves finding crosshairs projected on the floor with the Kinect's visible light camera. 3D coordinates are obtained for these points from the Kinect and the transform can be calculated. Again, this process must only be performed if the projectors or Kinect are moved.

### A.6.3 Decoupling the Kinect

During rehearsals, a software bug inside the OpenNI framework was discovered that caused the projection software to crash intermittently. As the scenes in the

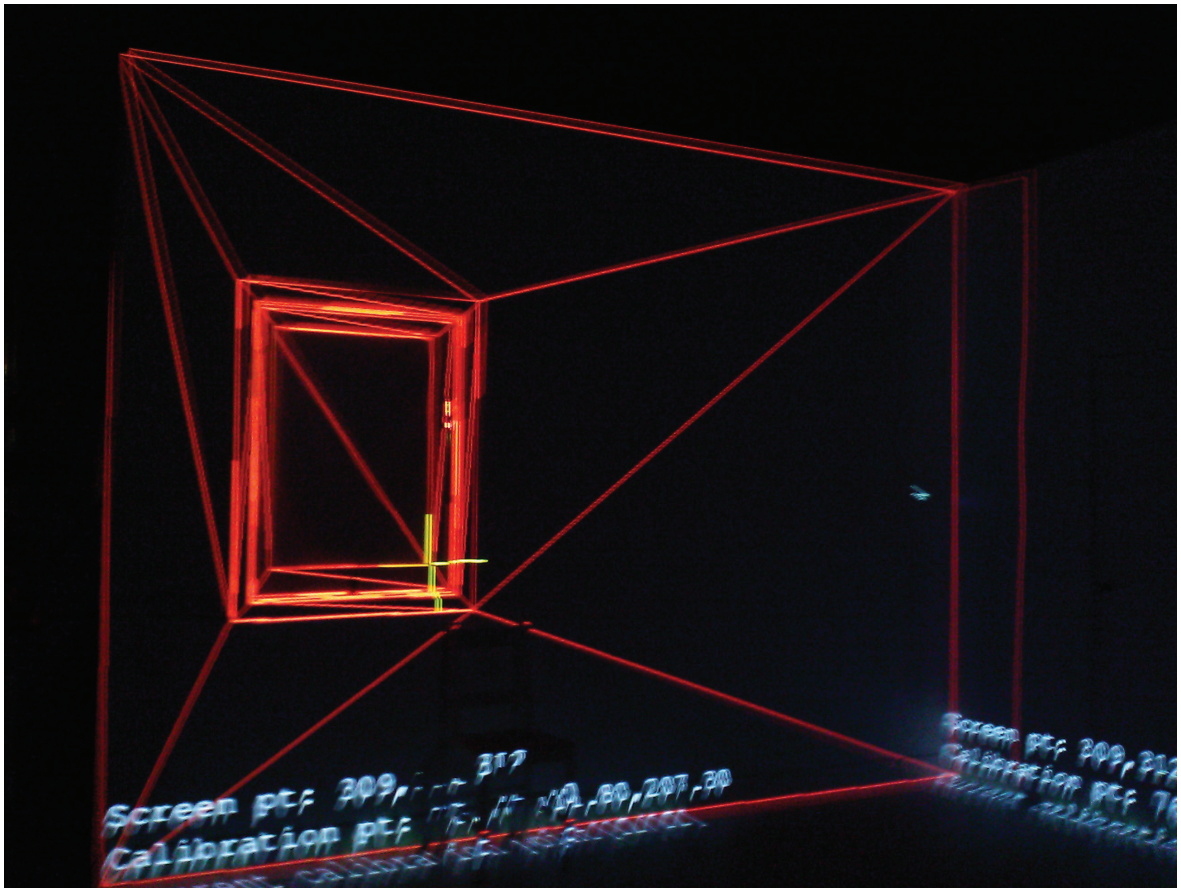


**Figure A.7:** The operator performs projector calibration by marking points on the set with a projected crosshair.

show are made entirely of projected artwork, the set would go dark if the software crashed. This was unacceptable for a live performance. The Kinect interface was rewritten to decouple the Kinect from the projection software.

The solution implemented involved two processes running simultaneously. The first process, tracker, handled interacting with the Kinect hardware via OpenNI. This process was run in a shell script that restarted the process whenever the software bug was encountered and the process crashed.

The second process was the main projection system. It accepted data from the Kinect via a TCP connection. This connection was automatically re-established as required. This solution resulted in a robust projection system. If the tracker process crashed, tracking would stop for approximately ten seconds, with the performance

continuing regardless. Virtual information would be reattached to actors once tracking restarted, based on their last known location.

## A.7 Summary

Half Real provides a case study in using spatial augmented reality in live theatre, enabling more dynamic projected content, and audience interactivity. Half Real successfully completed a tour of regional South Australia, before playing a three week, sold out season as part of the Melbourne Festival in 2011. This achievement can be seen as proof the technology and software developed was a success. However, as with any production there are lessons learned and room for improvement.

One of the major issues that had to be overcame was reliability and robustness. In Half Real, if the projection software crashed, the stage went dark. The system had to function correctly day after day, for extended periods of time. Decoupling subsystems was one of the most important factors in making the system robust. For example, it was important that the projection system kept running if the tracking system stopped responding. Another issue was sequencing content to be projected in each scene. The projection system used XML files for each scene. This effectively meant there was one scene description for the projection, and another for lighting and sound. In the future, the flexibility of the system could be improved by making it interoperable with existing stage management software, such as QLab[5], which would reduce the duplication, and make modifying the sequences of projected content much easier.

While Half Real has made an important step in using SAR for interactive performance art, there are many more possibilities to be explored. For example, using projectors to simulate physical light sources, such as follow spot lights that automatically track the actor. Or, using the projectors to project directly onto the actors in

[5]http://figure53.com/qlab/

order to change their appearance. I would like to extend the technology developed for Half Real to create much more sophisticated dynamic projection environments for performance and interactive art into the future.

# B

# Attachments

A number of additional materials are provided to support this dissertation.

## B.1 CD-ROM

The attached CD-ROM provides electronic copies of the following materials:

- demonstration videos of the applications presented in this dissertation,
- an electronic copy of this dissertation in portable document format (PDF), and
- an archive of all the current author publications to date (PDF).

## B.2 Internet

The information provided on the CD-ROM can also be accessed online, along with updates to this project, from the following website:

- http://www.20papercups.net/phd-thesis