THE UNIVERSITY OF CALGARY

Proxemic Interactions in Ubiquitous Computing Ecologies

by

Nicolai Marquardt

A DISSERTATION
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA
JULY, 2013

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Proxemic Interactions in Ubiquitous Computing Ecologies" submitted by Nicolai Marquardt in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Supervisor, Dr. Saul Greenberg
Department of Computer Science, University of Calgary

Dr. Sheelagh Carpendale
Department of Computer Science, University of Calgary

Dr. Frank Maurer
Department of Computer Science, University of Calgary

Dr. Thomas Patrick Keenan
Faculty of Environmental Design, University of Calgary

External Examiner – Dr. Andreas Butz
Institut für Informatik, Ludwig-Maximilians-Universität

Date

Abstract

In this dissertation, I explore how the knowledge of people's and devices' spatial relationships – called *proxemics* – can be applied to the design of ubiquitous computing (ubicomp) interactions. Edward Hall's proxemics theory describes how people use spatial relationships – such as varying their distance or orientation – to mediate their interactions with other people around them. But in spite of the opportunities presented by people's natural understanding of proxemics, only a relatively small number of ubicomp installations incorporate proxemic information within interaction design. Therefore, my goal in this dissertation research is to inform the design of future proxemic-aware devices that – similar to people's natural expectations and use of proxemics – allow increasing connectivity and interaction possibilities when in proximity to people, other devices, or objects. Towards this goal, I explore how the fine-grained knowledge of proxemic relationships between the entities in small-space ubicomp ecologies can be exploited in interaction design. In particular, I provide the following three major contributions:

First, I operationalize proxemics for ubicomp interaction with the Proxemic Interactions framework that serves to guide the design of ubicomp applications. The framework describes how designers can consider fine-grained proxemic information to mediate people's interactions with digital devices, such as large digital surfaces or portable personal devices. I identify five key dimensions of proxemic measures (distance, orientation, movement, identity, and location) to consider when designing proxemic-aware ubicomp systems. I also identify the gradual engagement design pattern as one particular strategy that allows designing system interactions that move from awareness, to reveal, to interaction.

Second, I design the *Proximity Toolkit* allowing ubicomp developers to rapidly prototype proxemic-aware ubicomp systems. The toolkit simplifies the development process by supplying higher-level information about proxemic relationships between the entities in ubicomp ecologies through an event-driven API and visual inspection tools.

Third, I explore the design of three case studies of proxemic-aware systems that react continuously to people's and devices' proxemic relationships. The case studies explore the application of proxemics in small-space ubicomp ecologies by considering first *person-to-device*, then *device-to-device*, and finally *person-to-person & device-to-device* proxemic relationships. Together, they validate the toolkit's versatility and the application of the Proxemic Interactions framework.

Publications

Materials, ideas, and figures from this dissertation have appeared previously in the following publications. On the following pages I provide a written and visual summary of which material is used in the thesis chapters.

Conference Publications

Marquardt, N., Ballendat, T., Boring, S., Greenberg, S. and Hinckley, K. (2012) Gradual Engagement between Digital Devices as a Function of Proximity: From Awareness to Progressive Reveal to Information Transfer. In *Proceedings of the ACM Conference on Interactive Tabletops and Surfaces – ACM ITS 2012.* (Boston, MA), ACM, pages 31-40.

Marquardt, N., Hinckley, K. and Greenberg, S. (2012) Cross-Device Interaction via Micro-mobility and F-formations. In *Proceedings of the ACM Symposium on User Interface Software and Technology – ACM UIST 2012.* (Cambridge, MA), ACM, October 7-10, pages 13-22.

Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011) The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the ACM Symposium on User Interface Software and Technology – ACM UIST 2011.* (Santa Barbara, CA, USA), ACM, October 16-18, pages 315-326.

Ballendat, T., **Marquardt, N.** and Greenberg, S. (2010) Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In Proceedings of the ACM Conference on Interactive Tabletops and Surfaces – ACM ITS 2010. (Saarbruecken, Germany), ACM, 10 pages, November 7-10.

Journal and Scientific Magazine Articles

Marquardt, N. and Greenberg, S. (2012) Informing the Design of Proxemic Interactions. In *IEEE Pervasive Computing*, 11, 2. April 2012. Joe Paradiso, Trevor Pering, Albrecht Schmidt, Eds., pages 14-23. © 2012 IEEE.

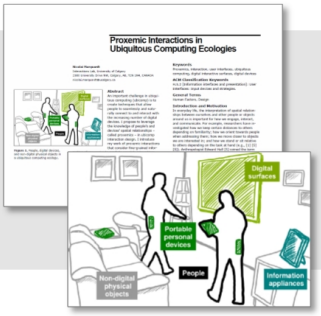
Reprinted with permission. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Greenberg, S., **Marquardt, N.**, Ballendat, T., Diaz-Marino, R. and Wang, M. (2011) Proxemic Interactions: The New Ubicomp? In *ACM Interactions*, 18(1):42-50. ACM, January-February. Invited cover story.

Extended Abstracts

Marquardt, N. (2011) Proxemic Interactions in Ubiquitous Computing Ecologies. In *CHI Extended Abstracts: ACM CHI Doctoral Symposium.* (Vancouver, BC, Canada), ACM, 4 pages + poster, May 7-12.

Marquardt, N. and Greenberg, S. (2010) Applying Proxemics to Mediate People's Interaction with Devices in Ubiquitous Computing Ecologies. In *Doctoral Symposium at ACM Conference on Interactive Tabletops and Surfaces – ACM ITS 2010.* (Saarbruecken, Germany), 4 pages, November 7-10.

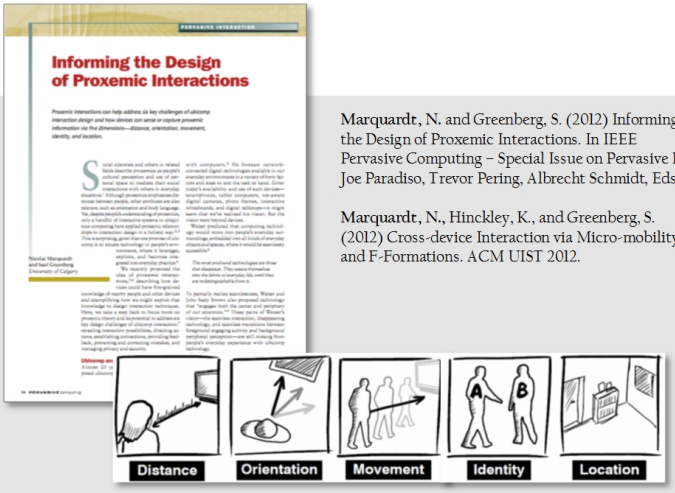


Marquardt, N. (2011) Proxemic Interactions in Ubiquitous Computing Ecologies. In ACM CHI 2012 Extended Abstracts: Doctoral Symposium. ACM, May 7-12.

Marquardt, N. and Greenberg, S. (2010) Applying Proxemics to Mediate People's Interaction with Devices in Ubiquitous Computing Ecologies. In Doctoral Symposium at ACM ITS 2010, November 7-10.

PART I

PROXEMICS IN UBIQUITOUS COMPUTING

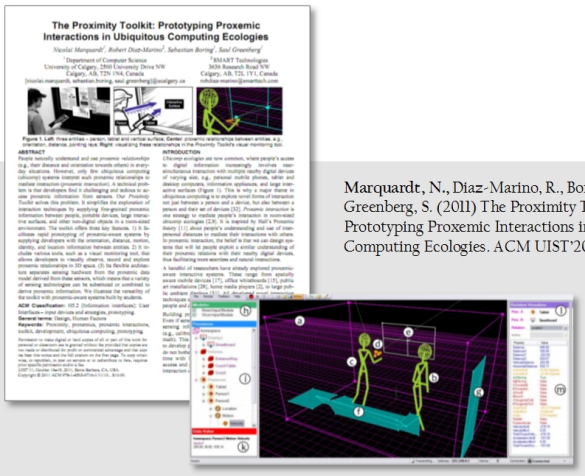


Marquardt, N. and Greenberg, S. (2012) Informing the Design of Proxemic Interactions. In IEEE Pervasive Computing – Special Issue on Pervasive I/O. Joe Paradiso, Trevor Pering, Albrecht Schmidt, Eds.

Marquardt, N., Hinckley, K., and Greenberg, S. (2012) Cross-device Interaction via Micro-mobility and F-Formations. ACM UIST 2012.

PART II

RAPIDLY PROTOTYPING PROXEMIC INTERACTIONS



Marquardt, N., Diaz-Marino, R., Boring, S., Greenberg, S. (2011) The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. ACM UIST'2011.

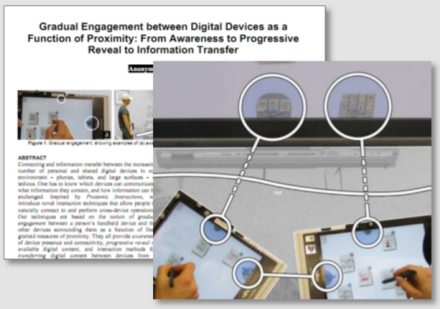
PART III

Exploiting Proxemics in Ubicomp Ecologies

7. Person/People-to-Device Proxemic Interactions

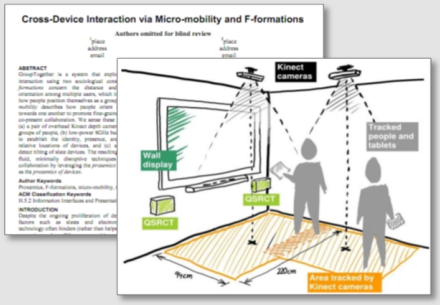
Ballendat, T., **Marquardt, N.** and Greenberg, S. (2010) Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. ACM ITS 2010.

8. Device-to-Device Proxemic Interactions



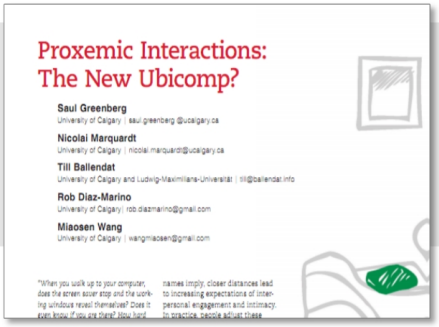
Marquardt, N., Ballendat, T., Greenberg, S., Hinckley, K. (2012) Gradual Engagement between Digital Devices as a Function of Proximity: From Awareness to Progressive Reveal to Information Transfer. ACM ITS 2012.

9. Considering Person-to-Person and Device-to-Device Proxemics



Marquardt, N., Hinckley, K., and Greenberg, S. (2012) Cross-device Interaction via Micro-mobility and F-Formations. ACM UIST 2012.

10. The Proximity Toolkit in Action



Greenberg, S., **Marquardt, N.**, Ballendat, T., Diaz-Marino, R., Wang, M. (2011) Proxemic Interactions: The New Ubicomp?. ACM Interactions, 18(1):42-50. ACM, January-February. Invited cover story.

11. Conclusion

Acknowledgments

My research and this dissertation would have been an impossible endeavour without the support and encouragement of many people.

First, I would like to thank my advisor Saul Greenberg: I am very grateful for all your support, encouragement, and guidance throughout the years of my research journey. I realize more and more how fortunate I am to have you as my mentor and friend. It is impossible to list everything I learned from you in the past years – not only about research and teaching. But what I do hope and strive for is to be as good a supervisor for my future students as you have been for me. Thank you!

I would like to thank the members of my committee: Sheelagh Carpendale, Frank Maurer, Jonathan Sillito, Tom Keenan, and Andreas Butz: thank you for your constructive and supportive feedback that informed my research.

I was very fortunate to work with Ken Hinckley at Microsoft Research further exploring proxemics in HCI. Thank you so much for all your advice and the wonderful collaboration – even beyond the end of my internship. Thank you to Andy Wilson, Hrvoje Benko, Bill Buxton, Michel Pahud, Andrew Bragdon, Merrie Morris, and Cati Boulanger for their support during my internship. I also thank the CML group at Microsoft Research Cambridge: Alex Taylor, Abigail Sellen, Richard Banks, Nicolas Villar, Steve Hodges, Stuart Taylor, Shahram Izadi, Tim Regan, James Scott, Ken Wood, and all my fellow interns during the three internships in Redmond and Cambridge.

I would like to thank my collaborators on the projects related to this dissertation, in particular: Rob Diaz-Marino, for all his work and an amazing collaboration of developing the Proximity Toolkit; and Till Ballendat, for the fascinating joined work of exploring proxemics in ubicomp ecologies. I would also like to thank Sebastian Boring and Miguel Nacenta for their help and support during their time as postdocs in the Interactions Lab.

I also thank all the students of the ubicomp graduate courses in 2010 and 2012 for allowing me to survey their amazing projects built with the Proximity Toolkit.

Thank you to all my friends and colleagues at the Interactions Lab. You are a fantastic group and I truly enjoyed getting to know all of you. I want to thank Helen He for helping me through the up and downs of a PhD, Petra Isenberg for being my lab buddy when I started in the iLab, Jim Young for enjoyable debates at the lunch table, and Charlotte Tang for always being there when I needed someone to talk to. I want to thank all my other friends in the iLab that made my time there such a wonderful experience: Bon Aseniero, Anthony Chen, Chris Collins, Marian Doerk, Matthew Dunlap, Cheng Guo, Jonathan Haber, Uta Hinrichs, Elaine Huang, Tobias Isenberg, Joe Kiemer, Ricardo Jota, Paul Lapides, David Ledo, Lindsay MacDonald, Andre Miede, Carman Neustaedter, Ehud Sharlin, Gerry Straathof, Tony Tang, Edward Tse, Steve and Amy Voida, Jagoda Walny, Martin Weigel, and all other former and current iLabbers. I also thank everyone in the CPSC department, in particular Claudia Maurer, June Au Yeung, Camille Sinanan, Susan Lucas, Beverley Shevchenko, and Mary Lim for helping me through the jungle of university administration.

My research was made possible with the support of: Alberta Innovates – Technology Futures (AITF), NSERC, iCORE, SMART Technologies, SurfNet, Microsoft Research, and the Department of Computer Science at the University of Calgary.

I would like to thank my family for all their support. Thank you to my parents for all their encouragement, providing help when I needed it, and their support for my graduate studies. Finally, and most importantly, I would like to thank my wife Kristin: thank you for being so patient, understanding, and for brighten up my days during the most stressful times in the past years. Thank you for your love and for believing in me!

Research Acknowledgments

Parts of my dissertation research were conducted in collaboration with other students and researchers in the Interactions Lab and at Microsoft Research. In this section I acknowledge and thank my collaborators for their support of my research:

First, the development of the Proximity Toolkit (Chapter 5 and 6) was done in collaboration with Rob Diaz-Marino, who was employed at the Interactions Lab for his work on the Proximity Toolkit. The following aspects of this research were done in collaboration with him (together with Saul Greenberg): concept of the toolkit architecture, programming of core toolkit components, programming of code examples, and documentation. The referenced toolkit publications were written by myself, as well as all later programming of the toolkit when Rob Diaz-Marino left after 2 years. One of the toolkit's plugin modules (OptiTrack) was developed in collaboration with the University of Konstanz (Roman Rädle and Stephan Huber).

Second, the work of two of the case studies of proxemic-aware systems (Chapter 7 and 8) was done in collaboration with Till Ballendat from the LMU Munich, who was a co-supervised visiting student of Saul Greenberg and myself. Part of the programing of the implemented systems was done collaboratively, but the majority of this programming work was done by Till Ballendat. The conceptual foundation of proxemic interactions, the interaction techniques, the Gradual Engagement Design Pattern, and the writing of both publications was done by me.

Third, the GroupTogether system (Chapter 9) was performed during my time as an intern at Microsoft Research in Redmond supervised by Ken Hinckley. The conceptual work, pre-study, programming, and evaluation was performed by me under Ken Hinckley's supervision.

Fourth, students of two Ubiquitous Computing graduate classes and other students of the Interactions Lab developed projects using the Proximity Toolkit. The selected projects that I describe as examples in Chapter 10 all include the attributions to the individual students.

Last, other researchers in the Interactions Lab advised me for several of my dissertation projects: this includes my supervisor Saul Greenberg, but also Sebastian Boring who was a postdoc in the Interactions Lab during that time.

My deepest thanks to all my collaborators and advisors; and to the funding agencies and companies supporting my research: Alberta Innovates – Technology Futures (AITF), NSERC, iCORE, SMART Technologies, SurfNet, Microsoft Research, and the Department of Computer Science at the University of Calgary.

To Kristin

Contents

List of Tables

List of Figures

List of Acronyms

2D	2-dimensional
3D	3-dimensional
AI	Artificial Intelligence
API	Application Programming Interface
AUI	Attentive User Interfaces
DMVC	Distributed Model-View Controller
GHz	Gigahertz
GPS	Global Positioning System
GUI	Graphical User Interface
HCI	Human-Computer Interaction
IR	Infrared
MVC	Model-View Controller
PARC	Palo Alto Research Center
PC	Personal Computer
QSRCT	Qualcomm Short Range Communication Technology
RFID	Radio Frequency Identification
TCP	Transmission Control Protocol
TUI	Tangible User Interface
Ubicomp	Ubiquitous Computing
WCF	Windows Communication Framework
WIMP	Windows, Icon, Menu, Pointer
WLAN	Wireless Local Area Network
WPF	Windows Presentation Framework

Chapter 1. Introduction

"When you walk up to your computer, does the screen saver stop and the working windows reveal themselves? Does it even know if you are there? How hard would it be to change this? Is it not ironic that, in this regard, a motion-sensing light switch is 'smarter' than any of the switches in the computer [...]?"

Bill Buxton "Living in Augmented Reality" (Buxton 1997)

Over the last two decades, Mark Weiser's (1991) vision of *Ubiquitous Computing* (ubicomp) as the next era of interacting with computers has increasingly become commonplace through the rising number of digital devices present in people's everyday life. *Ubicomp ecologies* are emerging (e.g., Figure 1.1), where people now use their portable personal devices (e.g., phones, tablets), interact with information appliances (e.g., digital picture frames, game consoles), and collaborate with large surfaces (e.g., digital whiteboards) within a given context. But Weiser's vision went beyond the mere *individual* devices – it predicted seamlessly accessible technologies of calm computing that "*weave themselves into the fabric of everyday life, until they are indistinguishable from it*" (Weiser 1991) and "*engage both the center and periphery of our attention*" (Weiser and Brown 1996). There are, however, still considerable problems that make interaction with devices in such ubicomp ecologies far from seamless. Using multiple devices in *concert* is often tedious and requires executing complicated interaction sequences (Cooperstock et al. 1997).

For example, consider the digital ecology of the living room shown in Figure 1.1. While most of the present devices are network-enabled, actually interconnecting and transferring content between these devices is painful without extensive knowledge, and

it requires time to configure and debug. Even when devices are connected, performing tasks among them is usually tedious—for example, navigating through network and local folders to find and exchange files. In practice, this means that, from a person's perspective, the vast majority of devices are blind to the presence of other devices. What makes this even more problematic is that these devices are also blind to the non-computational aspects of the ubicomp ecology which may affect their intended use. For example, devices do not recognize *people*, such as if only a single person is interacting with the device or a group of people that could work collaboratively. They do not recognize *non-digital objects*, such as if a person sitting on a chair to watch a screen from a distance, or if the person holding any object in their hand that could determine the intended interaction with the device. And devices also do not recognize the *spatial layout of the environment* (e.g., position of walls or doorways), which could help to determine if another wirelessly connected device is in the same or an adjacent room, or to know when a person is entering the room through a door so the system can activate.

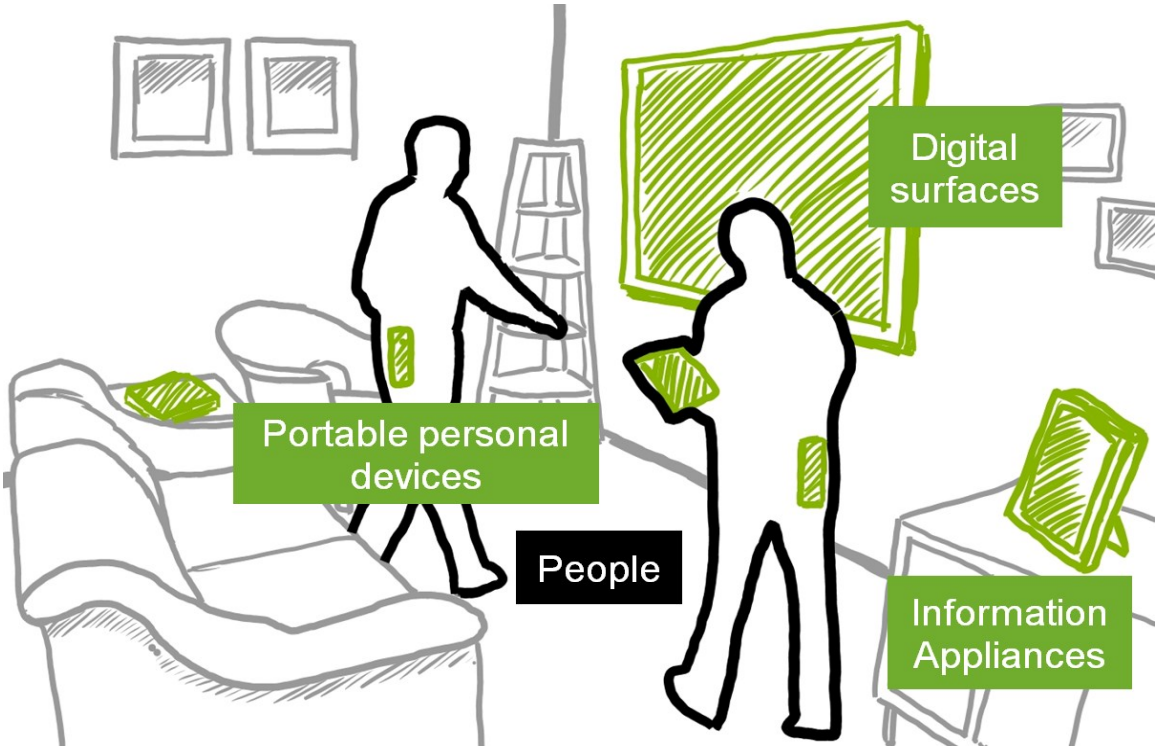


Figure 1.1 People, devices, and non-digital objects are part of a small space ubiquitous computing ecology.

To explore how the knowledge of such spatial relationships between people and the devices or objects around them could be leveraged in ubicomp interaction design, we first need a better understanding about how people use the space around them. A seminal theory analyzing and describing people's use of interpersonal space when interacting with others is Edward Hall's *proxemics*, introduced here but presented in more detail in Chapter 3.

Figure 1.2 People often implicitly adapt proxemic variables (e.g., distance or orientation) when interacting with others, as shown in these small group formations during conversations.

1.1 Proxemics

In everyday life, the spatial relationships between ourselves and other people or objects around us are important for how we engage, interact, and communicate. People often use changes of spatial relationships – such as distance or orientation – as an implicit form of communication. For instance, we keep certain distances to others depending on familiarity, we orient towards people when addressing them (e.g., see the informal circles of collaboration in Figure 1.2), we move closer to objects we are interested in, and we stand or sit relative to others depending on the task at hand. Proxemics – a term coined by anthropologist Edward Hall – is one of the seminal theories about people's perception

and use of interpersonal distances to mediate their interactions with other people (Hall 1966).

Hall's studies revealed patterns in how certain physical distances correlate to social distance when people interact. Other observations further refined this understanding of people's use of spatiality. For example, spatial features of the environment (e.g., location of walls, doors, furniture) influence people's use of proxemics. A person's orientation relative to others is another driving factor in how people greet and communicate with one another. Overall, proxemics mediates many aspects of social interaction. For example, it influences casual and serendipitous encounters (Kraut et al. 1988), is a nuance in how people greet one another (Kendon 1990), and is a major factor in how people arrange themselves for optimal small group collaboration via spatial-orientational maneuvering (Sommer 1969; Kendon 2010).

1.2 Proxemics Applied to Ubicomp Interactions

My thesis is that we can leverage information about people's and devices' fine-grained proxemic relationships for the design of novel interaction techniques in ubicomp ecologies.

To address this thesis, the overarching goal of this dissertation is to inform the design of future proxemic-aware devices that – similar to people's natural expectations and use of proxemics – allow increasing connectivity and interaction possibilities when in proximity to people, other devices, or objects. Towards this goal, I explore how the fine-grained knowledge of proxemic relationships between the entities in small-space ubicomp ecologies (people, devices, objects) can be exploited in interaction design (e.g., see example Figure 1.3).

I am not the first one to consider spatial information in ubicomp interaction design. My research directly builds on earlier explorations of considering proxemic relationships to drive people's interactions with technology (most importantly earlier work by Vogel and Balakrishnan, 2004; Ju et al., 2008). I am interested in further exploring the application of proxemics to ubicomp interaction design, as despite the contextual rich information

of proxemics and the opportunities presented by people's natural understanding of them, so far only a relatively small number of research installations incorporate knowledge about spatial relationships within ubicomp interaction design. Of those systems that do, most do not yet consider the fine nuances of distance, orientation, movement, location, and identity in people's and devices' proxemic relationships.

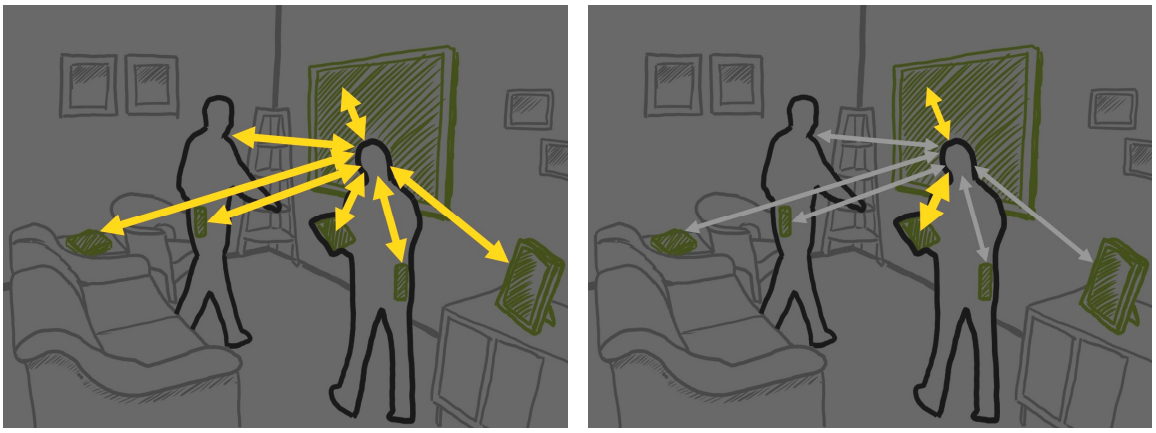


Figure 1.3 Interactions in ubicomp ecologies (cf. to Figure 1.1): (left) many possible interaction possibilities around a person, where (right) knowledge about proxemic relationships can be leveraged to identify devices more likely for possible interactions.

Towards my overarching goal of exploring the potential of applying proxemics in ubicomp interaction design, I focus in particular on the study of applying proxemics to interactions in *ecologies of people and devices* in *small space ubicomp environments* (i.e., small to medium sized indoor rooms, such as a living room at home or meeting room in the office). In my research, I explore how we can exploit the fine-grained knowledge of proxemic relationships (which I operationalize in Chapter 3 as distance, orientation, movement, location, and identity) between people, digital devices, non-digital objects, and the surrounding environment to mediate ubicomp interactions. In short, my research provides the following three major contributions:

(1) the operationalization of proxemic theories for ubicomp interaction design in the framework of *Proxemic Interaction*;

(2) the design of adequate development tools (the Proximity Toolkit) that facilitate rapid prototyping and exploration of proxemic-aware ubicomp systems; and

(3) the design and implementation of three explorative case studies probing into the design space of proxemic interaction in small space ubicomp ecologies – and therewith applying the operationalized proxemic theories and validate the application in practice of the rapid prototyping toolkit.

In the remainder of this chapter[1], first I outline the research context and describe the audience for the results of this dissertation (§1.3). Then, I introduce the research problems that I am addressing (§1.4). Next, I outline the objectives and corresponding research methodology to address these problems (§1.5). I close with an overview of the dissertation (§1.6).

1.3 Research Context and Audience

The research context of this dissertation is illustrated in Figure 1.4. The dissertation topic lies in the area of *Human-Computer Interaction*, the study of people's interaction with computers and design of novel interface approaches. More explicitly, my research is in the area of *Ubiquitous Computing* – interactive systems comprising one or multiple computing devices situated in people's everyday environments. Within this area, my work is targeted towards investigating the application of the social science theory of proxemics to the design of device interactions in the ecology of *small space ubicomp environments*. These small spaces are, for instance, rooms inside of a building, with small groups (2-4) of people being co-located in the same environment. These spaces define the context of what people do within them. In these small space ubicomp environments,

---

[1] Portions of this chapter are published as:

Marquardt, N. (2011) Proxemic Interactions in Ubiquitous Computing Ecologies. In CHI Extended Abstracts: ACM CHI Doctoral Symposium. (Vancouver, BC, Canada), ACM, May 7-12.

Marquardt, N. and Greenberg, S. (2010) Applying Proxemics to Mediate People's Interaction with Devices in Ubiquitous Computing Ecologies. In Doctoral Symposium at ACM Conference on Interactive Tabletops and Surfaces – ACM ITS 2010. (Saarbruecken, Germany), November 7-10.

I investigate how the knowledge of fine-grained proxemic relationships between people's and devices' can inform interaction design to mediate people's interactions.

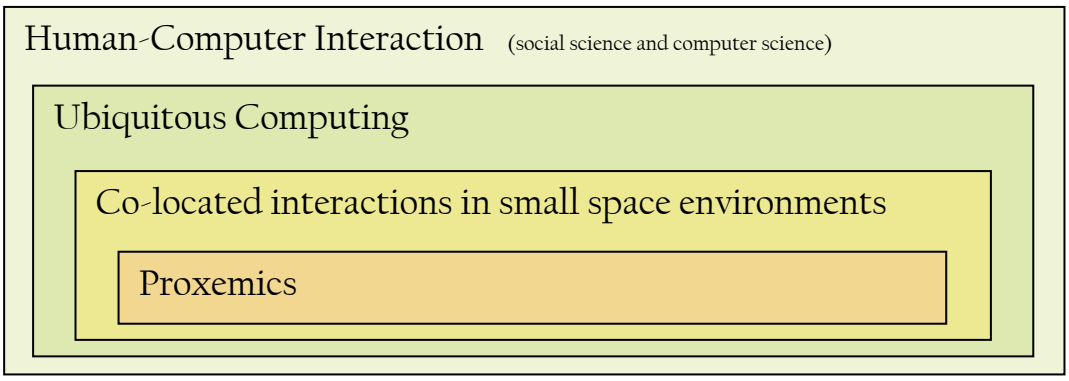


Figure 1.4 Research context.

In order to keep the scope of this dissertation manageable, I restrict my research to small space indoor ubicomp environments, as exemplified by living rooms in the home, and meeting rooms in the office. This excludes ubicomp systems designed for public spaces, building- and city-wide ubicomp deployments, or a person switching between those spaces. Furthermore, while considering a variety of essential dimensions for measuring proxemic relationships (as introduced in Chapter 3), my research will not consider many other dimensions influencing proxemic perception, such as cultural differences as discussed in Hall (1966) or Aiello (1987).

The primary audience for the work presented in this dissertation are ubicomp developers, researchers, and interaction designers. The operationalization of proxemics for ubicomp interaction design as a foundation and the insights of the three case study designs address the broader audience of ubicomp researchers, where my goal is to inspire and inform design process of building ubicomp systems. The toolkit design and technical details of building proxemic-aware applications address both developers of ubicomp applications, and designers that create toolkits for ubicomp development.

1.4 Dissertation Problems

My overarching long term goal is to explore proxemics in the context of interaction design in small space ubicomp ecologies. Within this goal I identify and pursue the following three research problems.

1. **We do not know how proxemic theories and measures of proxemic relationships can be applied to interaction design in ubicomp ecologies.**

   While the social science theories of proxemic theories describe and analyze people's perception and use to mediate human-to-human interaction, it is unclear how proxemics can be applied to person-to-device or device-to-device interactions. What is missing is a structured synthesis of relevant proxemic theories, their operationalization to interaction design, and the identification of essential dimensions of measured proxemic information that are important to allow systems to determine the proxemic relationships between people, devices, and objects present in ubicomp ecologies.

2. **We do not have adequate developer tools for rapidly prototyping and exploring proxemics aware interfaces in ubicomp ecologies.**

   Ubicomp applications based on proxemics will require sensing technology to track people's and devices' spatial relationships. Yet, such systems are difficult to build, which might be partially the reason for why proxemic relationships have so far been ill-explored in ubicomp design. It is difficult for developers to integrate sensing hardware (assuming such hardware is available). It is also difficult to access the sensed information from within the software in a form that can be readily applied to develop the interactive system. For example, a few toolkits exist that facilitate access to low level sensor data (e.g., Ballagas et al. 2003; Lee et al. 2004; Hartmann et al. 2006; Marquardt and Greenberg 2007). However, even with these toolkits, integrating the low level sensor input into proxemic-aware applications introduces many challenges for the developer, as it is often necessary to do complex mathematical 3D processing and interpretation of low level raw

sensor data into a higher, more usable form. These challenges prevent developers from rapidly prototyping, exploring, or refining proxemic ubicomp applications (Greenberg 2007).

3. **We do not know how people's interactions with devices in small space ubicomp ecologies can be mediated by the system's knowledge of proxemic relationships.**

   So far, only a handful of research projects have explored interaction techniques that consider proxemic relationships to mediate interaction. Their developed systems demonstrated the potential of proxemics when applied to interactions with devices; for example, for mediating collaboration on a digital whiteboard (Ju et al. 2008), grouping pairs of mobile phones for sharing content (Kray et al. 2008), or facilitating access of personal calendar information on large public displays (Vogel and Balakrishnan 2004). It is unclear, however, how devices in the richer ecology of multiple surrounding people, objects, and other devices can leverage fine-grained knowledge of proxemic relationships in order to mediate interactions in small space ubicomp ecologies such as a living room or meeting spaces.

1.5 Objectives and Methodology

My dissertation research is targeted to answer the question of how to leverage the system's knowledge of proxemic relationships to mediate interactions in the *complete small-space ubicomp ecologies* of people, objects, devices, and their surrounding environment. To make this tractable, I will focus on proxemic relationships between the following entities (illustrated in Figure 1.5):

- **People** (single person to small groups, i.e., 1-4 people)
- **Large interactive digital surfaces** (e.g., whiteboard)
- **Information appliances** (e.g., digital picture frames)
- **Personal portable devices** (e.g., phone, tablet computer)

- **Non-digital objects** (e.g., magazines, pens)
- **Fixed features** (e.g., walls) and **semi-fixed features** (e.g., furniture) of the environment

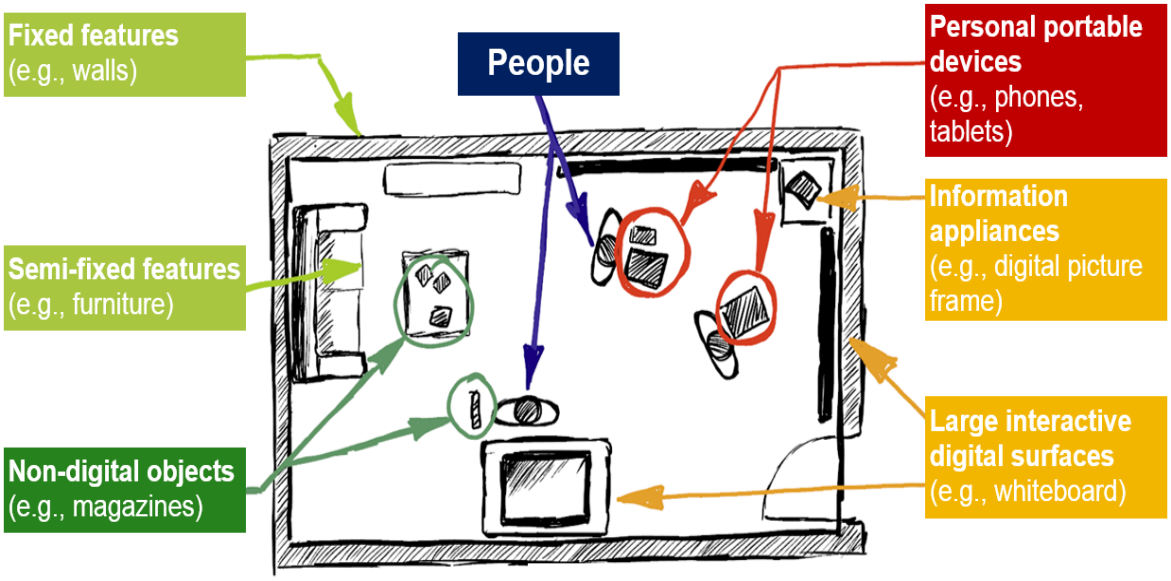


Figure 1.5 Ubicomp ecology with multiple people interacting with personal portable devices, information appliances, large digital surfaces, and non-digital objects.

In this context, I will work towards the following research objectives that address the three identified problems from Section §1.4:

1. **I will operationalize proxemics for ubicomp interaction design. I will describe a framework of Proxemic Interactions that informs the design proxemic-aware interactive systems.**

   I will identify and distil seminal theories of personal space and proxemics from the literature in environmental psychology, social psychology, anthropology, and ubiquitous computing to outline how proxemic information can be applied to the design of interaction techniques in ubicomp systems. I will translate proxemic theories for people's interactions with technology in ubicomp ecologies, with the goal to inform and guide the design process of ubicomp developers in form of a framework – called *Proxemic Interactions* – in order to let them create, invent, or discover novel interaction techniques. Similar to *generative theories* (Rogers 2004),

the Proxemic Interactions framework will enable "*practitioners to create or invent or discover something new*" by (a) allowing thinking in structured ways during the design process, (b) providing a clear vocabulary for discussing designs, and (c) allowing generating novel ideas through design dimensions and constructs (Shneiderman 2006; Beaudouin-Lafon 2004; Rogers 2004). The intention is not to provide a prescriptive set of guidelines or rules for the design, but to introduce the Proxemic Interactions framework as a first-order approximation of how to apply proxemics to interaction. I will discuss how this framework can inform the design of proxemic-aware ubicomp systems.

As part of the operationalization of proxemics for ubicomp, I will identify key dimensions of measured proxemic relationships that can be used to mediate people's interactions with ubicomp systems. Based on these identified dimensions, I will later illustrate the application of proxemic interactions to mediate in particular to *people-to-device*, *device-to-device*, and *person-to-person/device-to-device* interactions in ubicomp ecologies.

2. **I will design and implement rapid prototyping tools that make these proxemic measurements between people/devices/objects part of ubicomp ecologies easily accessible for developers.**

   Towards this objective, I will design and implement the *Proximity Toolkit*. The toolkit will particularly focus on facilitating the developer's access to essential aspects of proxemic relationships in ubicomp ecologies, such as:

   - Provide an application programming interface (API) that allows easy access to the accurate distance, orientation, movement, identity, and location information of people, objects, and devices in the ubicomp ecology
   - Support for people or devices entering *discrete* proxemic zones or for observing their *continuous* movements
   - Allow easy definition and later access to relationships between all entities

The above proxemic information is highly dependent on the sensor technology used. To allow for flexibility of the developed system and the particular tracking technology used, I will allow the integration of diverse tracking hardware along the low-/high-fidelity spectrum, such as marker-based motion capturing systems (high fidelity) or infrared depth cameras using structured light (low fidelity). The API will be consistent and independent from the underlying tracking technology.

Based on the Proxemic Interactions framework and essential dimensions (Objective 1) I will identify the most appropriate programming building blocks that can function as a language influencing programmer's creative thoughts (Greenberg 2007). I will validate the toolkit by designing and implementing proxemic-aware ubicomp applications (Objective 3) and by reflecting on the use of the toolkit by other developers creating applications. This will inform the further development and refinement of the Proximity Toolkit.

3. **I will design and implement proxemic-aware ubicomp systems that integrate concepts of proxemic interactions and explore the design space of proxemic interactions in ubicomp ecologies.**

   These developed ubicomp systems will illustrate the application of the Proxemic Interaction framework that is part of Objective 1, and also validate the creative potential of the development tools that are the outcome of Objective 2. I plan to develop three proxemic-aware ubicomp systems that illustrate how people can interact with the ecology of devices available in ubicomp environments: personal portable phones and tablets, interactive large displays, and information appliances. The three case studies and designed interaction techniques begin with a focus on *people-to-device* interaction, followed by *device-to-device* techniques, and finally variations of *person-to-person/device-to-device* interactions in ubicomp ecologies.

These three research objectives addressing the research problems are inter-related. I decided to address the research problems of proxemic relationships in ubicomp ecologies from three fronts: the operationalization of proxemics for ubicomp interaction, a toolkit

and developer tools for rapidly prototyping proxemic-aware systems, and case studies of proxemic interactions – particularly in small space ubicomp ecologies.

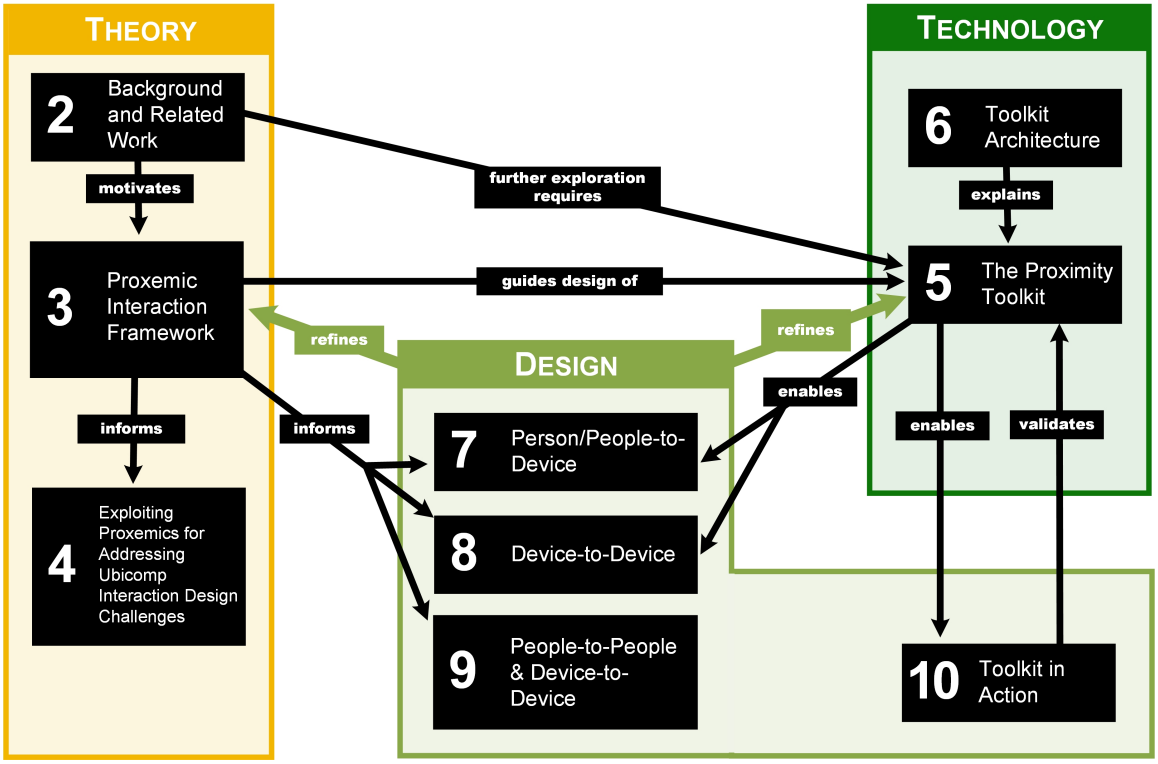


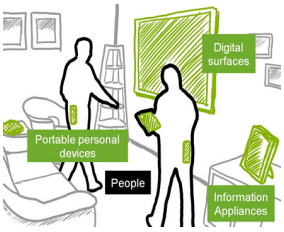
Figure 1.6 Overview of the dissertation chapters and their inter-relation.

1.6 Organizational Overview

The dissertation is structured into the following three major parts – all directly corresponding to the three research objectives. For literary convenience the structure of the chapters in this dissertation suggests a rather sequential exploration of proxemic interactions. In reality, all research segments heavily overlapped and were sometimes done concurrently, where results of each research segment affected the further process and refinements of other segments and parts of the thesis. Figure 1.6 shows a visual overview of the dissertation and the inter-relation between chapters.

PART I – Proxemics and Ubiquitous Computing

The first part of the thesis investigates the operationalization of proxemics for ubicomp interaction design.

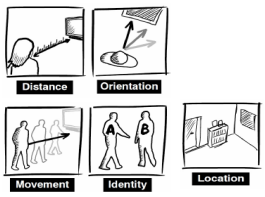


Chapter 2 – Background and Related Work

In this chapter I survey seminal research in ubiquitous computing and context-aware computing. I also review previous work considering spatial information about device, people, or both in ubicomp interaction design, which is relevant for the further exploration of proxemics in ubicomp ecologies.

Chapter 3 – Proxemic Interaction Framework

This chapter lays the foundation of proxemic interactions in ubicomp. I distil seminal theories of proxemics and personal space, operationalize proxemics for ubicomp in a framework of *Proxemic Interactions*, where I identify five key dimensions of proxemic measures most relevant for ubicomp interaction design.

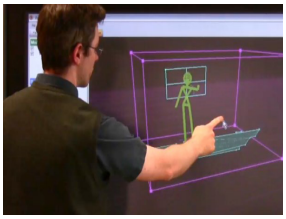


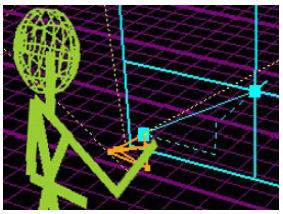
Chapter 4 – Exploiting Proxemics for Addressing Ubicomp Interaction Design Challenges

In this chapter I describe six ubicomp interaction challenges and discuss design strategies how to consider proxemics in system design to mitigate these problems. I refer to both earlier related ubicomp research described in Chapter 2, and to my own proxemic-aware case studies detailed in Chapters 7-9.

PART II – Rapidly Prototyping Proxemic Interactions

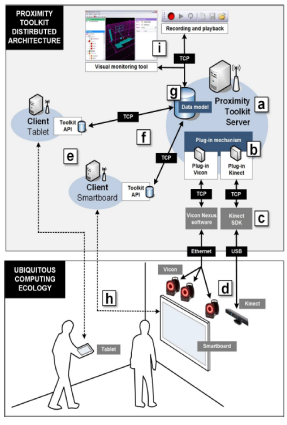
The second part introduces the required development tools for rapidly prototyping and exploring proxemic interactions in ubicomp ecologies.





Chapter 5 – The Proximity Toolkit

This chapter introduces the Proximity Toolkit that encapsulates the proxemic dimensions of Chapter 3 in a programming library. I explain how the toolkit's visual monitoring tool is a starting point for developers to explore proxemic relationships and their relevance for their envisioned designs. I then demonstrate – using a running programming example – how the event-driven API allows programmers to easily access the essential proxemic relationships between people, devices, objects, and the environment from within their code. By revisiting each of the five proxemic dimensions introduced in Chapter 3, I explain how they can be integrated into a proxemic-aware ubicomp application.

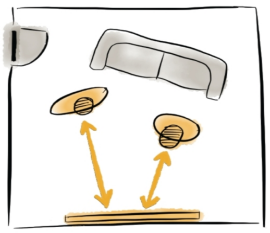


Chapter 6 – Toolkit Architecture

In this chapter I describe technical implementation of the toolkit architecture and the design of the programming library. I explain how the toolkit's flexible plugin architecture allows integration of several diverse tracking technologies along the high- and low-fidelity spectrum. This includes a discussion of decoupling the API from the tracking hardware, substitution of tracking technologies, handling uncertain input, and options for merging results of multiple tracking technologies in the toolkit.

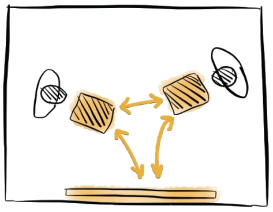
PART III – Exploiting Proxemics in Ubicomp Ecologies

The third part explores proxemics for interaction design in small space ubicomp ecologies. The three case studies illustrate the application of proxemic interaction concepts from Chapter 3, and demonstrate the versatility of the Proximity Toolkit (Chapter 5 and 6) for rapid prototyping and exploration of proxemics in ubicomp.



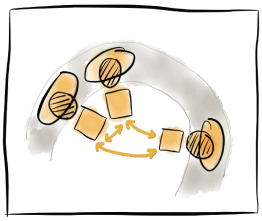
Chapter 7 – Person/People-to-Device Proxemic Interactions

In the first case study I focus on applying proxemics to mediate interactions of one person or multiple people with a large interactive display in a ubicomp ecology. I introduce a set of novel interaction techniques for people's interaction with applications on large interactive screens, where the system interaction is affected by people's spatial relationship to the display, to other objects, and the environment.



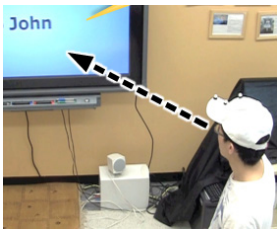
Chapter 8 – Device-to-Device Proxemic Interactions

In the second case study I consider proxemics to mediate device-to-device interaction – both between personal (e.g., tablets) and semi-public devices (e.g., digital whiteboards). Novel interactions techniques focus on mitigating cross-devices operations (e.g., transferring content between devices).



Chapter 9 – Considering Person-to-Person and Device-to-Device Proxemics

In the third case study I consider both person-to-person proxemics (through small group F-formations) and device-to-device proxemics in order to facilitate sharing of digital content between digital devices.



Chapter 10 – Proximity Toolkit in Action

Students explored the design of proxemic-aware ubicomp applications by using the Proximity Toolkit. Their creative designs illustrate the potential of proxemics in ubicomp and evaluate the versatility of our Proximity Toolkit. This chapter samples some of this work and discusses the student's use of the toolkit and proxemic interaction concepts.

Chapter 11 – Conclusion

The thesis concludes in Chapter 11 with a summary of the thesis contributions and pointers to future work.

PART I – PROXEMICS AND UBIQUITOUS COMPUTING

In this first part of the dissertation we[2] investigate the operationalization of proxemics for ubicomp interaction design. First, in Chapter 2 we survey related work in ubiquitous computing and context-aware computing, and review previous work considering spatial information for ubicomp interfaces. Next, in Chapter 3 we lay out the foundation of proxemic interactions in ubicomp, with a survey of seminal theories of proxemics and personal space, the operationalization of proxemics for ubicomp with the framework of Proxemic Interactions, and the identification of five key dimensions of proxemic measures most relevant for ubicomp interaction design. Last, in Chapter 4 we describe how to leverage proxemics in system design to mitigate six particular ubicomp interaction design challenges.

---

[2] The use of the plural 'we' in Chapters 2 to 10 refers to Nicolai Marquardt and the co-authors and collaborators acknowledged at the beginning of each of the individual chapters.

Chapter 2. Background and Related Work

"The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life, until they are indistinguishable from it"

Mark Weiser (1991)

This chapter[3] provides a two-part introduction to interactive ubicomp ecologies, which in turn frames the research proposed in Chapter 1. The goals of this chapter are threefold: introducing and motivating our research in ubiquitous computing system design; and demonstrating the potential of considering fine-grained proxemic information in ubicomp.

Section §2.1 introduces ubiquitous computing. Because ubicomp is a large area, this introduction quickly narrows to work that relates to the thesis focus of interactions in ubiquitous computing ecologies. In particular, it briefly surveys several seminal concepts in ubiquitous computing: embodied interaction and context awareness. Section §2.2 describes the state of the art in realizing interactive ubicomp ecologies. In particular it reviews those research projects that incorporate some kind of spatial or proxemic information to mediate people's interaction with ubicomp systems.

---

[3] Portions of this chapter are published as:

Marquardt, N. and Greenberg, S. (2012) Informing the Design of Proxemic Interactions. In *IEEE Pervasive Computing*, 11, 2. April 2012. Joe Paradiso, Trevor Pering, Albrecht Schmidt, Eds., pages 14-23. © 2012 IEEE.

2.1 Envisioning Ubiquitous Computing

Almost twenty years ago, in his *Scientific American* article Mark Weiser characterized the past, present and future of modern computing (Weiser 1991). He described how computer usage had already evolved from mainframe computing (one computer shared by many people) to personal computing (one person sits in front of one computer). He then predicted that the next major shift of how people would use computers would be towards ubiquitous computing, where each person has access to many digital devices (Figure 2.1). He foresaw that these digital technologies would be linked by networks, where devices would be available in a variety of form factors and sizes that would suit the task at hand. As Figure 2.1 illustrates, the number of devices available to people had – and would – increase over time.

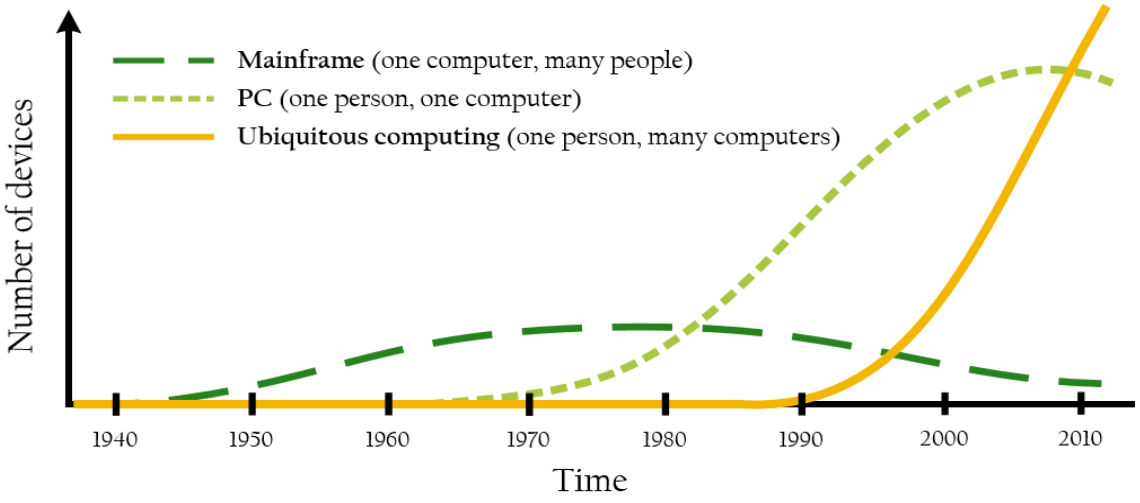


Figure 2.1 Graph representing the characteristics of the three major areas of modern computing: mainframe, personal computer, and ubiquitous computing[4].

Weiser and his colleagues at PARC designed a set of devices to illustrate his ubicomp concepts, and to serve as a sandbox for further exploration. Notably, he described device characteristics as arising in part from their quite different size scales: the yard-scale immovable large interactive *LiveBoards* (Figure 2.2, left), the foot-scale portable

[4] Source: reproduction of Mark Weiser's graph in (Want 2010) and on Mark Weiser's archived personal website: http://www.ubiq.com/weiser/UbiHome.html

notebook-sized *ParcPads* (Figure 2.2, bottom left), and the smaller handheld sized *ParcTabs* at the inch-scale (Figure 2.2, right). All were linked via a wireless network. Weiser's basic idea was that each device was designed and made readily available so that people could choose the kind of technology that best fit the task at hand, e.g., using the board for discussing digital content in a group, or using the pad to add private annotations to a document.

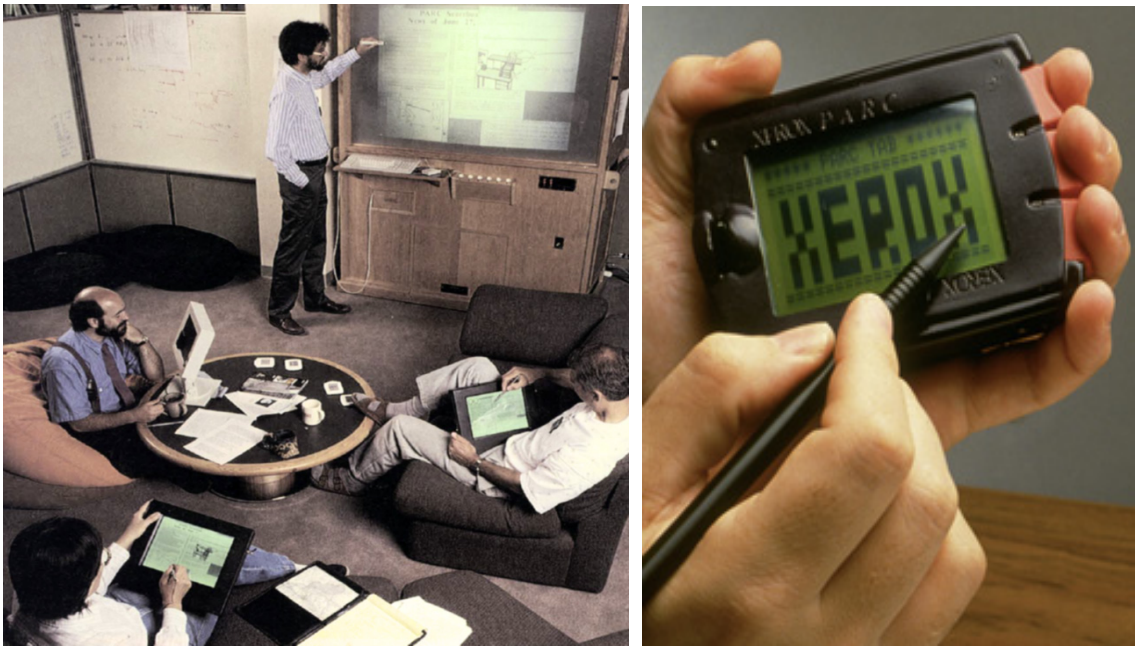


Figure 2.2 PARC's early test bed for ubicomp exploration: (left) LiveBoard in the background and a person using the ParcPad in the foreground; (right) the ParcTab handheld device[5].

It may seem that Weiser's vision has been realized given today's availability and use of devices such as smartphones, tablet computers, net-aware digital cameras, photo-frames, interactive whiteboards, digital tabletops, and so on. Yet this vision went beyond device availability.

Importantly, Weiser predicted the move of computing technology into people's everyday surrounding, embedded in all kind of everyday objects and spaces, and eventually

[5] Source: Scientific American article by Weiser (1991) and Xerox PARC, Brian Tramontana, available in the PARC newsroom media library: http://www.parc.com/newsroom/media-library.html

becoming invisible tools. The characteristic of invisibility in this context meant that the tool does not intrude on people's consciousness so that they could "*focus on the task, not the tool*" (Weiser 1994). A second key concept of ubicomp was to allow seamless interactions. As defined by Ishii et al. (1994), this describes both the need for seamlessness *(continuity)* with existing work practices, and for seamlessness *(smooth transitions)* between functional spaces. Later, Ishii and Ullmer (1997) summarize this concept as the seamless couplings between the physical and the digital world (or short: "*seamless couplings of bits and atoms*"). To partially realize seamlessness, Weiser and Brown proposed "calm" technology that "*engages both the center and periphery of our attention*" (Weiser and Brown 1996). That is, according to Weiser's vision, information technology should be accessible around people at the place *where* it is needed, and should reside invisibly in the background until the moment *when* it is required. Weiser emphasized that ubiquitous computing "*takes into account the natural human environment and allows the computers themselves to vanish into the background*" (Weiser 1991).

It is these parts of Weiser's vision – the seamless interaction with the disappearing and calm technology, the fluent transitions between foreground engaging activity and background peripheral perception – that is still missing from people's everyday experience with ubicomp technology. People carry cell phones between them. Desktop and laptop computers abound. Large displays and digitally controlled appliances are increasingly common place. Yet they largely exist as separate devices. For those that can be interconnected, the interface to do those connections are, at best, quite awkward. Still, progress has been made. As the next few sections show, researchers have developed refined and nuanced concepts of ubicomp.

2.1.1 Situating Computing in People's Everyday Environments

The vision of ubiquitously available technology in our environments and embodied interaction was highly influential for later technology explorations. For example, researchers further refined interaction concepts in so called multi-display environments (MDEs), where displays of diverse form factors allow access to (and interaction with) digital information in everyday environments. Previous research has shown how such an

"*ecosystem of displays*" (Terrenghi et al. 2009) can support various collaborative activities – mostly in office environments. Because of their device heterogeneity, the MDEs can be beneficial for group collaboration, for example by allowing the division and organization of tasks across devices, and choosing the type of device/screen that best fits to the task at hand. A core idea between these systems is that information can be easily moved between and across these displays in a near-seamless manner.

Figure 2.3 Multi-display environments: (top) i-Land, (bottom left) ARIS, and (bottom right) WeSpace[6].

[6] Source: i-Land by GMD-IPSI; ARIS (Biehl and Bailey 2004); WeSpace (Wigdor et al. 2009).

For example, *interactive landscape (i-Land)* by Streitz et al. (1999) was one of the early explorations of interactions in multi-display environments, where it considered people's interaction in the environment as a whole (Figure 2.3 top). As part of the interactive furniture they introduced chairs with integrated computers (*ComChairs*), multiple connected interactive walls (*DynaWall*), and tabletops for collaboration (*InteracTable*) (Streitz et al. 1999). A significant focus in their research was the seamless transfer of digital content between all entities. Within the three displays comprising the DynaWall, people could move information across the displays as if it were a single unit. Moving information between the interactive furniture was done differently. A person used physical objects identified by the system, where each object could act as a virtual container to represent digital data. To bring digital information from one device to another, a person placed the physical object next to any of the digitally-augmented furniture (on the so called *Bridge*) to associate information with the object. When bringing the object then to another device, the system would open the corresponding linked digital information on the screen (Streitz et al. 2002). A later addition to i-Land was the ConnecTable (Tandler et al. 2001), a pen-based small table. While each ConnecTable could be used individually, two people wishing to do tightly-coupled work could create a single homogeneous interactive digital workspace simply by abutting two ConnecTables together (as shown in the Figure), which would automatically connect the displays. The way ConnecTables leveraged their physical spatial relationship to establish a digital connection makes them a notable early example of proxemic interactions.

Later techniques further explored interaction challenges in MDEs. Nacenta et al. addressed the challenge of cross-display cursor movements with the perspective cursor technique (Nacenta et al. 2006), and later categorized related techniques used for cross-display interaction (Nacenta et al. 2009). One important idea of this work is to create perspective-corrected views on all surrounding devices, so that it becomes easier for a person to perform cursor movements across the different visible screens. Others introduced techniques for cross-device access and transfer of information. For example, Biehl and Bailey (2004) developed the *ARIS* interactive space windows manager that

allows people to easily relocate application windows across the multiple screens (Figure 2.3 bottom left) with the use of proxy icons around the screens. Wigdor et al. (2006) later refined this technique of cross-device access and control by using telepointers, multiple world-in-miniature visualizations, and other visualizations connecting the displays. More recently, WeSpace (Figure 2.3 bottom right) was designed for a more seamless integration of people's personal portable devices (e.g., notebook, tablets) into these multi-display ecologies (Wigdor et al. 2009). Terrenghi et al. (2009) present both a survey, and a design space taxonomy for these kind of multi-display environments organized along two axis: form factor scale and the degree of individual engagement.

MDEs, including the systems introduced above, provide good examples that address a particular niche in ubicomp technology, where it attempts to minimize the seams between the displays of multiple but different devices. Some of our own work in proxemic interaction, as described in later chapters, can be considered as contributing to MDEs, but – as with the ubicomp vision – is broader than that.

2.1.2 Embodied Interaction

Dourish's (2001b) theory of *embodied interaction* expands upon the ubicomp concept of situating technology in people's everyday environment. He brought together the core ideas of phenomenology theory, social computing, and tangible user interfaces, where he emphasized the importance of designing technology that exploits human skills and experiences that take place in their world (Dourish 2001b). Extending the ubicomp vision, the goal of embodied interaction is to build technology that is seamlessly integrated into people's everyday *practices*. People should not act *on* technology but instead *through* the technology, to perform their task at hand. The technology should be seamlessly integrated not only into the physical environment but also *embedded in people's social practices*. A fundamental concept of embodied interaction is therefore the technology's "*presence and participation in the world*" (Dourish 2001a), and the consideration of the associated meanings of the actual *place* in *space* where the interaction takes place (Harrison and Dourish 1996).

Dourish (2001b) emphasises that embodied interaction's notion of seamless integration requires bridging the gap between the digital and physical world. He makes specific reference to Ishii and Ullmer's (1997) concept of *tangible user interfaces*, or TUIs (Ishii and Ullmer 1997). TUIs integrate both digital input and output into graspable, physical objects. When well designed, these interfaces draw on people's natural skills and abilities when interacting with physical world objects. They emphasize that an important characteristic of TUIs is the seamless integration of technology with the physical environment (where they refer to Weiser's use of *invisibility* of technology), but also that the systems allow for "*seamless transition of the user's interaction between background and foreground information*". Under the covers, these systems use a variety of *sensors* (e.g., motion, touch) to gather input, and *actuators* (e.g., motors, solenoids) to manipulate the physical object to form output. For example, Ishii and his students developed *inTouch* to provide haptic interpersonal communication over distance (Brave et al. 1998). Each *inTouch* device comprises three cylindrical wooden rollers. When a person moves the rollers on one device (detected by position sensors), that motion is replicated on the other device (actuated by high precision motors). These devices are two-way, where both people can manipulate their device concurrently, yet feel the other's movement via force feedback. The concept of TUIs, however, goes beyond digitally connecting two physical devices, as TUIs can also mediate interaction between digital and physical entities. For instance, Ullmer et al. (1998) uses physical tokens to allow easy transfer of digital media between devices. Underkoffler and Ishii (1999) uses the placement of physical miniature buildings to control and augment a digital urban planning simulation on a tabletop display.

Dourish emphasizes other aspects of embodied interaction. He believes that embodied interaction recognizes multiple people, where he points to the field of Computer Supported Cooperative Work (CSCW). Furthermore, he emphasises how embodied interaction is a way of looking at the world: "*embodied interaction is not a technology or a set of rules. It is a perspective on the relationship between people and systems. The question of how it should developed, explored, and instantiated remain open research questions*" (Dourish 2001b).

Our own exploration of proxemics in ubicomp is a part of embodied interaction, in that we ground our designs on theories about people's implicit understanding and use of proxemics in everyday situations, and carefully translate these principles to ubicomp system design.

2.1.3 Context-Aware Computing

Context-aware computing relates to embodied interaction and ubicomp. The basic idea is that some kind of context-aware sensing method provides devices with knowledge about the situation around them. Using that knowledge, devices infer where they are in terms of social action, and then act accordingly to that context (Schilit et al. 1994).

Early research in context-awareness began with the integration of sensing capabilities, ultimately to give ubicomp systems sufficient information to recognize and react to situational context changes (Dey et al. 2001; Antifakos and Schiele 2002). An example of a context-aware device would be a mobile phone that can decide whether to ring depending on a person's current location (e.g., avoid ringing when in the cinema or a meeting in the office) (Coulouris et al. 2011). Often, context-aware systems infer the context information (e.g., location) from relatively simple measured properties such as noise levels, temperature, light, time, or acceleration. Strategies have been applied to fuse these diverse sensor measurements together in order to get more reliable results for inferring context (Antifakos and Schiele 2002). Schilit et al. (1994) identified three important aspects of context: "*where you are, who you are with, and what resources are nearby*". This extends the understanding of context to not only include location information of the person or device itself, but to consider the presence of people and resources in the environment as well.

ActiveBadge (Figure 2.4 left) was one of the early enabling technologies for the exploration of context- and location-aware computing concepts and how to apply them in practice (Want et al. 1992). The sensing aspect of the system is relatively simple: it determines the room people are in by transmitting and receiving signals via infrared to the tags – called ActiveBadges – that people wear (Figure 2.4). Yet even this basic information can be used to good effect. In particular, Schilit et al. (1994) later describe four novel

techniques that consider this information about people's location (e.g., as sensed by the Active Badge sensor, as shown in Figure 2.4 right) to drive interactions. First, the *proximate selection* technique filters nearby devices based on their location (e.g., showing all nearby devices that are in the same room as a person). Second, *contextual reconfiguration* change a device's configuration based on its current location (e.g., automatically making a nearby printer the default one). Third, with the *contextual information* technique the device's interface changes automatically when entering a new location (e.g., showing a list of discounted products when a person enters a store). Fourth, *context triggered actions* can be set to activate commands when entering a pre-defined location (e.g., reminding a person to look for a particular book the next time they are in a library). Overall, these interactions strategies summarize the core of interactions applied in many context-aware computing systems. As with ActiveBadges, many research projects primarily focus on *location* information (Oulasvirta and Salovaara 2009), either by sensing it directly or by inferring location information from other sensed properties.

A particular category of context-aware systems are *reactive environments* (Cooperstock et al. 1997; Buxton 1997). The fundamental idea of reactive environments is to design spaces that – by sensing people and device presence and movement – can infer the context of use and leverage that information to pro-actively perform certain system actions. Buxton illustrates the concept of reactive environments with an example that maps a simple sensed state in the physical world to control behaviours in the digital world (Buxton 1997). The *DoorMouse* sensor detects whether the door of an office is currently open or closed. This current state of the physical door (open/closed) is then directly mapped to the digital world, where it either allows or prevents a person to be interrupted with incoming messages or video calls on their computer. This simple design allows the technology to preserve some of the social protocols of the physical world (i.e., closing the door for not being disturbed) across to the digital realm. In another example, Cooperstock et al. (1997) built a reactive meeting room that automatically adapts the lights to a person's preference, displays a calendar overview, and reconfigures the audio and video equipment to address the presenter's needs. *Sentient computing* describes a similar concept for environments that reconfigures devices in reaction to the people

using the device (Addlesee et al. 2001). As one example, the system determines when a person is in close proximity to a desktop computer and automatically opens the last desktop session of that user – and closes it automatically when leaving (a concept introduced earlier as teleporting application states across devices Bennett et al., 1994).

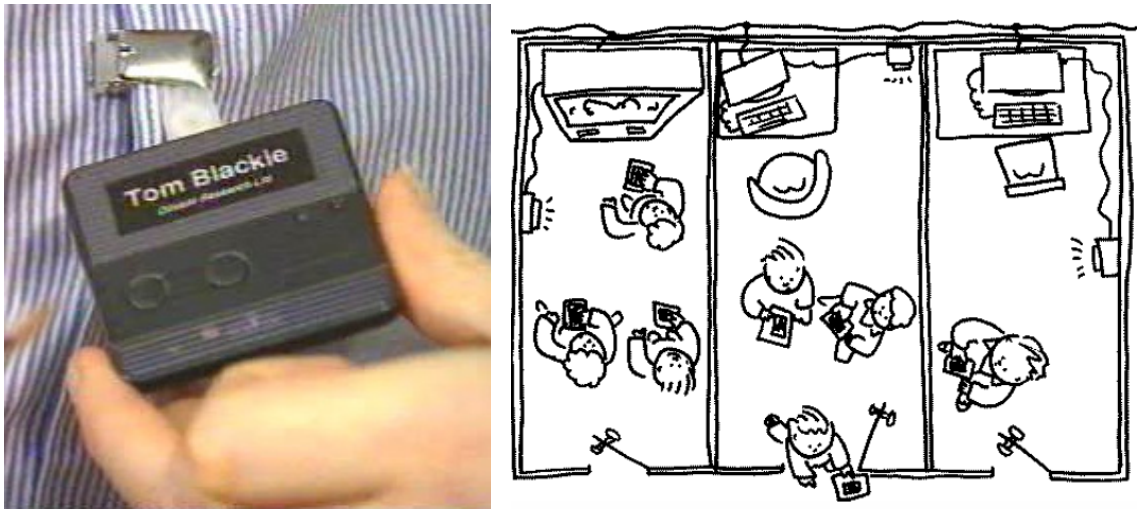


Figure 2.4 Context-aware computing: (left) Active Badge and (right) its application in practice: people's badges are detected to determine their presence in three different rooms[7].

The question of how to implement such reactive environments or context aware applications, and how to design adequate rules of behaviour, remains an active challenge of ubicomp research. Creating context-aware applications that match the environment and people's understanding of the situation is a critical yet highly difficult task for ubicomp developers. For example, Greenberg emphasizes that context is a dynamic construct that it is not always stable, and that similar-looking contextual situations may actually differ dramatically in their meaning to the people involved (Greenberg 2001). He states how this is partially due to the fact that not all information that defines a certain social context can be sensed by the system, such as: people's history of interaction, their emotions, or their current objectives (Greenberg 2001). Consequently, creating the rules of behaviour for context-aware systems (i.e., the rules that determine the system actions

[7] Source: Xerox PARC, Want et al. (1992), and Schilit et al. (1994).

based on sensed properties) is not just difficult, but sometimes even impossible (especially when errors cannot be tolerated). Greenberg warns not to "trivialize context", which could lead to inappropriate and frustrating applications. As partial solutions, he suggests that the rules of behaviour should avoid invoking risky system actions that the system should provide clear feedback of what it is doing, and that manual override should be possible in case the system gets it wrong. Others have also questioned the overly ambitious goals of using sensors to infer comprehensive context models fully describing social situations (e.g., Oulasvirta and Salovaara 2009; Rogers 2006). The challenges faced by context-aware system designers was even compared with the problems encountered in *strong* artificial intelligence research (AI), where the goal was to build intelligent computer systems that match or exceed human intelligence (Erickson 2002; Rogers 2006).

As we will further explain in Chapter 3, our research towards proxemic interactions relates in several regards to the research in ubiquitous computing, embodied interaction, and context-aware computing. We also envision systems that, in part, react pro-actively to sensed properties in the environment, and believe that these system designs can lead to more seamless and fluent interactions of people with their surrounding devices.

In the following section of this chapter we now sample related work that considered some form of spatial sensing (either devices, or people, or both) to drive people's interaction with their surrounding technology in such context-aware systems.

2.2 Ubicomp Systems Considering Spatial Relationships

We are not the first researchers that apply sensing of spatial relationships and proxemics to ubicomp system design. In this section, we review related ubicomp system designs that considered some form of spatial or proxemic information in the design of interactive applications. It is important to note that the primary focus of this section is to provide a structured general overview of the field, More detail will emerge in other chapters, where we will refer back to many of these systems later in Chapter 4 and throughout this

dissertation to discuss how these systems considered particular nuances of proxemic information in interaction design.

Because this dissertation focuses on systems that operate in indoor spaces ubicomp environments, we exclude broader area ubicomp deployments, such as interactive systems using Global Positioning System (GPS) receivers to determine the location of a person in a city (which was, for example, an integral part of the CyberGuide location-aware tour guide by Abowd et al. 1997). Most of the systems and interaction techniques we do survey in this section focus on a particular subset of the entities comprising ubicomp ecologies (e.g., interactions between devices, or between one person and a device). We see these works as providing fundamental building blocks for creating interaction designs considering the full ubicomp ecology.

This survey is structured into three major parts as shown in Table 2.1. The first part (Table 2.1 a) lists related work that considers the spatial relationships of devices. From bottom to top the systems or techniques are ordered according to their fidelity of tracked spatial information: detecting device presence at discrete distances (Table 2.1 b), continuous distance between devices (Table 2.1 c), or continuous distance and orientation (Table 2.1 d). The next part (Table 2.1 e) lists systems that sense people's presence. From left to right the projects are again ordered according to their tracking fidelity: detecting people's presence at discrete distances (Table 2.1 f), continuous distance (Table 2.1 g), or continuous distance and orientation (Table 2.1 h). Finally, in the third part we survey projects considering the spatial relationships of both people and devices in the full ubicomp ecology.

2.2.1 Sensing Devices

A major problem in Ubicomp is how to control the inter-connectivity of devices. This is especially problematic for mobile devices that may appear and disappear over time in an unpredictable manner, and that may not be known to the system before its first appearance. Consequently, various researchers have developed methods that involve sensing the close proximity of one device to other device(s) to mediate the establishment

of inter-device connections and (typically) to then transfer digital content between these devices over that connection[8].

This section reviews such device-to-device work, as summarized in part (a) in Table 2.1. The review progresses from devices that just sense each other's presence at discrete distances (Table 2.1 b), to those that recognize continuous distances (Table 2.1 c), and finally to those that sense and react to devices continuously changing distance and orientation (Table 2.1 d).

Sensing devices' presence at discrete distances (Table 2.1 b). A first class of techniques uses one or multiple discrete spatial zones – which often depends on the sensing technology used – to both initiate connectivity and to mediate the information exchanged between devices. A connection is automatically triggered when the spatial regions between devices overlap, i.e., to trigger the presence of one another. For example, Want et al. introduced a method that lets a device react to the presence of nearby devices or non-digital physical objects (Want et al. 1999). By attaching RFID tags to books, paper, or watches, a digital device equipped with an RFID reader is able to trigger certain activities as soon as these tagged objects come into sensor range. Similarly, the *Siftables* (small micro displays) detect proximity of other nearby devices when stacking them in a pile or placing them next to each other (Merrill et al. 2007). These techniques are powerful for connecting devices that are in very close proximity or – like in many cases – are even directly touching one another. Other techniques introduced techniques for sensing devices presence from a larger distance. For example, Rekimoto et al. (2003) combined RFID and infrared for establishing seamless device connectivity. Swindells et al. (2002) introduced a technique that worked from a larger distance, where he applied it to the *gesturePen* for initiating remote pointing for device selection (i.e., by pointing the pen directly at a device selects it).

[8] These approaches do require some form of limited *a priori* connectivity to coordinate this recognition, perhaps between devices or as mediated by a cloud network synchronization.

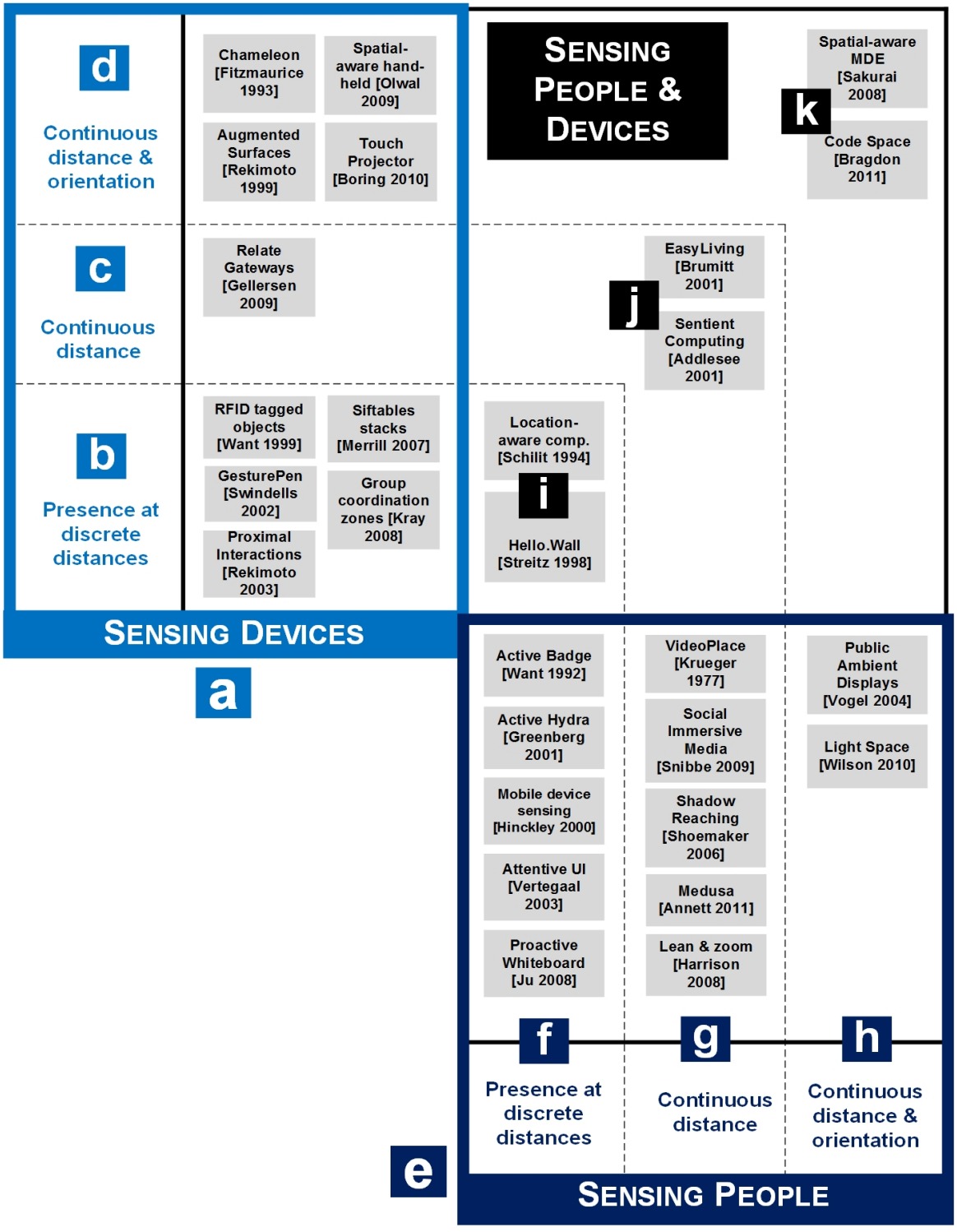


Table 2.1 Overview of related work of ubicomp research considering spatial information, categorized by type of tracked entity (people, device, and people + devices) and fidelity of sensed spatial information.

Instead of using one distance threshold determining inter-device connectivity, later research explored using multiple discrete zones. For example, Kray et al.'s (2008) *group coordination negotiation* introduced multiple spatial regions around mobile phones. Their scenario exploited these regions to negotiate exchange of information with others and to visualize the regions on a tabletop. Depending on how devices were moved in and out of three discrete regions, the transfer of media data between the devices is initiated.

Sensing devices' continuous distance (Table 2.1 c). A second class of techniques uses distances as a continuous measure, but does not sense the orientation of devices. For example, Gellersen et al.'s (2009) RELATE Gateways provided a spatial-aware visualization of nearby devices, which included their approximate distance and direction relative to the device. A graphical map showed the spatial room layout, and icons indicated the position of other nearby devices. Alternatively, and similar to (Rekimoto et al. 2003), icons at the border of a mobile device screen represented the type and location of surrounding devices.

Sensing devices' continuous distance and orientation (Table 2.1 d). A third class of techniques uses continuous measures of distances and orientation. Researchers for example considered how a spatially-aware mobile device would interact with other surrounding devices.

Notably, Chameleon (Fitzmaurice 1993) was a palmtop computer aware of its position and orientation (with six degrees of freedom). Fitzmaurice explored the use of the Chameleon device to access 3D information spaces; such as to support people's interaction in libraries with digitally tagged book shelves that the device could sense and react to. The Chameleon also allowed spatial navigation in a local virtual space by moving the handheld device in a two dimensional area (e.g., for panning a digital map that is larger than the display of the handheld device).

Olwal and Feiner (2009) later refined this technique where they explored the use of spatially-aware handhelds for high-precision interaction on large displays, with the advantage of having higher resolution visual output on the mobile device and a more consistent task performance. Similarly, TouchProjector (Boring et al. 2010) also tracks

the precise distance and orientation of a mobile device relative to other nearby digital surfaces. By doing so, it enables people interact with remote screens through a live video displayed on their mobile device.

Augmented Surfaces (Rekimoto and Saitoh 1999) demonstrate how the tracking of spatial relationships and orientation between devices allows techniques such as *hyperdragging* of content across devices, where a person can begin a mouse drag operation on one device, and continue the operation seamlessly onto another device to drop the information.

2.2.2 Sensing People

Next, we review related work where systems sense and react to the presence of a nearby person or multiple people (Table 2.1e).

Sensing people's presence in discrete zones (Table 2.1 f). A first class of projects in ubicomp react to the sensed presence of people as a binary state, i.e. if a person is in a particular room or not. One of the earliest of such systems is *ActiveBadge* (Want et al. 1992). As we mentioned earlier, a person wears a small electronic name tag that communicates its position through infrared signals to surrounding receivers (e.g., mounted to the ceiling). This made it possible to build applications that leverage the fact that the system knows the presence of an individual within a particular room. For instance, an application could forward phone calls appropriately to another room when a person is not at their desk (Want et al. 1992) or guide a person through a building (Abowd et al. 1997).

Hinckley et al. (2000) built another example of a device capable of detecting a person's presence – though at a smaller scale. They integrated a front facing proximity range sensors into a mobile phone, allowing the device to determine the close presence or absence of a person's head. The display was then deactivated when the device senses the close proximity to a person's head. Researchers have also considered a person's eye gaze direction as a measure that indicates a person's presence and focus to a particular device (Vertegaal and Shell 2008). *Attentive User Interfaces* (AUIs) describe this research

approach, where a system is monitoring a person's eye gaze to determine what device the user is attending to. This technique allows designing systems that only become activated (or receive input from a user) when the person is directly looking towards them. Therefore, attentive user interfaces are a suitable approach to direct a person's multimodal commands (like speech and hand gestures) to the correct device receiving these commands.

Other projects track people's presence in one of multiple discrete zones around a device to driver interactions. Greenberg and Kuzuoka (2001) designed the *ActiveHydra device* to demonstrate a responsive media space detecting people's presence. The device determines a person's distance (in one of three discrete zones) to the communication device to control the fidelity of the audio and video link between two remote collaborators. When looking at the device from a large distance, the screen updates at a low frame rate and only gives glimpses of the remote collaborator's location. When moving closer, the video changes to normal quality, but leaves the audio deactivate to preserve privacy. The audio channel is activated only when both people move directly in front of the device – and thus emulating a face to face conversation.

In a related approach, the *Range* system (Ju et al. 2008) divides the interaction space around a digital whiteboard into four discrete interaction zones (Figure 2.5). These zones correspond to certain transitions of how the system implicitly reacts to a person standing and interacting with the digital whiteboard from a particular distance: for example, ink strokes are clustered when standing at a distance, and the whiteboard clears up space in the center of the display when a moving approaches a board to add new content. Ju discusses a framework that categorizes these transitions along the dimensions of *implicit to explicit* and *foreground to background* interaction. From a technical point of view, the Range system uses four IR proximity sensors mounted in front of the display to approximate the distance of a person to the screen.

Sensing people's continuous distance (Table 2.1 g). A second class of systems sense and react to the continuous distance of nearby people to mediate interactions.

For example, some systems allow full body interaction with a large surface through continuous location sensing. With Kruger's (1985) *Videoplace*, people use their silhouettes (captured by a vision system onto the display) to directly interact with display's digital content. Later, Snibbe and Raffle (2009) built *social immersive media* installations – letting people playfully interact with digital projections on a large wall display or on the floor. This was developed further with the *shadow reaching technique* (Shoemaker et al. 2007), that allows similar interaction through real or virtual shadows. The shadows of a person can function as a magic lens modifying displayed content. In all three projects, the presence and movement of the person's body directly in front of the interactive screen is an essential part of the interaction technique itself.

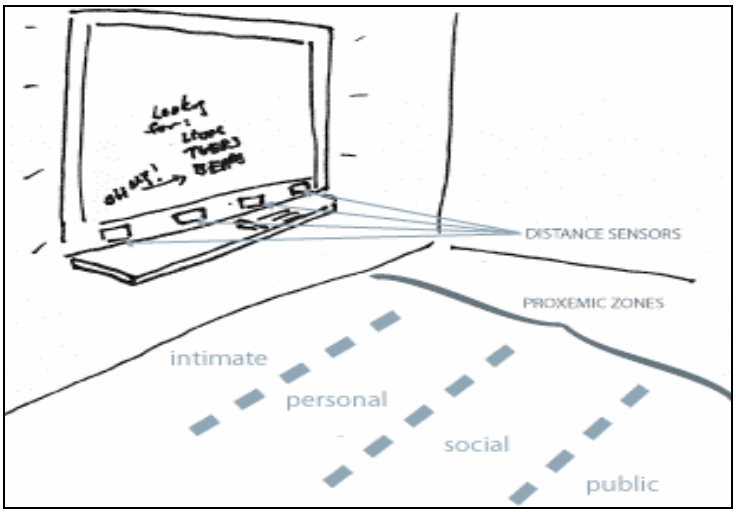


Figure 2.5 Digital whiteboard interaction and four proxemic zones[9].

The *Medusa* tabletop (Annett et al. 2011) introduced a method for continuous proximity sensing for detecting nearby people standing around a horizontal tabletop display. Inferring the location of where people stand around the tabletop allows to build applications that automatically reorient content on the screen to the direction of the person, show control widgets when approaching the screen with a hand, or hiding personal content once a second person approaches. Remarkably, the system is built by

[9] Source: Ju et al. (2008).

using an array of 138 IR proximity sensors; mounted in several layers around the tabletop (Figure 2.6).

At a desktop computer, the *lean and zoom* technique (Harrison and Dey 2008) illustrates how to use continuous measurements of a person's distance to a desktop computer screen adapting the view of displayed content. The smaller the distance becomes between a person's head and the screen, the more the application zooms into the displayed content.

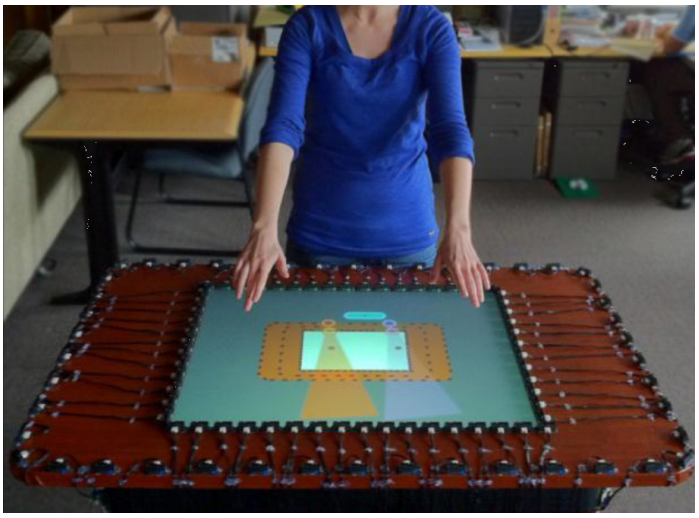


Figure 2.6 Proximity-aware multi-touch tabletop[10].

Sensing people's continuous distance and orientation (Table 2.1 h). A third class of devices explored the design of systems recognizing the continuous distance and orientation of one person or multiple people in space. For example, Vogel and Balakrishnan (2004) designed a public ambient display reacting to a person's distance and body orientation relative to the display. They map Hall's proxemic zones to four modes of interaction, moving from ambient display, to implicit, subtle, and personal interaction (Figure 2.7 left). The project explores the possible ways of how a person's interaction with the ambient display can be mediated by moving in and out of these discrete zones, but also by sensing a person's body orientation and a set of 3D hand

[10] Source: Annett et al. (2011).

gestures for explicit control of the displayed content (Figure 2.7 right). A major idea in their work is that interaction from afar is public and implicit, and becomes more private and explicit as people move towards the surface. They illustrate their concept with a digital calendar application that reveals more detailed and personal information when a person moves closer, that hides the information immediately when the person turns around, and that recognizes the presence of multiple people in front of the display and changes the displayed content accordingly.

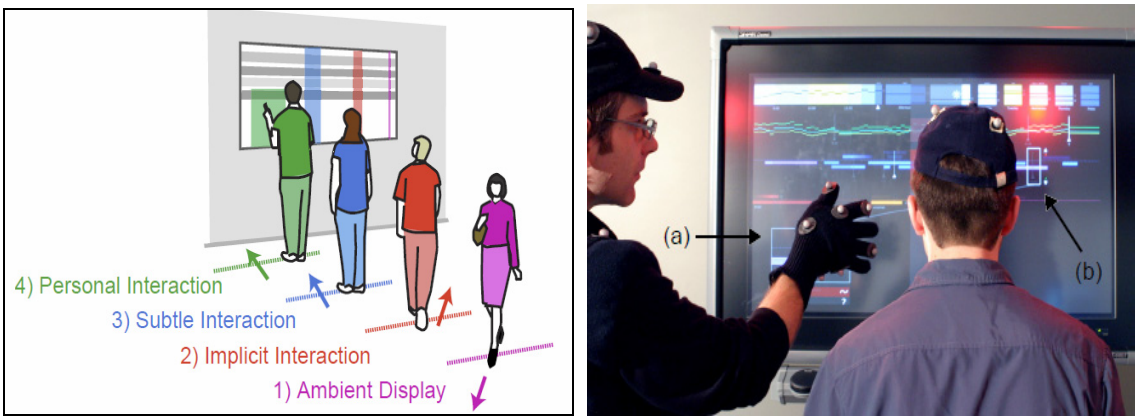


Figure 2.7 Interaction with public ambient displays[11].

More recently, *LightSpace* describes novel interactions of one or multiple people in a ubicomp ecology (Wilson and Benko 2010). Because the system tracks the people moving in space with ceiling-mounted cameras (Figure 2.8 left), a person can, for example, transfer a digital picture from one display to another by simply touching both the digital object and the destination surface simultaneously (Figure 2.8 right). Alternatively, a person can pick up virtual objects by sweeping them into their hands, dropping the virtual object onto another digital surface by touching it, or even passing an object to another person by dropping it into their hands. Importantly, the project explores interactions that leverage people's position and gestures *between* the interactive multi-touch surfaces – "*the room becomes the computer*" (Wilson and Benko 2010).

[11] Source: Vogel and Balakrishnan (2004).

2.2.3 Sensing Both People and Devices

The last category of systems (Table 2.1 i-k) that considers both people's and devices' spatial relationships, is in particular closely related to our own research goal of mediating people's interactions in a *complete ubicomp ecology*.

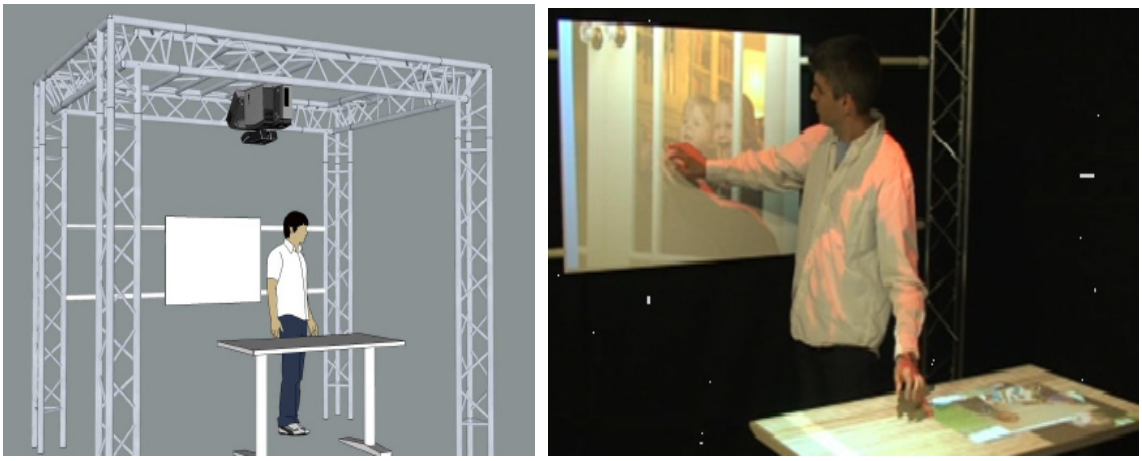


Figure 2.8 Tracking people's movements in a ubicomp environment with LightSpace[12].

Sensing people and devices' presence at discrete distances (Table 2.1 i). A first class of systems recognizes the presence of devices or people in an environment. Schilit et al.'s (1994) location-aware interaction techniques extends the earlier work of ActiveBadges (that focused on tracking people), where the system's knowledge of a person's location is combined with the knowledge of surrounding mobile and stationary devices. The system then, for example, uses this information to facilitate selection of nearby devices (e.g., the closest printer), or to reconfigure the technology nearby (e.g., dim the lights according to personal preferences). Other research projects track people and their devices' presence in *discrete distance zones* around large displays to adapt the modes of interaction. For example, *Hello.wall* (Streitz et al. 2003) introduces the notion of 'distance-dependent semantics', where the distance of a person to the interactive wall defines the possible forms of interaction and the kind of information shown on both the wall display and a person's mobile device. The project technically detects people and their devices in three discrete

[12] Source: Wilson and Benko (2010).

spatial zones around the display (using a sensing mechanism of RFID tags), and moves from ambient information, to notification, and direct interaction that links the mobile device to the large surface (shown in Figure 2.9).

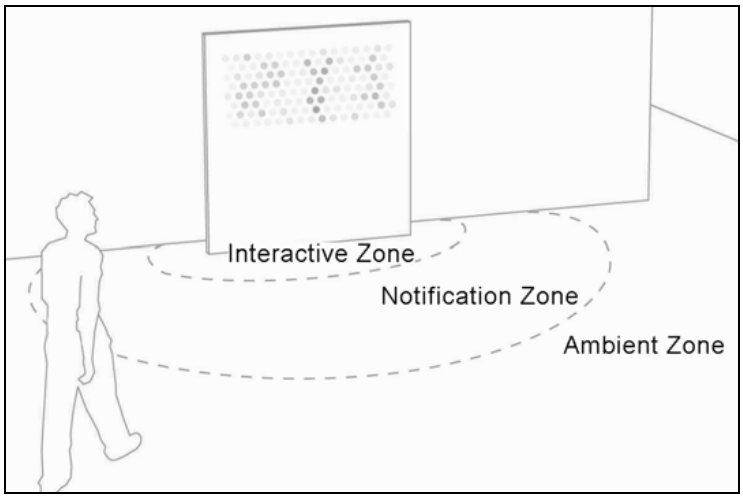


Figure 2.9 Presence of people and devices in discrete distance zones affect the interaction with Hello.wall[13].

Sensing people's and devices' continuous distances (Table 2.1 j). A second class of systems considers continuous information about people's and devices' position. The intelligent home environment of the *EasyLiving* project (B. Brumitt et al. 2000) leveraged information about the current location of people and devices to, for example, provide a customized interface on a person's mobile device to control nearby devices (e.g., adjusting the lights), or automatically activating devices based on a person's presence (e.g., playing the preferred music of a person on nearby speakers). To allow the design of such interactions, the project introduced the concept of creating a geometric model between entities in space that is updated with data gathered by fusing multiple sensor sources (e.g., computer vision and radio sensing). The software then takes this geometric model to check the location of a particular person, and updates interfaces or starts and stops services accordingly (B. Brumitt et al. 2000). As mentioned earlier in Section §2.1.3, the sentient computing strategy applied a similar technological design to

[13] Source: Streitz et al. (2003).

facilitate people's interactions in office spaces (Addlesee et al. 2001). For example, mobile devices automatically reconfigure themselves depending on who picks them up.

People's and devices' continuous distance and orientation (Table 2.1 k). A third class of devices consider both people's and devices' continuous distance and orientation to mediate interactions.

First, systems leveraged knowledge about these precise relationships to facilitate people's navigation and use of multiple screens in ubicomp ecologies. For example, Sakurai's (2008) middleware for MDEs generates perspective-corrected output on all screens surrounding a person to facilitate mouse cursor navigation across these screens. The cross-display mouse movement technique itself is related to hyperdragging (Rekimoto and Saitoh 1999) we mentioned earlier, except that only the knowledge about the exact relationship between a person's head and the surrounding displays & devices enables the accurate perspective correction of this approach (Figure 2.10 left).

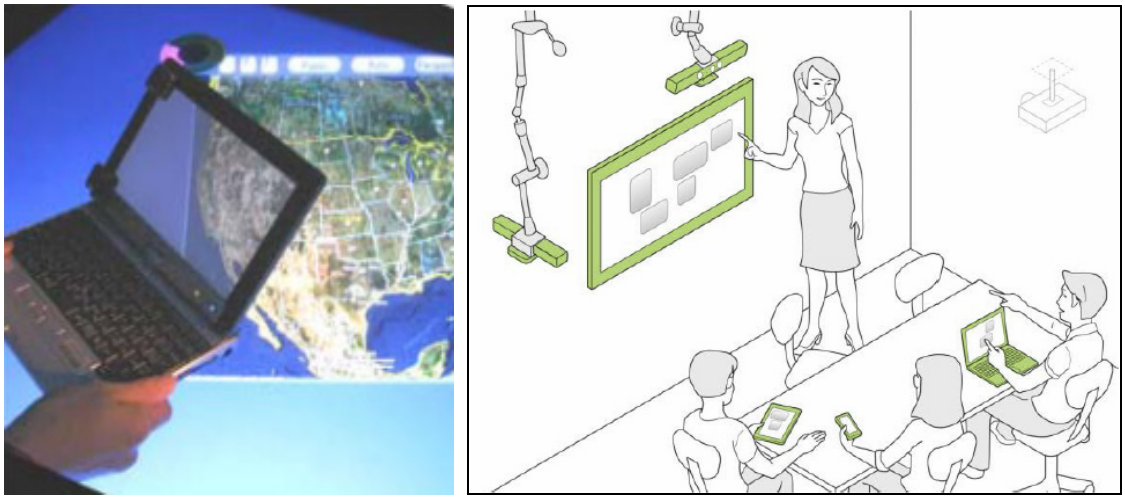


Figure 2.10 Tracking continuous distance and orientation between people and their devices in ubicomp ecologies: MDE middleware and CodeSpace[14].

Second, Bragdon et al. (2011) build on the concept of spatially aware environments, where they contribute the Code Space system (Figure 2.10 right) supporting developer's

[14] Source: MDS middleware by Sakurai et al. (2008); CodeSpace by Bragdon et al. (2011).

code review meetings. Their novel *touch + air hybrid gestures* allow people to access, control, and share information across multiple personal devices and large surfaces. As part of their set of cross-device interactions a person can, for example, share content from a notebook computer onto a large display by touching the notebook's screen with one hand while pointing towards large wall display with their other hand. Most importantly, these hybrid techniques only become possible because the system considers the spatial relationships between people and their devices.

As described shortly in Chapter 3, we are interested in bringing this line of research further, by exploring ubicomp interaction design considering proxemics between the complete ecology of people, digital devices, and non-digital objects, and the surrounding environment.

2.3 Conclusion

In this chapter we surveyed related work in the research area of ubiquitous computing most relevant to our proposed research of proxemic interactions. Similar to many of the projects surveyed in this chapter, we also envision systems that, in part, react pro-actively to sensed properties in the environment, and believe that these system designs can lead to more seamless and fluent interactions of people with their surrounding devices.

Our research does, however, differ in three important aspects to earlier research in context sensing, embodied interaction, and ubicomp.

First, instead of a general model of context through sensing, we focus only on very specific aspects relevant to proxemics: the distance, orientation, and other aspects defining the spatial relationships between people, devices, non-digital objects and the environment they are in. With this, the proxemic dimensions we identify in Chapter 3 are a *focused subset* of context-aware information. Our three case studies of proxemic-aware systems – while all considering the fine-grained knowledge about entities in ubicomp ecologies – then each focuses on a particular segment of the design space we

discussed based on Table 2.1. First, we investigate person/people-to-device proxemics (Figure 2.11a), then device-to-device proxemics (Figure 2.11b), and finally consider both the proxemics between people and proxemics between devices (Figure 2.11c) in ubicomp interaction design.

Second, our proxemic interactions model tries to leverage social expectations of people as described by proxemic theory (to be discussed later), i.e., that system reactions are in accordance to people's expectations.

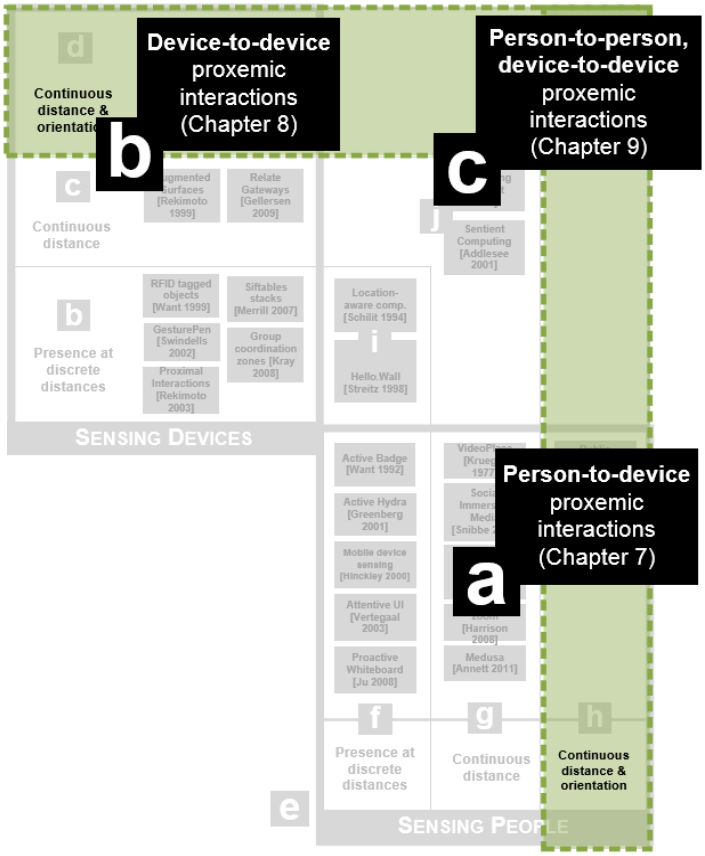


Figure 2.11 Our three case studies of proxemic-aware systems and their relation to the design space of related work (cf. to Table 2.1).

Third, our research differs in the granularity of spatial information we are interested in. Instead of knowing that a person or device is "in a room" as common in many context-aware systems, we are interested in finding out what technology designs are possible with more accurate, fine-grained measures of proxemics defining the relationships

between entities: How close are people? Where exactly are devices located? How are people holding their devices? What is the orientation of people and devices, and are they facing each other? How fast is a person approaching a particular device? Similar to the richness of how these proxemic relationships affect our everyday interactions with other people, our thesis is that we can mediate interactions with devices by considering proxemic relationships in ubicomp ecologies.

To gain a better understanding of exactly what kind of proxemic relationships are most relevant to consider, and how these could be measured in an interactive ubicomp system, our next chapter distils essential proxemic theories, which we then operationalize as proxemics applied to ubicomp interaction design.

Chapter 3. Proxemic Interaction Framework

In the previous chapter we illustrated the potential of considering people's or devices' spatial relationships when designing interactive systems of ubicomp ecologies. The idea of integrating information of people's or devices' spatial relationships in interaction design is not new. However, this integration was only rarely done in the context of complete ubicomp ecologies comprising people, devices, and objects. Furthermore, the earlier work focused on distance information (often only divided into discrete spatial regions), and had not factored in other fine-grained and nuanced aspects defining spatial relationships.

Before delving into the specifics of how different research projects have sensed various parameters, we need to better understand what kinds of information about spatial relationships might be relevant for interaction design. To begin this exploration, we review several seminal theories in sociology, psychology, and ethnology about people's understanding and use of the personal space around them. Based on this understanding we then operationalize these theories for ubicomp interaction design, and discuss the potential of applying nuances of these theories to address various ubicomp interaction design challenges.

Within this context, the goal of this chapter is to inform and guide the design process of ubicomp developers in form of an operationalization of proxemics – that we call the *Proxemic Interactions* framework – in order to let developers create, invent, or discover novel proxemic-aware interaction techniques. The strengths of this framework are similar to generative theories (Rogers 2004) or interaction models (Beaudouin-Lafon

2004), which (a) allow thinking in structured ways during the design process, (b) provide clear vocabulary for discussing designs, and (c) allow generating novel ideas through design dimensions and constructs (Shneiderman 2006; Beaudouin-Lafon 2004; Rogers 2004). Rogers (2012) emphasizes that the appeal of generative theories "*is their ability to account for technology-augmented behaviors and to inform new interventions to change behaviors that people care about — compared with the scientific theories that were intended to test predictions, and to make generalizations about human performance under controlled conditions.*" (Rogers 2012). Our intention is not to provide a prescriptive set of guidelines or rules for the design of spatially-aware ubicomp systems, but to introduce the *proxemic interactions* framework as a first-order approximation of how to apply proxemics to ubicomp interaction design.

In the remainder of this chapter[15], we first survey seminal psychological and social theories about human spatial behaviour that are most relevant for ubicomp interaction design (§3.1). We then use the insights of these theories to operationalize proxemics (§3.2) as knowledge that can be sensed or captured by devices via five essential dimensions – distance, orientation, movement, identity, and location (§3.2.1). Last, we introduce a proxemic interaction model describing the nuanced use of these five dimensions in ubicomp interaction design (§3.2.2).

3.1 Theories of Personal Space and Proxemics

We begin by reviewing theories that we believe are relevant to inform ubicomp design. In particular, this research leverages insights from sociological concepts – most importantly the theories of personal space, proxemics, and F-formations.

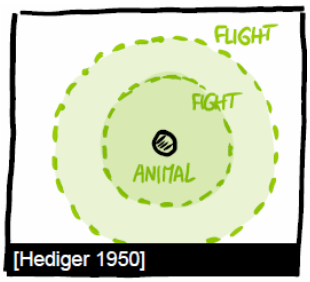
---

[15] Portions of this chapter are published as:

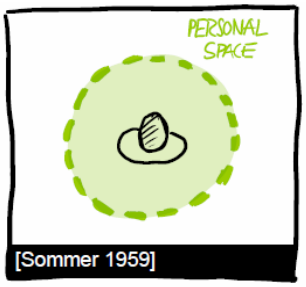
Marquardt, N. and Greenberg, S. (2012) Informing the Design of Proxemic Interactions. In *IEEE Pervasive Computing*, 11, 2. April 2012. Joe Paradiso, Trevor Pering, Albrecht Schmidt, Eds., pages 14-23. © 2012 IEEE.

Marquardt, N., Hinckley, K. and Greenberg, S. (2012) Cross-Device Interaction via Micro-mobility and F-formations. In *Proceedings of the ACM Symposium on User Interface Software and Technology – ACM UIST 2012.* (Cambridge, MA), ACM, October 7-10, pages 13-22.

3.1.1 Personal Space



The term personal space was initially used in zoology for describing animal reactive behaviour, and defined as the distance zone where animals perform complex greeting, courting, and care-soliciting behaviours (Katz 1937). The biologist Hediger later suggested that animals are surrounded by "bubbles or balloons" that affect their spacing relative to other animals around them (Hediger 1950). He observed particular distances affecting animal behaviour; most importantly the flight distance (that when entered by a predator causes the animal trying to escape), and the smaller critical or fight distance (when entered by a predator can lead to a fight). These distances were also influenced by the animal species, their age, size, and gender.



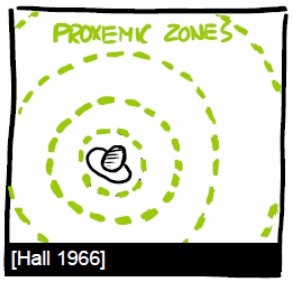
Researchers later adopted the term personal space to the study of human spatial behaviour, and in particular *interpersonal* relationships. Sommer (1959) describes personal space as the distance that one person places between themselves and other people around them. Personal space is often described as an invisible boundary or bubble of space surrounding a person (Altman 1975). People's perception of changes in the area of personal space around them influences how they engage, interact, and communicate with others. A major part of psychology research primarily focused on the *protective function* of personal space and people's use of this space as it is dictated by social rules and norms (Aiello 1987). People often react with resentment if these rules are broken – for example if a stranger enters the boundaries of their personal space (Altman 1975; Ciolek 1983).

Even though studies of personal space are related to those of *territories*, there are important differences (Sommer 1959; Altman 1975). While territories usually refer to a fixed geographical location with clear boundaries, the personal space does not have a fixed location or any clearly marked boundaries. Furthermore, personal space is defined

as being an area around the body of a person, while territories are not (Bechtel and Churchman 2002).

The definition of personal space as an invisible bubble surrounding a person has been controversial (Aiello 1987). For example, Patterson (1975) criticized this definition as "unnecessary and probably misleading", as it implies stability where actually a large variety of factors impact people's perception of personal space. Similarly, Knowles (1989) argues that studies of discrete personal space distances makes the questionable assumption that people's reactions to distance changes are *not* continuous. This is why further theoretical approaches were introduced for the study of human spatial behaviour – perhaps most importantly the theory of proxemics.

3.1.2 Hall's Proxemics



The anthropologist Edward Hall introduced proxemics[16] to define the "interrelated observations and theories of man's[17] use of space" (Hall 1966), and he focused in particular on the measurable distances between people as they interact. Most importantly, Hall went beyond the sole analysis of people's use of personal space as a protective measure, but instead conceptualizes personal space as a form of non-verbal and implicit communication – that he also refers to as the silent language. His theory – while emphasising social and cultural differences – generally describes how people perceive, interpret, structure, and (often unconsciously) use the micro-space around them, and how this affects their interaction and communication with other nearby people. Hall believed that the theory of proxemics does not only contributes to the study of people's interactions in daily life, but also to

[16] From the Latin root *prox-* (proximitas, meaning nearest or closest) and the suffix *-emic* (a term used by anthropologists to reflect a point of view in terms of the individual within a culture, and that also indicates links to terms in linguistic structuralism, such as systemic and phonemic); see (Nöth 1995)

[17] In his writings, Hall uses the term 'man' to refer to both men and women, as it was a convention at that time (now almost 50 years ago).

better understand "the organization of space in his houses and buildings, and ultimately the layout of his towns" (Hall 1963).

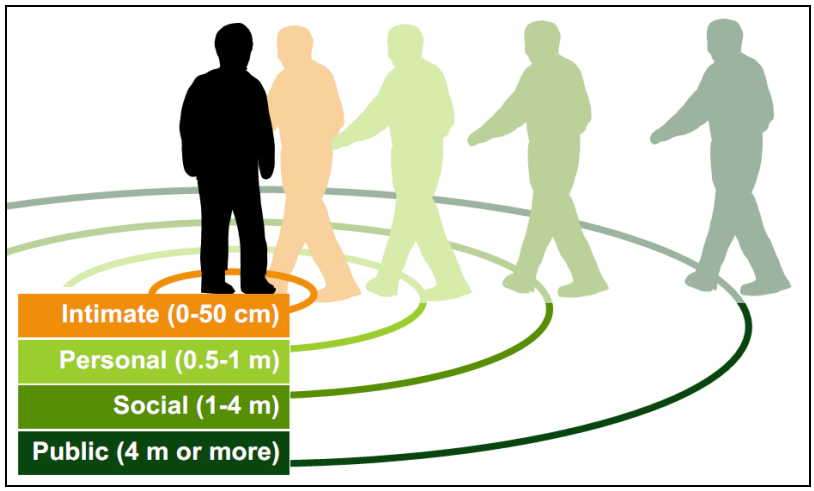


Figure 3.1 Hall's four discrete proxemic zones.

Hall details how people perceive, interpret and use proxemics cues, especially distance, to mediate relations to other people. In particular, he correlates physical distance to social distance between people. As illustrated in Figure 3.1, he categorizes this into four discrete distance zones that are also directly linked to people's perceived sensory information, and describes the primary types of activities corresponding to them[18]:

- **Intimate** (0 - 50cm), e.g., the distance of individuals in close relationship (or in an engaged argument as shown in Figure 3.2a). This is the distance that addresses the most sensory inputs: people can see the other close person, hear even a whisper, feel the heat or smell another person, and touch them. Normally people enter the intimate distance of another person only with permission (with exceptions due to environmental constraints, such as people standing very close in an elevator).

[18] This particular categorization applies mostly to North American culture, but Hall later also describes cultural differences of proxemics (and the zone distances in particular) that he observed with people of different cultures (e.g., Latin-American, Asian, and European).

- **Personal** (0.5 – 1.2m), e.g., when interacting with friends or family (Figure 3.2b). At this distance "within arm's length" it is still possible to touch the other person, and people can speak at lower volume to each other.
- **Social** (1.2 – 3.5m), e.g., the interaction in a more formal setting (Figure 3.2c). At this distance it is harder to reach out to touch another person, the voice is louder, and interactions are often more formal.
- **Public** (› 3.5m), e.g., the distance of a speaker to an audience (Figure 3.2d). At this distance people have to speak louder to address others, and the other primary sensory input is vision.

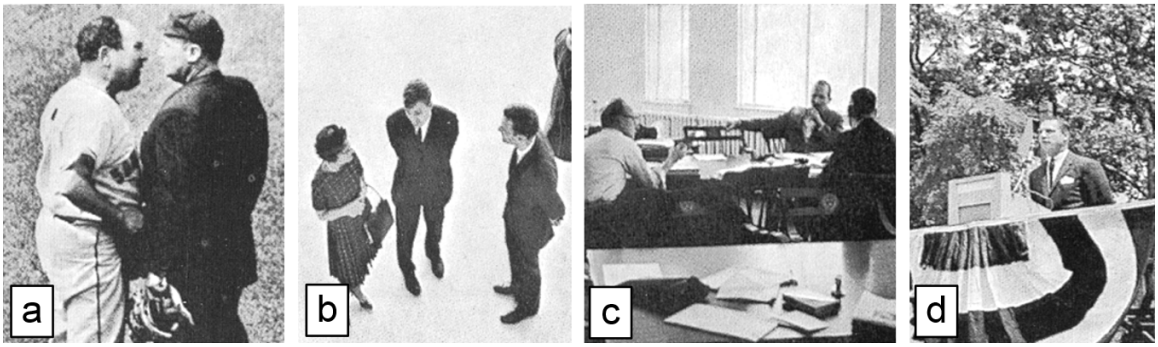


Figure 3.2 Proxemic distances: (a) intimate, (b) personal, (c) social, (d) public[19].

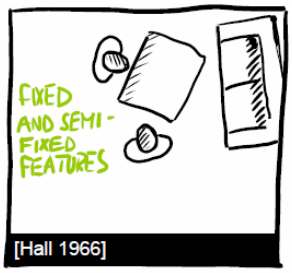
These collective distances, which Hall further divides into a near and far distance and collectively calls the *dynamic space*, characterize a progression of interactions ranging from highly intimate, to personal, to social and to public (Hall 1966). The four zones are also suggestive of the kinds of relationship people consider within them: intimate distance for intimates (e.g., partners), personal for people who are close (e.g., friends), social for people in a professional relationships (e.g., colleagues), and public for addressing an unfamiliar or unknown group of people (e.g., audience of a talk).

Even though the boundaries of proxemic zones are invisible and imaginary, people's social responses to violations of expected behaviour corresponding to each zone are

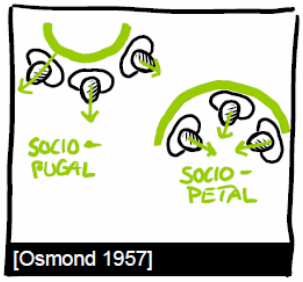
[19] Source: Hall (1996).

perceived as real and concrete. Even through Hall emphasized distance as the most important factor in regards to proxemic behaviour, we also acknowledged the effect of other variables such as orientation (explained shortly). While Hall mostly derives his theory from qualitative observations of human spatial behaviour, later empirical studies confirmed the validity of Hall's qualitative ideas (Aiello 1987).

3.1.3 Environment: Fixed and Semi-Fixed Features



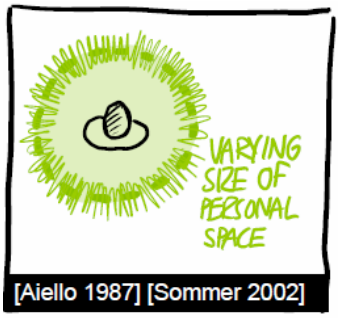
Proxemic behaviour is not only affected by people's use of the immediate space around them, but also "the organization of space in his houses and building, and ultimately the layout of his towns" (Hall 1963). In this context, Hall identified two other factors that influence people's use of proxemics (Hall 1966). *Fixed features* include the immobile properties of the space: the layout of buildings and rooms, the walls, doors, and windows. *Semi-fixed features* include the spatial layout of elements in the space that can be moved (like furniture, chairs, or tables).



Hall noticed that the layout of the fixed features as well as the arrangement of elements in the semi-fixed feature space influence our use and perception of personal space, where particular layouts can be *sociofugal* (separating people) and *sociopetal* (bringing people together) – where he refers to earlier studies by Osmond (1957). A simple example is how chairs in a living room can be brought together into a sociopetal small circle to encourage intimate chat, or to be placed opposing each other to enforce a sociofugal dynamic.

Although others have critiqued Hall's classification of personal space as being overly simple (e.g., comments in Hall, 1968), his work has become an influential seminal theory of studying personal space and is still being widely applied in psychology and sociology research to study, understand, and consider people's use and perception of personal space (Bell et al. 2005; Aiello 1987). Since then (and summarized next), other theories added new perspectives that go beyond Hall's original distance-focused view of personal space.

3.1.4 Size and Shape of Interpersonal Distance Zones



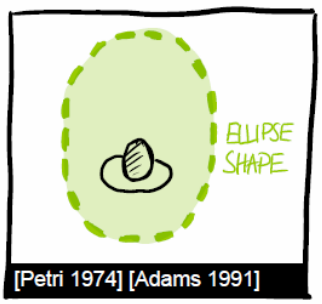
The exact ranges of interpersonal distance zones (or the single personal space distance) are not static. Even Hall noted that "*the measured distances vary somewhat with differences in personality and environmental factors*" (Hall 1966). Later experimental psychological studies observed that the size of the personal space boundaries depend on a variety of factors, and that those are "continually open to modification" (Hayduk 1985). For example, the following experiments (or surveys of multiple study results) observed changes of the personal space distance depending on:

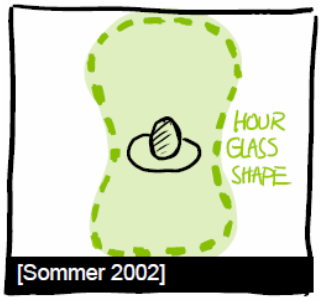
- **the environment:** factors that impact people's use of proxemics are, for example, room size, spatial layout, and room density (Evans et al. 1996) or the current lighting conditions (interpersonal closeness caused significantly less discomfort in high illuminated settings than in darker settings) (Adams and Zuckerman 1991).

- **the cultural background:** empirical studies mostly confirmed Hall's qualitative observations that people from contact cultures (Mediterranean and Latin Backgrounds) sit and stand closer than people from noncontact Anglo Saxon cultures (Aiello 1987).

- **the gender and age:** studies showed that female pairs often stand closer than male pairs (Price and Dabbs 1974), and that interpersonal distance increases from childhood age to adults (Aiello and Aiello 1974).

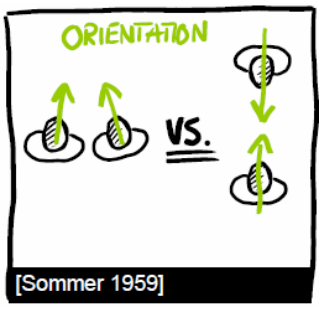
- **the relationship to the people around:** for example, people that are extroverted or affiliative tend to work in closer proximity (Gifford 1997), and friendship and acquaintanceship decreases interpersonal distance (Sommer 2002).





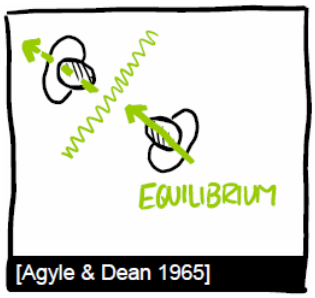
Further experimental studies also confirmed that the shape of personal space is not necessarily circular around a person, as it was often described in earlier work. For example, an elliptical shape was suggested, with longer distance at the front smaller distances at the sides (Petri et al. 1974). This ellipse shape also appears as result in other related studies about people's reaction to other people entering their perceived personal space (Adams and Zuckerman 1991; Duke and Nowicki 1972). Later, Sommer suggested an hourglass shape for the personal space, with the wider areas in front and back of a person, and narrower sides of the hourglass shape at the side of the person (Sommer 2002).

3.1.5 Orientation



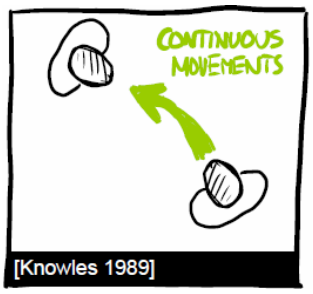
Orientation generally describes how people face towards or away from each other, and this too affects proxemic relationships. As we described earlier, orientation matters for people's perception of their surrounding personal space. Sommer (1969) later studied people's preference of spatial seating arrangements and relative orientation around a table depending on the task at hand. Depending on the task, the majority of people tended to particular seating positions: face-to-face seating for competitive tasks, side-by-side for cooperative tasks, and side-by-side or corner-to-corner during conversations. Sommer concludes that spatial arrangements and relative orientation that people choose during small group interactions are "functions of personality, task, and environment". Thus, structuring the semi-fixed feature space can have a "profound effect on behaviour and [...] this effect is measurable" (Hall 1966). Others identified patterns, where people's orientation to one another depended on the type of conversion and social status (Ciolek 1983). Overall, these and other study experiments demonstrated the importance of considering orientation (and not just relative distance) when analyzing human interactions in close proximity.

3.1.6 Compensation, Balance, and Privacy



People constantly adjust their use of space to fit the presence of, and interactions with, others. This includes how people react to and try to overcome 'invasions' or 'violations' of their personal space. Some theories describe people's adaptation to given spatial circumstances, and how they try to maintain a certain comfort level or equilibrium in these situations (Baldassare 1978). For example, the *intimacy equilibrium model* (Argyle and Dean 1965) assumes that when people interact they always strive to maintain an overall balance towards a desired optimal proxemic distance. To achieve this balance, people might try to adapt proxemic variables such as distance, orientation, or eye contact, which the model describes as "inverse relationship between mutual gaze, a nonverbal cue signaling intimacy, and interpersonal distance" (Ciolek 1983). For example, when a person stands too close to us, we might step back to maintain the equilibrium. If any of the variables cannot be changed in this particular situation (such as standing very close to others in an elevator), the change of *another variable* can be used to compensate (in the elevator example: changing orientation to face away while avoiding eye contact). Another predictive model formalizes equilibrium as an optimal proxemic distance, where it adds proxemics variables including identity and familiarity of the other person, and the type of interaction (Sundstrom and Altman 1976). People also use personal space as a method to protect a certain level of privacy. Altman (Altman 1975) reframes this use as a dynamic boundary regulation process that controls privacy.

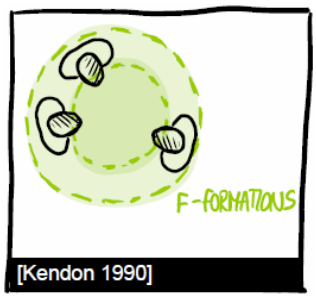
3.1.7 Discrete vs. Continuous Distances



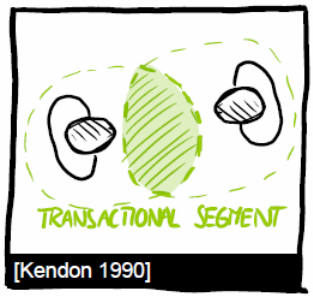
The analytical observations of human spatial relationships do not necessarily have to be classified into discrete zones (such as the single zone of personal space, or Hall's four proxemic zones). Aiello did not find evidence that people's reactions to changes in spacing/orientation happen at the zonal transition points (Aiello 1987). This is in line with Knowles, who argues

that any discrete category system makes the assumption that people's reactions to distance changes are *not* continuous (Knowles 1989). Instead, Aiello suggests that "experiences occur more *gradually* as sensory inputs change from one distance zone to another". For example, the continuous position/orientation adjustments and movements that people do according to the intimacy equilibrium model (Argyle and Dean 1965) explained earlier fit into this understanding of continuous changes of proxemic variables.

3.1.8 The Focused Encounter: F-formations



Distance in and of itself is not a complete description of the social connectedness of co-located persons. Whereas Hall's notion of proxemics primarily concerns the impact of *distance* on the perceptions available to the organism—and hence the types of communications afforded—the study of *F-formations* further considers the *physical arrangements* that people adopt when they engage in focused conversational encounters. Specifically, F-formations (*Face-or Facing-Formations*) are a macro-level theoretical lens through which one observes small-group interactions (Kendon 1990; Kendon 2010; Ciolek and Kendon 1980).



F-formations consider the spatial relationships that occur as people arrange themselves during face-to-face interaction for optimal task-dependent communication and clarity of perception. A typical arrangement is a roughly circular cluster that contains 2-5 persons who are actively part of the group (see Figure 3.3 and Figure 3.4). The inner space of that circle (called *o-space*) is reserved for the main activity the group is pursuing. This inner space is formed by the overlapping area of the transactional segments of each individual person that is part of the formation. These transactional segments are defined as the "space extending in front of a person which is the space he is currently using in whatever his current activity may be" (Kendon 1990). In many cases this is the area in front of a person's body that can be reached with their hands, but can extend beyond that reach to

further distances. The ring of space occupied by the people (*p-space*) determines group membership and it is where people are located. The surrounding region (*r-space*) buffers the group from the outside world. Thus persons who are nearby but not in p-space are excluded from the fine-grained social circle that defines the F-formation. Still, the group monitors the r-space to see others who may be trying to join.

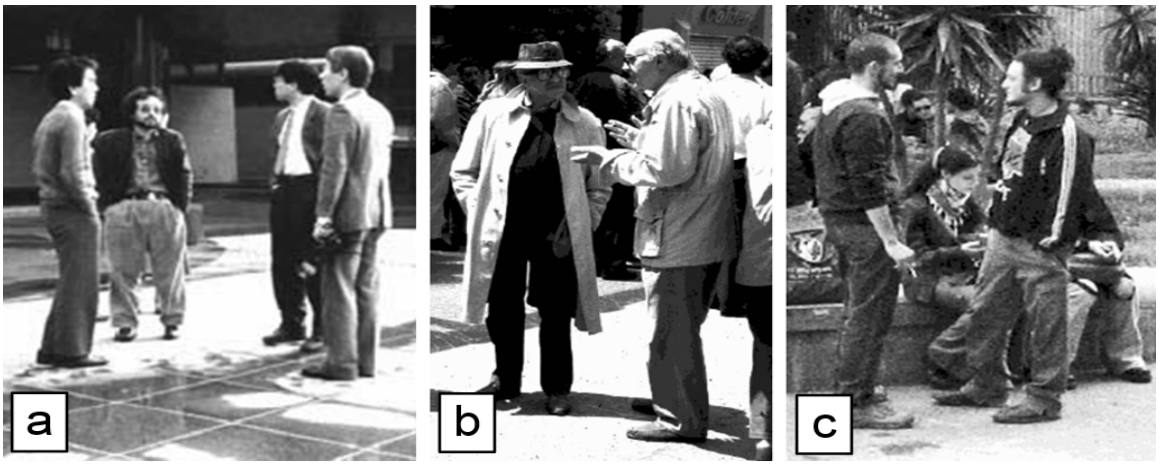


Figure 3.3 F-formations: (left) circular, (center) corner-to-corner, (right) face-to-face[20].

Kendon describes specific procedures that take place when a person joins an existing F-Formation from within the r-space. For example, an approaching person in r-space may be greeted via eye contact, while a person who is facing away, even if close to the group, is not treated as a potential member.

F-formations are nuanced and not necessarily circular. Different relative body orientations—face-to-face, side-by-side, or corner-to-corner (see Figure 3.3)—afford different types of collaborative tasks: competitive, collaborative, or communicative, respectively (which directly relate to earlier observations by Sommer, 1959). As illustrated in Figure 3.5, further more fine-grained spatial arrangements in F-formations can be differentiated, and classified into open and closed formations (the letters used to describe these formations are derived from the lines formed by the major box axis

[20] Source: Kendon (2010).

connecting the left and right shoulder of a person, and the connection line between the two people – for example, the letter 'H' for two people standing face to face). Group size varies, but tends to be small. Freely forming groupings rarely surpass five persons; 95% are four persons or less, and more than half are dyadic (Dunbar et al. 1995). Gestures made or objects held within the o-space become the focus of conversation, whereas objects held down (in p-space) or outside the circle (in r-space) are excluded (Ciolek and Kendon 1980).

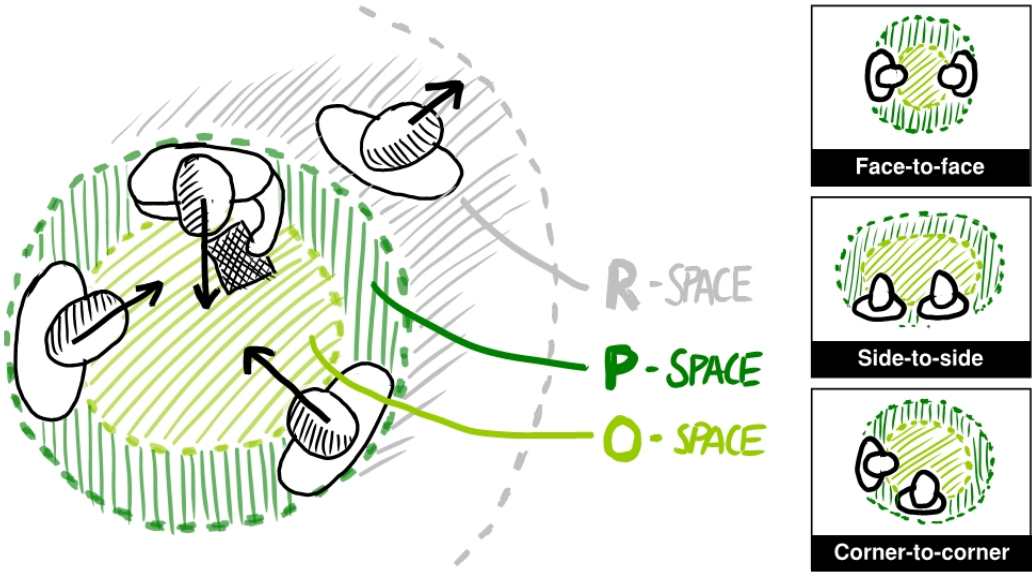


Figure 3.4 An F-formation consists of two or more persons engaged in joint activity. Their bodies define three, roughly circular, regions: the inner o-space, the ring of p-space, and the surrounding r-space.

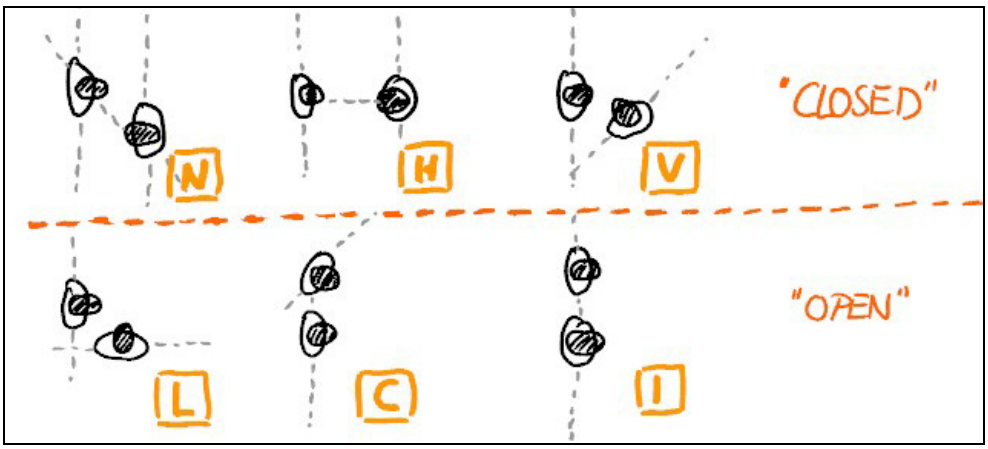
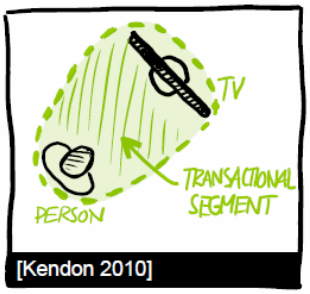


Figure 3.5 Types of F-formations.



Semi-fixed features around a person can also be part of the transactional segment of a person or multiple people. Kendon explains how semi-fixed features of a room (e.g., a TV that a person is looking at) or moveable objects (e.g., a piece of paper the person is writing on) can be the focus point of a transactional segment and influence the structure of F-formations (Kendon 2010). In this case, "spatial and postural orientation are [...] excellent clues to major junctures in the flow of behaviour" (Kendon 2010).

3.1.9 Proxemic Theories as Analytical Lenses in Interaction Design

In HCI research, theories of proxemics and F-formations have both been used as lenses to analyze individual or group interactions with interactive technology.

For example, F-formations have been used to analyze social interactions in crowded environments, for example by using the F-formations theory as a conceptual lens for analyzing social interactions of visitors in a tourist information centre (Marshall et al. 2011). Later, the theory was also fundamental part to classify touch-less gestural interactions of medical doctors reviewing patient's digital images during surgeries (Mentis et al. 2012). So far, however, the F-formations theory has only rarely been applied to interaction technique design.

In the context of interaction design, researchers have taken Hall's proxemic concepts and defined discrete zones around devices with implications of how these zones affect people's interactions. For example, as reviewed earlier in Chapter 2, four distance zones defined the possible interactions around an ambient display from ambient to personal (Vogel and Balakrishnan 2004), and similarly people's presence in four zones affected the implicit reactions of an interactive whiteboard application (Ju et al. 2008).

3.1.10 Summary

In summary, the reviewed sociological and psychological theories (in particular personal space, proxemics, and F-formations) formalize an important phenomenon of people's everyday interactions: the perception and use of personal space during everyday

encounters. They describe how factors such as distance and orientation, as well as how these factors change over time, play an important role in how people mediate their interactions with others.

Yet in spite of proxemics being a fundamental part of social behaviour, the fine-grained nuances of proxemics are only rarely considered in ubicomp interaction design. A few researchers have taken Hall's proxemic concepts and defined discrete zones around devices with implications of how these zones affect people's interactions. For example, as reviewed earlier in Chapter 2, four distance zones defined the possible interactions around an ambient display from ambient to personal (Vogel and Balakrishnan 2004), and similarly people's presence in four zones affected the implicit reactions of an interactive whiteboard application (Ju et al. 2008). However, only very few systems have considered these varied aspects of proxemics (e.g., distance, orientation, motion) between all entities of the ubicomp ecology: the people, devices, objects, and the surrounding environment.

In the context of ubicomp, the implication is that proximity becomes, in part, an estimation of people's desire to communicate with one another via the devices they carry and – by extension – a desire of one or more people to interact with particular devices that surround them. In the next section of this chapter we discuss how proxemic theories can be operationalized to inform the design of novel techniques for people's co-located interactions with digital devices in ubicomp ecologies.

3.2 Operationalizing Proxemics for Ubicomp Interaction

The proxemic theories above describe many different factors and variables that people use to perceive and adjust their spatial relationships with others. It is important to note that most of these theories describe people's relations to people, and not to devices. Even so, in this dissertation we argue that we can use these theories as a first-order approximation to apply proxemics to ubicomp design. As part of this approximation and our proxemic interaction framework, we identify five device-oriented proxemic dimensions – inputs and states that devices can hold about proxemics relationships –

which we believe are most relevant to operationalizing proxemics in ubicomp interaction (see Figure 3.6). That is, they describe not only the fine-grained nuances in the relationships between people, but with all entities in ubicomp ecologies: people, digital devices, non-digital objects, and the features of the surrounding environment.

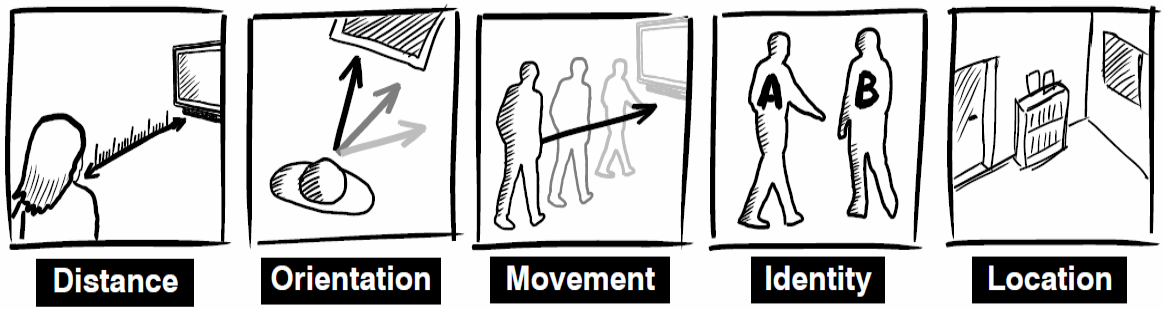
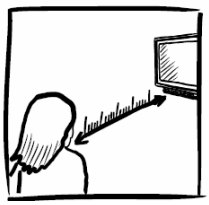


Figure 3.6 Five key proxemic dimensions relevant for ubicomp interaction design.

3.2.1 Proxemic Dimensions

The following five proxemic dimensions are directly derived from the theories we reviewed at the beginning of this chapter. We note that these dimensions are a starting point, where they determine what we consider the foundational information that can be sensed computationally and applied to proxemic interactions. We anticipate that other dimensions will be identified in future work that can contribute to the nuances of designing proxemic interactions.

3.2.1.1. Distance



As a fundamental dimension of describing spatial relationships (relating directly to the theories of personal space and proxemics we reviewed earlier), *distance* describes the measurable distance between entities in the space: people, devices, objects, and fixed/semi-fixed features in the environment (Figure 3.7). Distances can be represented in many ways. For example, they can be precise measurements (e.g., 120 centimeters) or crude categorizations (e.g., zone 1). Distance can be described as absolute positions or as relative distances between entities.

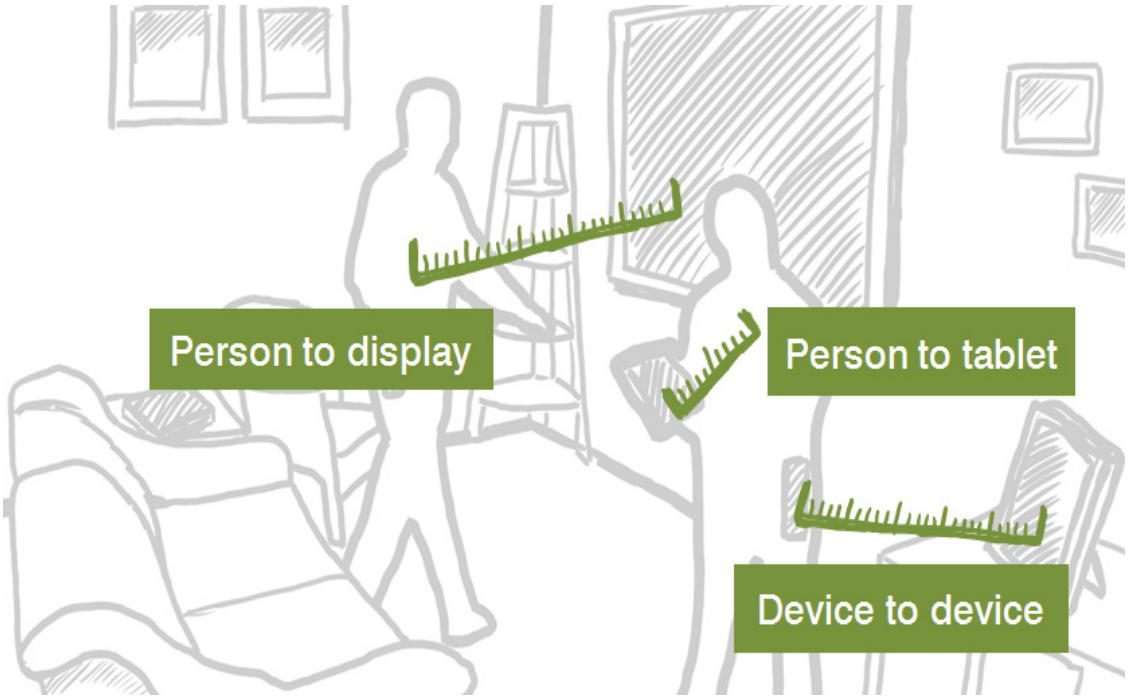


Figure 3.7 Examples of distance relationships in ubicomp ecologies (cf. to Figure 1.1).

Furthermore, distance can be updated at discrete levels or continuously. For example, a designer can consider discrete distance updates at a single threshold when in very close proximity (Figure 3.8a), when person or device enters a distance threshold at a larger distance (Figure 3.8b), or when an entity passes from one zone to another (Figure 3.8c) – such as with Hall's four proxemic zones (Hall 1966). Alternatively, distance can be updated *continuously* as entities move in space (Aiello 1987; Knowles 1989), either as a linear mapping of distance to engagement (Figure 3.8d), or any other mapping such as a logarithmic function (Figure 3.8e). Discrete and continuous distances can also be intermixed, for example, such as defining discrete zones at far distances but shifting to continuous measures when a close threshold has been reached (such as shown in Figure 3.8 f and g).

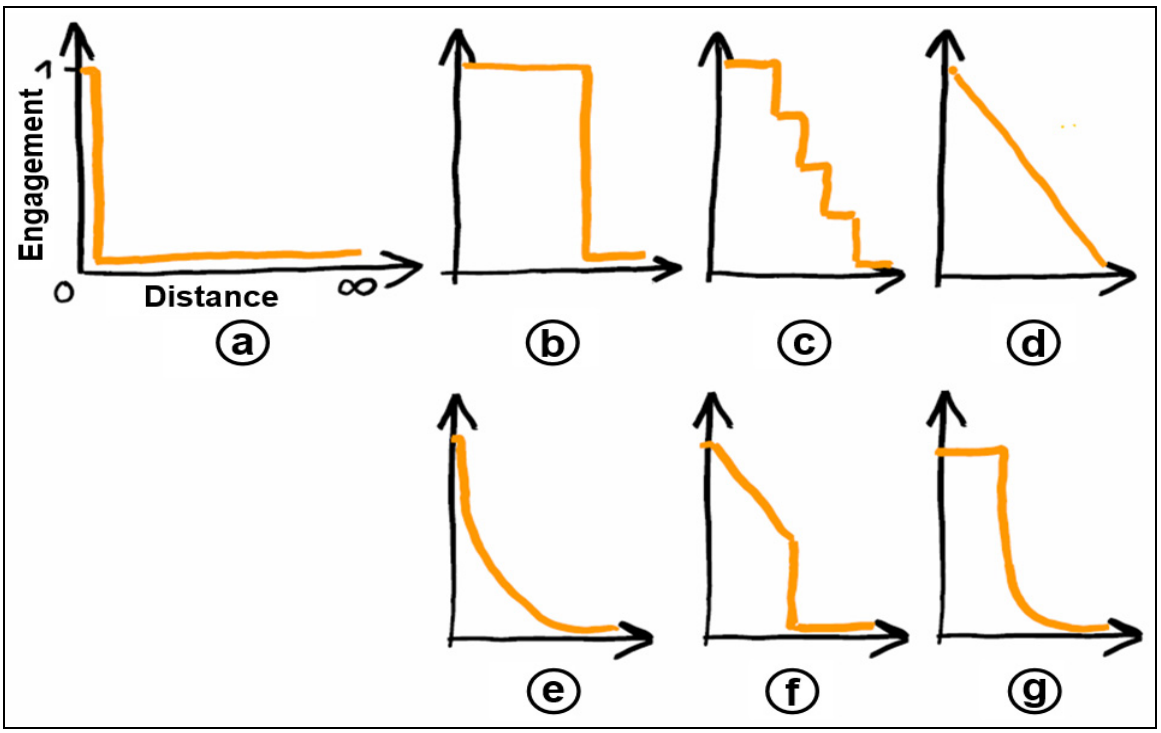
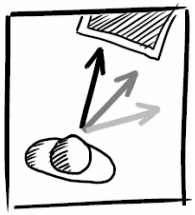


Figure 3.8 Using distance measures to determine a person's level of engagement with a ubicomp system: from discrete to continuous measures.

3.2.1.2. Orientation



Orientation provides information about which direction an entity is facing. For people, this includes a person's body orientation (Sommer 1969), but also the orientation of the face and limbs like arms or legs (which can be summarized in a body posture). It can also include the orientation of a person's gaze towards another entity (Argyle and Dean 1965). For devices or objects, this might require a well-defined front (e.g., the front-facing side of a display), and can also include further more refined measures of orientation (e.g., the body, face, and gaze orientation of a humanoid robot).

Orientation can be relative between two or more entities, or absolute when relative to a fixed point in the environment. They can be described in both qualitative (e.g., "facing towards" or "facing away", Figure 3.9a) and quantitative terms (e.g., the measured angle, Figure 3.9b). Such quantitative descriptions can, for example, be expressed as pitch/roll/yaw angle of one object relative to another.

Given a known quantitative orientation, it is possible determine where a ray cast from one entity would intersect with another entity (*ray casting*). Ray casting is useful, for example, to determine what a person is pointing at as they stretch out their arm to point at an object in the environment (Figure 3.9c), or even what they are looking at (i.e., the focus of one's gaze).

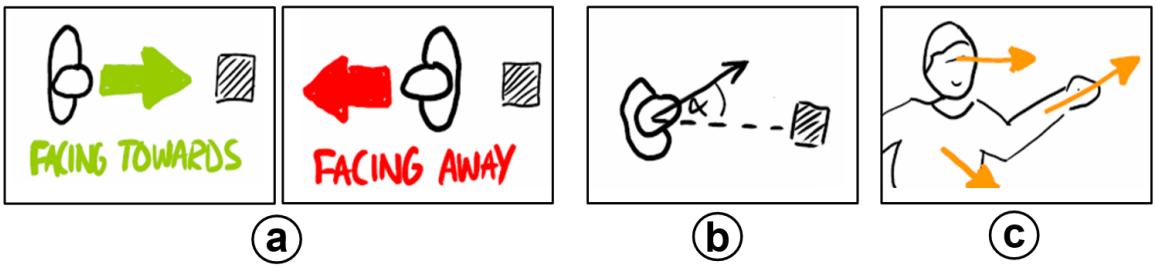
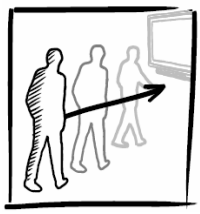


Figure 3.9 Considering orientation: (a) facing direction, (b) exact orientation angles, (c) orientation of a person's body, head, and arms.

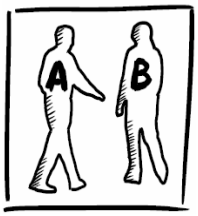
3.2.1.3. Movement and Motion



Measurement of *movements* and *motion* lets us understand the changes of position and orientation of an entity over time (e.g., a person that is walking towards a device, or a device that is moved closer to another device).

If we know one's motion over time, we can also derive *velocity* and *acceleration*. These changes in movements, for example, reveal how a person is approaching a particular device or object. Different actions of the system can then be taken depending on (for example) the speed of motion, and/or whether one entity is moving and turning towards vs. away from another entity.

3.2.1.4. Identity



Identity uniquely describes the entities in the space. The most detailed information provides the exact identity of a person or object (e.g., "Fred", "Person A", "Fred's Cell phone"), as well as data associated with that entity. Other less detailed forms of identity are possible, such as identifying a category precisely (e.g., "book", "person"), or roughly ("non-

digital object"), or even affiliation to a group (e.g., "family member", "visitor"). The identity information is important to discriminate one entity from another, where the required granularity of identity information depends on the envisioned system functionality. For example, crude identity categories (e.g., "a person") are sufficient if a device is supposed to react to any person approaching it, but unique identification (e.g., a person's name) might be necessary to display personalized content.

3.2.1.5. Location

In contrast to the earlier mentioned *distance*, the dimension *location* describes the qualitative aspects of the place where the interaction takes place. That is, it characterizes the location (such as home vs. office settings), describes features in the fixed (e.g., room layout, entryways) or semi-fixed (e.g., furniture positions) feature space, and provides meta-information such social practices and context of use of that space by the entities seen within it. This location information is important, as the meaning applied to the four other inter-entity measures may depend on the contextual location.

3.2.2 Applying Dimensions to Ubicomp Interaction Design

As part of designing proxemic interactions, developers can then use the information provided across these five dimensions to drive possible interactive behaviours of the system. The following summary of the Proxemic Interactions framework (Figure 3.10) demonstrates a possible sequence of using all five dimensions in practice for interaction design.

First, a sensing technology (for example one of the tracking systems reviewed in Chapter 2) provides spatial sensing data of tracked entities (Figure 3.10a) in the ubicomp ecology: people, devices, and features of the environment (left side of Figure 3.10). Next, considering the location information (Figure 3.10b) provides context of the setting (e.g., home vs. office), but also details about semi-fixed features (e.g., a large interactive display) and fixed features (e.g., the door of the room). Next, the identity information about entities allows the system either to differentiate between people and devices that

are present in the ubicomp ecology, or – at a finer granularity – gives unique names of the two people and devices (Figure 3.10c). These two last steps together provide us with a list of entities currently recognized by the system. Then, a system can determine the relative distance between those entities (Figure 3.10d), for example, how close any of the tablet devices to each person, or the distance between a person and the large screen.

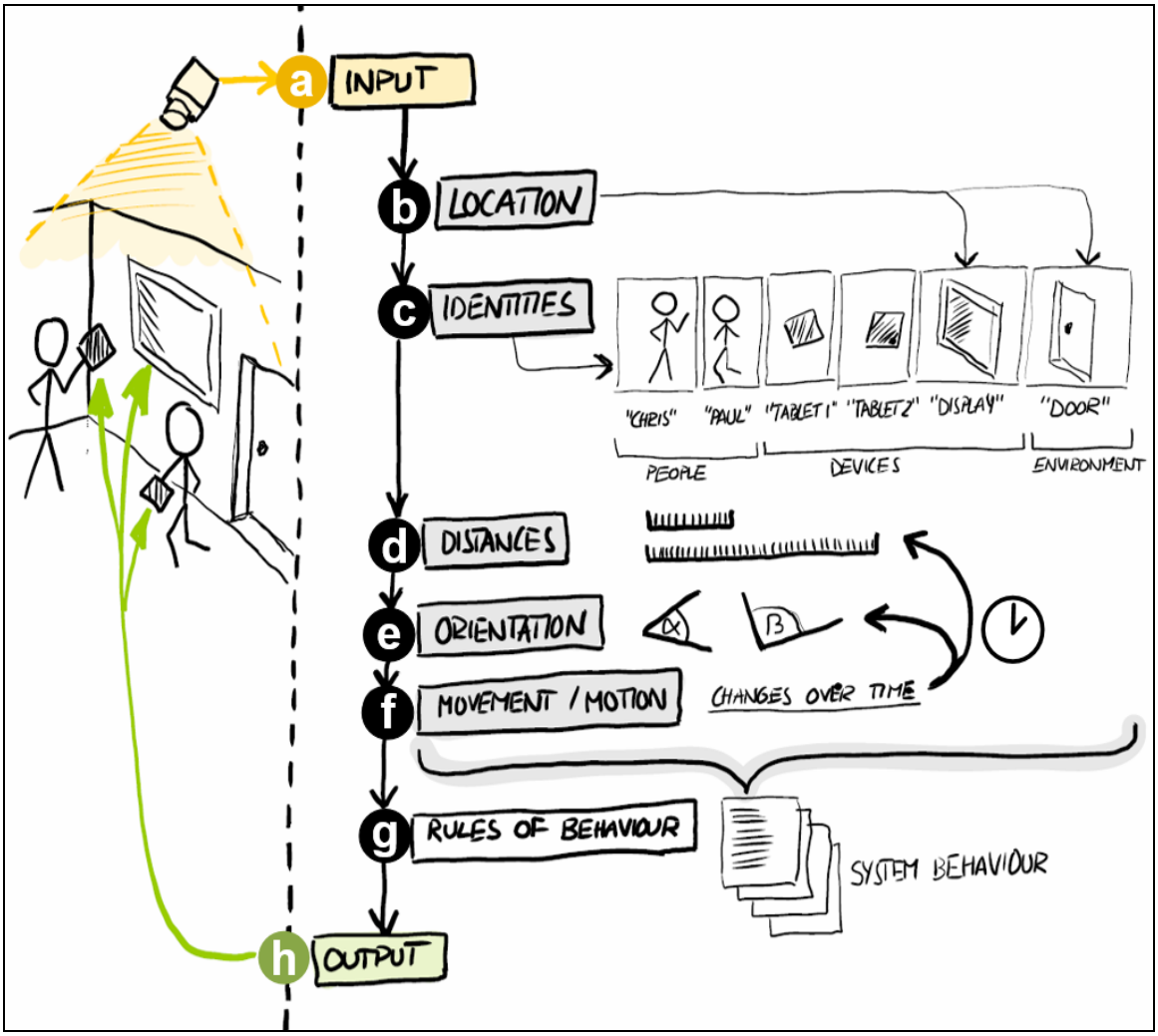


Figure 3.10 Proxemic Interaction framework.

The next step determines orientation between entities (Figure 3.10e); for example, if a person is facing towards the display or away from it. By determining changes of these two dimensions over time, we can determine the movement and motion information (Figure 3.10f). This provides, for instance, information about a person entering through

the door and moving quickly towards the display. The information of all the five stages are then interpreted in the next step that defines the rules of behaviour (Figure 3.10g). These rules define how the system interprets proxemic relationships between entities, and how they are translated into system actions (i.e., the reactive behaviour). Depending on these rules, the system then triggers appropriate output actions, e.g., changing the content displayed on any of the tablets or the large interactive screen (Figure 3.10h).

3.3 Conclusion

This chapter explained the operationalization of proxemics for ubicomp interaction design in form of the Proxemic Interaction framework. The intention is to inform ubicomp designers implementing proxemic-aware devices about important proxemic dimensions to consider for the design and review methods of how those can be applied to challenges in ubicomp interaction. By focusing in particular on how the knowledge in the five proxemic dimensions (distance, orientation, movement, identity, and location) can be applied to ubicomp interaction, our intention is to open up a new perspective onto how proxemics can be considered when designing new ubicomp systems that react seamlessly and appropriately to people's expectations.

The next chapter will explain of how to consider information in each of the five dimensions (step b-f) to address ubicomp design challenges. This will also provide examples of the potential rules of behaviour interpreting proxemic input. Later, in Part II of this dissertation, we explain how the Proximity Toolkit actually translates raw input into the five proxemic dimensions (i.e., it explains step (a) in Figure 3.10). The three chapters of Part III will then explain how to use all five dimensions together in the design of proxemic-aware systems.

Chapter 4. Exploiting Proxemics for Addressing Ubicomp Interaction Design Challenges

Other ubicomp systems have exploited a subset of the five dimensions of proxemic interactions we introduced in Chapter 3. Yet very few make use of all of them, let alone consider them as characterizing the interplay between entities in a ubicomp ecology. In this chapter[21] we describe six ubicomp interaction design challenges, and revisit some of the related work from Chapter 2 and our own[22] applications from Chapter 7, 8, and 9 to detail how particular systems use aspects of these five dimensions to address these challenges. We categorize these possible interactions into a series of proxemic interaction themes as part of each of the six design challenges. The main purpose of this chapter is to give the reader a better understanding of the kinds of possible proxemic interactions, but also emphasize how to use proxemic interactions as a lens to categorize prior work.

In Section §4.1 we describe the six ubicomp interaction challenges. Next, we show the potential of using proxemic interactions to address six key design challenges of ubicomp interaction: revealing interaction possibilities (§4.2), directing actions (§4.3),

---

[21] Portions of this chapter are published as:

Marquardt, N. and Greenberg, S. (2012) Informing the Design of Proxemic Interactions. In *IEEE Pervasive Computing*, 11, 2. April 2012. Joe Paradiso, Trevor Pering, Albrecht Schmidt, Eds., pages 14-23. © 2012 IEEE.

[22] For literary reasons and the flow of this dissertation we decided to anticipate some techniques that we describe in more detail later in Chapter 7, 8, and 9

establishing connections (§4.4), providing feedback (§4.5), preventing and correcting mistakes (§4.6), and managing privacy and security (§4.7). We relate both the theory and dimensions (described in Chapter 3) to the design challenges, and partially situate a sampling of prior systems summarized in Chapter 2 within that setting. Finally, in Section §4.8 we discuss the use of the interaction themes and conclude.

4.1 Ubicomp Interaction Design Challenges

Designing ubicomp applications (such as those reviewed in Chapter 2) that seamlessly fit into people's environments and social practices remains a difficult task for developers. From an end user's perspective, we identify six core challenges related to designing embodied and seamless ubicomp interactions, inspired by Bellotti et al.'s (2002) design considerations for sensing systems and augmented by other analytical and reflective ubicomp discussions (e.g., Dey, 2010; Bardram and Friday, 2010). While this selection does not cover all ubicomp design challenges (e.g., scalability, graceful degradation, evaluations, see: Abowd and Mynatt, 2000; Leahu et al., 2008), we focus in particular on the ones with the highest relevance to proxemic interactions.

Challenge 1. Revealing interaction possibilities. Norman (1988) appropriated Gibson's (1977) notion of *affordances* to describe how an object's visuals can 'suggest' how it might be used. Traditional GUIs exploited visual affordances to design interface elements that suggested their use and possible actions; they worked, because they could assume that they were in the foreground of a user's attention, i.e., the person was watching the screen. Yet this cannot be directly applied to ubicomp, as ubicomp assumes that technology can be integrated into the everyday environment in a way that it 'disappears', or is present in the just-perceptible periphery of our attention, and that it is able to fluently move into the center of our attention as needed (Weiser 1991; Bardram and Friday 2010; Buxton 1995; Cooperstock et al. 1997). This introduces the challenge: how can technology be designed to reveal the interaction possibilities appropriate when it is not only in the background of a person's attention, but during the transition of it moving into the foreground?

Challenge 2. Directing actions. Input to a single traditional device is straightforward, as it usually comes through a dedicated input device (e.g., a mouse, keyboard, touch surface). Yet ubicomp can be different. Input may be detached from a particular device. Possible actions can be performed through speech, gestures, eye gaze and other alternative options. One problem is that the device has to somehow discern whether that action is actually a directive to the system, or whether it should be ignored because it is just part of a person's everyday actions (e.g., a voice command *vs.* social talk; a command gesture *vs.* a gesture or movement made in the course of doing other things) (Ju et al. 2008; Bellotti et al. 2002; Dey 2010). The problem of directing the actions to a particular device is even more problematic when there are large quantities of devices present in the local ecology, for the system has to discern which device (or set of devices) should respond to a person's directed action.

Challenge 3. Establishing connections. Device connectivity is a significant challenge in ubicomp. Technical issues aside, the ad-hoc nature of ubiquitous computing means that people need to somehow control (albeit seamlessly) how one device connects to another device in a way that reflects their interaction needs while still safeguarding privacy and security (for example to transfer digital content from a personal smartphone to a large public screen) (Bardram and Friday 2010). This challenge is compounded by the potential and perhaps unpredictable interplay between a large numbers of digital devices. Some may be personal (a smart phone), others may belong to the inhabitants of a space (a home's picture frame), and others may be public (e.g., a public wall display). Their form factor also affects their mobility, which in turn can suggest different factors affecting how they should establish connections.

Challenge 4. Providing feedback. Appropriate feedback is a mainstay of traditional GUI interaction design. Yet as ubicomp interfaces move away from the traditional desktop computer setting, it becomes even more important to provide feedback about the current status of the application, its interpretation of user input, or the occurrence of errors (Ju et al. 2008; Bellotti et al. 2002; Abowd et al. 2002; Dey 2010). To complicate matters, ubicomp systems have to consider that people's attention in regards to the ubicomp technology might switch between foreground and background.

Challenge 5. Avoiding and correcting mistakes. When mistakes or errors happen, the system should provide options for a person to correct these mistakes (Ju et al. 2008; Bellotti et al. 2002). As many ubicomp systems use some kind of sensing technology to monitor people's actions, such errors and misinterpretation of sensor data are even more likely to occur in ubicomp settings than with traditional computers.

Challenge 6. Managing privacy and security. A large issue in ubicomp is that as the number of potential interactions with technology increase, so too do the risks to privacy and the need for greater security (Langheinrich 2010). The question is how can the system protect privacy sensitive information and handle the access to information, while at the same time not get in the way of all the positive offerings of ubicomp mentioned in Challenges 1-5?

We now revisit each of these design challenges, where we speculate – with examples drawn from the literature reviewed in Chapter 2 – how knowledge of proxemics as gathered by the 5 dimensions described in Chapter 3 can mitigate problems inherent in each challenge. These examples are merely a starting point, where their contributions are re-framed within each challenge, and where they hint at the potential of future proxemic interaction designs (such as those discussed in Part III).

4.2 Revisiting Challenge 1: Revealing Interaction Possibilities

To address this challenge, a system must offer possible actions (Ju et al. 2008) that afford *seamless transitions* from background to foreground interaction (Buxton 1995). This concept is somewhat similar to how people approaching each other exchange greet and begin communicating through various signals (eye gaze, body language and talk), where signals and possible actions vary appropriately across this greeting phase. Similarly, ubicomp should 'greet' other entities by revealing interaction possibilities that match what is possible at the moment. Several strategies to accomplish this are described below.

4.2.1 Reacting to the Presence and Approach of People

At the most basic level, if a system can sense the presence and approach of people, it can use that information to reveal possible interactions.

Various prior systems do this, but only as a binary measure: if it detects a person it marks them as 'present', otherwise 'absent'. In response to this binary measure, systems would trigger an appropriate action. Buxton (1995) describes examples of smart light switches used motion detectors to infer presence and then turn lights on and off in response. Greenberg et al. (2011) brings up the example of a desktop computer screen that uses a proximity sensor to capture a person's distance from the display, and from that either activated the screen or went into a power-save mode. Both systems 'reveal' interaction possibilities implicitly: the first by illuminating the room, and the second by showing the desktop computer is on and ready to go.

Other systems detect and use presence information to explicitly reveal interaction possibilities. Consider *ActiveBadges* – identity tags worn by individuals – where badge (and thus a particular person's) location is tracked at a room-level within a building (Want et al. 1992). Its inventors exploited this presence and identity information to offer personalized computing services at that person's current location, e.g., where their desktop computer display would 'follow' them to other rooms and appear on nearby screens. Similarly, the EasyLiving system (Barry Brumitt et al. 2000) selects custom media content when sensing the presence of a person in a particular room at home. Another popular example is a large screen that senses when a person enters a room, where the display not only turns on but tailors its contents to suggest its offerings (Vogel and Balakrishnan 2004). In our media player example (introduced in Chapter 7), for example, when a person crosses a threshold into a room, a splash screen appears revealing that the large display is a media player, and then offers several videos the person could select for watching. The system intentionally displays only a small number of videos using large graphics, to make it appropriate for viewing at a distance. If one person seated on a couch is already viewing a video while another person enters, the system displays its information differently, where it reveals what is being viewed with minimal disruption to the primary viewer. In terms of our dimensions, this media player

exploits both relative distance and identity to reveal appropriate interaction possibilities compared to just a binary notion of presence, i.e., it considers the room as an ecology. It uses people's *approach* across a threshold (the doorway), their *distance from the screen*, and their *presence relative to other fixed features* in the room (e.g., the couch).

4.2.2 Transition from Awareness to Interaction

In real life, people exploit proxemics cues as they greet and engage in social interaction. One may have peripheral awareness of the other while at a distance, become increasingly aware and engaged as the other turns towards and approaches them, and then begin to interact when within an appropriate proxemic region. Some public ambient displays apply a similar mechanism to engage people, where they trigger actions to attract a passer-by's attention, and progressively show more information and interaction possibilities as the person approaches and attends the display, ideally leading to foreground interaction by direct touch (Prante et al. 2003; Vogel and Balakrishnan 2004). The idea is that the passer-by notices the public display as it *implicitly* reacts to their presence, where it captures their attention and interest (as discussed in Challenge 1). Their attention is realized by moving closer and facing the display; the system also detects and react to that interest (Vogel and Balakrishnan 2004). In the media player (Chapter 7), for example, the number of videos, their size and associated text is adjusted as the person approaches the display, where it reveals more video selections and more information about those videos. A system such as this exploits distance, orientation and movement to infer a person passing by at a *larger distance*, then *turning towards* the display, then *approaching*, and finally standing directly *in front* of it.

4.2.3 Spatial Visualizations of Ubicomp Environments

In the physical world, we often know what is available simply by looking around us. To make this work in Ubicomp, we need to explicitly visualize otherwise hidden offerings on a device's screen(s), such as when one device is within range of another, its relative location, and what can subsequently be done between them. For example, Relate Gateways (Gellersen et al. 2009) place icons at the devices' screen border to represent

the type and location of surrounding devices relative to that device's position. Our cross-device techniques introduced in Chapter 8 also visualize the spatial relationships to nearby personal devices: if a person points their device towards the large screen, a graphic appears as a ray-cast 'projection' on that screen indicating its position and orientation. As the mobile device approaches and is oriented towards the large display, increasing detail about that device, its contents and its interaction possibilities are revealed. This possible reveal of information can, however, differ depending on the context of use and the fact of whether other devices are seen as public or semi-public vs. personal ones.

4.3 Revisiting Challenge 2: Directing Actions

While Challenge 1 concerns how a ubicomp system can reveal interaction possibilities to a person, Challenge 2 addresses how a person can in fact *direct* their input actions to a particular device.

4.3.1 Discrete Distance Zones for Interaction

Similar to how people tend to move closer to others when interacting (say, to begin a conversation), systems might accept user input only when the person has a certain distance relative to the device. Thus, to address a particular system, a person may have to approach and move closer to it. Some ambient display systems do this by realizing Hall's discrete proxemic zones as thresholds that adjust interaction possibilities according to which zone a person is in (Vogel and Balakrishnan 2004; Prante et al. 2003). Hello.Wall (Prante et al. 2003) introduced the notion of *distance-dependent semantics*, where the distance of an individual from the wall defined the kinds of interactions possible. While information on the large display can be seen from afar (Challenge 1), a person had to move closer to actually interact with it (e.g., to transfer information from a mobile device). Vogel et al. (2004) extended this concept, where they defined four proxemic zones of interaction around the large display. From far to close, these ranged from ambient display, to implicit, then subtle, and finally to personal interaction with the interactive calendar application. Each of these zones allowed particular kinds of

interaction with the display's contents. Similarly, Ju (2008) also defined four zones around an interactive whiteboard, where she allows certain actions only when a person is standing close to it. Our cross-device interaction techniques discussed in Chapter 8 show yet another promising approach, where each interaction zone explicitly support *different input modalities* that are appropriate to the person's distance from the display. When afar, people interact via pointing (ray casting) and by hand gestures direct touch when in close distance.

4.3.2 Considering Attention and Orientation

Instead of relying on only distance, the system can use other measures to infer a person's attention to it. This is the premise of *attentive user interfaces* (AUIs) that are designed to "support users' attentional capacities" (Vertegaal and Shell 2008). In one class of AUIs, the system reaction depends on whether a person is directing his or her attention to the device as detecting eye gaze (Vertegaal and Shell 2008), which in turn can be considered a very fine-grained measure of orientation. Our media player also exploits orientation as a measure of attention. When a person turns away from the video screen (to, say, read a magazine or talk to another person), the system pauses video playback, and resumes when they turn back towards it. Wang's proxemic-aware presenter (Chapter 10) also uses orientation as an indication of attention. If the presenter is facing towards the audience and away from the large display, a standard slide deck is shown. However, when the presenter turns towards the display, small navigation controls and speaking notes become visible at the side of the screen closest to the presenter.

4.3.3 Considering Location Features

Ubicomp systems are often embedded in people's everyday environments, surrounded by other physical objects and social meanings that comprise the ecology of that place. Inspired by research in context- and location-awareness (Schilit et al. 1994), our next concept emphasizes the importance of interpreting the physical setting where an interaction takes place (Dourish 2001b). In particular, people's relationships to fixed and semi fixed features (as defined by Hall, 1966) can be indicators for directing actions to a

particular ubicomp system. In Brumitt et al.'s (2000) EasyLiving project the geometric relationship of people to semi fixed features (e.g., a couch) is considered to determine which screen is activated to display information to a person. Similarly, in our media player (Chapter 7), the ubicomp system not only monitors a person's proxemic relationship towards a device, but also to that person's distance to other fixed and semi-fixed features in the ecology. If a person selects a video and then sits on the couch, that is interpreted as an indicator that she is ready to watch the currently selected video and thus video playback begin. However, if the person instead moves to the doorway, that is interpreted as an indicator that she is no longer interested, and the system shuts down. In both cases, the distance from the person to the screen is the same, but her location in the room's ecology is different. These examples are a starting point of how to consider people's relationships to fixed and semi fixed features in the environment.

4.3.4 Considering Motion Trajectories

Going straight towards another person – or instead quickly passing by – are also proxemic cues that we implicitly interpret in everyday interactions with others. Similarly, ubicomp systems can interpret people's and device's motions for directing actions. For example, Vogel and Balakrishnan's (2004) ambient display ignores people quickly passing by, but reacts to (and gathers input) from people walking straight towards it. Motion cues can be quite fine-grained, where it can exploit distance, orientation and velocity as well as how each changes over time.

4.3.5 Adapt to Number of Nearby Devices

A system's interpretation of a person's actions can also depend on the number of other nearby devices that it can sense. To illustrate, a user of Swindells et al.'s (2002) gesturePen triggers interaction between two devices by pointing his device to the chosen one. We can extend this to help one choose between a large number of devices by applying distance-or identity-based filtering technique to limit the number of possible pointing targets, e.g., the system could require the person to move closer to their target up to the point where it can discriminate the desired one.

4.4 Revisiting Challenge 3: Establishing Connections between Devices

As suggested by our last example, people need to somehow control how one device connects to another device within a potentially large ecology of devices in a way that seamlessly supports their interaction needs while still safeguarding privacy and maintaining security. We do this naturally – the way we greet and move closer to one another via proxemics is essentially a negotiation to establish connections for communication.

4.4.1 Connection as a Consequence of Close Proximity

We can exploit distance, identity and even orientation to determine proxemic relationships between devices, and then establish connections between only those that are in close proximity. As opposed to directly connecting two devices with a cable, such wireless connections facilitate the spontaneous and lightweight transfer of information. Existing systems now do this, although most do so as a binary function (e.g., close = connected). Rekimoto et al.'s (2003) combination of near-field RFID communication and wireless networks allows inter-device communication only when two mobile devices are in close proximity. Alternately, physically bumping two devices together can activate a connection: the accelerometer signal produced by bumping identifies the devices (Hinckley 2003), and bumping can only occur as a consequence of direct touch. Another strategy exploits people's proximity to on another, where they communicate to synchronize an act that establishes the connection. One example is both simultaneous shaking their handheld device (Holmquist et al. 2001). Similarly, a stitching gesture can be used, where one person start a gesture on one device, which is then continued on the other; this can only be done if the devices are nearby (Ramos et al. 2009).

4.4.2 Progressive Connection Process

While the above systems are binary in nature, progressive connection processes are also possible. Kray et al.'s (2008) *group coordination negotiation* introduced spatial regions around mobile phones to establish and break device connections or initiate data transfer.

As a device moves across three discrete regions, a preview of a media transfer is first display, where transfer begins only after moving into a closer region. Our cross-device interaction techniques (Chapter 8) are somewhat similar, but it uses a continuous rather than discrete progression over distance. When a person holds a handheld media player in her hand, a subtle notification on the large screen indicates the connection possibility. As he moves closer to the screen, he sees the two devices connect, where the large display progressively reveals more information about the handheld's video content as icons. As the two devices move within touch distance, a touch interface appears that allows the person to transfer digital media either through pick and drop or by touching the handheld to one of the icons revealed on the large display.

4.5 Revisiting Challenge 4: Providing Feedback

Next, we discuss how to leverage proxemics for providing continuous feedback about a system's status or any errors that occur.

4.5.1 Adjusting Feedback Output

Due to the embedded nature of many ubicomp systems, there is often no graphical display for showing feedback to the user. Instead, output can be via visual lights, audible sounds, speech, or physically moving objects (like in many tangible user interfaces). Assuming a system knows the physical orientation and distance of a person, it can adjust the provided output to the person that it is addressing. The Listen Reader (Back et al. 2001), for example, adjusts the volume of the audio output depending on a person's proximity to a digitally augmented book. Similarly, in our media player (Chapter 7) a person sees large preview thumbnails of available videos when at a distance. The screen continuously shows more content as the person moves closer (and thus, can read more information).

4.5.2 Selecting Appropriate Feedback Modality

Furthermore a system can select the most appropriate output modality to a person (e.g., visual *vs.* audible) based on their proxemic relationship. For example, when the person is

facing away from a large screen, the system might use an audible signal as a notification. When the person is standing closer to the system facing the screen, visual output may be used instead.

4.5.3 Proxemic-dependent Reveal of Feedback

Details presented to a person can vary depending on the distance and/or orientation of the person relative to the system. He (2010 chapter 3), for example, introduced distance-dependent semantic zoom in an augmented reality energy viewer for the home. The feedback of energy use is adjusted based upon the viewer's proximity to rooms or appliances within a room (distance and orientation are detected through fiduciary tags). When holding the viewer outside a room's doorway, the energy use of that room as a whole is displayed. When the person moves into the room, the energy use of each appliance is seen as a coloured glow around it; as he moves closer to a particular appliance, details of that usage appear first as a text overlay and then as a graph.

4.6 Revisiting Challenge 5: Preventing and Correcting Mistakes

Our next design challenge addresses the question of how a person can correct errors, such as those that result from the system misinterpretation a person's action, or by the person performing an unintended action.

4.6.1 Inverting Actions

One technique allowing a person to correct a mistake (and thus undo a system's action) is by performing the *inverse/opposite* action. The system implicitly responds by reverting to the prior state. For example, in Vogel's and Balakrishnan's (2004) ambient display setting, when a person moves closer to the screen, personal calendar information is revealed. If the person didn't want this information made public, he just steps back (and thus performs the opposite action): the personal information disappears immediately.

Other proxemic dimensions can be exploited as well. For example, an action triggered by the person facing a screen can be stopped (or reverted) simply by turning away.

4.6.2 Explicit Action to Undo

Ju et al. (2008) present an opposing explicit strategy to undo actions. Her application runs on the interactive whiteboard, where it implicitly responds to people's actions. This can easily result in an unwanted action (for example: automatically moving a cluster of ink strokes to the side of the display to free up space). To correct this, the person moves closer to the screen (instead of stepping back, like in Vogel's system) and grabs the cluster of ink strokes to keep it from moving.

Of course, both the above techniques can be combined to override the system. In fact, Vogel used both in his system: a person can either use a set of simple hand gestures to trigger or stop certain system functions, or just step back from the screen to have the same effect.

4.6.3 Proxemic Safeguards

As a safeguard mechanism, actions with a high impact (e.g., deleting information, or resetting the system) could be restricted to occur only when a person is in very close proximity to a device. For example, while a person can manipulate information on an interactive whiteboard from a large distance by using remote gestures, she would have to move directly in front of the screen to delete data by (say) direct touch. Alternatively, such actions with high impact could even require a certain proxemic relationship in *multiple* dimensions. For example, the delete action could require a person to stand in close proximity to the screen *and* being oriented towards it *and* look at the screen simultaneously. The action could also be tentative and undoable as the person remains close by, where the person then has to manually commit the changes that otherwise would be reset as they move away.

4.7 Revisiting Challenge 6: Managing Privacy and Security

Next, we review techniques that apply proxemics for managing privacy and security in ubicomp systems (Langheinrich 2010).

4.7.1 Proximity-dependent Authentication

Access to ubicomp systems can be granted depending on the sensed proximity of people, devices, or other objects. Bardram (2005) discussed proximity-based user authentication allowing access to computers when approached by a person. The system is implemented through authentication tokens (e.g., pens) that wirelessly authorize the access of a person once in close proximity to the computer (i.e., the person stands in front of it). The *tangible security* (Chen and Sinclair 2008) approach uses the measured proximity between pairs of tokens to authenticate access. For example, a person obtains access to a cell phone only as long as the physical *security token* he carries remains in close proximity. If the phone is lost, strangers cannot access its contents as they do not have the security token. Mayrhofer et al. (2007) took this concept further, where his system leverages the shared knowledge (between the person and device) about spatial references to other devices in close proximity to authorize access. Furthermore, Rekimoto et al. (2003) combine near-field sensing techniques (such as RFID or Infrared) with wireless network communication to seamlessly establish device to device connections. Near-field communication initiates the wireless communication channel. That is, a person must not only bring his device close to the other device, but also make sure they are in line of sight before the connection is established.

4.7.2 Distance-dependent Information Disclosure

Another strategy uses distance between entities to determine the amount of information that is shared between them. This approach suggests that "*distance implies distrust*" (Fishkin et al. 2005), and vice versa: *closer proximity implies trust*. For example, the *distance-dependent disclosure RFID tags* (Marquardt et al. 2010) vary information transmitted between the tag and the reader depending on the distance between them. The closer the

tag is to the reader, the more information is revealed. Similarly, Vogel and Balakrishnan's (2004) public calendar reveals a person's personal calendar information only when the person is moving very close to the display. The information disappears immediately once the person steps back away from the display.

4.7.3 Proxemic-aware Privacy Mechanisms

While these approaches consider *distance* as a factor affecting access, the techniques could be further refined by considering other proxemic dimensions such as *orientation*, *identity*, or *location*. A person's body, face, or gaze orientation can affect the amount of information shared. For example, privacy-sensitive information shown on the display of a proxemic-aware mobile device could be visible as long as the person is looking at the screen, but hidden once looking away. Alternatively, the information might disappear once the system notices *another person* looking at the display. By considering the *identity* dimension, a system would be able to use relaxed privacy and security settings when a person is alone, but switch to more restrictive privacy and security settings when it detects any other people or devices around them (e.g., in a crowded setting). By considering *location*, a mobile ubicomp device could adjust its security setting depending on the type of environment; using higher level settings in an open office (where strangers may come by and try to access the device), but lower security level when at home (which is usually a much more trusted setting).

4.7.4 Considering People's Expectations of Personal Space

Altman's (1975) theory considers personal space as a protection mechanism for maintaining a certain level of privacy. This could be leveraged to design systems that respect people's expectations of personal space. That is, the ubicomp system can influence the simultaneous interaction of multiple people in a way that maintains such levels of privacy for everyone involved. To illustrate, let us revisit Vogel's (2004) public ambient display. When people move closer to the display, they get more details about their own personal calendar visible on the screen. Thus, people stand next to each other viewing their personal calendars. When considering Altman's theory of balancing

privacy through proxemics, the system could be designed to *separate* the large screen interaction areas of the two people. For instance, the areas for viewing personal calendars could be displayed where it depends upon a minimum distance between those people.

4.8 Discussion and Conclusion

Overall, we concentrated on a few example systems and techniques to illustrate how the ubicomp interaction design challenges can be addressed by considering the five proxemic dimensions identified earlier. These were chosen to inspire ubicomp interaction designs. They are not meant to be a complete review, nor as a catalog of solutions.

We also recognize that a single technique can serve different purposes across these challenges. For example, the idea of progressive reveal of information as a person approaches a display reveals interaction possibilities (Challenge 1), affords actions being directed to it (Challenge 2), is used to establish a connection (Challenge 3), provides feedback that it is responding to the person (Challenge 4), can be used to prevent and correct mistakes by inverting actions (Challenge 5), and helps people manage privacy and security simply by moving to adjust what information is visible (Challenge 6). We believe this to be one of the strengths of proxemic interactions: if techniques are developed with social expectations of proxemics in mind, they can likely be applied as a universal way to mediate many challenges in ubicomp.

PART II – RAPIDLY PROTOTYPING PROXEMIC INTERACTIONS

The above techniques of Proxemic Interactions all depend on the system's ability to sense proxemic relationships of people, devices, and the environment. A variety of technologies for sensing proxemic relationships exist (surveyed in Chapter 6): using computer vision, radio frequency standards, ultrasonic, infrared, and a variety of other approaches. But all of these technologies require corresponding software infrastructures that interpret the sensed data. In the ubicomp research area, a variety of such infrastructures has been developed in the past; such as for fusing sensor data from different sources (Antifakos and Schiele 2002), processing low level data and aggregate to higher level events (Salber et al. 1999), or providing infrastructures for distributed access to sensor data (Krumm and Hinckley 2004). However, an important challenge remains to provide developers easy access to this sensing technology (and therewith any derived proxemic features) from within their developed applications.

To address this challenge, in this next part of this dissertation we introduce the Proximity Toolkit that simplifies access to proxemic information from within ubicomp software, so that ubicomp designers can focus on the actual exploration of Proxemic Interaction techniques, rather than dealing with low level hardware issues or sensor processing.

Chapter 5. The Proximity Toolkit

In Chapter 3, we argued that we can design proxemic interaction systems that will let people exploit their natural understanding of their proxemic relations with their nearby digital devices, thus facilitating more seamless and natural interactions. As explained earlier in Chapter 4, a handful of researchers have already explored proxemic-aware interactive systems. These range from spatially aware mobile devices (Kortuem et al. 2005), office whiteboards (Ju et al. 2008), public art installations (Snibbe and Raffle 2009), to large public ambient displays (Vogel and Balakrishnan 2004). This previous research in proxemic interaction opened up a promising direction of how to mediate people's interaction with ubicomp technology based on proxemic relationships. The caveat is that they are all just starting points to how we can integrate proxemic measures into interaction design. Further explorative research – including the development and evaluation of actual proxemic-aware systems – will help to refine our understanding of how proxemic theories apply to ubicomp.

Building proxemic-aware systems, however, is difficult. Off the shelf technologies rarely come equipped with the necessary hardware or software for sending even rudimentary proxemic relationships. Even if sensing hardware is incorporated into our technologies (e.g., via camera-based tracking systems), writing software to translate low-level sensing information into proxemic information is difficult: this task often requires challenging calibrations of sensing input and complex mathematical calculations to process sensor data, as well as developing a reusable API. This introduces a high threshold for those wishing to develop proxemic interaction systems. As a result, most do not bother. Of the few that do, they spend most of their time with low-level implementation details to

actually access and process proxemic information *vs.* refining the interaction concepts and techniques of interest.

One of my thesis goals is to facilitate the rapid exploration of proxemic interaction techniques. To achieve this goal and to mitigate the above problems, we built the *Proximity Toolkit*. The Proximity Toolkit:

- transforms raw tracking data gathered from various hardware sensors (e.g., infra-red motion capturing systems, depth sensing cameras) into rich high-level proxemic information accessible via an event-driven object-oriented API;
- includes a *visual monitoring tool* that displays the physical environment as a live 3D scene and shows the proxemic relationships between entities within that scene (see Figure 5.1);
- contains a tool to record events generated by entities for later playback during testing and other tools to quickly calibrate hardware and software; and
- works with commonly available computers and software development kits (Windows development with Microsoft Visual Studio, .NET, and C# (Microsoft MSDN 2013)).

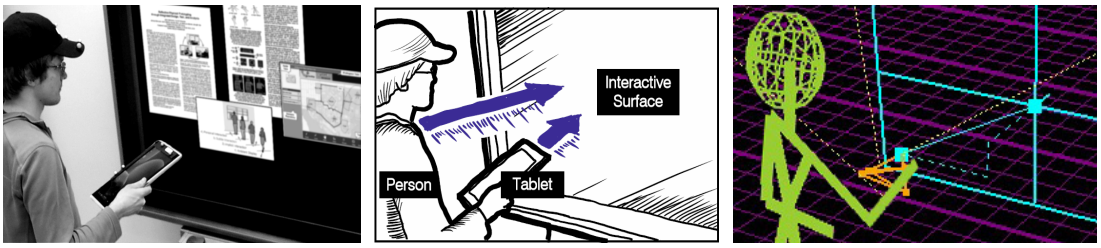


Figure 5.1 Left: three entities – person, tablet and vertical surface; Center: proxemic relationships between entities, e.g., orientation, distance, pointing rays; Right: visualizing these relationships in the Proximity Toolkit's visual monitoring tool.

The remainder of the chapter[23] is structured as follows: First, we discuss the motivation and requirements of ubicomp prototyping toolkits for proxemics and briefly review related toolkit work in HCI (§5.1). We then describe the challenges developers are facing when building proxemic-aware applications (§5.2). In Section §5.3 we introduce the design of the Proximity Toolkit, where we explain how the toolkit's visual monitoring tool is a starting point for developers to explore proxemic relationships and their relevance for their envisioned designs. Next, by using a running programming example we demonstrate how its event-driven API allows programmers to easily access the essential proxemic relationships between people, devices, objects, and the environment from within their code (§5.4). By revisiting each of the five proxemic dimensions from Chapter 3, we explain how these dimensions can be exploited in a proxemic-aware ubicomp application. In Section §5.5 we explain how additional tools help to facilitate the development process, and conclude in Section §5.6. The Proximity Toolkit video figure[24] demonstrates the toolkit in action.

Later in this dissertation, we will illustrate the versatility of the toolkit: first with proxemic-aware systems built as explorations that are part of this dissertation (Chapter 7 and 8), and second with projects built by students in ubicomp graduate courses (Chapter 10).

5.1 Ubicomp Prototyping Toolkits

In this section, we motivate the need for prototyping tools for ubicomp development, and for proxemic interactions in particular. As we will see, building proxemic

---

[23] Portions of this chapter are published as:

Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011) The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the ACM Symposium on User Interface Software and Technology – UIST'2011.* (Santa Barbara, CA, USA), ACM, October 16-18, pages 315-326.

[24] Video available for download at:

http://grouplab.cpsc.ucalgary.ca/grouplab/uploads/Publications/Publications/2011-ProximityToolkit.UIST.mov

interaction systems is difficult to do even when sensing hardware is available. We then sample and review work in three areas related to toolkit design: toolkit support for post-GUI development, ubicomp development architectures, and 3D spatial tracking.

5.1.1 Motivating Ubicomp Prototyping

One important goal of prototyping is to allow developers to rapidly construct working systems of their envisioned design ideas, and – maybe even most importantly – to rapidly iterate over *alternative* concept designs without investing too much effort into the implementation of only a single idea (Buxton 2007; Greenberg 2007). Buxton (2007) describes this process with the mantra of "*getting the right design*" (i.e., exploring alternative designs) which serves as a sharp contrast to "*getting the design right*" (i.e., refining one particular design).

Prototypes can be built in varying refinement along a spectrum from low fidelity to high fidelity. Low fidelity prototypes can be in the form paper and pencil sketches that, for example, show the layout of a novel interface, or the design of a new input sensing device. The more refined medium fidelity prototypes might consist of a cardboard-built model of that novel device. High fidelity prototypes may be in the form of a limited working system implemented to the point that a person can fully interact with particular features of interest, and where the system responds accordingly. Each prototyping technique has value in different parts of the development cycle. Low fidelity techniques are usually more important in the very early stages (especially to allow ideation and broad consideration of those ideas), while higher fidelity prototypes become crucial during the further fine-grained exploration of a particular design. Even so, high fidelity prototyping can be challenging. It is often time consuming and expensive (Rudd et al. 1996). A high fidelity prototype – especially in novel areas – may require new infrastructure to be built. Furthermore, once a programmer successfully built a system tackling all the low level implementation issues, he or she is less likely to explore alternative ideas, as it would require significant additional effort. This results in less evolutionary development and dissemination in any novel interactive systems research area (Greenberg 2007).

As earlier explained by Greenberg (2007), Gaine's BRETAM phenomenological model of how science technology develops over time can help to better understand the importance of this evolutionary and replicative research in the early stages of a new technology exploration. Essentially, the model (shown in Figure 5.2) illustrates the life cycle of technology-oriented research. First, it begins with a *Breakthrough* describing an inventor's new concept. For example, novel multi-touch screen technology were first invented around 1981/1982, where it opened up a new way for people to interact with on-screen digital content (Buxton 2013). *Replication* occurs when other researchers begin to re-create and iteratively refine that breakthrough idea. Due to the complexity of using cutting edge technology, this can be a long process, where it is often driven by research labs who are willing to take on the burden of exploring these technologies. Returning to multi-touch, for example, many research labs replicated multi-touch technology throughout the 1990's and 2000's, and at the same time developed a variety of interaction techniques based on it. As researchers gained experience, they were able to generalize the lessons learnt as *Empirical* design rules, for example, what comprises a good multi-touch gesture set. As knowledge progresses, *theories* are developed that encapsulate the gained insights done through this empirical research (for example, the theory of bimanual input as applied to multi-touch). Finally, the combination of both empirical and theoretical models make it possible to *Automate* manufacturing processes, leading to a *Mature* technology. As for multi-touch technology, these last steps took place mostly during the first decade of the twenty-first century, leading to mature multi-touch technologies as used in mobile phones, and in other large interactive surfaces (Buxton 2013).

Returning to prototyping, it is of special importance to the Replication stage: without the right developer tools (i.e., the right building blocks), this stage of the process is throttled, as it is too difficult for developers to explore their envisioned ideas and refinements of the technology (Greenberg 2007).

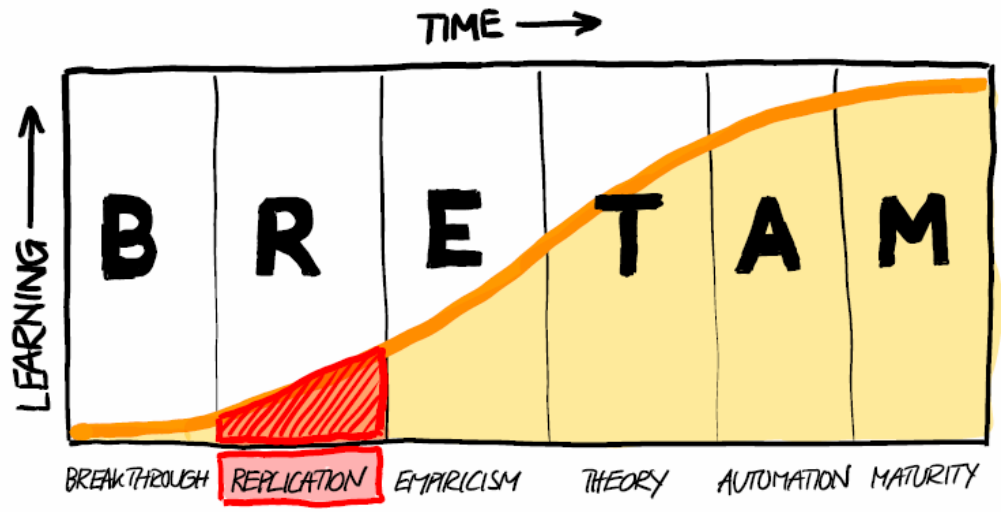


Figure 5.2 BRETAM model of technology-oriented research (Gaines 1991), with the highlighted Replication stage where prototyping techniques are most relevant.

5.1.2 Toolkits in Human-Computer Interaction

Prototyping toolkits have a long history of enabling the rapid creation and exploration of interactive systems – from desktop applications to groupware to ubiquitous computing installations. Greenberg (2007) emphasizes that "*by removing low-level implementation burdens and supplying appropriate building blocks, toolkits give people a 'language' to think about these new interfaces, which in turn allows them to concentrate on creative designs*". With a flexible and powerful toolkit, a programmer can rapidly create new envisioned systems, replicate existing ideas by others, and refine these designs iteratively. Simplicity and the earlier mentioned low threshold are key for toolkit design.

As a lesson learned from their extensive toolkit research, Myers et al. (2000) explain "*that in practice high-threshold systems have not been adopted because their intended audience never makes it past the initial threshold to productivity, so tools must provide an easy entry and smooth path to increased power.*" By hiding the low level complexities of a new technology, a toolkit can effectively lower the barrier for a programmer to dive into an iterative exploration of novel system designs using that technology. Myers et al. (2000) introduced the terms *threshold* and *ceiling* to describe essential characteristics of toolkit designs. The threshold describes how difficult (or easy) it is for a programmer to learn and use the toolkit, while

the ceiling describes how much can be done with the system (i.e., if sophisticated and comprehensive applications can be built). One challenge in toolkit design is to aim for a *low threshold*, but simultaneously maintain a relatively *high ceiling* (Myers et al. 2000). Resnick et al. (2005) later added a third goal for effective tool design: *wide walls*. That is, the prototyping tools should "*support and suggest a wide range of explorations*" and different kind of projects (Resnick et al. 2005). As a strategy to achieve all three of these goals when designing prototyping tools, the authors suggest to include elements and features that can be used in many different ways and combinations. While these principles apply to HCI toolkit design in general, we will now focus on ubicomp toolkits in particular.

5.1.3 Sensor-Based Ubicomp Toolkits

Several ubicomp development toolkits facilitate the prototyping of physical and tangible user interfaces that bridge the connection between the digital and physical world (Ishii and Ullmer 1997). For example, *Phidgets* (Greenberg and Fitchett 2001) and the *iStuff* toolkit (Ballagas et al. 2003) provide physical building blocks (buttons, sensors, motors) that programmers can easily address from within their software. In practice, this usually means that developers connect a variety of sensors and actuators to their computer and easily access the sensed information (or control actuators) through an easy-to-use application programming interface (API). *Shared Phidgets* took this concept further by simplifying the prototyping of *distributed* (i.e. remote located) physical user interfaces and by providing an environment that displays an interactive visual representing the running system (Marquardt and Greenberg 2007). *VoodooIO* also provides hardware building blocks to transform everyday surfaces into interactive control areas: programmers can stick sensors and actuators into a layered substrate surface and directly access them from within their software (Villar and Gellersen 2007).

The visual authoring environment in *dTools* (Hartmann et al. 2006) brought similar concepts to interaction designers. This authoring environment provides visual proxy representations for each connected sensor and actuator, and allows designers to visually compose the envisioned behaviour of the interactive system by connecting the proxy components and specifying how they should react to input events. Similarly, the Calder

toolkit makes physical components easily accessible from within design prototyping tools such as Macromedia Director (Lee et al. 2004). Other toolkits simplified the integration of computer vision techniques into novel user interfaces, such as *PapierMache* (Klemmer et al. 2004).

In summary, these toolkits (and others not mentioned here) were essential for many of the recent explorations in the design of physical user interfaces and ubicomp interactions. They drastically simplified access to physical sensors and actuators from within developed ubicomp programs. Similar to the Proximity Toolkit's design goals, they lowered the learning threshold while trying to maintain a high ceiling of what can be possibly built. However, none of these toolkits directly support proxemic interactions.

5.1.4 Ubicomp Development Architectures

On a somewhat higher level of abstraction, Dey et al. (2001) introduced the *Context Toolkit* as an architecture to compose context-aware ubicomp systems. They provide *context widgets* as encapsulated building blocks, working in conjunction with *generators*, *interpreters*, or *aggregators*. Generators are the providers of incoming sensor data and generate events for further processing. Interpreters filter and transform this incoming data. Aggregators take input from multiple sources to trigger higher level events in case certain conditions are met. The context toolkit allows the composition of new applications through a concatenation of these basic higher level components – and thus facilitates scaffolding approaches. Matthews et al. (2004) later applied similar concepts to the programming of peripheral ambient displays.

Other systems facilitate access to location information of devices in ubicomp environments. For example, Hightower et al.'s (2002) *Location Stack* fuses the input data from various sources to a coherent location data model. Krumm and Hinckley's (2004) *NearMe* wireless proximity server derives the position of devices from their 802.11 network connections (without requiring calibration), and thus informs devices about

any other devices nearby. Li's (2004) *Topiary* introduced prototyping tools for location-enhanced applications.

In summary, these architectures and toolkits have been essential building blocks for the exploration of ubicomp applications, such as peripheral displays, augmented whiteboards, or context-aware reminders (e.g., Dey et al. 2001). Yet, only few of the proxemic dimensions from Chapter 3 have been considered so far within these systems and toolkits – most focus on measuring a person's current location or relative distance, and even then most have limited fidelity and accuracy.

5.1.5 3D Spatial Tracking Toolkits

Few development toolkits support the exploration of novel interfaces considering the presence, movements, and orientation of people, objects, and devices in 3-dimensional (3D) space. For example, some toolkits allow development of augmented reality (AR) applications. To illustrate, Feiner's (1997) prototyping system allows exploration of novel mobile augmented reality experiences (e.g., with a head mounted 3D display, or a mobile tablet like device). This was developed further in *Open Tracker* (Reitmayr and Schmalstieg 2005), *DART* (MacIntyre et al. 2004), and Sandor and Klinker's (2005) prototyping environment for handheld-based AR applications. These toolkits mostly focus on supporting augmented reality applications running on mobile devices, and not on ubicomp ecologies in small rooms. Some commercial systems track 3D data of objects. For example, the Vicon Nexus software gives access to 3D spatial information of tracked objects (Vicon Motion Systems 2013). This information, however, only includes low level position data, which developers need to process manually in order to gain insights into proxemic relationships.

The Proximity Toolkit builds on this prior work. Like post-GUI toolkits, it bridges the connection between the virtual and real world, but in this case by tracking proxemic information. Similarly, it extends ubicomp architectures and 3D spatial tracking by capturing and providing fine-grained information about 3D proxemic relationships in small ubicomp spaces (i.e., not only location, but also orientation, pointing, identity, etc.). Like the best of these, it supplies an API that, in our case, makes the *five essential*

proxemic dimensions easily accessible to developers (Chapter 3). Like the more advanced tools, it also provide additional development tools, such as a monitoring tool for visualizing proxemic relationships, a record/playback tool to simplify testing; templates, documentation, and examples.

5.2 Derived Challenges for Developers

Building proxemic-aware systems such as the ones described earlier in Chapter 3 and 4 is difficult and tedious. This is mostly due to the serious technical challenges that developers face when integrating proxemic information into their application designs. Several challenges are listed below.

1. *Exploring and observing proxemic measures between entities in the ecology*. Developers need to do this to decide which measures are important in their scenario.

2. *Accessing proxemic measurements* from within software that is developed to control the ubicomp system. Developers currently do this through very low-level programming against a particular tracking technology, requiring complex 3D transformations and calculations, and often resulting in brittleness.

3. *Support for proxemic concepts* is created from scratch by developers, e.g., when considering distance of spatial zones or the properties of fixed and semi-fixed features (e.g., the spatial arrangement) in applications.

4. *Debugging and testing* of such systems is difficult due to a lack of sensing and/or matching monitoring tools.

5.3 Design of the Proximity Toolkit

The Proximity Toolkit directly addresses these challenges. It facilitates programmers' access to proxemic information between people, objects and devices in a small ubicomp environment, such as the room shown in Figure 5.4 and visualized in Figure 5.3. It contains four main components.

a) **Proximity Toolkit server** is the central component in the distributed client-server architecture, allowing multiple client devices to access the captured proxemic information.

b) **Tracking plug-in modules** connect different tracking / sensing systems with the toolkit and stream raw input data of tracked entities to the server.

c) **Visual monitoring tool** visualizes tracked entities and their proxemic relationships (Figure 5.3).

d) **Application programming interface (API)** is an event-driven programming library used to easily access all the available proxemic information from within developed ubicomp applications.

We explain each of these components in more detail below, including how each lowers the threshold for rapidly prototyping proxemic-aware systems.

However, we first introduce a scenario of a developer creating a proxemic interaction system. Through this scenario, we will illustrate how the Proximity Toolkit is used in a real programming task to create a prototype of a proxemic-aware ubicomp application. The example is deliberately trivial, as we see it akin to a *Hello World* illustrating basic programming of proxemic interaction. Still, it shares many similarities with more comprehensive systems built for explorations in earlier research discussed in Chapter 3 and 4, e.g., (Ju et al. 2008; Vogel and Balakrishnan 2004).

Scenario. Developer Steve is prototyping an interactive announcement board for the lounge of his company. In particular, Steve envisions a system where employees passing by the display are: attracted to important announcements as large visuals from afar; see and read more content as they move closer; and post their own announcements (typed into their mobile phones) by touching the phone against the screen. To create a seamless experience for interacting with the large ambient display, Steve plans to recognize nearby people and their mobile devices. Steve builds his prototype to match the room layout shown in Figure 5.4.

Figure 5.3 Proximity Toolkit monitoring tool: (a) tracked ubicomp environment; (b-g) visual representation of tracked entities in Figure 3; (h) list of available input modules; (i,k) list of all tracked entities; and (l,m) relation visualizer.

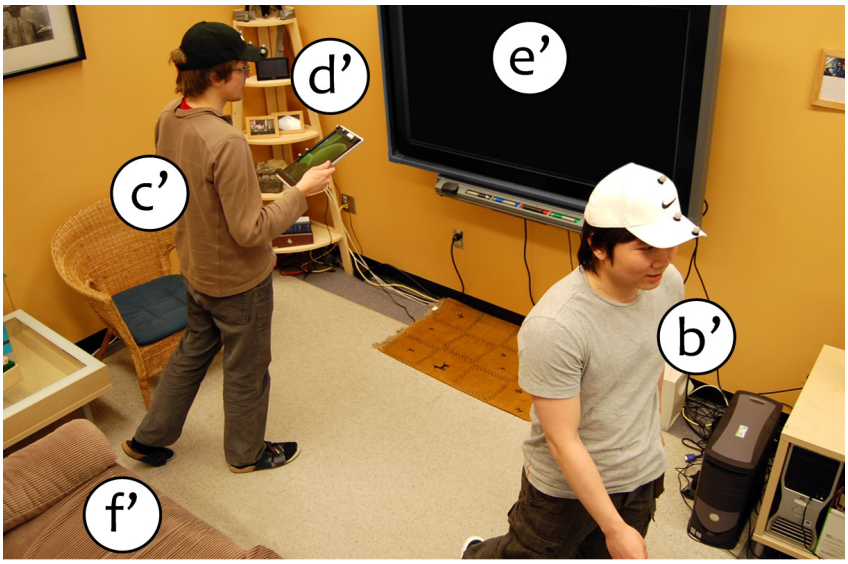


Figure 5.4 The Proximity Toolkit captures proxemic relationships between: (b' and c') people, (d' and e') devices, and fixed- and (f') semi-fixed features.

5.3.1 Proximity Toolkit Server

The Proximity Toolkit Server is the central component managing proxemic information. It maintains a hierarchical data model of all fixed features (e.g., walls, entranceways), semi-fixed features (e.g., furniture, large displays), and mobile entities (e.g., people or portable devices). This model contains basic information including identification, position in 3D coordinates, and orientation. The server and toolkit API then perform all necessary 3D calculations on this data required for modeling information about higher-level proxemic relationships between entities.

The server is designed to obtain raw data from various attached tracking systems (Figure 5.5 top). For flexibility, each of the tracking systems is connected through a separate plugin module loaded during the server's start-up. These plugins access the captured raw input data and transfer it to the server's data model. The current version of the toolkit contains three plugins: the marker-based Vicon (Vicon Motion Systems 2013) and OptiTrack (NaturalPoint 2013) motion capturing systems, which both allow for sub-millimeter tracking accuracy, and the Kinect sensor, which allows tracking of skeletal bodies (Microsoft 2013). In Chapter 6 we discuss the implementation, integration, and combination of these tracking technologies, and how to set up the server to match the

environment. Importantly, the server's unified data model is the basis for a *distributed Model-View-Controller (dMVC)* architecture (Boyle and Greenberg 2005). Using dMVC, the server stores the data as a model, which other clients – even those residing on different computers - can then access and update as needed. The dMVC architecture is accessed by the toolkit client API, the monitoring tool, and is used to calculate proxemic relationships between entities (Figure 5.5 bottom).

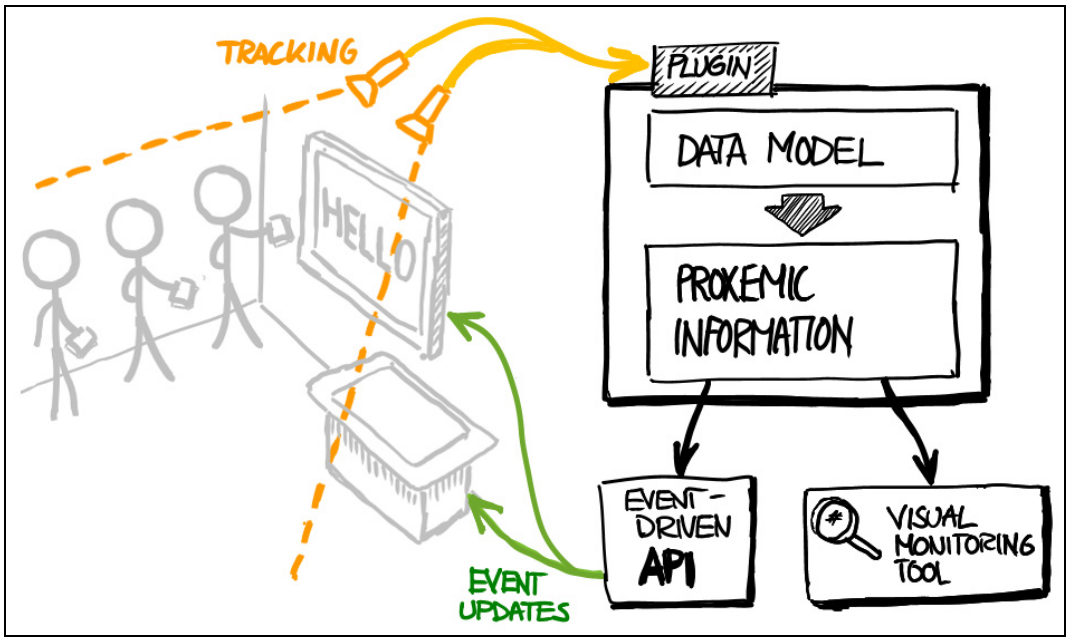


Figure 5.5 Architecture overview of the Proximity Toolkit.

Scenario. Developer Steve begins by starting the server. The server automatically loads all present tracking plugins. Based on the information gathered from these plugins, it populates and updates the unified data model in real-time. By default, the toolkit already includes a large pre-configured set of tracked entities with attached markers (such as hats to identify a person, gloves to identify the position of the person's hands, and portable devices such as tablet computers) and definitions of fixed and semi-fixed features (such as a large interactive surface and surrounding furniture in Figure 5.3). To add a new tracked object, Steve attaches markers to it and registers the marker configuration as a new tracked entity. This process takes minutes.

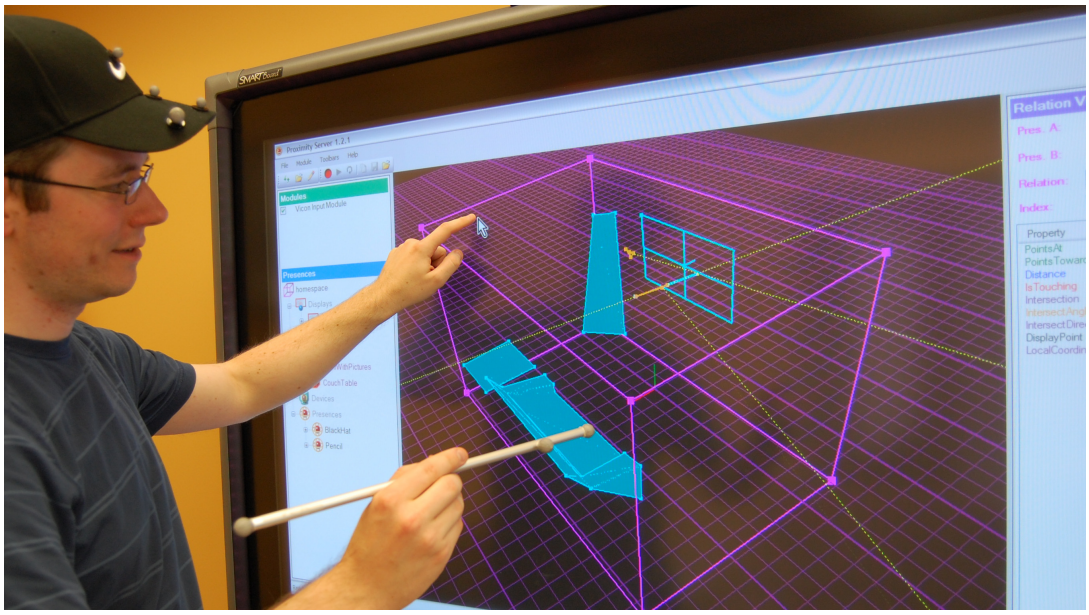


Figure 5.6 Developer using the visual monitoring tool of the Proximity Toolkit: viewing the real time 3D visualization of the environment and all tracked entities.

5.3.2 Visual Monitoring Tool: Tracked Entities

The visual monitoring tool helps developers see and understand in real time what entities are being tracked and how the data model represents their individual properties (Figure 5.6). Figure 5.3 is a screenshot of this tool: the visualized entities in (b-f) correspond to real-world entities captured in Figure 5.4 (b'-f').

Specifically, the visual monitoring tool connects to the server (through TCP) and presents a 3D visualization of the data model (Figure 5.3 centre). This view is updated in real-time and always shows:

- the approximate volume of the tracked space as a rectangular outline (Figure 5.3a)
- position and orientation of people (Figure 5.3b,c)
- portable digital devices, such as a tablet pc (Figure 5.3d)
- digital surfaces, such as the large wall display (Figure 5.3e)
- fixed and semi-fixed features, such as a table, couch (Figure 5.3f), and entranceway (Figure 5.3g).

					Dimensions				
		Property name	**Description**	**Data type**	Distance	Orientatio	Movemen	Identity	Location
A Individual entity	**I1**	`Name`	Identifier of the tracked entity	string				■	
	I2	`IsVisible`	True if entity is visible to the tracking system	bool				■	
	I3	`Location`	Position in world coordinates	Point3D					■
	I4	`Velocity`	Current velocity of the entity's movement	double			■		
	I5	`Acceleration`	Acceleration	double			■		
	I6	`RotationAngle`	Orientation in the horizontal plane of the space	double		■			
	I7	`[Roll/Azimuth/Incline] Angle`	The orientation angles (roll, azimuth, incline)	double		■			
	I8	`Pointers`	Access to all pointing rays (e.g., forward, backward)	Array [ ]		■			
	I9	`Markers/Joints`	Access individual tracked markers or joints	Array [ ]					■
B Relationships between two entities A and B	**R1**	`Distance`	Distance between A and B	double	■				
	R2	`ATowardsB, BTowardsA`	Whether A is facing B, or B is facing A	true		■			
	R3	`Angle`	Angle between front normal vectors	double		■			
	R4	`HorizontalAngle`	Angle between A and B in the horizontal plane	double		■			
	R5	`Parallel, ATangentalToB, ...`	Geometric relationships between A and B	bool		■			
	R6	`[Incline/Azimuth/Roll] Difference`	Difference in incline, azimuth, or roll of A and B	double		■			
	R7	`VelocityDifference`	Difference of A's and B's velocity	double			■		
	R8	`AccelerationDifference`	Difference of A's and B's acceleration	double			■		
	R9	`[X/Y/Z] VelocityAgrees`	True if X/Y/Z velocity is similar between A and B	bool			■		
	R10	`[X/Y/Z] AccelerationAgrees`	True if X/Y/Z acceleration is similar	bool			■		
	R11	`Collides`	True if the two volumes collide	bool	■				■
	R12	`Contains`	True if volume A contains volume of B	bool	■				■
	R13	`Nearest`	The nearest point of A's volume relative to B	Point3D	■				■
C Pointing Relationships between A and B	**P1**	`PointsAt`	Pointing ray of A intersects with volume of B	bool		■			
	P2	`PointsToward`	A points into direction of B (w/ or w/o intersection)	bool		■			
	P3	`IntersectionDegree`	Angle between ray and front facing surface of B	double		■			
	P4	`DisplayPoint`	Intersection point in screen/pixel coordinates	Point2D	■	■			
	P5	`Intersection`	Intersection point in world coordinates	Point3D	■				■
	P6	`Distance`	Length of the pointing ray	double	■				
	P7	`IsTouching`	A is touching B (pointing ray length = 0)	bool	■				

Table 5.1 Accessible proxemic information in the Proximity Toolkit: individual entities, relationships between two entities, and pointing relationships. This information is accessible through the toolkit API and the toolkit monitor visualization.

The left side of the monitoring window shows a list of the activated input tracking plugins (Figure 5.3h) and another list with an overview of all currently tracked entities (Figure 5.3i). Clicking on any of the items in this list opens a hierarchical list of properties showing the item's current status (e.g., its location, or orientation). When Steve selects any of these properties, the monitoring window shows the corresponding value (e.g., the current position as a 3D Vector, or the velocity; Figure 5.3k). Part A of Table 5.1 shows an overview of the most important available properties.

Scenario. Before Steve starts to program, he explores all available proxemic information through the visual monitoring tool. He inspects the currently tracked entities where he moves some of them to see how they are represented in the scene (Figure 5.3 left, also displayed in the center), as well as which entity properties are available for him to use. Steve finds this visual overview particularly important to his initial design, as he is still investigating the possible mappings of proxemic relationship to system behaviour. In later stages, he will also use this monitoring tool to test and debug his program.

5.3.3 Visual Monitoring Tool: Relationships

Another major feature of the visual monitoring tool is to let people set and observe particular proxemic relationships *between* entities, where developers will use these relationships to define particular proxemic interaction behaviours. Specifically, the *Relation Visualizer* panel (Figure 5.3, l-m, and Figure 5.7, b-d) allows a developer to select a type of relationship between entities, and then to observe the values of all related properties. The complete list of proxemic relationships that are available to observe are summarized in part B/C of Table 5.1.

Scenario. Steve wants to observe a relationship between *Person1* (representing the first person entering the space) and the *Smartboard* display. Steve drags the two entries from the list of tracked entities (Figure 5.7a) to the top of the *Relation Visualizer* panel (Figure 5.7b). Next, Steve selects one of the following relationship categories from a drop down menu (Figure 5.7c).

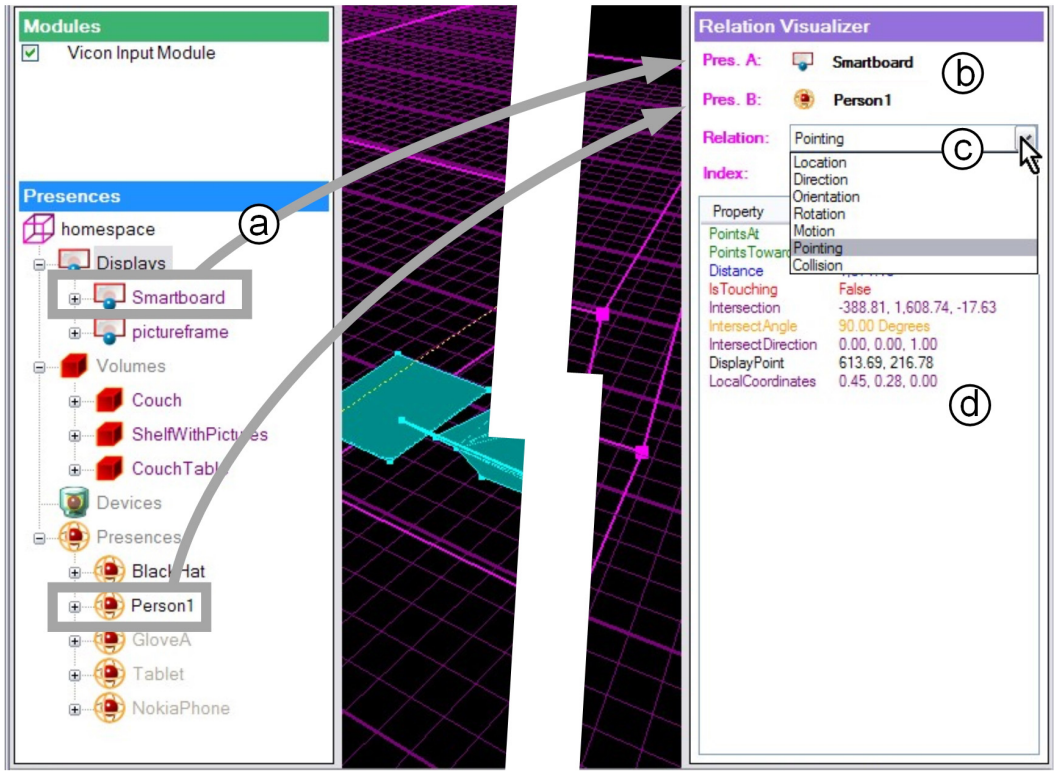


Figure 5.7 Using the relationship visualizer: (a) dragging two entities from the 'Presences' list to the (b) visualizer panel, (c) selecting one of the relationships from the dropdown list, and (d) viewing the measured relationships in the panel.

- **Location** (e.g., changes in distance between the person and the smartboard, Figure 5.8a)
- **Direction** (e.g., if the front of the person's body faces towards the screen, Figure 5.8b)
- **Motion** (e.g., acceleration or velocity, Figure 5.8c)
- **Collision** (e.g., if the volumes of two tracked entities are so close that they collide, Figure 5.8d)
- **Orientation** (e.g., angles between entities, Figure 5.8e)
- **Pointing** (e.g., the display intersection point of the right arm pointer of the person, Figure 5.8)

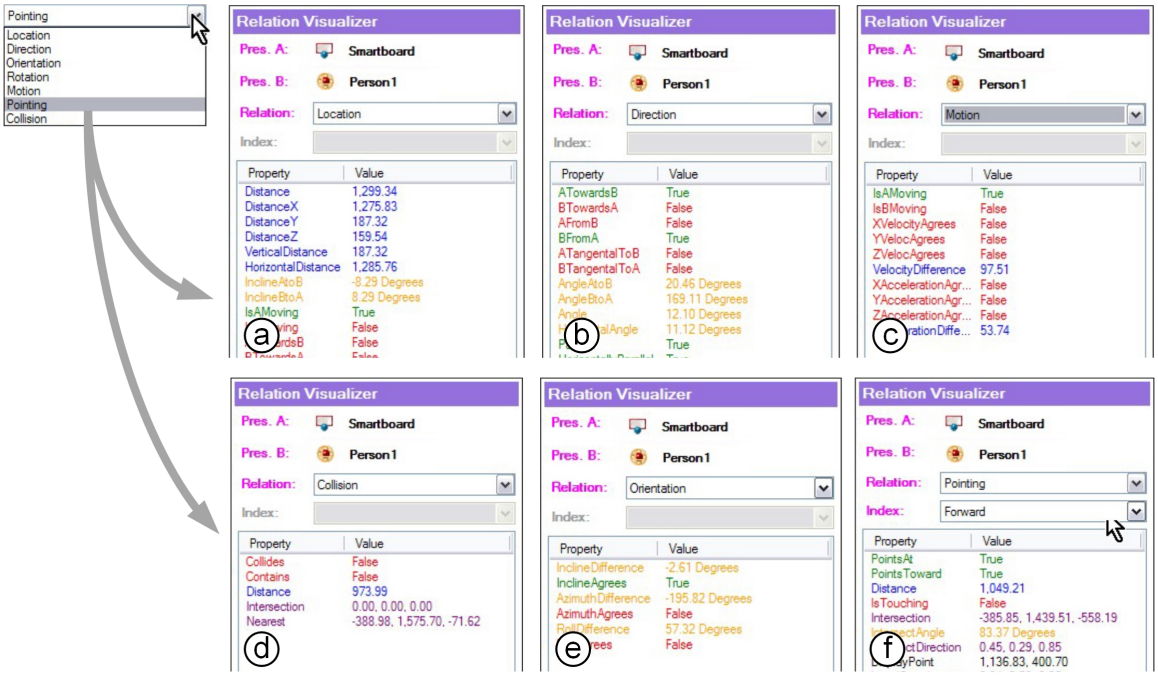


Figure 5.8 Exploring different categories of proxemic information: (a) location, (b) direction, (c) motion, (d) collision, (e) orientation, and (f) pointing.

Steve can now observe how those entities relate to each other. The panels shown in Figure 5.8 displays the numeric values of any properties belonging to this category. The categories plus the properties within them operationalize the five essential elements of proximity mentioned previously.

With his public announcement application in mind, Steve is interested in knowing when a person is in close distance to the display. He selects the *Location* category, and looks at the values of the *Distance* property, which – in this case – measures the distance of the person's body to the board (Figure 5.8a). Next, he wants to know when the person is facing towards the screen. He selects the *Direction* category from the menu, and immediately sees the related proxemic properties with their current values and their graphical appearance in the visualization (Figure 5.8b). He is particularly interested in the *ATowardsB* property, which is *true* if the person [A] is facing towards the smartboard [B]. He decides to use the information about direction and distance to adapt the content shown on the announcement board.

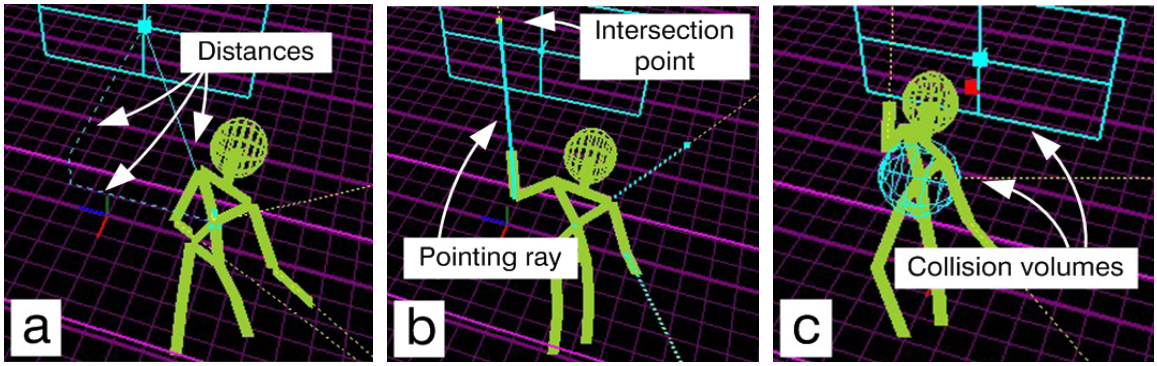


Figure 5.9 Visualizing proxemic relationships: (a) distance, (b) pointing and (c) collision (also demonstrated in the video figure).

Steve continues exploring other proxemic relationship categories and makes note of the types of relationships that he will integrate into his application. As he selects these other categories, the 3D visual representation changes accordingly. Figure 5.9 illustrates three other visualizations of proxemic relationships that Steve explored: the distance between the person and the display (Figure 5.9a), the forward pointer of the left arm and its intersection point with the smartboard (Figure 5.9b), and the collision volumes (Figure 5.9c).

5.4 Simplified API Access to Proxemic Information

We now take a closer look at the development API, offered via an *object-oriented C# .NET development library*, which in turn is incorporated into the Microsoft Visual Studio development environment. We designed it to be fairly easy to learn and use (1) by taking care of and hiding low-level infrastructure details, (2) by using a conventional object-oriented and event-driven programming pattern, and (3) by including templates within Microsoft Visual Studio that embodies most of the house-keeping chores required to set up and use the API. Essentially, the API lets a developer programmatically access the proxemic data previously observed in the monitoring tool. We explain how this works by continuing the scenario.

Scenario. Steve adds the Proximity Toolkit API library to his own PC-based software project. The only criterion is that his PC needs network access to the proximity server.

Steve begins by initializing his software. To set up his software to use the server, he adds three lines of code (lines 1-3, Figure 5.10). First, he creates a new client connection object, then starts the connection to the server (at the given IP address and port), and finally creates a `ProximitySpace` object, which provides a high-level framework for monitoring the interaction of tracked presences, such as people and objects. The `ProximitySpace` object maintains a list of all available tracked entities, and is used to create instances of entities or for initializing event handlers to monitor relationships. The provided development templates already include the code for this setup procedure, and can be used as a starting point by developers. The toolkit offers a collection of diverse development templates, each including example source code illustrating a particular aspect of the toolkit's functionality (e.g., one template demonstrates how to use distance information, another template how to use motion, etc.).

```
01  ProximityClientConnection client = new ProximityClientConnection();                    Setup
02  client.Start("192.168.0.11", 888);
03  ProximitySpace space    = client.GetSpace();
04  PresenceBase person     = space.GetPresence("Person1");
05  PresenceBase smartboard = space.GetDisplay("SmartBoard");
06  PresenceBase tablet     = space.GetDisplay("Tablet");
07  RelationPair relation   = space.GetRelationPair(person, smartboard);                   Events
08  relation.OnDirectionUpdated += new DirectionRelationHandler(OnDirectionUpdated);
09  relation.OnLocationUpdated  += new LocationRelationHandler(OnLocationUpdated);
10  void OnDirectionUpdated(ProximitySpace space, DirectionEventArgs args) {              Callbacks
11      if (args.ATowardsB) { [... person is facing the display, show content ...] } else { [...hide...] }
12  }
13  void OnLocationUpdated(ProximitySpace space, LocationEventArgs args) {
14      double distance = args.Distance; [... change visual content as a function of distance ...]
15  }
16  RelationPair relationTablet = space.GetRelationPair(tablet, smartboard);               Event
17  relationTablet.OnPointingUpdated += new PointingRelationHandler(OnPointingUpdated);
18  void OnPointingUpdated(ProximitySpace space, PointingEventArgs args) {                 Callback
19      if (args["forward"].PointsToward && (args["forward"].Distance < 500.0)) {
20          Point intersection = args["forward"].DisplayPoint;
            [... show awareness icon on smartboard display ...]
21          if (args["forward"].IsTouching) {
                [... transfer content from the tablet to the large display ...]
22  }}}
```

Figure 5.10 Partial source code for the proxemic-aware announcement board application.

Next, Steve initializes three of the entities he is interested in (lines 4-6, Figure 5.10): the person representing the first person entering the space, the smartboard, and a tablet, as

shown in Figure 5.11 (`PresenceBase` is a special object that represents individual tracked or static objects).

The following describes how Steve then monitors the relationships *between* these entities. We go through each of the five proxemic dimensions introduced earlier in Chapter 3 (albeit in a slightly different order), explaining how Steve writes his application to monitor changes in each of these dimensions, and how he uses that information to mediate interaction with his interactive announcement board.

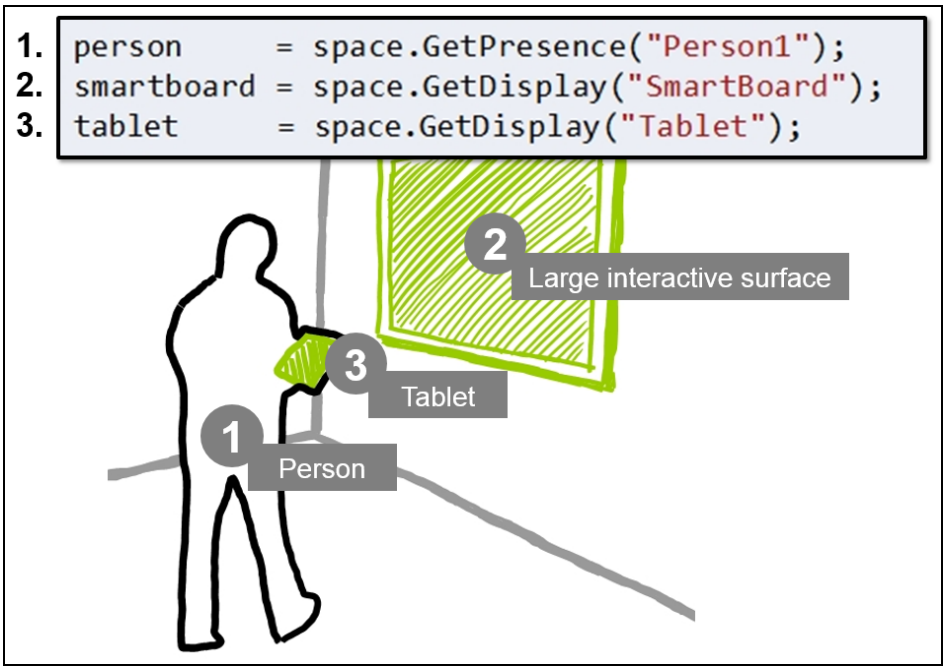
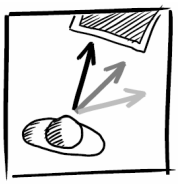


Figure 5.11 Initializing three entities with the Proximity Toolkit to monitor relationships between them.

5.4.1 Orientation



Monitoring orientation changes allows the programmer to (1) access the exact angle of orientation between two entities and/or (2) determine whether two entities are facing each other. Steve is mostly interested in the relationship between a person and the smartboard display. He adds line 7 (Figure 5.10), which creates a relationship between these two as indicated by their parameters. The system is now tracking both entities relative to each other. Steve is also interested in knowing when the orientation and location between these two changes.

For orientation, he initializes an event handler to receive updates of the *Direction* relationship between the person and the smartboard (line 8, Figure 5.10). The `OnDirectionUpdated` method is invoked when the system recognizes any changes in orientation between the person and the smartboard (line 10, Figure 5.10; and Figure 5.12). While Steve could access each entity's precise orientation values (e.g., angles of orientation, entries I6 and I7 in Table 5.1), he is only really interested in knowing whether a person is facing towards the smartboard. Consequently, he writes the event handler callback method (lines 10-12, Figure 5.10) to access the `ATowardsB` property (entry R2 in Table 5.1) in the event arguments: it is true only when the person is facing the smartboard (line 11, Figure 5.10).

Entries R2-R6 and P1-P3 in Table 5.1 give an overview of further orientation relationships that can be monitored (e.g., angle, geometric relationships, azimuth). As well, the programmer can access the absolute orientation of an individual entity at any time, such as rotation angle roll, azimuth, and incline relative to a plane (see entries I6 – I7 in Table 5.1). For example, the following property returns the current roll angle of the tablet: `tablet.Orientation.Roll;`

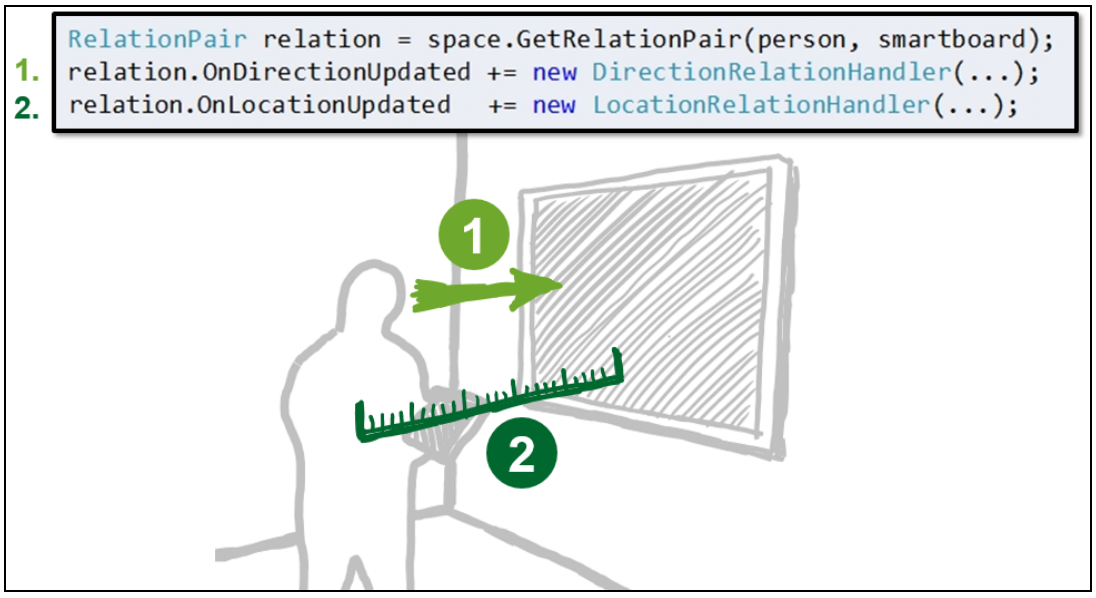
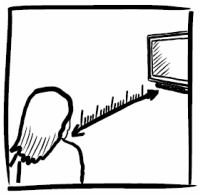


Figure 5.12 Subscribing for events of direction and location/distance changes between a person and a large display.

5.4.2 Distance, including Location, Pointing and Touching

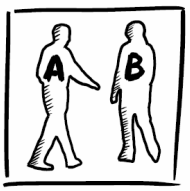


Similarly, Steve can monitor changes of distance between entities. We illustrate how Steve can receive updates about distance changes by adding another event callback for `OnLocationUpdated` events (line 9, Figure 5.10; and Figure 5.12). This callback method (line 13-15, Figure 5.10) is invoked whenever the location of at least one of the two entities changes. In line 14 (Figure 5.10) Steve accesses the current distance between the person and the smartboard, and uses this distance value to make the visual content on the announcement board vary as a function of the distance between the person and the display. The closer the person, the more content is revealed.

Other available properties relate to distance (Table 5.1). First, the actual location property of each entity, i.e, its position within the space, is accessible at any time. For example Steve can access the current coordinates of the person by accessing `this.person.Location` (Table 5.1, I3). Second, *pointing relationships* monitor orientation *and* distance simultaneously (Table 5.1, Part C). Pointing is similar to ray-casting. Each entity can have one or multiple pointers. Each pointer has a pointing direction, and the callback returns the intersection of that direction with the other entity. It also returns the length of the pointing ray between entities, which may not be exactly the same as distance. To illustrate, Steve tracks not only the close distance of a tablet computer to the smartboard, but where that tablet raycasts onto the smartboard. He initializes a second `RelationPair` between the tablet and the smartboard (line 16, Figure 5.10). He subscribes for `OnPointingUpdated` events that are triggered whenever any of the pointers of the tablet changes relative to the board (line 17, Figure 5.10). In the event callback method (lines 18 to 22, Figure 5.10) Steve first checks if the tablet's forward pointer faces the display (`PointsTowards,` entry P2 in Table 5.1) and if the ray length between tablet and board is smaller than 50 cm (line 19, Figure 5.10). If this is the case, he shows an icon on the ray's intersection point (line 20, Figure 5.10) on the smartboard to let the person know they can touch the surface to initiate a transfer.

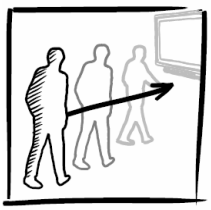
Third, Steve checks if the tablet is touching the surface - (`IsTouching`, line 21, Figure 5.10 and entry P7 in Table 5.1) – a distance of ~0. If so, he initiates transfer of the content on the tablet to the large display. By using the intersection point of the tablet with the screen Steve can show the transferred content at the exact position where the tablet touches the board.

5.4.3 Identity



The toolkit allows access to the identity information of all tracked entities. The Name property provides the identifier string of each entity, and `IsVisible` is true if the entity is currently tracked by the system (Table 5.1, I1 and I2). A developer can subscribe to events that raise notifications about any new tracked entities that enter the ubicomp space via the `space.OnPresenceFound` event. In the associated event callback method, the event arguments give information about the type and name of the detected entity. For example, Steve could have his system track and greet a previously unseen person with a splash screen on first appearance, and dynamically initialize any necessary event callbacks relating that person to other entities in a scene.

5.4.4 Motion



Motion events describe the changes of distance and orientation over time, e.g., to receive updates of changes in acceleration and velocity of any entity. For example, Steve can have his application ignore people moving quickly by the display, as he thinks they may be annoyed by any attempts to attract their attention. To receive such velocity updates, Steve would add an event handler (similar to lines 8 and 9, Figure 5.10) through `OnMotionUpdated` and then simply access the value of the `args.Velocity` property (Table 5.1, I4, R7, and R9). Based on that value, he would activate the display only if the velocity was less than a certain threshold. Of course, Steve could have determined a reasonable threshold value by observing the velocity value of a person rushing by the display in the visual monitoring tool.

5.4.5 Location: Setup of Environment

Using location, the toolkit lets one track the relationships of people and devices to the semi-fixed and fixed features in the physical environment. For example, the model may contain the fixed-feature position of the entranceway to a room, allowing one to know if someone has crossed that threshold and entered the room. It may also contain the location of semi-fixed features, such as the chairs and table seen in Figure 5.4. Monitoring event handlers for fixed and semi-fixed features can be initialized similarly to the ones we defined earlier.

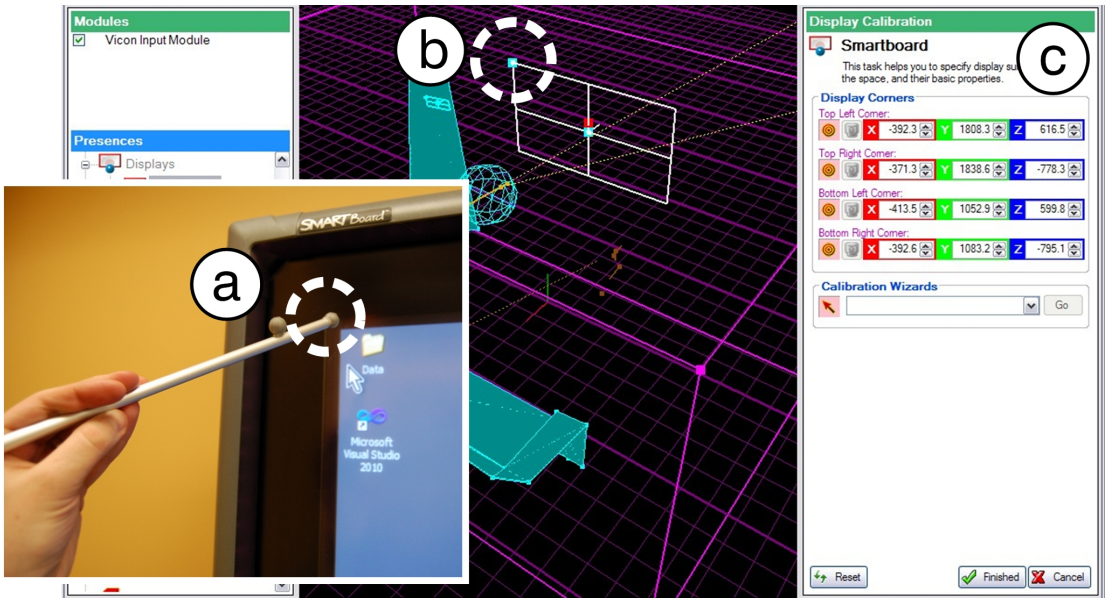


Figure 5.13 Defining new fixed and semi-fixed features (e.g., display) using (a) a tracked physical pointer, (b) visual feedback in the 3D view, and (c) a dialog for adjusting precise numeric measurements.

Steve had set up several fixed feature entities – the smartboard and the entrance-way – through several initial configuration steps. This only has to be done once. Using a physical pointer (the stick in Figure 5.13a), he defines each entity's volume by physically outlining them in space. Under the covers, the toolkit tracks the 3D tip location of this stick and builds a 3D model of that entity. Each location point of the model is confirmed by pressing a button (e.g., of a wirelessly connected mouse). Figure 5.13 illustrates how Steve defines the smartboard. After placing the pointer in the four corners of the display

plane (Figure 5.13a), the coordinates appear in the visualization (Figure 5.13b), and a control panel allows fine adjustments (Figure 5.13c). He saves this to the Proximity Toolkit server as a model. Similarly, Steve defines the entrance-way by outlining the door (Figure 5.3g), and the couch by outlining its shape (Figure 5.3f). The editor allows precise adjustments of all definitions of semi-fixed features added to the toolkit's space model (Figure 5.14). For example, for the selected model of the couch (Figure 5.14a), Steve can refine the created volume in the volume editor window (Figure 5.14).

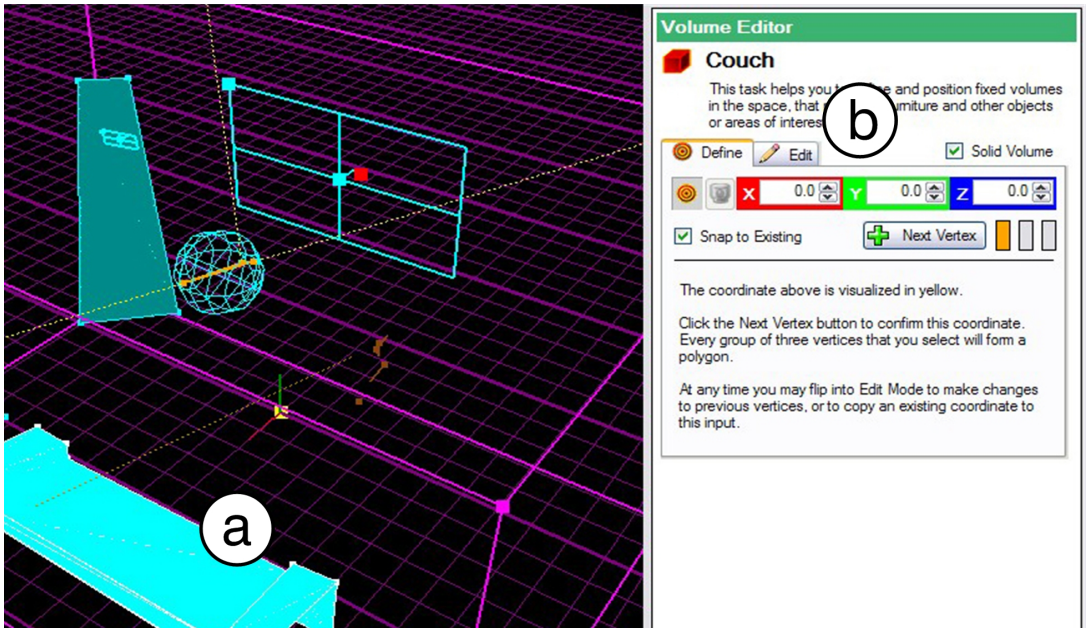


Figure 5.14 Change shape and position of (a) semi-fixed features using the (b) volume editor dialog.

Steve can now monitor proxemic relationships between all moving entities and these new defined features. For example, he can create an event handler to receive notifications when a person passes through the entrance-way (by using the `OnCollisionUpdated` event) and when a person sits on the couch (using the Distance property of the `OnLocationUpdated`).

Semi-fixed features differ. While they are part of the environment, they are also movable. As with fixed features, a developer would model a shape by outlining it with the stick. Unlike fixed features, he would also add markers to that entity. The toolkit tracks those markers, and repositions the entity accordingly. For example, Steve could have modeled

a chair, tracked where it is in the room, and adjusted the presentation if a person was sitting on it.

We believe location should also include further contextual information about this particular environment, e.g., the meaning of that place. Such contextual information is not yet included in the toolkit, but could be easily added as metadata.

Scenario – next steps. The walkthrough example illustrated the easy-to-use mechanisms of integrating proxemic measurements into a ubicomp system. While simple, this starting point allows Steve to further extend the system functionality exploring proxemic interactions. Examples include: (1) subscribing to events of a second person to let the system react to both persons' movement to the display; and (2) monitoring additional tablet computers, and enabling content-sharing between them as a function of the device's distance. Overall, the toolkit minimizes the effort necessary for such extensions, and allows rapid exploration and alteration of interaction techniques.

5.5 Additional Tools Facilitating the Prototyping Process

The toolkit is more than an API, as it offers additional tools to lower the threshold for developing proxemic-aware systems. The already-discussed visual monitoring tool is one of these. Several others are described below.

5.5.1 Recording and Playback of Proxemic Sequences

To test applications, developers would need actors to perform the proxemic movements between entities every time. This is problematic for many reasons: it is tedious; the sensing equipment may not be calibrated or available (e.g., because others are using it); and it is difficult to repeat particular test sequences (e.g., such as those that test edge conditions). To alleviate this, the toolkit provides a record/playback tool within the visual monitoring tool. With the click of a button (Figure 5.15a), developers can record events generated by entities moving in the environment (Figure 5.15b,c). They can later play back these sequences for testing (Figure 5.15d), where the toolkit software

generates all events as if they were happening in real time and a person would be still tracked (Figure 5.15e). Under the covers, each individual sequence is recorded as an XML file, where the toolkit uses that record to recreate all events. Because the tracking hardware is not needed during playback, testing can be done anywhere, e.g., a standard desktop workstation located elsewhere. For example, Steve could have recorded test sequences such as: a person passing by the screen, a person approaching the display, or a device pointing towards the display. He would then replay these sequences while developing and testing his software at his desk.

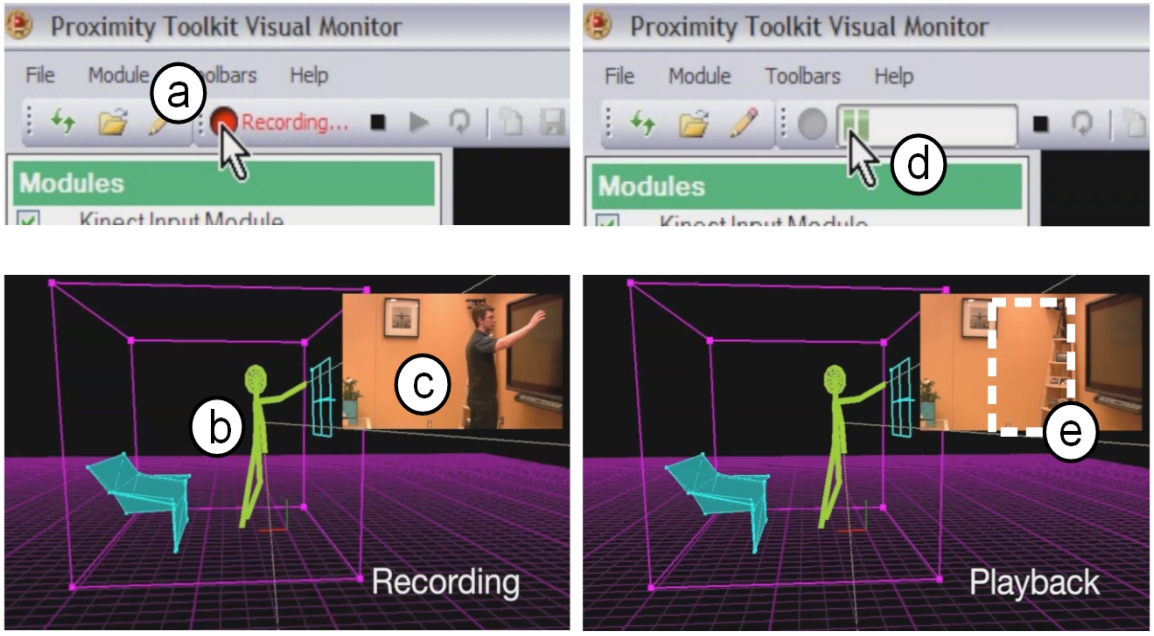


Figure 5.15 Recording and playback of proxemic sequences: (a) activating recording function to (b) record proxemic information of a (c) person interacting in space. (d) Playback later generates the same events as before, without the person being tracked anymore (e).

5.5.2 Component Library, Templates, and Examples

The toolkit also leverages developers' existing practices by seamlessly integrating the toolkit into the familiar capabilities of a popular IDE, Microsoft *Visual Studio* (but the ideas are generalizable to other IDEs).

First, the toolkit includes a library of *drag-and-drop* components (compatible with both *Windows Presentation Framework* and *Windows Forms*), where the programmer can view and set all their properties and generate event handlers for all available events via direct manipulation rather than coding. Included are matching components for the `PresenceBase` (i.e., all tracked entities), and `RelationPairs` – called `PresenceControl` and `RelationControl` respectively. Using these visual components as a starting point not only reduces tedium and coding errors, but also reduces the threshold for inexperienced developers (such as students) as all properties and events are seen. For example, Figure 5.16 demonstrates the four drag-and-drop steps that Steve has to perform when using the Windows Forms components to monitor relationships between entities: First, he drags the `PresenceManager` control (that functions as a managing container for all other controls) onto the application window (Figure 5.16a). Next, he drags two `PresenceControls` into the `PresenceManager`, and gives them the names "*Person1*" and "*Smartboard*" to link them to these two entities tracked in the toolkit (Figure 5.16b). Then Steve drags a `RelationControl` into the `PresenceManager` (Figure 5.16c). Last, he selects the events he is interested in (Figure 5.16d), and this automatically generates the corresponding event callback methods stubs. This is all done without requiring Steve to program a single line of code. This process can be repeated to monitor further entities or different relationships.

Second, we reduce start-up effort by including a set of templates containing the minimum required code. The default template includes the reference to the Proximity Toolkit library and the source code initializing the connection to the server, creating the `ProximitySpace` object (lines 1-3 in Figure 5.10), and subscribing for two example `PresenceBase` objects (similar to lines 4-15 in Figure 5.10). Using this template lets programmers begin coding with minimal effort.

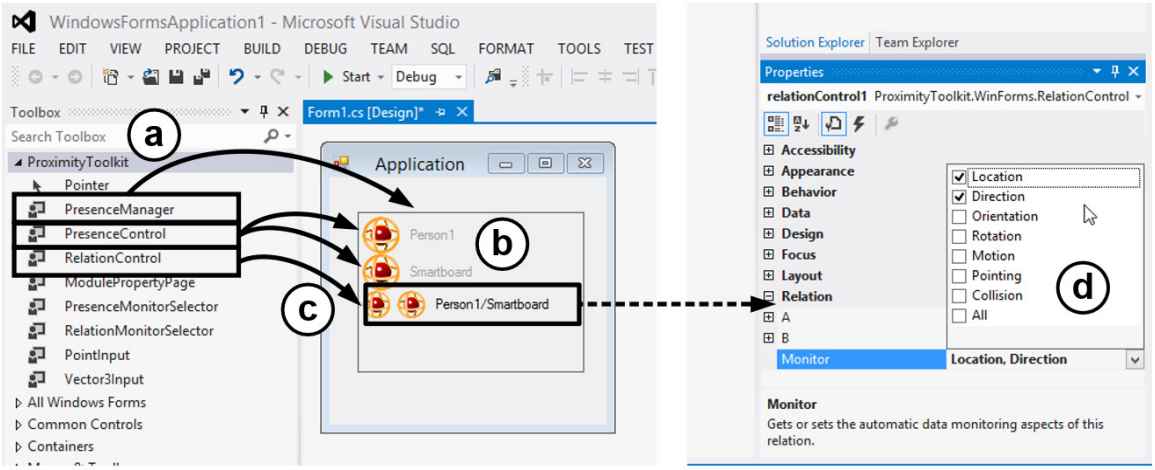


Figure 5.16 Using the drag-and-drop components in Visual Studio to monitor entities tracked by the Proximity Toolkit.

Third, to ease learning, we provide a large set of teaching applications illustrating standard programming patterns. The current set of 14 example applications illustrates the required programming commands to access a particular proxemic relationship (e.g., *how to use pointers*, *how to use motion events for recognizing gestures*, or *how to react to orientation changes*). An extensive API documentation and tutorial videos complete this starter package for developers. The Proximity Toolkit website (GroupLab 2013) includes the installer packages and example source code as downloads, detailed documentation, tutorials, and further links (Figure 5.17abcd). This website became an important central repository for information used by all developers and students both from within our lab but also from other universities using the toolkit.

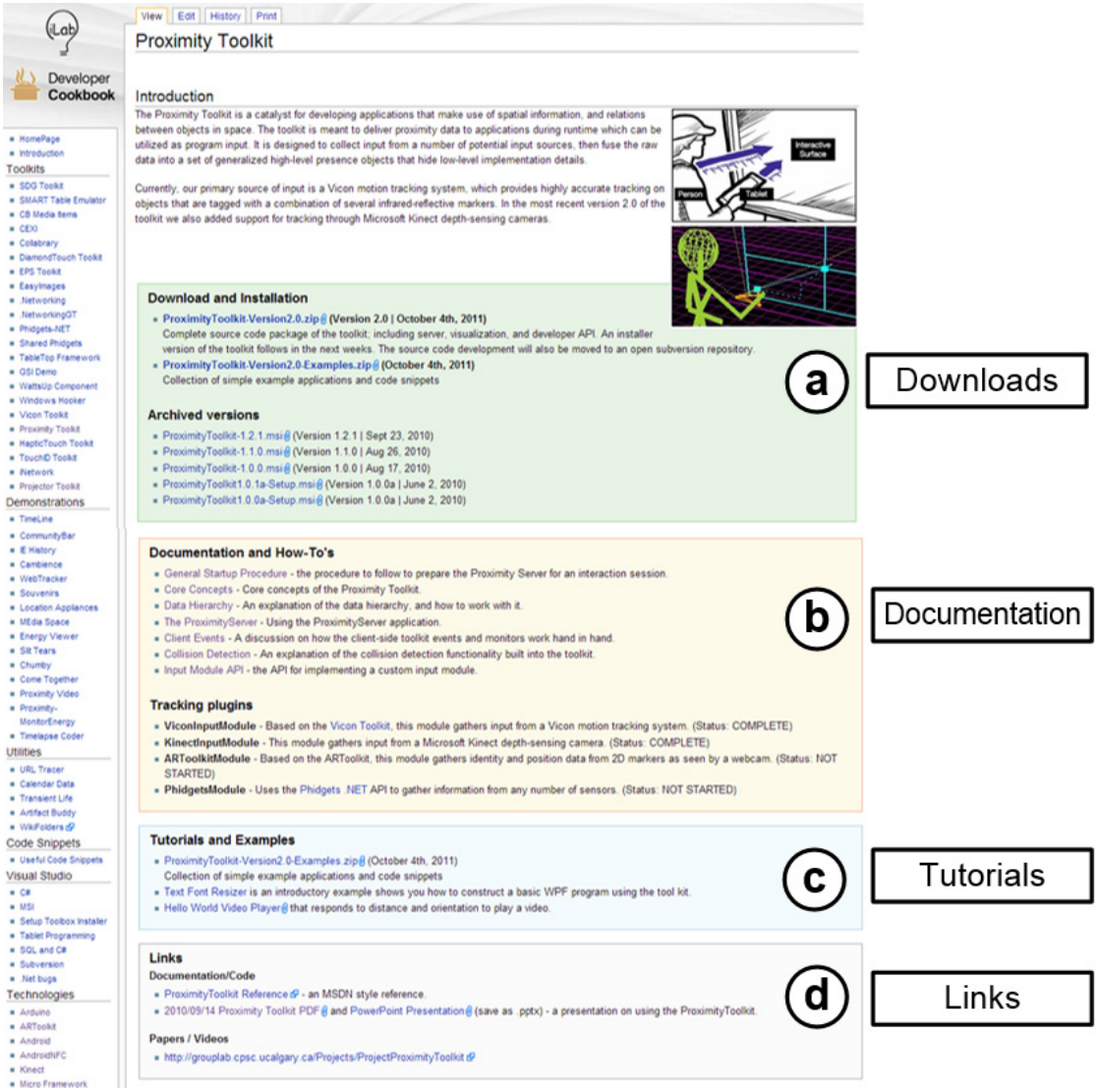


Figure 5.17 The Proximity Toolkit website, as part of the Interactions Lab Developer Cookbook: (a) downloads, (b) documentation, (c) tutorials, and (d) further links.

5.6 Conclusion

In this chapter we introduced the Proximity Toolkit that enables (and satisfies my thesis goal of) rapid prototyping and exploration of novel interfaces that incorporate the notion of proxemic relationships. Through hiding most of the underlying access to tracking hardware and complex 3D calculations, the toolkit lets developers concentrate on the actual design and exploration of novel proxemic interaction. The toolkit was invaluable for my own explorations of proxemic-aware systems (discussed shortly in Chapter 7 and

8), but also made it possible for students in our lab and ubicomp classes to explore proxemic interactions through rapidly prototyping ideas (Chapter 10). To address an even wider audience, we decided to make the toolkit open source[25], so that laboratories at other university (and anyone else interested in this field) can download the toolkit as a starting point for their own explorations of proxemics in ubicomp. HCI labs at the University of Copenhagen (Denmark) and University of Konstanz (Germany) are two of the groups already using the toolkit for their own research explorations. They even went beyond this use where – as we explain in the next chapter – they programmed a custom plugin integrating another tracking technology.

To guarantee flexibility for different setups, environments, and requirements that programmers might have, the extensible architecture of the toolkit allows the integration of diverse tracking technologies for sensing people's and devices' spatial relationships. The details of this underlying technical architecture are usually hidden from the individual programmer, because the plugin architecture guarantees that the API stays consistent independent from the actual technology used. In the next chapter we delve into these details: how the toolkit allows the integration of diverse tracking systems, how to address the unique properties of the raw tracking data, and how to fuse the data in a unified data model.

[25] Available for download at: http://grouplab.cpsc.ucalgary.ca/cookbook/index.php/Toolkits/ProximityToolkit

Chapter 6. Toolkit Architecture

The previous chapter demonstrated the Proximity Toolkit mostly from the application developer's point of view: how its architecture makes low level proxemic information available, and how its API and accompanying tools facilitates rapid exploration of proxemic-aware ubicomp applications (Figure 6.1 bottom). The focus of this chapter[26] is delve into details from the toolkit developer's point of view, i.e., someone who wishes to understand and perhaps extend the toolkit itself, or to replicated its basic workings. In particular, this chapter sheds light on the technical implementation of the toolkit architecture, and most importantly how the toolkit supports different tracking technologies with varying capabilities for tracking people and devices in ubiquitous computing ecologies (Figure 6.1 top).

We begin by briefly introducing key technologies for tracking spatial relationships between people and devices in ubicomp spaces (§6.1). We discuss their strengths and limitations, and argue for a flexible support of such technologies in in the Proximity Toolkit. Next, we explain technical details of the toolkit's architecture (§6.2). In that section, we explain how the toolkit's flexible plugin architecture allows integration of several diverse tracking technologies along the high- and low-fidelity spectrum. We discuss the toolkit's data hierarchy and the use of a distributed model-view-controller

---

[26] Portions of this chapter are published as:

Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011) The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the ACM Symposium on User Interface Software and Technology – UIST'2011.* (Santa Barbara, CA, USA), ACM, October 16-18, pages 315-326.

architecture. We also explain methods for substitution of tracking technologies, handling uncertain input, and merging data of multiple tracking technologies with the toolkit. Finally, we discuss the limitations of the current toolkit and the diversity of available tracking fidelity, and conclude (§6.3).

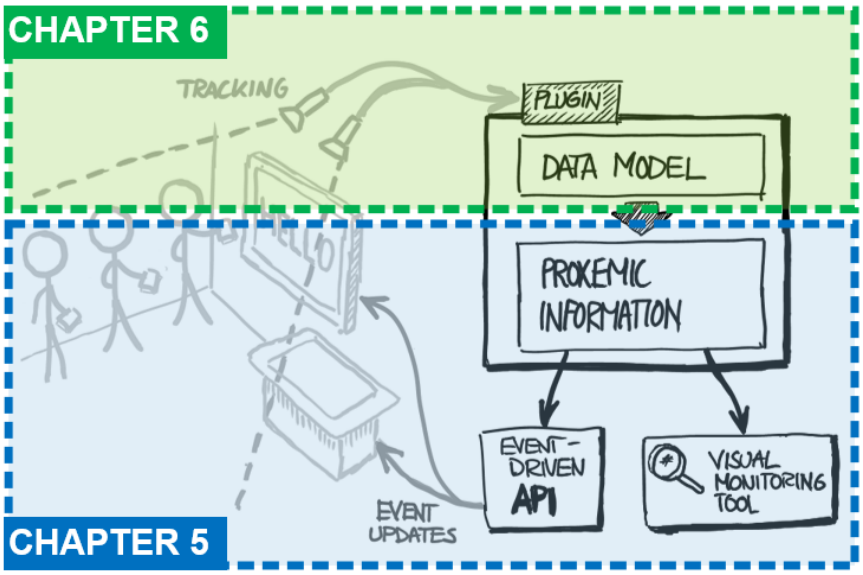


Figure 6.1 Structure of the two chapters discussing the Proximity Toolkit (the architecture shown in the background is identical to Figure 5.5).

6.1 Tracking Technologies for Ubicomp Spaces

In this section we review five different technical approaches (with some variations in each) that are commonly used for determining people's or devices' position, orientation, or identity in ubicomp systems (also see surveys of Hightower and Borriello, 2001; and more recently of Varshavsky and Patel, 2010). Most importantly, we compare these technologies by discussing how they differ in technical capabilities, limitations, and cost (see Table 6.1 for comparison). We selected these five particular techniques as they are commonly used approaches in ubicomp systems, but want to emphasize that various other techniques (e.g., acoustic sensing) are possible to track people and devices in ubicomp spaces.

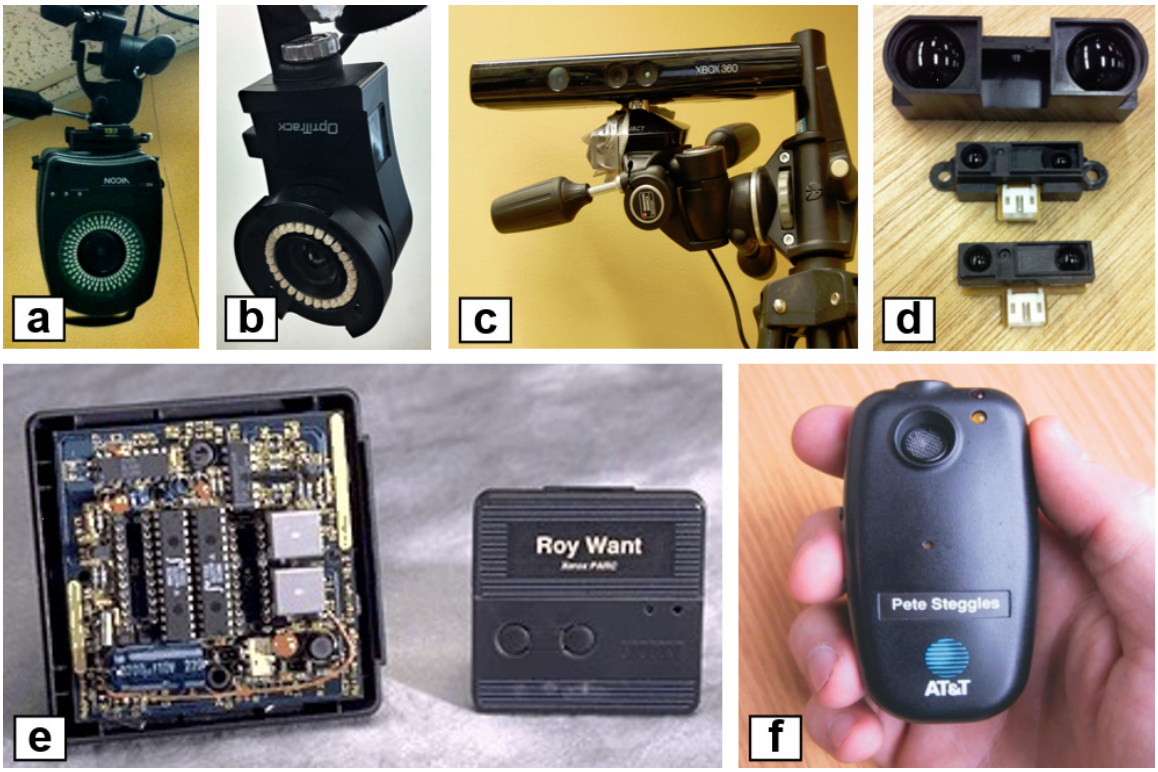


Figure 6.2 Overview of tracking technologies for ubicomp spaces: (a) Vicon and (b) OptiTrack motion capturing, (c) Kinect structured light depth camera, (d) IR range finding sensors, (e) Active Badge IR tag positioning and (f) Bat ultrasonic tag positioning[27].

6.1.1 Infrared Marker-Based Tracking

At the high-fidelity end of the spectrum, infrared marker-based tracking systems allow fine-grained tracking of objects, people, and devices by providing highly accurate information about their position and orientation. These systems use a minimum of three (with usually six to eight) IR cameras to track the position of reflective markers that are illuminated with strong IR-light emitting strobes. For example, Vicon (Vicon Motion Systems 2013) provides a commercial marker tracking system (Figure 6.2a), which is commonly used in industry for motion-capture animation, and (to a lesser extent) as a rich input device for academic prototypes (e.g., Vogel and Balakrishnan 2004; Li et al. 2009). Vicon provides very high fidelity tracking information (~1 mm accuracy), unique

[27] Source: Figure 6.2e by Want et al. (1992); Figure 6.2f by Addlesee et al. (2001).

identifiers of all tracked entities, and updates at high update rates of around 240 Hz. Depending on the number of cameras used, a Vicon setup can be configured to track markers within an indoor room. However, the usual tracking setup is expensive, with a cost of more than $100k. A similar, but more affordable, tracking system is OptiTrack (NaturalPoint 2013), with similar specifications but smaller tracking volumes and lower update rates of around 120 Hz (Figure 6.2b). A major limitation are possible occlusions, as the cameras need line of sight to the tracked infrared markers.

6.1.2 Depth Sensing via Structured Light

Depth sensing cameras based on structured light can also sense people's location (and gestures) in small spaces. In ~2008, Microsoft Kinect (Microsoft 2013) in conjunction with PrimeSense (PrimeSense 2013) made these cameras commercially available at a very affordable price (around $250), where it was targeted for gaming (see a Kinect camera in Figure 6.2c). As a consequence, the once-esoteric depth sensing camera rapidly came in widespread use, where it was also adopted as an input mechanism for many academic research projects. Both the Kinect and PrimeSense track an area of approximately 2x4 meters, where they capture raw depth images. These are then analyzed to determine the skeletal body posture and gestures of up to two people. Their APIs let programmers access the depth image directly, or via the calculated skeletal postures. They are ideal for tracking people. However, depth sensing cameras are limited in comparison to infrared marker-based systems. The spatial tracking resolution of approximately 5 cm and update rate of up to 30 Hz is relatively low. While their APIs are tuned to analyze body postures, they do not identify people. Tracking the location of devices (rather than people) is non-trivial, as the complexity of programming increases drastically when processing raw depth images (e.g., to detect device shape, location and orientation). Similarly, while more than one camera could be used (e.g., to track larger volumes and/or to triangulate objects), such programming would be very difficult for those without a very string skill set in both computer vision and 3D graphics programming. Even so, notable recent research prototypes have used one or multiple of these depth cameras to build interactive ubicomp systems, for example the LightSpace (Wilson and Benko 2010) and CodeSpace (Bragdon et al. 2011) systems we mentioned earlier.

6.1.3 Radio-based Sensing (Signal Strength Trilateration)

Radio-based signal strength sensing (e.g., with 802.11 WLAN or Bluetooth radios) is a common approach in many ubicomp projects to determine the (coarse) location of devices. The core idea is to have multiple radio stations at fixed locations in the environment and sense the relative signal strength of these stations when received by a mobile receiver (Varshavsky and Patel 2010). Based on the signal strength, a system can then either perform a trilateration calculation to determine the current position of a mobile receiver, or compare the list of signal strength values to a database that contains so called fingerprint signatures that match to specific locations in the environment. For example, the EasyLiving project used WLAN signal strength trilateration (Barry Brumitt et al. 2000), as does the NearMe ubicomp location server (Krumm and Hinckley 2004). The hardware cost of such a setup is relatively low because most digital devices have WLAN transponders already built in. The location tracking is, however, often unreliable as many factors besides the distance affect the signal strength (e.g., walls, furniture), with a spatial resolution of around 1-2 m for WLAN and around 50 cm for Bluetooth (with update rates of 1 Hz or lower). One of the major advantages of this technique is that it provides unique identifiers for all tracked devices, which is considerably more difficult to achieve with the two earlier mentioned computer-vision based tracking technologies.

6.1.4 Range Finding Sensors

Modern range finding sensors use a variety of methods to sense the approximate distance between the sensor and any object in front of it. Common examples are sensors that emit pulses of infrared or ultrasonic frequencies and observe the reflections of these signals. Infrared range finders (e.g., the three sensors in Figure 6.2d) are commonly used as switches to trigger an event (e.g., as in bathroom sinks to turn faucets on and off). However, more advanced uses in academic prototypes include Ju's interactive whiteboard (Ju et al. 2008) and Annett's et al. proxemic-aware multi-touch tabletop (Annett et al. 2011), where multiple sensors are tracked to detect coarse motion information. Their advantage is that they are highly affordable (i.e., a few dollars). Yet

the spatial resolution and the typical tracked volume is rather low compared to the other techniques (~5 cm and 30 Hz). Additionally, they merely return a distance value to an object in front of it: they do not provide any identification of objects in front of the sensor, or any information about the orientation of that object.

Type of tracking technology	Example technology	Typical resolution (spatial, temporal)	Typical scale	Cost	Identity	Orientation	Technical limitations
Infrared camera marker tracking	Vicon	~ 1 mm, ~ 240 Hz	Room	> $100,000 per setup	X	X	Requires augmentation and line of sight
	OptiTrack	~ 1 mm, ~ 120 Hz	Room	~ $15,000 per setup	X	X	Requires augmentation and line of sight
Structured light depth sensing camera	Kinect	~ 5 cm, ~ 9-30 Hz	Small area 2x3m	~ $300 per camera		X	Complex raw data processing
Radio signal strength trilateration or fingerprinting	W-LAN 802.11	~ 1-2 m, ~ 1 Hz	Building	< $100 per transmitter	X		Tracking slow compared to other techniques
	Bluetooth	~ 50 cm, ~ 1 Hz	Room	< $100 per transmitter	X		Tracking slow compared to other techniques
Range finding sensors (e.g., ultrasonic or IR)	Phidgets, Samsung	~ 5 cm, ~ 30 Hz	Small area ~2x3m	~ $30 per sensor			Only limited sensing capabilities
Infrared or ultrasonic tag positioning	Infrared, ActiveBadge	Room	Building	Research prototype	X		Possible IR interference and low spatial resolution
	Ultrasonic, Bat	~ 3 cm	Room or Building	Research prototype	X		Expensive ceiling sensor grid

Table 6.1 Comparison of commonly used spatial tracking hardware for indoor ubicomp environments.

6.1.5 Infrared and Ultrasonic Tag Positioning

Tag positioning systems typically use infrared or ultrasonic signals to locate the presence and/or position of an active transponder. Transponders are, for example, realized within name tags people wear, and thus can be used to determine the location of people in a building. For example, the ActiveBadge system (Figure 6.2e) uses infrared signals to determine the location of each tag (and thus of the person wearing it) (Want et al. 1992). The spatial resolution is low: in most cases the system can only determine if a tag is

present in a particular room or not. The sentient computing Bat (Addlesee et al. 2001) system refined this technique by deploying a larger grid of wearable (Figure 6.2f) and ceiling-mounted ultrasonic sensors that allow more accurate location sensing (approximately 9cm resolution). Both systems provide unique identification of all tags but no information about the tag's orientation. Furthermore, they are both research prototypes with significant deployment and maintenance costs.

6.1.6 Summary

Table 6.1 summarizes the technologies presented in this chapter. When using any of these tracking technologies, a ubicomp system developer has to weigh the strength and limitations in context of the envisioned ubicomp systems. As seen from the table, no technology is ideal, where there is a large trade-off between equipment and/or deployment costs vs. information fidelity. Thus some technologies are particularly suitable for building specialized high-fidelity prototype explorations (e.g., infrared marker tracking), where the high cost and site setup are acceptable. Others are ideal for real world deployments because they don't require additional markers (e.g., Kinect or range finding sensors), the cost is low, and the lower detection power and resolution is sufficient for its purpose. In some cases, the combination of two or more tracking technologies might be beneficial, although such setups are still fairly rare.

Overall, we believe it is important to acknowledge the diversity of possible tracking hardware for ubicomp spaces, as the technology in a large part limits or affords what can be measured and how that technology can be deployed. In spite of the differences between these technologies, we believe the toolkit architecture should be designed to accommodate various means for sensing proximity. Throughout the next section of this chapter, we will emphasize the various methods of how the Proximity Toolkit supports several of these diverse tracking technologies.

6.2 Proximity Toolkit Architecture

The Proximity Toolkit is built as a collection of executable tools and programming libraries in C#. In this section we give a general overview of the toolkit architecture. We also introduce essential components of the toolkit that are somewhat out of the ordinary. In particular, we focus on both the server's *unified data model* (managed with the distributed Model-View-Controller pattern) and the flexible *plugin architecture*.

6.2.1 Overview and History

The following major technical components (that are explained in detail throughout this chapter) are part of the toolkit's flexible architecture (Figure 6.3). The central server manages the data model describing the current state of the tracked environment (Figure 6.3a). The server loads plugins to integrate raw data from different tracking hardware (Figure 6.3b). Depending upon the plugin, the tracking hardware may run on yet another computer or process connected to the server (Figure 6.3c); in most cases the communication between the tracking software and the plugin is implemented via TCP sockets. The tracking software (e.g., the Nexus software for the VICON motion capturing system) then connects to the corresponding sensor hardware (e.g., IR cameras) that tracks entities in the ubicomp ecology (Figure 6.3d). The server also manages network connections to diverse clients that access the server via the toolkit API. These clients are, for example, the proxemic-aware ubicomp applications that developers built to explore proxemic interactions (Figure 6.3e). The toolkit API connects with the server via TCP connections (Figure 6.3f), and internally handles the access to the server's data model which contains the proxemic information of the tracked ubicomp entities (Figure 6.3g). By using the methods and events provided by the toolkit API, and as demonstrated earlier with our walkthrough example of Chapter 5, programmers then create the client software that, for example, changes content on a tablet computer or Smartboard large display (Figure 6.3h) depending on a person's proximity to the device. The visual monitoring tool and the recording mechanism we introduced in Chapter 5 also take the information in the server's data model and visualize or record it (Figure 6.3i). The remainder of this chapter explains these components in more detail.

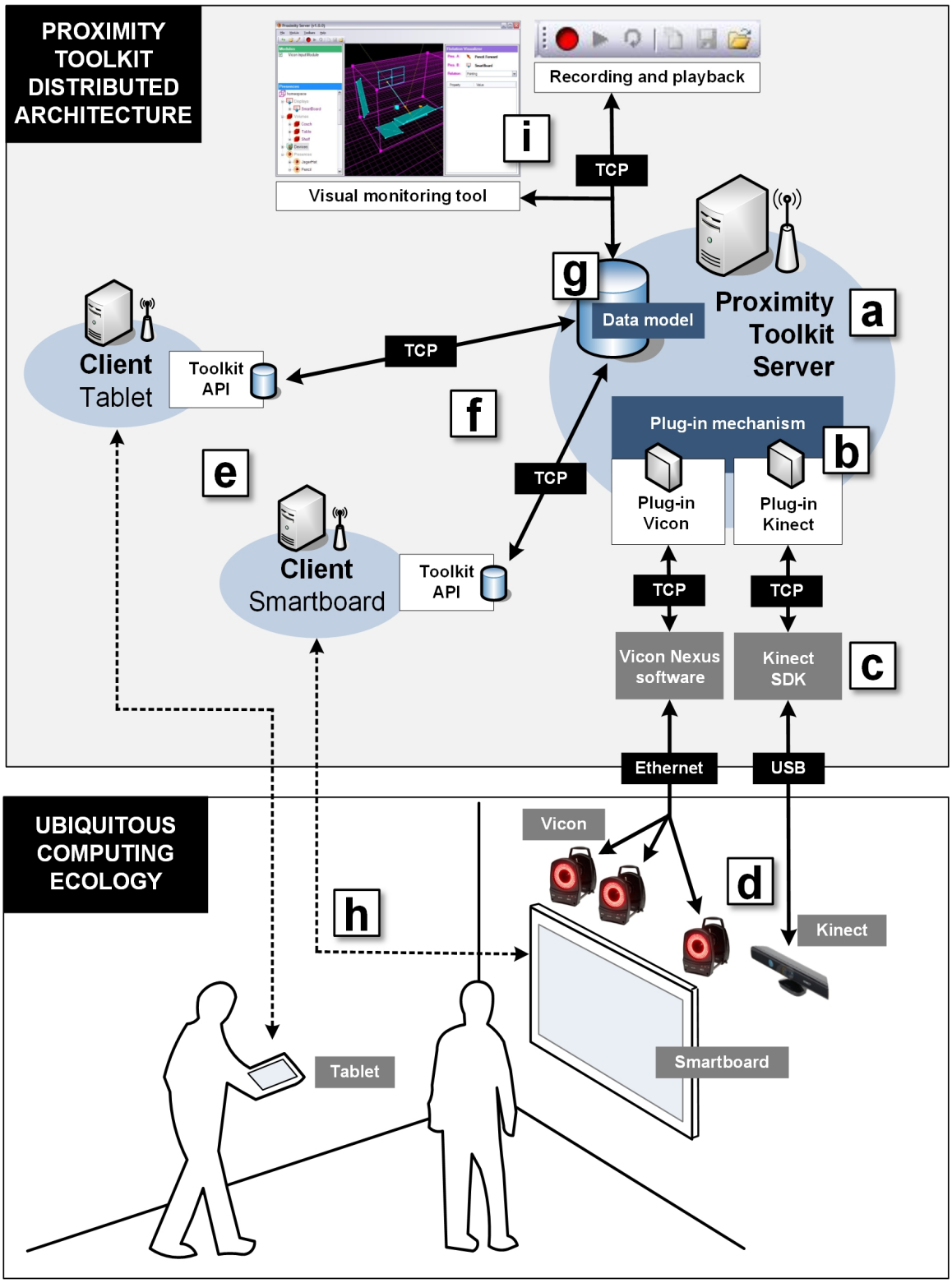


Figure 6.3 Architecture of the Proximity Toolkit.

The current Proximity Toolkit has evolved considerably over six years[28]. Its first version in 2007 went through several major development iterations, and was then significantly redeveloped. The early versions (Diaz-Marino and Greenberg 2010) were tightly linked to a particular tracking technology (i.e., the VICON motion capturing system). This meant that other technologies – such as those reviewed in Section §6.1 – could not be exploited. As a major change of the revised toolkit, the flexible architecture now separates sensing hardware from the proxemic data model derived from these sensors, which means that a variety of sensing technologies can be substituted or combined to derive proxemic information. The following sections further explain how the flexible architecture of the toolkit allows the integration of diverse tracking technologies.

6.2.2 Typical Technical Setups

Before we delve into the details of the toolkit architecture, we first briefly describe two typical technical setups of the Proximity Toolkit in ubicomp spaces – both shown in Figure 6.4 and Figure 6.5.

The first setup uses the Vicon motion capturing system (Vicon Motion Systems 2013) to track entities in a furnished (living room) space. The eight ceiling-mounted Vicon cameras (Figure 6.4a) are arranged in a circle around the room. All cameras are connected to the centralized *Ultranet* hub (Figure 6.4b) that is managing and powering the cameras. The *Ultranet* hub is also processing the captured raw images of all connected cameras. The hub is connected with a gigabit Ethernet link to a personal computer (PC) running Microsoft Windows 7, the Vicon *Nexus* software, and the Proximity Toolkit server (Figure 6.4c). The tracked area covered by the cameras is approximately 4x4x2

[28] The development of an early version of the toolkit (specifically built for the Vicon tracking system) was primarily done by Rob Diaz-Marino. The major redevelopment of the Proximity Toolkit was then done collaboratively by myself and Rob Diaz-Marino to include in particular: the API building blocks following the identified five proxemic dimensions in Chapter 3, the plugin architecture supporting diverse tracking systems, different plugins as input providers (VICON, Kinect, OptiTrack), the monitoring tools allowing visual real time inspection of proxemic information, and the development of additional tools, examples, and templates as described in Chapter 5 and 6.

meters (Figure 6.4d) and the space also includes a large interactive display (Smartboard, Figure 6.4e) connected to the PC.

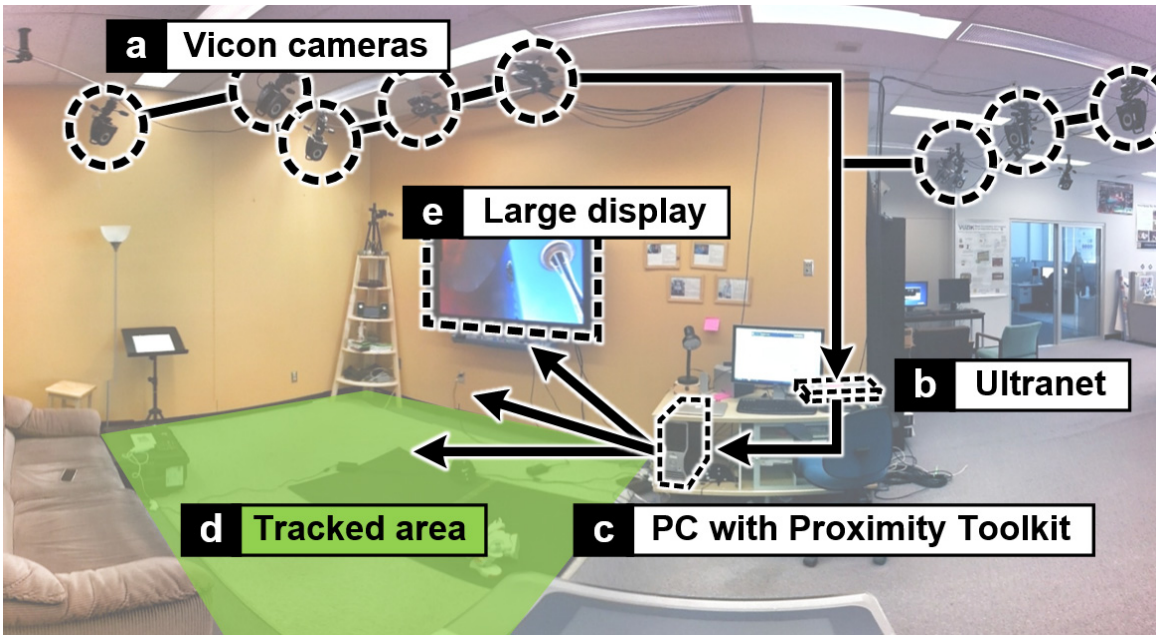


Figure 6.4 Typical physical setup of the Proximity Toolkit with the Vicon cameras.

The second setup uses the OpiTrack hardware (NaturalPoint 2013) to track a smaller physical volume around a table. The eight optitrack cameras (Figure 6.5a) are mounted on tripods and positioned around the table. Similarly to our first setup, these cameras are also connected to a hub (Figure 6.5b). The OptiTrack hub then in turn is connected to a PC running the OptiTrack software and the Proximity Toolkit server (Figure 6.5c). Again the space includes a large display (Figure 6.5d) as part of the tracked area on the table (Figure 6.5e). The overall physical volume tracked with this setup is significantly smaller as in our first setup – it is approximately 2x1x1 meters. A typical marker tracked by the cameras is shown in Figure 6.5f. The marker is composed out of a series of infrared reflective balls that are arranged in a particular pattern.

Other setup variations possible, depending on the available hardware, the layout of the physical space, and the requirements of the application to be built using the toolkit. But these two setups are typical for my (and students') projects built with the toolkit.

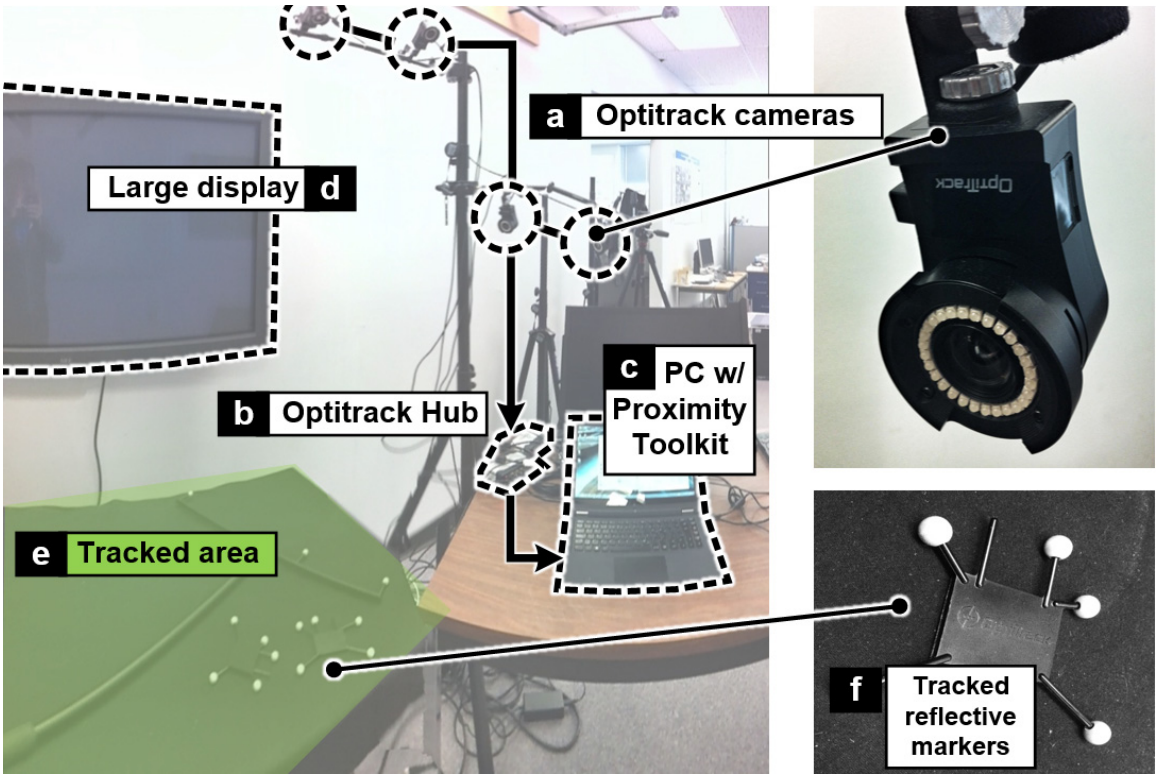


Figure 6.5 Typical physical setup of the Proximity Toolkit with OptiTrack cameras.

6.2.3 Major Building Blocks of the Proximity Toolkit API

The Proximity Toolkit has several major building blocks in its API whose purpose is to manage the proxemic information of the tracked environments and to give developers access to this information. These building blocks are implemented as classes in the toolkit API. We demonstrated how to use these classes/objects with the development walk-through example in Chapter 5, and will now focus on the higher level architecture.

The major object responsible for managing the data of the tracked environment is the `ProximitySpace`, as represented in the UML diagram in Figure 6.6a. The `ProximitySpace` maintains the global coordinate system, and the origin and dimensions of the tracking space. It also manages collections of all tracked entities. Because it is the root object managing all entities, it is implemented following the Singleton Pattern (Gamma et al. 1994) which guarantees that there is only a single instance of this object per running server.

There are three tracked entities (Figure 6.6c), all implemented as specializations of the `PresenceBase` class (Figure 6.6b). `TrackedPresence` tracks people, devices, and objects. The `DisplayPlane` defines all fixed digital screens such large wall displays. The `Volume` describes other fixed and semi-fixed features such as walls, doors, or a couch.

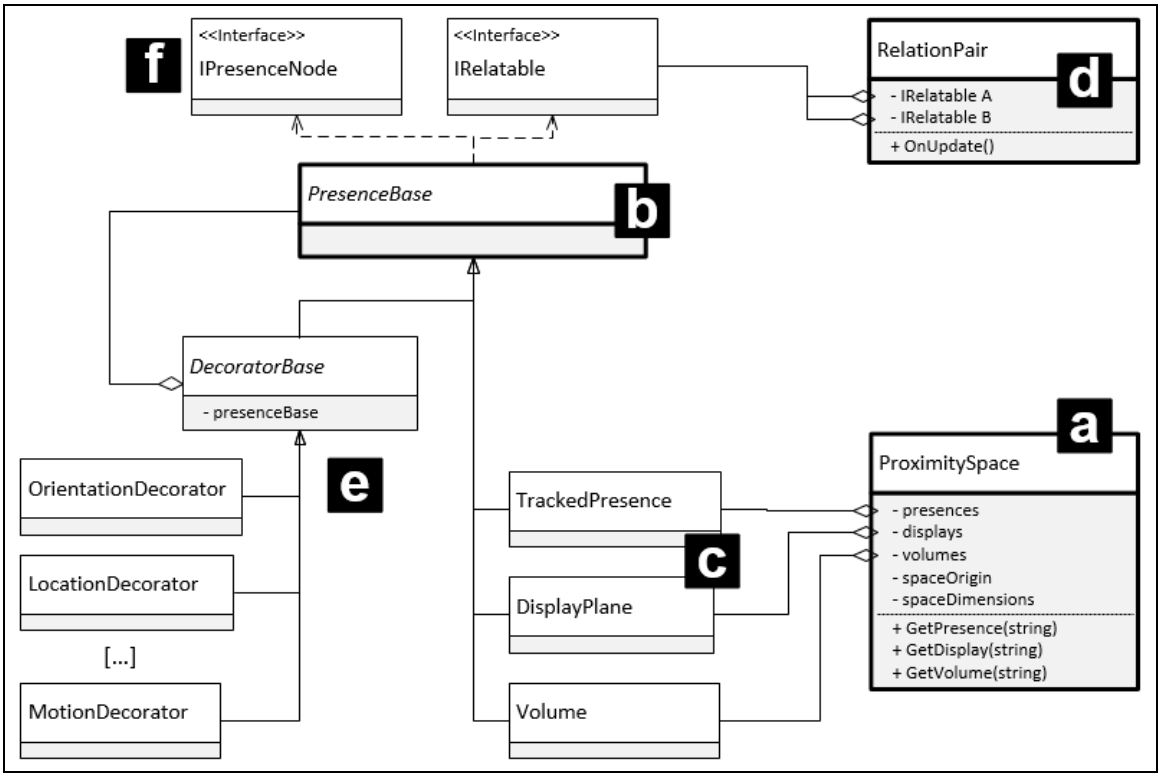


Figure 6.6 UML class diagram of essential building blocks of the Proximity Toolkit.

All `PresenceBase` derived classes also implement the `IRelatable` interface, which defines the necessary properties and access methods to compare the relationships between entities. This makes it possible to specify two entities and monitor their proxemic relationships with the `RealtionPair` class (Figure 6.6d). Any `RelationPair` object offers a series of events that other objects can subscribe for to receive notifications of changes in the proxemic relationship between tracked entities (also see the walkthrough example in Section §5.4).

These classes form the core of the available building blocks of the toolkit and expose the current state of the tracked environment and relationships between entities to the

developers. Two more aspects shown in the UML diagram that are essential for the flexibility of the architecture are explained shortly: the use of the decorator pattern to accommodate individual capabilities of the tracking hardware (Figure 6.6e) and the hierarchical tree structure being part of the distributed data model (with implementing the `IPresenceNode` interface (Figure 6.6f).

6.2.4 Plugin Modules for Supporting Diverse Tracking Hardware

The data providers of raw tracking input data are implemented as separate plugin modules, which are dynamically loaded into the proximity server at start-up. In this section we discuss the general implementation of plugin modules, and then explain the use of the decorator pattern to address diverse tracking capabilities.

The current toolkit includes three plugins to integrate different hardware that provide tracking data of entities in 3D space: the VICON motion capturing system (Vicon Motion Systems 2013) that tracks infrared reflective markers; the OptiTrack system (NaturalPoint 2013) that works similarly to the VICON but tracks smaller volumes[29]; and the Microsoft Kinect depth camera tracking people (Microsoft 2013). Each of these tracking systems comes with a software and API provided by the manufacturer that allows access to a stream of data providing the current position of tracked entities. This raw tracking data includes, for example, the 3D position of tracked infrared reflective markers (when using the motion capturing systems), or the 3D position of joints of a person's approximate skeleton model (when using the Kinect depth cameras). This raw data then forms the low level input that each of the three plugins inputs to the central data model of the server. The plugins then translate this information into higher level proxemic events made available by the Proximity Toolkit's API (see Table 5.1 for an overview of these events).

[29] The development of the OptiTrack plugin was done in collaboration with Stephan Huber and Roman Rädle from the University of Konstanz.

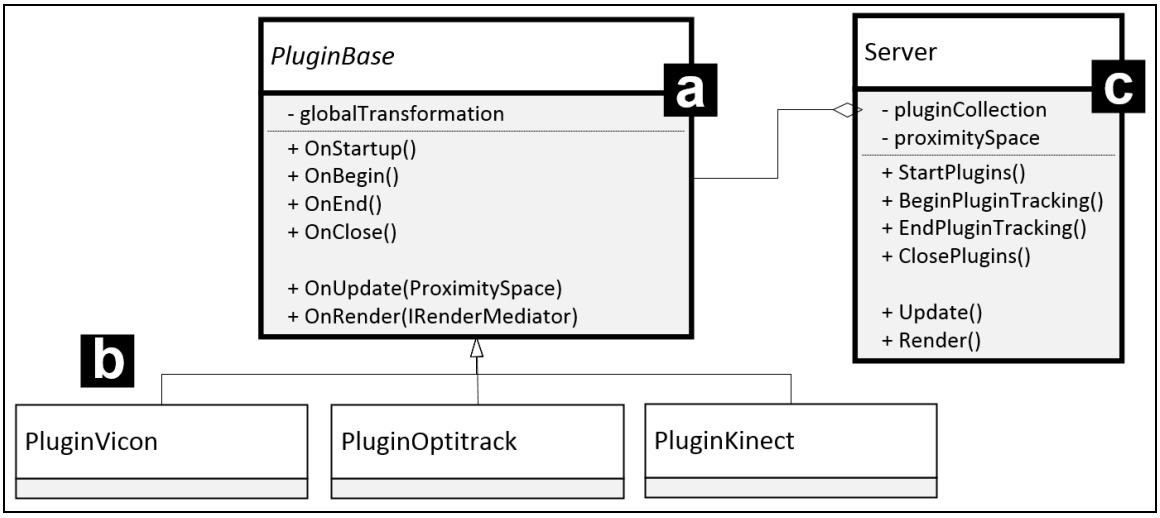


Figure 6.7 UML class diagram of plugin mechanism.

Templates, base classes, interfaces, and utility classes facilitate plugin development. Programmers begin with the plugin template, derived from the `PluginBase` class (Figure 6.7a). This base class provides a set of utility methods, such as one for affine transformations from the tracking system's local coordinate system to the Proximity Toolkit's unified coordinate system. This affine transformation matrix is determined through a one-time calibration process. To calculate the transformation matrix, the developer has to provide the calibration module with at least four 3D coordinates of points in space captured by both tracking systems simultaneously (e.g., the coordinates of a hand that is both tracked by the Kinect cameras and – by using a glove with attached IR markers – by the VICON infrared cameras). Next, developers implement several mandatory methods for each of the plugin class implementations (Figure 6.7b), including `OnStartup` (to initialize the tracking hardware), `OnBegin` (to begin the streaming of raw data), `OnEnd` (to stop the streaming), and `OnClose` (to shut down the tracking hardware). For example, in the current three plugin implementations, the `OnStartup` method causes the VICON plugin to initialize the underlying Nexus software (Vicon Motion Systems 2013), the Kinect plugin to initialize the OpenNI (PrimeSense 2013) software, and the OptiTrack (NaturalPoint 2013) plugin to initialize the Tracker software and start the NatNet streaming of tracking data to the plugin. All plugins are managed by the server (Figure 6.7c) in a pluginCollection object, and a set of methods

iterates over this plugin collection to start/close plugins, begin/end tracking, update tracking data, and render any visual elements.

Once initialized, plugins receive raw data of tracked people, objects, and/or devices in 3D space. The `OnUpdate` method of each plugin module is responsible to stream raw tracking data into the toolkit (Figure 6.4), and is called by the server once in each update cycle. The single parameter of the method is the `ProximitySpace` method mentioned above that is responsible for managing all tracked entities. The general structure of the code inside of this method is identical across most plugins: cycling through all entities tracked by the plugin (Figure 6.4, line 3), requesting the `TrackedPresence` object with the name of each particular entity (Figure 6.4, line 5), and changing the properties of that entity, such as the `IsVisible` property (Figure 6.4, line 6). The toolkit includes a flexible and extendable mechanism to address the presence of absence of proxemic tracking information in each of the different tracking technologies that is explained in the following section.

```
1.  private override void OnUpdate(ProximitySpace space)
2.  {
3.    foreach (Entity entity in this.trackedEntities)
4.    {
5.      TrackedPresence presence = space.GetPresence(entity.Name);
6.      presence.IsVisible = true;
7.      presence.[property] = [value];
8.      [...]
9.    }
10. }
```

Figure 6.8 General structure of the plugin's OnUpdate method.

The only remaining mandatory method that needs to be implemented for each plugin is the `OnRender` method. This allows the programmer to specify how the toolkit renders a tracked entity in the 3D view of the visual monitoring tool. The `OnRender` method the `IRenderMediator` interface is an interface providing basic graphic instructions (e.g., `DrawPoint`, `DrawSphere`, `DrawLine`, `SetColor`). Similarly to the implementation of the `OnUpdate` method, a programmer would again iterate over all currently tracked

entities of the plugin and specify the instructions to render each entity. For example, the VICON and OptiTrack plugins draw the outline of the marker layout of each entity (Figure 6.9a) or optionally add a 3D volume to indicate the physical shape of the entity (Figure 6.9b). In contrast, the Kinect plugin draws the approximate head and body position of the currently tracked person(s) (Figure 6.9c).

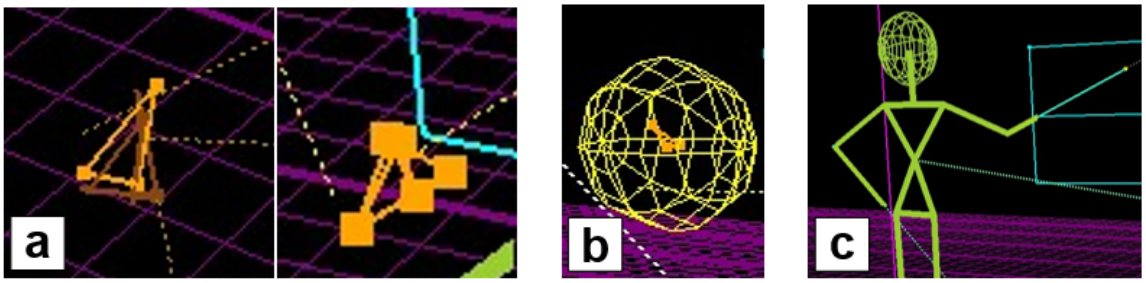


Figure 6.9 Different visualizations for tracked entities in the visual monitoring tool.

6.2.5 Applying the Decorator Pattern to Accommodate Individual Tracking Capabilities

In order to allow the integration of hardware with different tracking capabilities, the plugins specify the kinds of proxemic information they support. For example, a tracking system might gather information about the position of an entity, but *not* its orientation. Following the *decorator pattern* (Gamma et al. 1994), each plugin can specify exactly what kind of input data a particular tracking hardware provides. The decorator pattern describes a flexible mechanism to extend the functionality of objects at run-time. As a more flexible mechanism compared to sub classing, the pattern allows a programmer to add additional behaviour to a class, but also – most importantly – any possible combination of such additional behaviours. As shown in Figure 6.6e, the abstract `DecoratorBase` derives from `PresenceBase` and wraps a `PresenceBase` object (to keep its original behaviour). The plugin can then use decorator objects for *each* proxemic dimension of input data it supports, where it calls the update method on these decorators. For example, the `LocationDecorator` updates location of an entity and the `OrientationDecorator` its orientation. The tracked presences can contain any of the following data decorators (all specializations of the `DecoratorBase` class):

- **LocationDecorator** - Provides position in 3D space of the tracked entity, such as the 3D coordinates.
- **MotionDecorator** - Details about movement in 3D space, such as the acceleration or velocity when an entity is moving.
- **OrientationDecorator** - Details about where an entity is facing and what direction is defined as *up* and *forward*.
- **RotationDecorator** - Details about the rotation of the entity, such as the rotation of an entity along the three axis.
- **CollisionDecorator** - Details about the physical volume of the entity, such as the approximate shape.
- **PointerDecorator** - A set of defined rays that point in directions around the entity, such as the pointing arms of a person.
- **PoseDecorator** - The skeleton of an entity, such as the approximate skeleton pose of a person, or the shape of an object.

Plugins can add custom decorators for any proxemic information not yet supported by available decorators. To do this, developers need to create their own custom decorators for each class of additional properties they want to track with the plugin. For example, the VICON and OptiTrack plugins extend the toolkit with the following collection of decorators:

- **MarkerDecorator** - A set of detectable points that are subparts of the presence. These points are the location of all infrared-reflective markers that are tracked with the motion capturing systems.
- **DisplayDecorator** - The details of a display that is attached to a presence. This decorator is used, for example, to track the location of tablet computers or phones. The displays of these devices are then associated to the corresponding infrared-reflective marker tracked by the cameras.

Both of these decorators are useful to support the specific capabilities of the motion capturing systems, but would be meaningless for other tracking systems – such as the Kinect cameras, as they are not capable of tracking individual markers or digital displays.

During each update cycle (i.e., when the `OnUpdate` method mentioned earlier is called), the decorator objects update the proxemic information in the server's unified data model as the proxemic information of each entity. For example, Figure 6.6 shows the partial source code when using three different decorators to add information for a tracked entity in the `OnUpdate` method: adding *orientation* information (lines 1,2), *location* information (lines 3,4), and *pose* information (lines 5,6: updating the joint describing the torso position of a person). No high-level calculations on raw input data are required for the plugin implementation, as these are performed by the proximity server or directly on the clients via the toolkit's API.

```
1.  OrientationDecorator orientationDecorator =
     (OrientationDecorator)presence.GetDecorator(typeof(OrientationDecorator));
2.  orientationDecorator.Update(entity.Orientation);
    […]
3.  LocationDecorator locationDecorator =
     (LocationDecorator)presence.GetDecorator(typeof(LocationDecorator));
4.  locationDecorator.Update(entity.Location3D);
    […]
5.  PoseDecorator poseDecorator =
     (PoseDecorator)presence.GetDecorator(typeof(PoseDecorator));
6.  poseDecorator.UpdateJoint("Torso", torso.Location3D, torso.Direction);
```

Figure 6.10 Using multiple decorators to add information in the OnUpdate method; called in line 8 of Figure 6.8.

6.2.6 Data Hierarchy

All values corresponding to the presences and each decorator are managed by a hierarchical tree structure that corresponds to the entries in the shared dictionary model. Each of the above presence and decorator objects implement a common interface called `IPresenceNode` (Figure 6.6f). These objects form the traversable data tree that goes from the `Space` root object, right down to the individual data properties of the presence

and decorator objects. The `IPresenceNode.Parent` property holds the parent node in the tree structure, while the `IPresenceNode.Nodes` structure holds the list of child nodes. If necessary, the code from within each plugin can access the data for any sub tree by using an indexer on a node, and a dot-separated path string as follows:

```
object data = Node["NodeName.NodeName"];
```

or, if a particular data type is expected:

```
DataType data = Node.GetValue<DataType>("NodeName.NodeName");
```

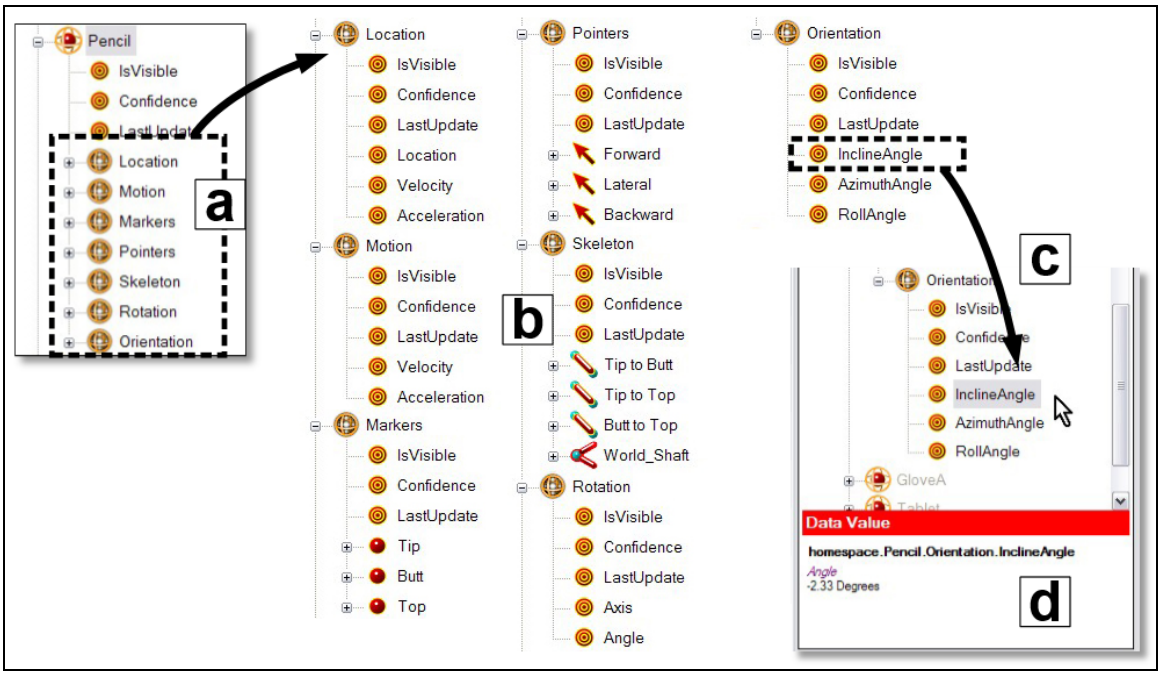


Figure 6.11 Viewing all available input dimensions of a tracked entity in the Proximity Toolkit visual monitoring tool.

All available dimensions of input data for each tracked entity are directly visible in the monitoring tool. For example, the list view (Figure 6.11) and 3D view (Figure 5.3) give direct feedback about the available proxemic dimensions. The listed properties in Figure 6.11 all directly correspond to the `IPresenceNode` tree structure. The developer can explore all properties by selecting any of the dimensions (Figure 6.11a), which opens a detailed view of all properties (Figure 6.11b) where they can select an individual item

(e.g., the `InclineAngle` in Figure 6.11c) and view the current value (Figure 6.11d). The available input dimensions can be also checked directly from the client API by using the `HasDecorator` method of the `PresenceBase` object and the `IsVisible` properties for each available input dimension. This allows developers of proxemic-aware applications using the toolkit to review the capabilities of the currently used tracking technology before beginning their programming.

The complete data of this hierarchical tree structure is directly mirrored in the toolkit's distributed data model, explained next.

6.2.7 Unified Data Model and Distributed Model-View-Controller

The Proximity Toolkit server's unified data model is the basis for a distributed Model-View-Controller (dMVC) architecture (Boyle and Greenberg 2005; Greenberg and Roseman 1999). Using dMVC, the server stores the tracking data as a unified data model, which other clients can then access and update as needed. This enables the prototyping of proxemic-aware applications. Of importance is that multiple clients – even those residing on different computers – can connect to the Proximity Toolkit server and receive updates about the proxemic relationships between entities. This data model is implemented through a shared hash table that is accessible through TCP connections (Boyle and Greenberg 2005). While application developers can access and modify this underlying data model if desired, this is rarely necessary and thus it is normally kept hidden from view. Instead, the dMVC architecture is accessed by the toolkit client API, the monitoring tool, and is used to calculate and visualize the higher level proxemic relationships between entities as used by developers.

Figure 6.12 describes the basic design of the dMVC architecture. Tracking hardware can act as *Controllers* (Figure 6.12a), where each hardware can update the server's unified data *Model* (Figure 6.12b) with information of the tracked entities in space (where the access is implemented with the plugins explained shortly). Different clients can then act as *Views* to that data model, where they subscribe for change events in this unified data model. For example, the visual monitoring tool described in Chapter 5 is one of these views, displaying all currently tracked entities in a graphical 3D visualization (Figure

6.12c). Or, the clients running on the tablet computer that a person is holding or the application running on a large wall display can be views to the data model, too, where they react to the changes of the proxemic relationships between entities (Figure 6.12d). Last, clients can be both a view and a controller. For example, the toolkit's recording and playback function acts as a viewer when recording data from the unified data model, and acts as a controller when playing back the recorded data, as it then sends the recorded data back to the data model (Figure 6.12e).

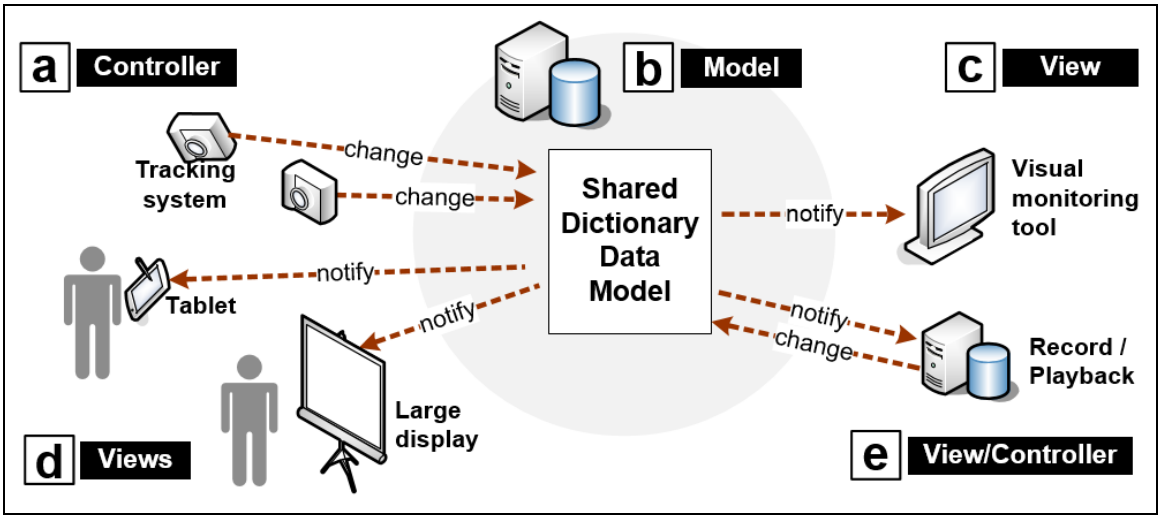


Figure 6.12 Unified data model and the distributed Model-View-Controller architecture of the Proximity Toolkit.

Under the covers, the server's unified data model is held as a collection of hierarchical key-value pairs representing the current state of the tracked environment (Figure 6.13). Part of these key-value pairs describe the setup of the environment, such as the volume of the tracked space, the location and shape of fixed features (e.g., doors, walls, wall displays) and semi-fixed features (e.g., couch, tables).

A second part of the key-value pairs describe all currently tracked entities. These keys are structured according to the following pattern:

```
/[space]/[presence]/[proxemic-dimension]/[identifier]
```

- **[space]** is the identification name of the `ProximitySpace` root object mentioned above.
- **[presence]** is the unique identification name for an entity (i.e., all possible classes derived from `PresenceBase`). This can be a person (e.g., `person1`), a mobile device (e.g., `tabletPC`) or a device at a fixed location (e.g., `smartboard`), a mobile object (e.g., `pen`) or an object at a fixed location (e.g., `couch`).
- **[proxemic-dimension]** is the identification name of the particular proxemic dimension (implemented by any of the decorators) that the data value belongs to (e.g., `orientationdecorator` for values about the orientation of an entity).
- **[identifier]** is the identification name of the particular value (e.g., `rollangle` for the current roll angle of the tracked entity)

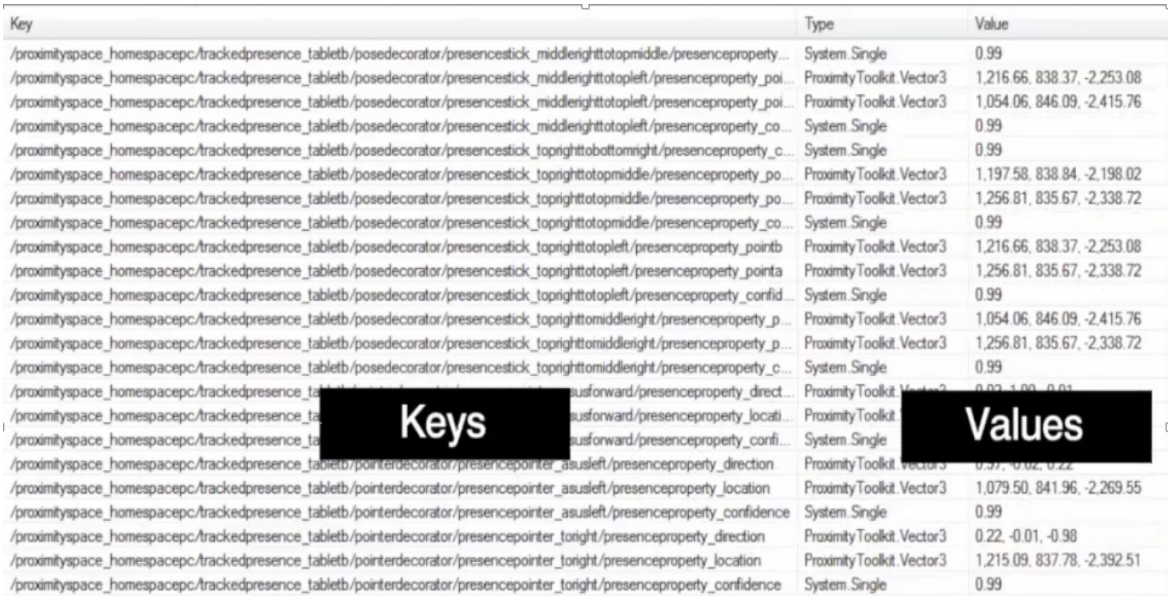


Figure 6.13 Key/value pairs of the Proximity Toolkit data model viewed with the shared dictionary explorer.

For example, the key-value pairs in Figure 6.14 are part of the data model of a tracked person (i.e., orientation, location, motion/movement). Table 5.1 provides an overview of the different types of proxemic data that is stored in the unified data model, and also lists

the data type of the values (Table 5.1, column 5). The server and the toolkit API calculate necessary proxemic relationships for the entities present in the data model. On the one hand, the calculations for an individual entity are done on the server and added to the unified data model (e.g., the acceleration and velocity data that is calculated based on the location data and timestamps). On the other hand, calculations of proxemic relationships *between* entities is done in the toolkit API, i.e., they are done on each client individually. To reduce computational overhead, the necessary 3D calculations are done only on demand, for example, when a client subscribes to events for a particular relationship between two entities. This is important, as calculating all possible relationships (which would likely include many that are not needed by the developer) would be prohibitively expensive.

```
/home/person1/orientationdecorator/rollangle        = -95.5
/home/person1/locationdecorator/location            = [12.4, 3.7, 8.2]
/home/person1/motiondecorator/velocity              = [0.1,  0.6, 20.5]
```

Figure 6.14 Example key/value pairs describing a tracked entity.

6.2.8 Substitution of Tracking Systems

Tracking systems/plugins can be *substituted*, providing that their hardware gathers similar tracking information. Due to the separation of tracking hardware and API, a programmer's access to this proxemic information via the toolkit API remains unchanged, regardless of the underlying tracking mechanism used.

For example, a programmer can use the OptiTrack motion capturing system instead of the VICON by simply selecting the corresponding plugin in the toolkit server start-up dialog. In this case, the transition is seamless, as both systems provide equivalent raw tracking information (although they have differences, this is mostly in how the low level tracking information is supplied, formatted, gathered and translated by the plugin, and in the tracking accuracy and maximum tracking volume as described in Section §6.1). In a similar fashion, a programmer can replace tracking systems that use different mechanisms to track entities in space. For example, instead of using the depth camera for tracking people's positions and postures (Figure 6.15a), a programmer can use the

infrared (IR) motion capture system instead by attaching IR reflective markers to a person's body (Figure 6.15b). This flexibility to substitute the underlying tracking technologies ideally gives the developers the freedom to experiment with alternative setups, where they can explore the particular strength and trade-offs from these systems. However, the differences between tracking systems cannot be entirely hidden, as discussed next.

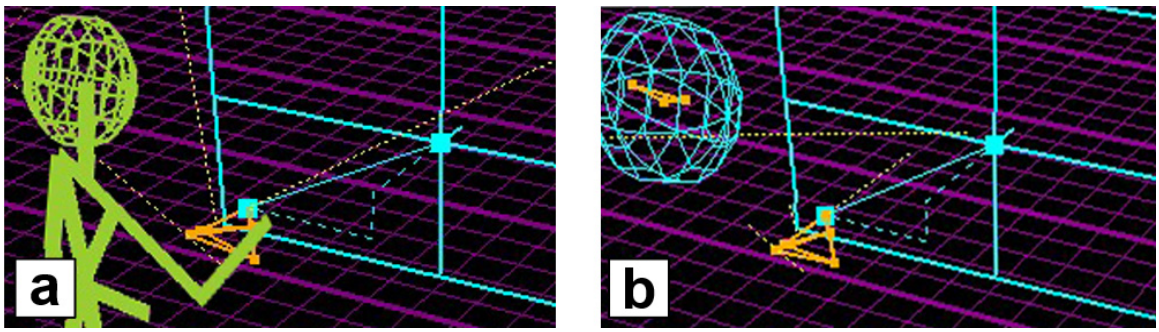


Figure 6.15 Substituting of tracking systems: (a) tracking a person with Kinect cameras or (b) with motion capturing cameras by attaching markers to a hat the person is wearing.

6.2.9 Dealing with Uncertainty of Tracking Information

All 3D tracking systems provide input with some kind of uncertainty. As tracking systems differ in precision of tracking data they provide (and whether they are even able to supply particular kinds of data), plugins are required to provide additional information about this uncertainty. In particular, the toolkit provides two values describing tracking uncertainty. First, the Precision value specifies how accurate the system tracks entities (normalized between 0.0 and 1.0). Precision is defined as *1 mm / [minimum resolution in mm]*, where the minimum resolution is measured in mm (e.g., minimum resolution is 1mm for the Vicon system, and 20mm for Kinect). Thus, the lower the resolution, the higher the precision value is. Second, the Confidence value indicates the estimated accuracy of the provided tracking information. It again ranges from 0.0 - 1.0, where 0 is 0% confidence (i.e., lost tracking), and 1 is 100% confidence. In our plugins, the Vicon and OptiTrack motion capturing software provide estimates accuracy information for all tracked markers, and this value is mapped directly to our Confidence

value. In contrast, the Confidence value of a person tracked by the Kinect depth cameras is calculated by dividing the recognized parts of the body (e.g., arms, legs) to the total number of possible parts to recognize (i.e., the Confidence is 1.0 if the full body of a person is tracked). These confidence and precision values are applied to each individually tracked entity. Furthermore, the precision value can differ depending on where in the 3D space an entity is tracked (e.g., precision is higher when a person stands closer to the depth sensing camera).

A developer can monitor the quality of input data with the visual monitor tool. A table view lists confidence and precision values, and the 3D view gives direct feedback of the precision (or absence) of tracking. Similarly, the API exposes the Confidence and Precision values of each entity. It also includes the `IsVisible` (false if lost tracking) and `LastUpdated` (timestamp of the last update) properties.

6.2.10 Merging Tracking Data

In cases where different plugins provide *complementary* tracking information of a single entity, the information can be merged in the proximity server's data model. For example, the Kinect and Vicon systems could both track a single person simultaneously: the Kinect system provides information about the person's body position in 3D space, and the Vicon system tracks a glove the person is wearing in order to retrieve fine-grained information of the person's finger movements. Both plugins then update the entity's data model in the server with their tracked information.

If two systems provide *overlapping/conflicting* tracking data (e.g., two systems provide information about an entity's location), the information will be merged in the server's data model. To do so, the server calculates a weighted average (taking the Confidence and Precision values) of all values received in a certain time frame (i.e., one update cycle) and updates the proxemic data model of that entity. This means, that the higher the confidence and precision value of a given entry, the more it affects the final merged value of that entity.

Our current implementation of the merging algorithm is only a first step towards combining the data of multiple tracking sources. As part of future research and extensions to the toolkit, other algorithms for tracking data fusion (e.g., Zhao et al. 2005) could be seamlessly implemented on the server level (thus not requiring any changes to the plugins or the API). Furthermore, it would be possible to extend the toolkit's uncertainty information via Schwarz et al.'s (2010) framework for handling ambiguous input, where this could track ambiguous information simultaneously and delay event triggers. Later in Chapter 9 we will come back to this problem of merging tracking data, where we introduce a sensor fusion approach to merge data from multiple low-fidelity sensors to gather information about proxemic relationships between people and devices.

6.2.11 Availability as Open Source Project

We believe that the architectural details provided in Part II are sufficient to allow a highly skilled toolkit developer to understand and replicate the basic ideas of the Proximity Toolkit. Even so, this would require significant effort, as the toolkit as a whole is quite complex. Consequently, we make the source, executables, templates, documentation and examples fully available at:

http://grouplab.cpsc.ucalgary.ca/cookbook/index.php/Toolkits/ProximityToolkit

The availability of the source code serves several purposes. First, it allows a toolkit developer to probe our toolkit for details not mentioned here, including run-time details such as performance. Second, it allows the toolkit developer to extend the existing toolkit (e.g., by adding plugins as mentioned above). Third, it helps the toolkit maintainer understand the system, which would help debug the system after the original developers are no longer working on it. Finally, it helps the application developer understand what is going on under the covers, which in turn would help them understand what can and cannot be done with the system.

6.2.12 Discussion

The Proximity Toolkit was built from the beginning as a prototyping tool facilitating exploration of proxemic interactions. Our primary reason for integrating expensive motion capturing tracking technology (i.e., VICON and OptiTrack via the plugins) as the dominant provider of tracking data was to allow the developers (and myself) to use the toolkit to explore very high-fidelity proxemic interactions. This allowed myself and others to develop and explore many novel interaction concepts (explained shortly in Part III), without being overly limited by the shortcomings of available tracking hardware. Nevertheless, it is important to note that the prototypes implemented with this high-end motion capturing systems are exactly that: prototypes. Due to the extensive hardware setup, high cost, and the necessary augmentations of tracked entities with reflective markers, they are not meant for out of the lab deployments. These setups also do no scale very well beyond room-sized environments and have limitations regarding the maximum number of tracked entities. They also suffer from typical problems of camera-based systems, such as losing tracking when markers are occluded (which can be partially addressed by increasing the number of cameras and varying the camera angles).

Even so, it is important to realise that this high-fidelity tracking information is not imperatively necessary for implementing proxemic-aware ubicomp applications. Instead, strategies of proxemic interaction can be adjusted and / or integrated into systems using hardware that ranges across the low- to high-fidelity spectrum of tracking fidelity. Depending on the sensing technology used, the fine-grained information about the mentioned proxemic dimensions might not be always available, or only available in low fidelity. In Chapter 9 we further explore the use of low-cost tracking sensors to gather proxemic information across the five identified dimensions.

6.3 Conclusion

In this chapter, we explained the underlying technical architecture of the Proximity Toolkit. Most importantly, the architecture allows the integration of diverse tracking hardware, allowing ubicomp developers to mix and match different tracking

technologies when prototyping their applications. We explained different mechanisms that enable this flexible architecture, such as the plugin modules using the decorator pattern to accommodate different tracking capabilities, and the shared data model implementing the distributed model-view-controller. What is also important is that the developer's access via the toolkit's API does not change regardless of what tracking system is currently used.

Part II introduced the mechanics of the Proximity Toolkit, explained how its toolkit API provides a set of higher level building blocks that facilitate prototyping of proxemic-aware applications (Chapter 5), and described how its extensible toolkit architecture allows integration of diverse tracking technologies (Chapter 6).

Part III of this dissertation will cover diverse explorations of proxemic interactions in ubicomp ecologies. As we will see, the toolkit proved not only invaluable for my own explorations of proxemic interaction concepts, but also for the explorations of many others including students (in our lab, our graduate courses, and from other universities).

PART III – EXPLOITING PROXEMICS IN UBICOMP ECOLOGIES

In this third part of the dissertation we explore the potential of considering proxemics for interaction design in small space ubicomp ecologies. The three chapters illustrate the application of proxemic interaction and the five dimensions from Chapter 3 for the design of novel interactions in *ubiquitous computing ecologies*. That is, in these chapters we introduce novel interaction techniques that take into account the proxemic relationships between the different entities in ubicomp ecologies – the people, devices, objects, and the environment – to mediate people's interaction with digital devices. These chapters also serve technological purposes: Chapters 7 and 8 demonstrate the versatility of the Proximity Toolkit for rapid prototyping and exploration of proxemics in ubicomp, while Chapter 9 investigates the use of low-cost tracking technologies to gather information about people's and devices' spatial relationships.

Chapter 7. Person/People-to-Device Proxemic Interactions

In this chapter[30] we focus on proxemic interactions of one person or multiple people with a large interactive display situated in a ubicomp ecology. We introduce a set of novel interaction techniques for people's interaction with large interactive screens. For the design of these techniques we consider relationships of different aspects of a small space ubicomp ecology: the relationships of a single person to a device, of multiple people to a device, and of non-digital objects to people and devices. In particular, we exploit the knowledge of distance, orientation, location, movement, and identity as part of the introduced dimensions of proxemic interactions (§3.2) to drive the possible interactions.

The remainder of the chapter is structured as follows. First, we introduce a scenario of people using a proxemic media player, an application we use as a running example throughout this chapter to explain the different interaction concepts (§7.1). This

---

[30] Portions of this chapter are published as:

Ballendat, T., Marquardt, N. and Greenberg, S. (2010) Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. In *Proceedings of the ACM Conference on Interactive Tabletops and Surfaces – ACM ITS'2010.* (Saarbruecken, Germany), ACM Press, 10 pages, November 7-10.

Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R. and Wang, M. (2011) Proxemic Interactions: The New Ubicomp? In *ACM Interactions*, 18(1):42-50. ACM, January-February. Invited cover story.

As described in the Technical Acknowledgements section, the development of the proxemic-aware media player application demonstrating the interaction concepts was done in collaboration with Till Ballendat.

example ubicomp application reacts to nearby people and their relationship to devices, objects, and the environment. Next, we introduce novel interaction concepts incorporating the knowledge of the fixed and semi-fixed feature space (§7.2), interpret a person's directed attention extending attentive user interfaces (§7.3), and support fine-grained explicit interactions (§7.4). We then demonstrate how proxemic information can regulate interaction based either on continuous movement or by movement in and out of discrete proxemic zones (§7.5). In §7.6, we introduce the gradual engagement design pattern, and then describe how we can apply the stages of the gradual engagement pattern to the interaction with the media player (§7.7). Next, we detail how the system can consider people's identity (§7.8). Finally, we explain how the techniques extend beyond pairwise interaction (§7.9).

7.1 Scenario: The Proxemic Media Player Application

Throughout this chapter, we use the example of people interacting with a home media player application located in a living room. Later sections, which present concepts for designing proxemic interactions, will use episodes from this scenario to anchor the discussion. Please note that the video figure[31] shows the media player application and the interaction techniques in action.

The scenario follows Fred, who is approaching the display from a distance. We explain how the system supports Fred's implicit and explicit interaction with the digital surface as a function of his identity and his distance and orientation relative to the surface and to other features in the room. We also explain how the system responds to other proxemic dimensions such as other objects and other people in that space.

[31] Video available for download and streaming at: http://grouplab.cpsc.ucalgary.ca/Publications/2010-ProxemicInteractions.ITS

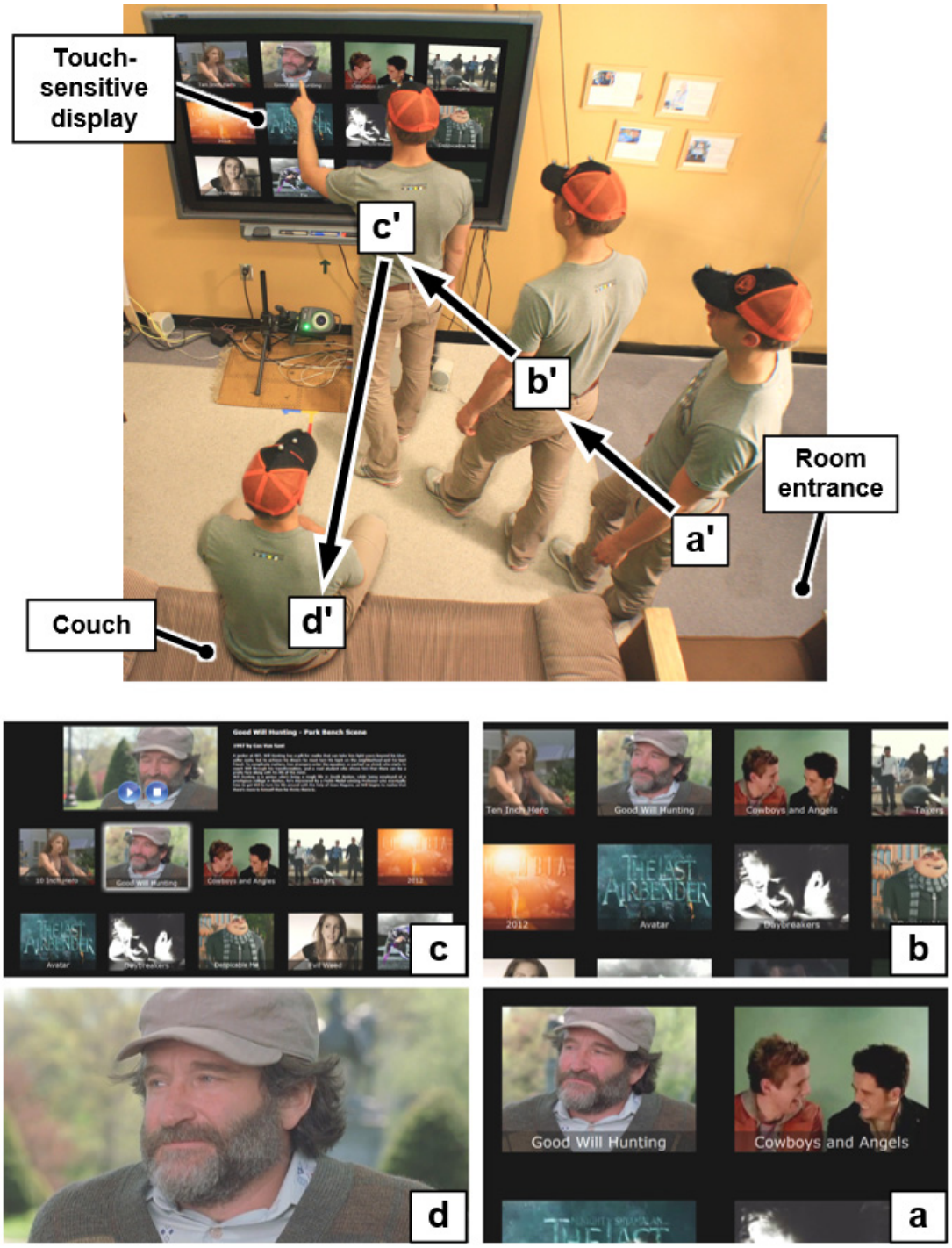


Figure 7.1 Proxemic Interaction: (a) activating the system when a person enters the room, (b) continuously revealing more content as the distance of the person to the display decreases, (c) allowing explicit interaction through direct touch when person is in close proximity, and (d) implicitly switching to full screen view when the person takes a seat.

The primary interface of the interactive media player application supports browsing, selection, and playback of videos on a large wall-mounted digital surface: a 52 inch touch-sensitive SmartBoard from Smart Technologies, Inc. (Figure 7.1 top left). The Proximity Toolkit software tracks (via the VICON plugin and using reflective infrared markers) the location and orientation of nearby people, objects, and other digital devices. All equipment is situated in a room that resembles a domestic living room.

Figure 7.1 (top) shows Fred approaching the display at four distances (a' – d'), while the four scenes below show what Fred would see at those distances. Initially, the proxemic media player is 'asleep' as the room is empty. When Fred enters the room at position a' in Figure 7.1, the media player recognizes Fred and where he is standing. It activates the display, shows a short animation to indicate it is activated, and then displays four large video preview thumbnails held in Fred's media collection (Figure 7.1a). As Fred moves closer to the display at b', the video preview thumbnails and titles shrink continuously to a smaller size, thus showing an increasing number of available videos (Figure 7.1b). When Fred is very close to the surface at c', he can select a video directly by touching its thumbnail on the screen. More detailed information about the selected video is then shown on the display (Figure 7.1c), which includes a preview playback that can be played and paused (Figure 7.1c, top left), as well as its title, authors, description and release date (Figure 7.1c, top right). When Fred moves away from the screen to sit on the couch located at d', his currently selected video track starts playing in full screen view (Figure 7.1d). If Fred had previously seen part of this video, the playback is resumed at Fred's last viewing position, otherwise it starts from the beginning.

Fred tires of this video, and decides to select a second video from the collection. He pulls out his mobile phone and points it towards the screen (Figure 7.2a). From its position and orientation, the system recognizes the phone as a pointer, and a row of preview videos appears at the bottom of the screen. A visual pointer on the screen provides feedback of the exact pointing position of Fred's phone relative to the screen. Fred then selects the desired videos by flicking the hand downwards, and the video starts playing. Alternately, Fred could have used a non-digital pen to do the same interaction (Figure 7.2b).

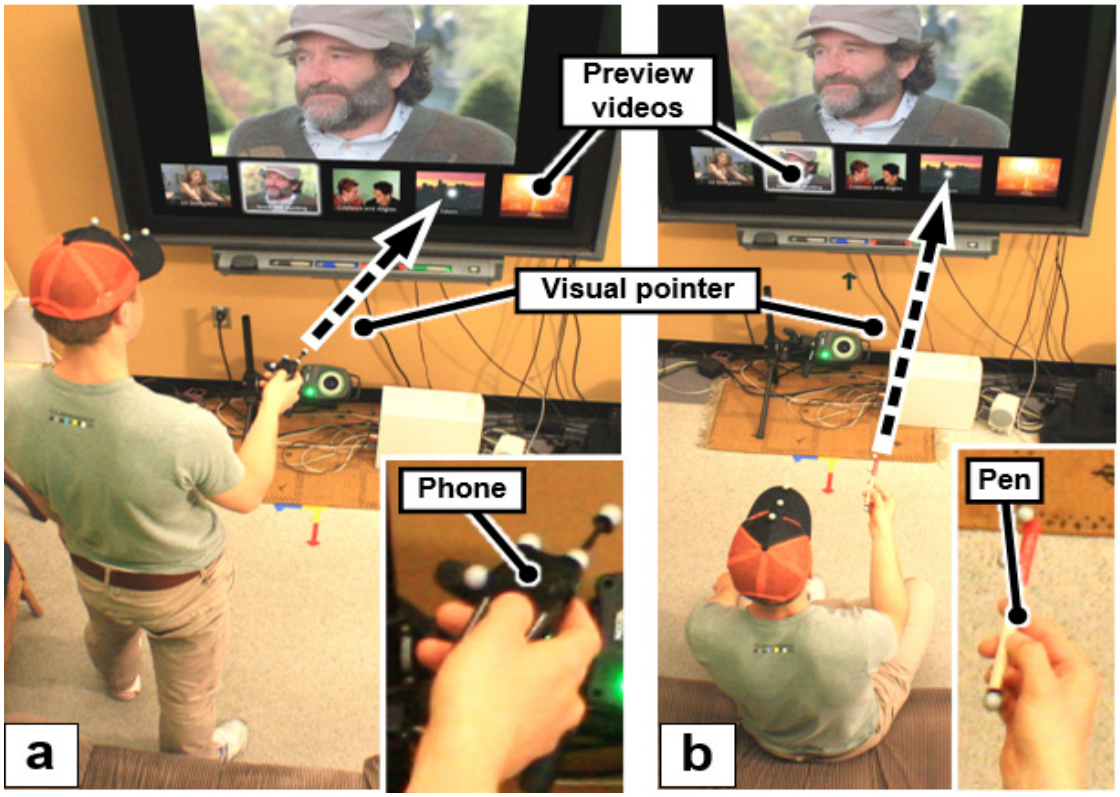


Figure 7.2 Explicit interaction triggered through distance and orientation between a person and digital / non-digital physical artefacts: (a) cell phone, (b) pen.

Somewhat later, Fred receives a phone call. The video playback automatically pauses when he answers the phone (Figure 7.3a), but resumes playback after he finishes the call. Similarly, if Fred turns away from the screen to, for example, read a magazine, the video pauses, but then continues when Fred looks back at the screen (Figure 7.3b).

As Fred watches the video while seated on the couch, George enters the room. The title of the currently playing video shows up to at the top of the screen to tell George what video is being played (Figure 7.4a). When George approaches the display, more detailed information about the current video becomes visible at the side of the screen where he is standing (Figure 7.4b). When George moves directly in front of the screen (thus blocking Fred's view), the video playback pauses and the browsing screen is shown (Figure 7.4c). George gets control of the application and can now select other videos by touching the screen. The view changes back into full screen view once both sit down to watch the video (not shown, but similar to position d' at Figure 7.1).

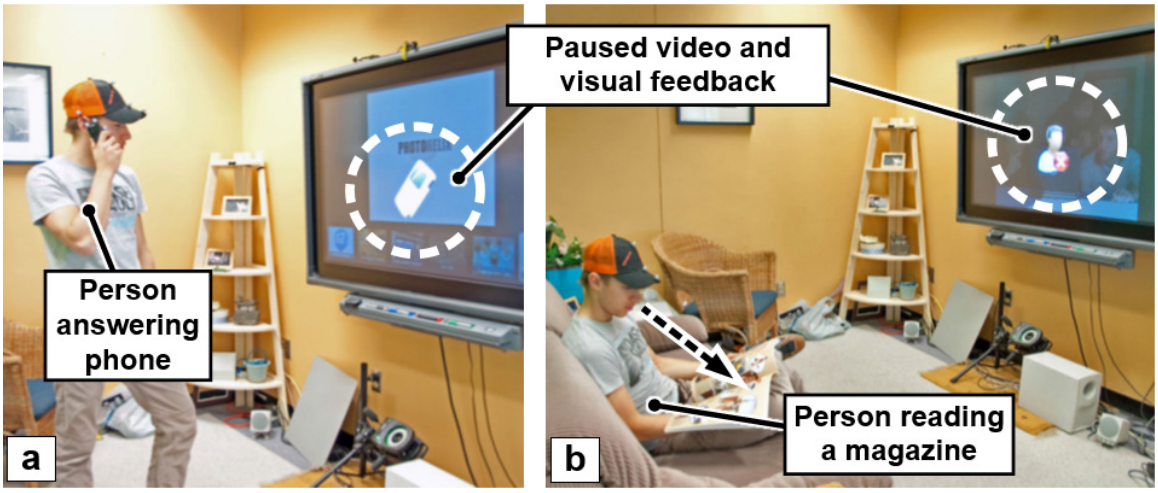


Figure 7.3 Integrating attentive interface behaviour: pausing the video playback when the person is (a) answering a call or (b) reading a magazine.

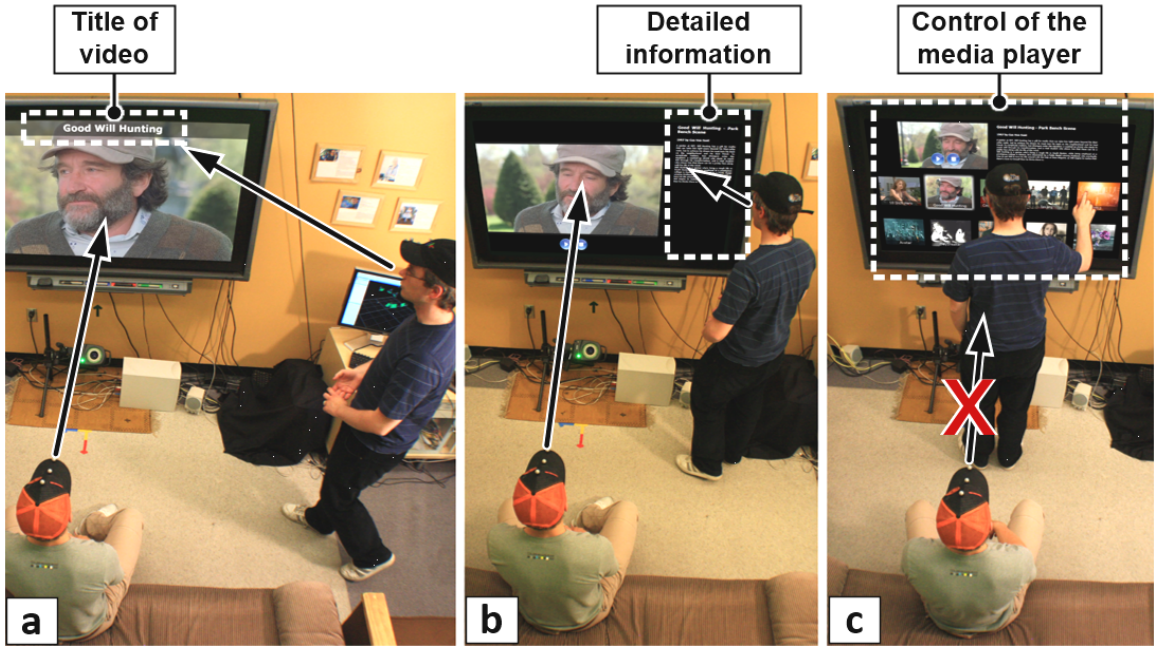


Figure 7.4 Mediating between multiple people: (a) incoming person sees basic information such as video title; (b) as one moves closer, the split view provides a more detailed video description; (c) when within reach of the display, the person gets full control.

If Fred and George start talking to each other, the video pauses until one of them looks back at the screen (Figure 7.5). When all people leave the room, the application stops the video playback and turns off the display (not shown).

While this media player is a simple application domain, it provided a fertile setting to develop and explore concepts of proxemic interaction[32]. In the next section we will discuss the details of proxemic interaction concepts associated with a single person or multiple people interacting with a large digital surface, where these concepts apply the five input dimensions (§3.2) in meaningful ways to people's proxemic interactions with ubicomp systems.

Figure 7.5 Integrating attentive interface behaviour: two people talking to another.

7.2 Incorporating the Fixed- and Semi-fixed Feature Space

One promise of Ubicomp is to situate technology in people's everyday environments, in a way that lets people interact with information technology in their familiar places, environment and within their social routines. Dourish framed this concept as *embodied interactions* (Dourish 2001b); technology that is seamlessly integrated into people's everyday practices, rather than separated from them (§2.1.2). Context-aware computing is one outcome of this, where some kind of context-aware sensing (Schilit et al. 1994) provides devices with knowledge about the situation around them (§2.1.3). This sensing

[32] We make no claim one way or another that our proxemic media player is somehow 'better' than existing media players.

usually involves measuring a coarser subset of the five dimensions discussed earlier (§3.2), e.g., very rough positions, and other factors such as noise, light, or tilting. In this section we contribute to these concepts by introducing the notion of context-aware systems that mediate embodied interaction by understanding the proxemic relationships of people (as defined by the five dimensions) to the fixed- and semi-fixed feature space surrounding them (§3.1.3).

For an interactive system (such as the interactive wall display in the media player application), knowledge about the *fixed feature space* includes the layout of the fixed aspects of the room, such as existing walls, windows, and territorial boundaries such as doors and entrance ways. A fixed feature space also includes knowledge about fixed displays – such as a digital surface – located in this environment. To elaborate, knowledge about the position of the fixed entrance doors allows the system to recognize a person entering the room from the doorway (Figure 7.6a), where (as in our scenario) it takes implicit action by awakening the fixed display from standby mode. Similarly, knowing the position of the fixed display means that the interface on that display can react as a person approaches it (Figure 7.6a).

Semi-fixed features in the environment include all furniture, such as bookshelves, chairs, and tables whose position may change over time. While it is somewhat object-dependent, semi-fixed features often remain at specific locations, but are per se movable objects that people rearrange to adapt to changed situations (such as moving a group of chairs around a table). Unlike fixed features whose position needs to be configured only once, knowledge about the positions of semi-fixed features must be updated over time as changes are noticed.

Knowledge of semi-fixed features can also mediate interaction. To illustrate this point, we compare two stages of a person relative to the media player's interactive surface: approaching from a distance (see Figure 7.1, position a') and watching the video when seated at the semi-fixed couch (Figure 7.1 position d'). The actual distance of the person relative to the surface is similar in both situations (Figure 7.6b), yet they suggest very different forms of interaction. The fact that the person is seated on a couch or chair

facing the display becomes an indicator for watching the video. Yet standing at the same distance and then moving closer to the screen is used to infer that the person is increasingly interested in getting more information about the available videos in the media collection.

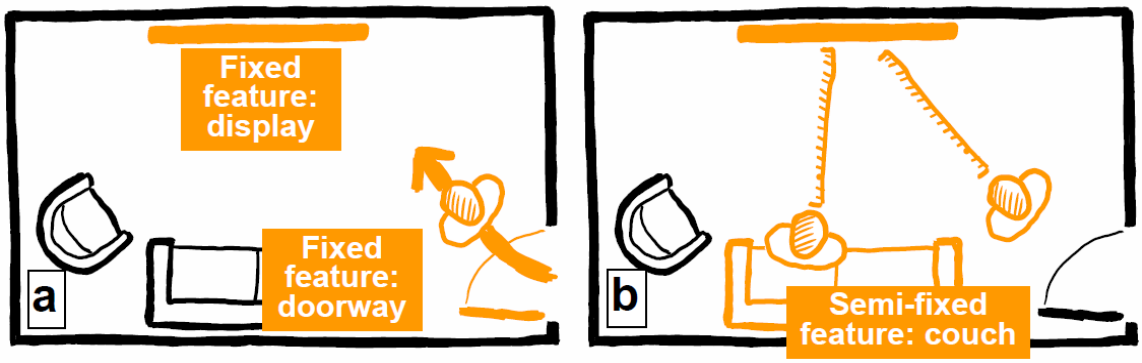


Figure 7.6 Fixed and semi-fixed features in ubicomp ecologies.

In summary, information about distance and orientation of a person *relative to* the fixed and semi-fixed feature space provides contextual cues that can mediate implicit interactions with the system. This nuanced view reflects Hall's notion that proxemics embeds measures such as distance within the context of entity relationships.

7.3 Interpreting Directed Attention to People, Objects, and Devices

Proxemic interactions can be used to extend the concept of attentive user interfaces (AUIs) that are designed to "support users' attentional capacities." (Vertegaal and Shell 2008). In AUIs, as explained earlier in Section §2.2.3, the system reaction depends on whether a person is directing his or her attention to the device that holds the system (usually through detection of eye gaze). With proxemic interactions we take this AUI concept one step further, where the system is designed to also incorporate information about: what entity a person is attending to, and the importance of not only orientation (an indication of gaze) but distance as well in that context.

Attending to the system itself occurs if the device reacts to how it is being looked at. This is how most traditional AUIs work. We include an example of this behaviour in the media player application: the system plays the video as long as at least one person faces the large display (Figure 7.7a), but pauses when that person looks away for a length of time (Figure 7.7b).

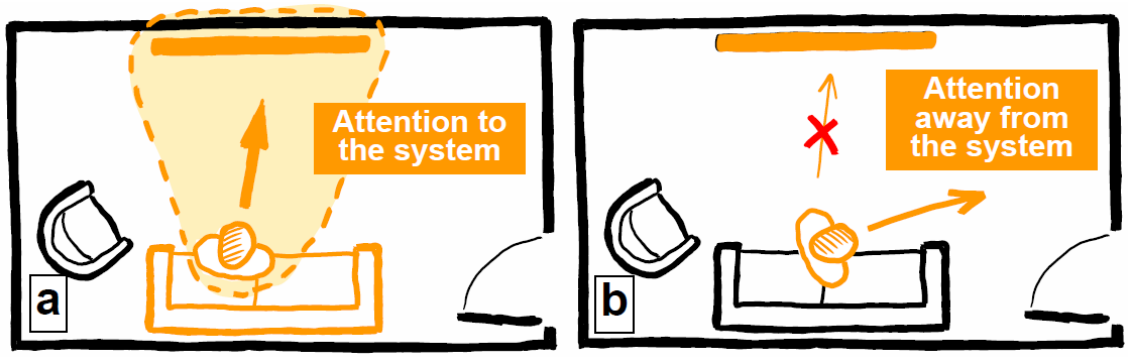


Figure 7.7 Interpreting directed attention to the system.

We can also differentiate these two situations by considering the transactional segment of a person (§3.1.8). Kendon (2010) explains how the fixed or semi-fixed features around a person can be the focus point of a transactional segment. For example, as long as the person is oriented towards the large display it becomes part of the current transactional segment (highlighted with a dashed outline in Figure 7.7a). Once the person turns their body and head into another direction away from the display, the transactional segment changes and does not include the display anymore.

Attention to other surrounding objects and devices. We enrich the concept of AUIs by including how a person's attention directed to *other* surrounding objects of the semi-fixed feature space can trigger implicit system reactions. In the media player system, the fact that a person is holding and facing towards a newspaper or magazine (shown in Figure 7.3b and Figure 7.8a) provides cues about the focus of this person's attention, i.e., the system infers that Fred is reading, and pauses video playback until Fred stops reading and looks back at the screen. The transactional segment shifts from the display towards the newspaper (transactional segment highlighted with a dashed outline in

Figure 7.8a). If Fred had a similar gaze towards, for example, a bowl of popcorn, the video would not have paused.

A shift of attention can also be suggested by the relative distance of an object to the person. For example, the media player system detects when Fred is holding his mobile phone close to his ear (as shown in Figure 7.3a). It infers that Fred is having a phone conversation, and pauses the video until Fred moves his phone away from his head. The measurement of the relative distance of phone to the person's head, as well as their orientation towards each other, provides the necessary information for the system to implicitly react to this situation.

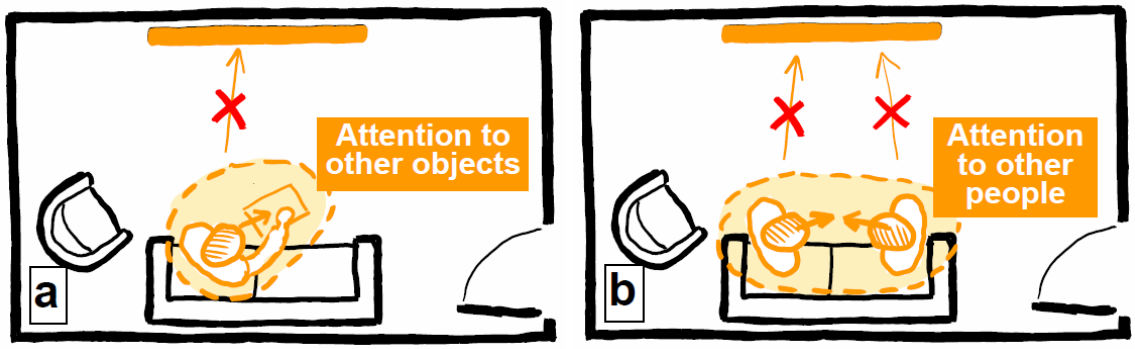


Figure 7.8 Interpreting directed attention to other objects or other people.

Attention to other people. We can discriminate how one person attends to other people as a means to trigger implicit system reactions. For example, consider Fred and George when they turn towards each other to converse (Figure 7.5 and Figure 7.8b). The scenario illustrates how the system implicitly reacts to this situation by pausing the video. The transactional segments of both people on the couch overlap in a face-to-face formation as illustrated in Figure 7.8b (highlighted as the area with a dashed outline). Alternatively, by knowing that both people are in a conversation (rather than just knowing that they are looking at each other), the system could have simply turned down its volume instead of pausing the video. Of course, these reactions beg the question of whether the system response is an appropriate one, or whether alternate reactions (or perhaps none at all) would have been more appropriate. This will be returned to later at

the end of this chapter, when we discuss issues with the 'rules of behaviour' regarding such implicit actions.

7.4 Supporting Fine-Grained Explicit Interaction

Aside from *implicitly* reacting to a person's proxemic relation to other semi-fixed environment objects, these relationships can also facilitate a person's *explicit* forms of interaction with the system. In this section we introduce the concept of using physical objects as *mobile tokens* that people can use to mediate their explicit interaction with an interactive surface. The meaning of these tokens is adjusted based upon the token's identity and its distance and orientation to other entities in the space.

To illustrate this concept, consider the explicit interaction in our scenario where Fred pointed his cell phone or a pencil at the surface to view and select content (Figure 7.9). The way this works is that all mobile tracked objects are interpreted as *mobile tokens*. Three units of information cause the media player system to interpret that token as a pointing device: it is held in *front* of a person, it is roughly *oriented* towards the display, and it is within a particular *distance* from the display. Indeed, two quite different devices can serve as similar tokens: the phone in Figure 7.2a, and the pen in Figure 7.2b. It is important to note that the system is not using any of the digital capabilities of the mobile digital phone to make this inference. Rather, (and as with the physical pen) the system uses only the knowledge of its identity, position, and orientation to switch to a certain interaction mode. Thus, the particular proxemic relationship between a person and a mobile token is interpreted as a *method of signaling* (Clark 2003), as discussed in Clark's theory of pointing and placing as forms of communication. Further, the specific orientation and distance of the token to other devices (e.g., the large display) are interpreted to establish an *intrinsic connection* (Clark 2003) to control that particular device.

A key advantage is that the use of these mobile tokens as identifiers can disambiguate similar looking gestures. For example, a gesture recognition system cannot tell if the

intent of a person pointing their hand towards the screen is to interact with the screen, or if it is just a gesture produced as part of a conversation. Mobile tokens, on the other hand, create a specific context to disambiguate and interpret gestures, where the distance and location of the objects relative to the person and other objects are used to infer a certain explicit interaction mode.

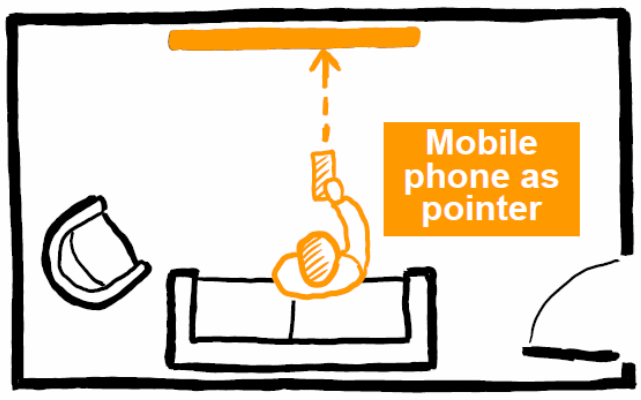


Figure 7.9 Using a mobile phone as a pointer.

Many of these behaviours can be triggered by approximate knowledge of proxemic relationships. Yet having exact knowledge is helpful for minimizing errors that can occur where the system misinterprets a person's manipulation of a mobile token as an explicit action. For example, consider a person playing with a pen in their hand vs. pointing the pen at the screen to select an item. If proxemic measures are reasonably precise, the triggering event could rely solely on the pen being a specific distance from the person's body and a specific orientation towards the screen for a particular length of time.

Another example includes the multiple meanings held by a mobile token. Consider how the meaning of the mobile phone depends on its proxemic relation to its holder and to the display. The distance of the phone to a person's head indicates an ongoing phone conversation (Figure 7.10a), while stretching the arm holding the phone at a larger distance (relative to the person's body) and moving it towards the display shifts its meaning to an interaction pointer (Figure 7.10b). Holding the phone at a distance between these two extremes (Figure 7.10c) lets the person interact with the touch screen of the device.

In our scenario, the actual explicit interaction with the digital video content displayed on the large surface is triggered by the person through several means. Moving the position of the mobile token across the screen highlights potential media item selections. Changes of the orientation angle allow fine-grained positioning of a pointer icon on the screen, while fast downwards acceleration is used for selection. Of course, other gestures are possible.

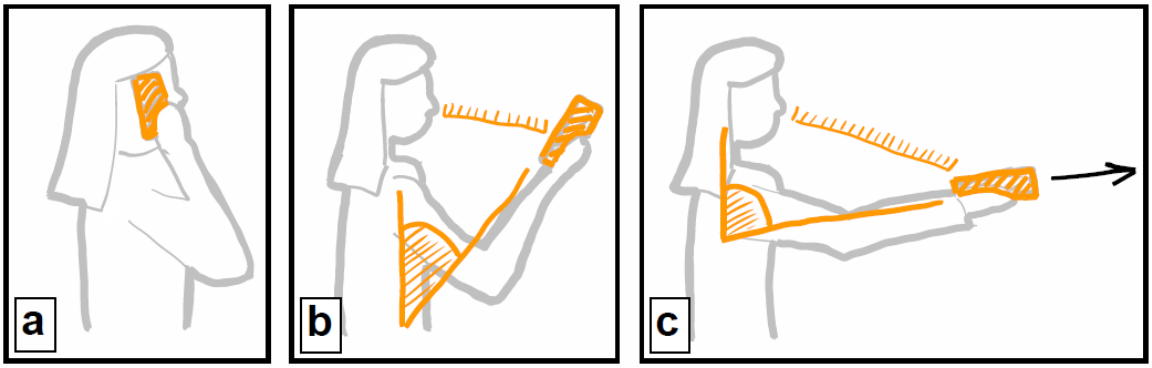


Figure 7.10 Using proxemic relationships between a person and objects (e.g., mobile phone) to infer interaction modes and disambiguate gestures.

7.5 Interpreting Continuous Movements or Discrete Proxemic Zones

Another concept is that the behaviours of proxemic interfaces can react to the position and distance of the surrounding entities as either *continuous movements*, or as movements in and out of *discrete proxemic zones*.

For *continuous movement*, the calculated distances between people and devices function as input variables that continually affect the interactive system's behaviour (Figure 7.11a). For example, as a person approaches the screen of the media player application, the number of visible video preview thumbnails shown continually increases with distance (see Figure 7.1a,b). To do this, the system gradually resizes the preview images to a smaller size (i.e. an animated zoom out effect); thus more content is visible as the person approaches the screen. Depending on the situation, an inverse behaviour might be

applied, where the system actually *zooms into* the content to make it larger when the person is approaching the screen (similar to *Lean and Zoom* by Harrison and Dey, 2008).

With *discrete proxemic zones* we can divide the space into discrete regions (Figure 7.11b). When a person enters or leaves the thresholds of these zones, certain actions are triggered in the system. Indeed, the use of zones is inspired by the inter-personal proxemic distance zones defined by Hall (Hall 1966), and others have applied zones as a way to mediate interaction with public ambient displays (Vogel and Balakrishnan 2004) and digital whiteboards (Ju et al. 2008).

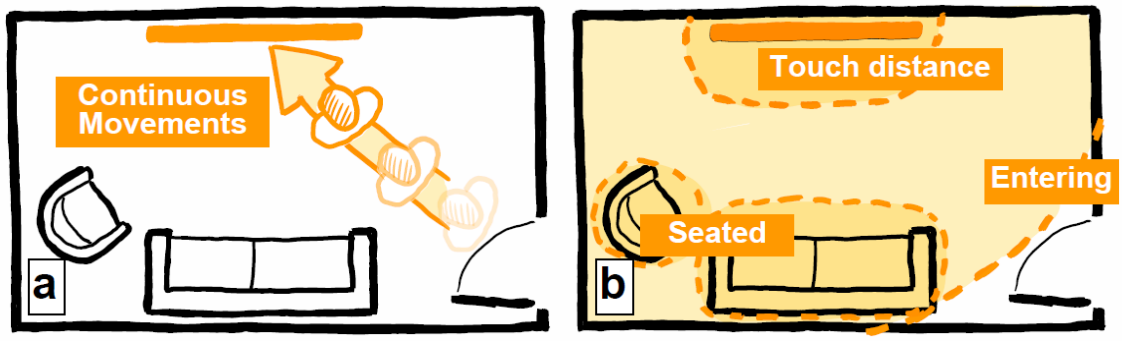


Figure 7.11 Interpreting (a) continuous movements or presence in (b) discrete zones around the large interactive display.

The media player uses discrete zones in several ways (Figure 7.11b). It uses zones to trigger an associated implicit action (e.g., activate a display screen when entering the room in zone "Entering" of Figure 7.11b). The media player also use zones to allow certain forms of explicit interaction (e.g., switching to an interface that allows direct touch interaction when the person is standing in close proximity to the screen in zone "Touch distance" of Figure 7.11b). Finally, the media player also uses zones around semi-fixed features as explained earlier in §7.2 (e.g., switching to full screen view of the video when sitting on the couch or a chair in zone "Seated" of Figure 7.11b).

A problem associated with discrete zones occurs when the interface rapidly switches back and forth between two states; this occurs when the person stands exactly along a border of one of the discrete zones. This jitter is solved via the concept of a *hysteresis tolerance*: the entry and exit point of each region are not at the same distance, but are two

separate distances. For example, our system uses a 15-20% hysteresis tolerance for proxemic regions around the interactive wall display (percentage of the region dimension) to avoid this rapid switching.

7.6 The Gradual Engagement Pattern

The vast majority of ubicomp interfaces are premised on the notion that a person is fully attending to them, i.e., the systems are designed to support *foreground* activities and tasks. However, a variety of systems also recognize that the person may not be directly attending to them (i.e., they are in the *background* of their attention), where they still try to be helpful by presenting an interface that selectively informs the user of information of interest.

One class of examples includes ambient displays (Mankoff et al. 2003) embedded in a physical environment. The display usually presents non-critical information unobtrusively, which a person can monitor at a distance and at the periphery of their attention (thus providing basic awareness). The display often contains a way for the person to easily transition to more in-depth information exploration if the person decides to engage with it; this normally occurs by the person approaching and directly interacting with that display. That is, such displays implicitly incorporate a binary notion of proximity: from afar, and within interaction reach.

A few early proxemic interaction systems provide another class of examples that use a much more refined notion of proxemics. Many of these systems commonly interpret decreasing distance and increasing mutual orientation between a person and a device within a bounded space as an indication of a person's gradually increasing interest in interacting with that device. Influential earlier work (discussed in Section §2.2.4) considered such gradual increasing engagement between a person and large interactive displays (Ju et al. 2008; Vogel and Balakrishnan 2004). For example, Vogel et al. directly applied Hall's theory to a person's interaction with a public display (Vogel and Balakrishnan 2004). They defined four discrete zones around the display that affect a person's interaction when moving closer: from far to close, interactions range from

ambient display of information, to implicit, subtle, and finally personal interaction. Similarly, Ju's interaction techniques with a digital whiteboard remain public and implicit from a distance, and become increasingly more private and explicit when the person moves closer to that display.

We generalize the sequence inherent in these (and other) systems as a design pattern called *gradual engagement*[33]. The basic idea is that:

- **STAGE 1:** background information supplied by the system provides awareness to the person about opportunities of potential interest when viewed at a distance;
- **STAGE 2:** the person can gradually act on particular opportunities by viewing and/or exploring its information in more detail simply by approaching it; and
- **STAGE 3:** the person can ultimately engage in action if so desired.

This pattern is, of course, directly inspired by proxemic theory (§3.1) and by the systems that reflect that pattern (§2.2). The pattern also characterizes what we thought was the 'best' of how proxemics was previously applied to ubicomp design. The intention of this gradual engagement pattern is to characterize how we can facilitate interactions between a person or multiple people and the devices surrounding them by leveraging fine-grained proxemic measurements (e.g., distance, orientation, identity) between all entities. As a design pattern (Borchers 2001; Tidwell 2005), its strengths lie in (1) unifying prior work in proxemic interaction, (2) synthesizing essential, generalizable interaction strategies, and (3) providing a common vocabulary for discussing design solutions. Most importantly, the pattern informs and inspires future designs, and also allows for variations of the pattern applied to different domains. In this chapter we focus on the application of the gradual engagement pattern to person-to-device interactions, but as we will see later in Chapter 8, the pattern can be also applied to other relationships (e.g., device-to-device).

[33] Jan Borchers describes an even more general pattern titled 'Attract-Engage-Deliver' (Borchers 2001). The difference is that our pattern incorporates proxemics as a first-class element.

7.7 Applying the Gradual Engagement Pattern: From Awareness to Interaction

In our work, we combine both continuous movements and discrete proxemic zones to design system interfaces that move fluently from awareness, to progressive reveal, to direct explicit interaction following the gradual engagement pattern. One example illustrates this combination.

The media player begins by providing peripheral awareness information about its capabilities and content when a person enters the room. The system detects the presence of the person at a distance (around 4m), activates the display, displays a welcome animation, and plays a subtle acoustic signal. This indicates to the person that the system is active. Here, the system uses a discrete proxemic zone around the digital display that triggers this activation behaviour. At this point, if the person just walks past the display, or does not face the display, the media player application reverts to sleep mode. If, however, the person does move closer, the system shows preview images of video content, where it progressively reveals more preview items on the screen as the person approaches the screen. Here, the system uses the continuous mapping of distance to the size and quantity of preview items shown. When the person stands within reach of the screen, we enter another discrete zone: direct touch interaction. At that distance, the person can use their hands for direct touch interaction with the screen content; thus the continuous resizing of the displayed preview thumbnails stops as it would otherwise make selection difficult. We will return to other examples where we apply the gradual engagement pattern in Chapter 8.

7.8 Leveraging People's Identity

The concepts introduced so far only require knowledge about "a person" approaching the display, but they do not require the actual identity of a person. We now discuss examples that leverage the knowledge about the actual identity of individuals.

History. Knowing which person is interacting with the system is used to continue activities that this person began in the past. For instance, when a person enters the room and immediately sits down, the media application will resume playback of the last video that a person previously watched but did not finish.

Personalization. The media player could save one's settings as a personal profile (not implemented in the current version of the proxemic media player). This could include personal configurations, idiosyncrasies of how the system should respond to that particular person, and that person's media content. For example, if a particular person were to approach the display, the media player would then display content out of that person's media library. This also raises the question of where such information is stored. While it could be on the server, privacy concerns suggest that it could also be placed on a person's mobile device. For example, our media player would only have access to the personal identity information of only those people who come in close proximity to it, where that accessed information is erased as soon as that person leaves.

Safeguards. Identifying the person interacting with the system can also function as a safeguard to restrict access (not yet implemented). For instance, children may only be allowed to access the media player application during pre-defined time slots, or access to available media content could be restricted to movies rated for their age. To take this one step further, children's viewing could be mediated by requiring the presence of others (e.g., adults or parents) in the room for either accessing particular content or for going beyond previously specified restrictions (such as the amount of time they are allowed to watch movies).

7.9 Mediating People's Simultaneous Interaction

Proxemic interactions should also mediate the interaction of multiple people in the same space. In the simplest case, as long as all people are in the same proxemic state relative to the display's surface, the system's behaviour could be similar to the proxemic interactions introduced for a single person interacting with the surface. In reality, however, we expect people to be in different proxemic stages, where the system would

need to reason about how it should mediate its behaviour to reflect people's simultaneous interaction possibilities.

7.9.1 Merging multiple proxemic distances

In situations where people have different proxemic distances to the interactive display of the media player application, the system can be designed to individually address people's diverse proxemic needs, albeit as a compromise.

For example, in the example scenario we saw George enter the room while Fred was watching a video. George wants to know what is currently playing, while Fred wants to keep watching. To compromise between these needs, the system displays the title of the currently playing video at the top of the screen, thus subtly informing George while still letting Fred watch without too much distraction (Figure 7.4a and Figure 7.12a). If George sits at the couch or on a chair, the title disappears.

If George approaches the screen instead of sitting down, the display animates and splits off a small region of the screen. This region provides further information of the video being played: its description, author information, and the release date (Figure 7.4b and Figure 7.12b). The positioning of this region also depends on George's spatial relation to the display – if he moves from the right side to the left, the information panel smoothly animates to that side of the display (Figure 7.12c).

Such a system could be annoying if, for example, that information was displayed whenever a person just happened to be approaching the screen from anywhere. Instead, our system only shows this information when a person is approaching from the fixed feature of the doorway, and only when there is another person seated.

When both people are in the same proxemic state, the views merge. For instance, both people can watch the video in full screen when seated, or both can explore and choose from the videos available when standing in front of the display.

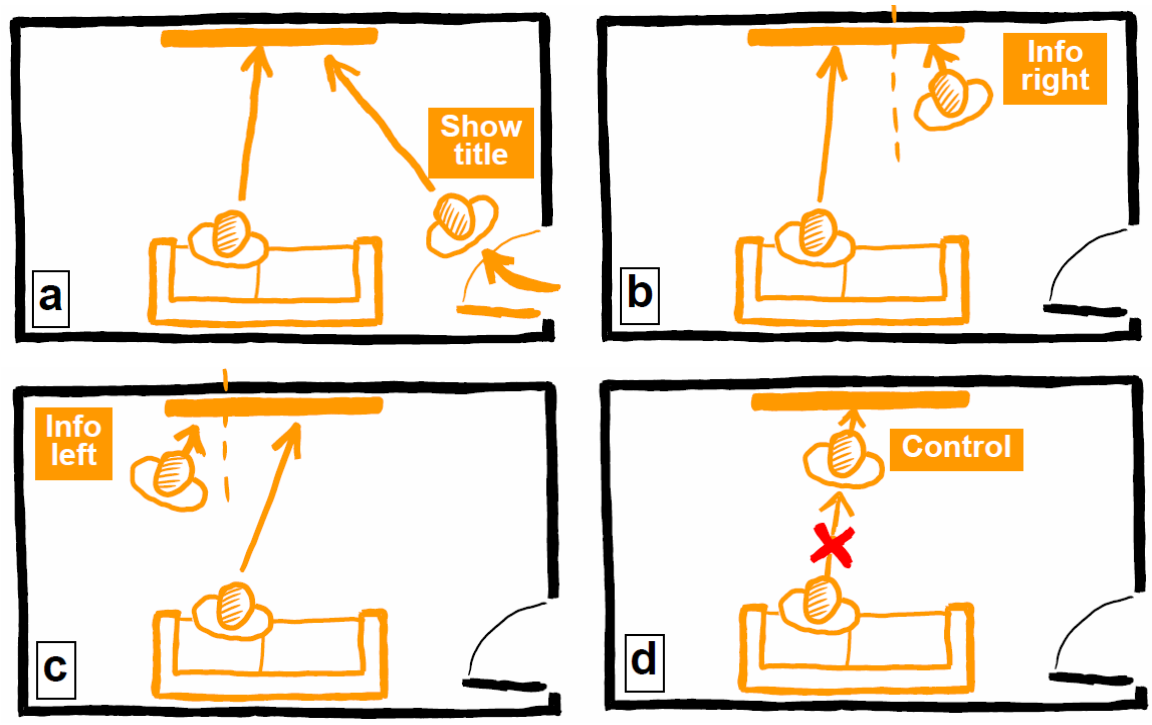


Figure 7.12 Merging multiple proxemic distances to mediate people's simultaneous interaction with the display.

7.9.2 Handling conflicts

When multiple people are present within a proximity-aware application, situations will arise where the system has to handle two *conflicting* individual possibilities. For example, consider the scenario situation of Figure 7.4c (and Figure 7.12d): Fred is sitting in front of the large display watching a movie, while George moves directly in front of the display to browse a media collection.

Several strategies are possible to handle these situations. The system could favour the person in closer proximity; e.g., George standing directly in front of the display would have priority over Fred sitting at a larger distance. This is the solution shown in Figure 7.4c and Figure 7.12d, where George gets full access to the media library to select videos; a strategy that makes sense as Fred's view is already blocked. Alternately, the system could have given the video player priority, disallowing George's interaction, leaving the two to resolve this through social means (e.g., both standing up to make a selection). Or the system could create some kind of composite view, i.e., by moving the video so that

Fred could still see some of it, while still giving George interactive controls in the blocked part of the screen. There is no perfect solution. These (and other possibilities) need to be considered carefully in the design of such systems.

To support the simultaneous interaction of multiple people in our scenario, the media player further refines the discrete zones around the large display that trigger actions once a person enters these zones (Figure 7.13). For example, the "Touch distance" zone shown in Figure 7.11b is now divided for multi-person interaction into three discrete zones (Figure 7.13 top): displaying the title when the second person enters the room, the video information when that person stands either on the left or right side of the screen, and the touch controls when the person stands directly in front of the display.

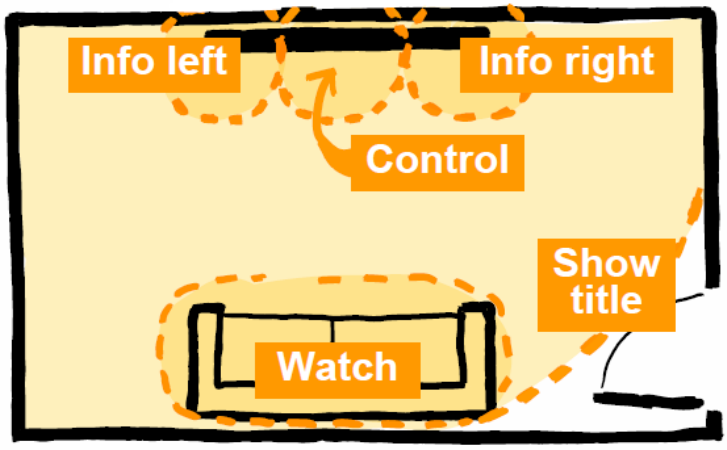


Figure 7.13 Refined discrete zones around the large display to support simultaneous interaction of multiple people.

7.10 Discussion and Conclusion

In this chapter, we showed how proxemic interactions enable a multitude of implicit and explicit interactions with an interactive vertical display. We explained these interaction techniques through a scenario of people interacting with a media player application displayed on that interactive display. In particular, we explained how knowing the continuous movement of an approaching identified person along with the position, orientation, and usage of identified objects can be exploited in interface design, e.g., how

the system should implicitly respond to proxemic entities and how the system can afford opportunities for explicit interactions.

The presented implementation of the proxemic-aware media player – while fully functional – serves primarily as an example that illustrates design possibilities for proxemic interactions between one person or multiple people and their surrounding devices. We do not suggest that the rules of behaviour of the media player application are ideal, nor that they achieve the perfect balance between adjudicating proxemic information and implicit or explicit interaction. Some of the presented media player rules are clearly problematic, where the action taken may not be appropriate in certain social situations. As a simple example, the instance where the system pauses when two people turn to one another may be opposite of what those people want (e.g., when they are perhaps talking excitedly about the movie's climax). Problems such as these are likely endemic to systems that try to infer people's intentions. Instead, we see the presented techniques as a set of novel ways to consider proxemics in interaction design that can inspire future explorations of proxemic-aware technology; designing and debugging appropriate interactions based upon implicit acts, or developing design patterns such as the gradual engagement pattern comes under this exploration.

While all of the presented techniques focused on the interaction of a person (or two people) with a *single* device, we simultaneously began the exploration of how to consider proxemics to mediate device-to-device relationships. This exploration led to a series of techniques for facilitating the use of multiple devices in concert, and is the focus of the following chapter.

Chapter 8. Device-to-Device Proxemic Interactions

The increasing number of digital devices in our environment, ranging from personal mobile phones to semi-public large stationary surfaces, enriches how we interact with digital content. Yet, cross-device information transfer – which should be a common operation – is surprisingly difficult. One has to know which devices can communicate, what information they contain, and how information can be exchanged.

To mitigate this problem, in this chapter[34] we focus on applying concepts of proxemic interactions to mediate device-to-device operations – both between personal (e.g., tablets) and semi-public devices (e.g., digital whiteboards). In particular, we refine the gradual engagement pattern from Section §7.6 to ease the information transfer task. As we will see, the refined pattern suggests how devices can gradually engage the user by disclosing connectivity and information exchange capabilities as a function of inter-device proximity. That is, as people move and orient their personal device towards other surrounding devices, the interface progressively moves through three stages affording gradual engagement: (a) *awareness* of device presence and connectivity, (b) *reveal* of exchangeable digital content, and (c) interaction methods for *transferring* digital content

[34] Portions of this chapter are published as:

Marquardt, N., Ballendat, T., Boring, S., Greenberg, S. and Hinckley, K. (2012) Gradual Engagement between Digital Devices as a Function of Proximity: From Awareness to Progressive Reveal to Information Transfer. In *Proceedings of the ACM Conference on Interactive Tabletops and Surfaces – ACM ITS 2012.* (Boston, MA), ACM, pages 31-40.

As described in the Technical Acknowledgements section, the development of the cross-device interaction techniques was done in collaboration with Till Ballendat.

between devices tuned to particular distances and device capabilities (these three stages correspond to the three parts of Figure 8.1 – we discuss the details later in this chapter).

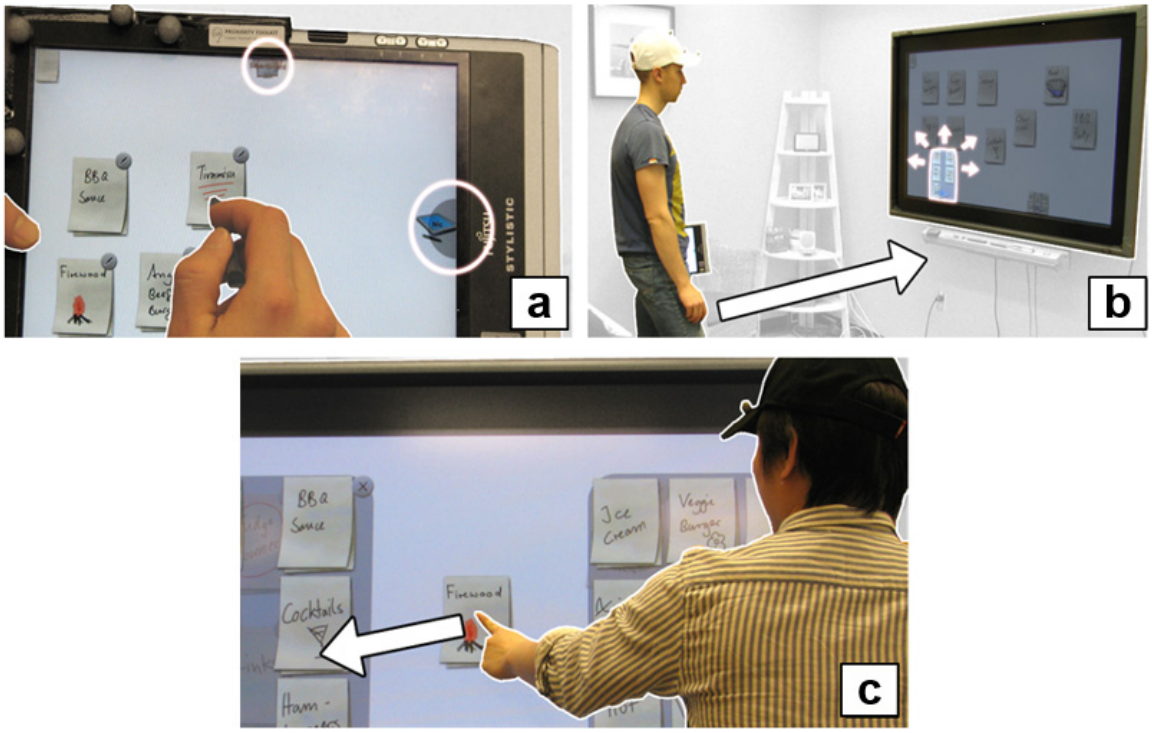


Figure 8.1 Gradual engagement pattern applied to mitigate cross-device operations: showing examples of (a) awareness, (b) progressive reveal, which (c) leads to information transfer.

The chapter is structured as follows. First, we describe how to refine the gradual engagement pattern to address cross-device interactions (§8.1). Next, we briefly review related work that relates to cross-device transfer and the gradual engagement pattern (§8.2). The next three sections then follow the three stages of the refined design pattern: *awareness* of device presence and connectivity (§8.3 and Figure 8.1a), *reveal* of exchangeable digital content (§8.5 and Figure 8.1b), and interaction methods for *transferring* digital content between devices tuned to particular distances and device capabilities (§8.6 and Figure 8.1c). In each of these sections we describe particular interaction techniques, where their presentation is structured according to the three stages of the gradual engagement design pattern. We also explain how gradual engagement techniques may differ when the other device seen is personal (such as a

handheld) vs. semi-public (such as a large display). Then, in Section §8.7 we briefly describe the implementation, and discuss sensing requirements and privacy aspects in Section §8.7. A video figure[35] illustrates the applications and the dynamics of the various methods described in this chapter.

8.1 Applying Gradual Engagement to Cross-Device Information Transfer

Personal mobile devices (e.g., phones, tablets) and semi-public stationary devices (e.g., information appliances, interactive surfaces) are an increasingly commonplace way for people to ubiquitously interact with digital information (see examples in §2.1 and §2.2). Most of these devices are optimized for a seamless user experience when one uses them *individually*. Yet, using multiple devices in *concert* (such as for transferring information from a mobile phone to the device of a nearby person) is often tedious and requires executing complicated interaction sequences. This is why several projects in the area of ubiquitous computing (reviewed in §2.2.1 and §2.2.2) began introducing new techniques to facilitate transfer of content between nearby devices, e.g., (Hardy and Rukzio 2008; Hinckley 2003; Rekimoto 1997). However, significant challenges remain. People do not know *which* devices can communicate with one another, *what* information they contain that is exchangeable, and *how* information can be exchanged in a controlled manner.

To mitigate these problems, we began exploring how to apply proxemic interaction concepts and – in particular – the gradual engagement pattern (§7.6) to inter-device operations. The previous application of the gradual engagement pattern (§7.7) primarily focused on people's interaction with large displays, where the display's content changes as a function of proxemic variables (§3.2). We decided to refine the *gradual engagement* design pattern by considering fine-grained proxemic relationships *between multiple devices*

[35] Video available for download at: http://grouplab.cpsc.ucalgary.ca/grouplab/uploads/Publications/Publications/2012-GradualEngagement.ITS.mp4

allowing seamless transitions from awareness to information transfer. Specifically, engagement increases continuously across *three stages* as people move and orient their personal device towards other surrounding devices (Figure 8.2):

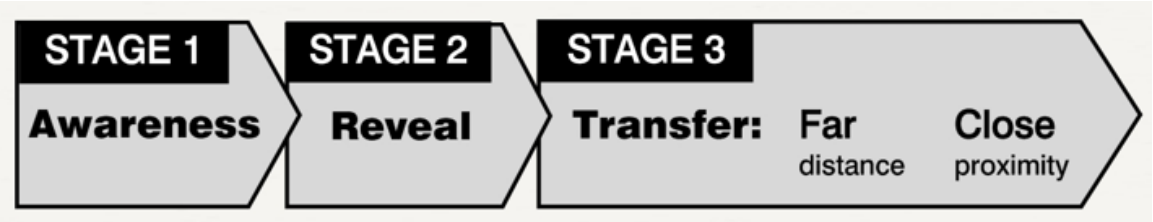


Figure 8.2 Three sequential stages of the refined gradual engagement pattern applied to cross-device operations.

Stage 1. Awareness of device presence and connectivity is provided, so that a person can understand what other devices are present and whether they can connect with one's own personal device. We leverage knowledge about proxemic relationships between devices to determine when devices connect and how they notify a person about their presence and established connections.

Stage 2. Reveal of exchangeable content is provided, so that people know what of their content can be accessed on other devices for information transfer. At this stage, a fundamental technique is progressively revealing a device's available digital content as a function of proximity.

Stage 3. Transferring digital content between devices, tuned to particular proxemic relationships and device capabilities, is provided via various strategies. Each is tailored to fit naturally within particular situations and contexts: from a distance *vs.* from close proximity; and transfer to a personal device *vs.* a semi-public device.

We illustrate the application of this gradual engagement pattern between devices as a suite of interaction techniques, all based on providing a seamless transition leading from awareness, to reveal, to interaction. We use three example applications throughout this chapter to illustrate how these various techniques leverage proxemic interaction and follow gradual engagement to facilitate access to digital information:

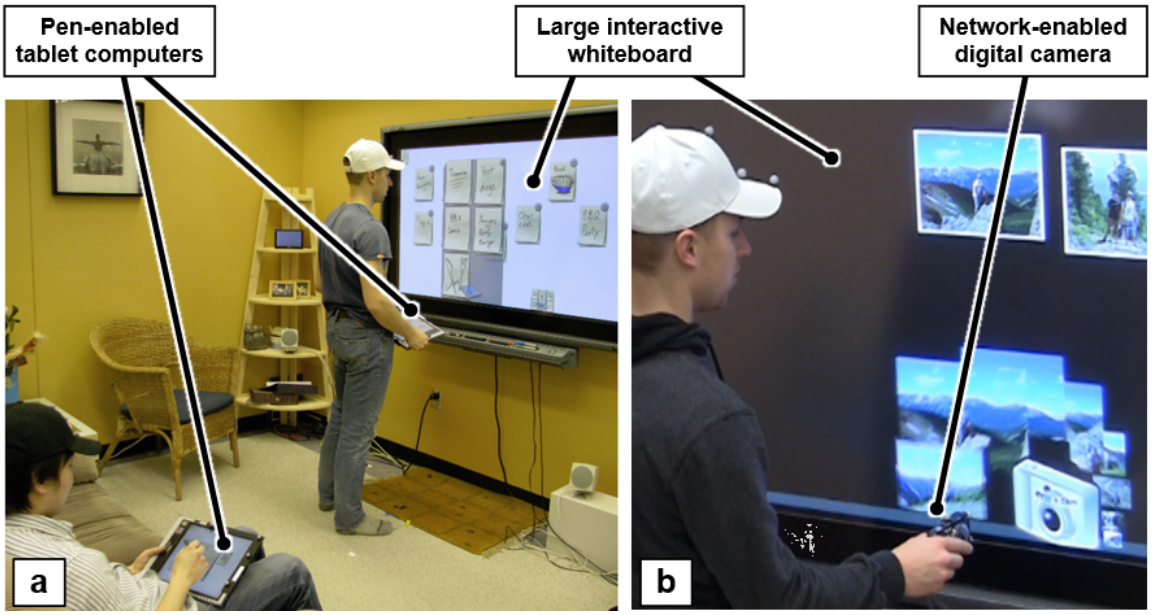


Figure 8.3 Example applications illustrating interaction concepts: (a) proxemic brainstorming, and (b) proxemic photo canvas.

- First, *proxemic brainstorming* is an interactive digital brainstorming groupware tool (Figure 8.3a). Its users can individually create, change, and manage virtual sticky notes on their personal pen-enabled tablets (e.g., see Figure 8.1a). A large whiteboard provides a public sharing space for interacting over notes, and different techniques (explained shortly) allow temporary or permanent transfer of the digital notes between all devices and thus all people in the group.

- Second, *proxemic photo canvas* (Figure 8.3b) facilitates transfer of digital photos from a network-enabled digital camera to other devices, such as a large display or a digital photo frame.

- Third, we refer back to the *proxemic media player* application from Chapter 7 to illustrate interaction concepts. We extended the functionality of the media player so that it recognizes nearby portable digital media players a person is using and also supports the transfer of digital media between devices.

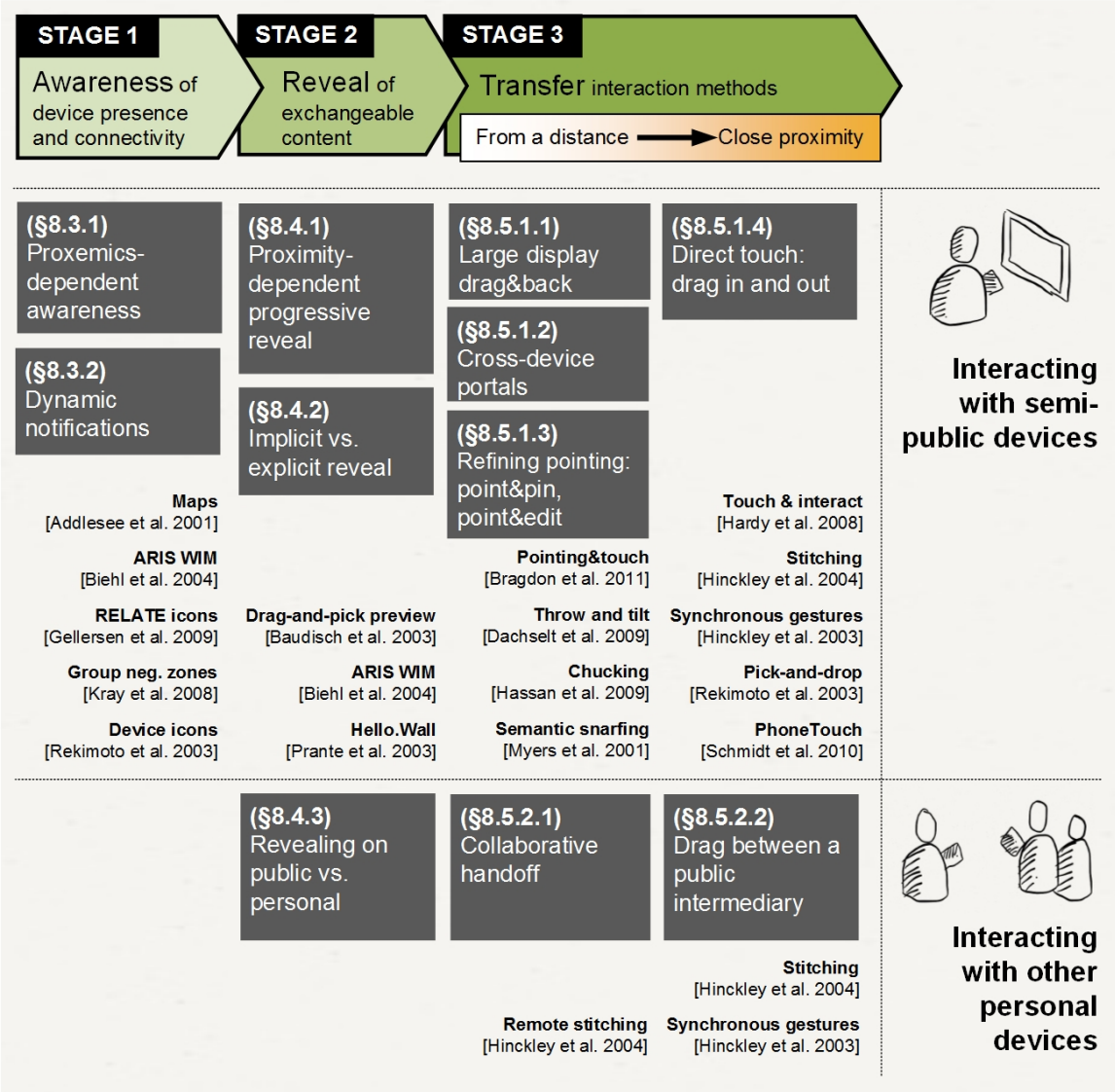


Figure 8.4 The stages of the refined gradual engagement pattern (top row) and interaction methods derived from the pattern supporting awareness and interaction in each stage: person interacting with semi-public devices (middle row) or personal devices of other people (bottom row).

In the main part of this chapter we will revisit each of the three stages of the refined gradual engagement pattern to introduce these techniques, where we use scenarios from each of these three applications to illustrate the techniques in action. Figure 8.4 provides an overview of all interaction techniques across the three stages of the design pattern. We list our novel or refined interaction techniques (grey boxes in Figure 8.4) and refer to techniques of related work that fit into each of the three stages. Each section explain

details of each particular technique. First, however, we review prior work that relates to our derivation of the gradual engagement pattern.

8.2 Prior Work Applied to Gradual Engagement

We briefly sample prior work that contributed to our derivation of particular stages of the gradual engagement pattern. In our review we also refer to ubicomp systems discussed earlier in Section §2.2, but primarily focus on projects providing awareness, reveal content, or facilitating cross-device operations. Beyond the review, we also later explain how many of these prior techniques can be applied to people's interactions across all three stages of gradual engagement. Figure 8.4 summarizes how these works fit within and thus contribute to the pattern.

8.2.1 Awareness of Device Presence and Connectivity

Most systems define a discrete spatial region around devices, where a connection is established (and information transfer possible) once the distance becomes smaller than a certain threshold. Often, this distance depends on the actual sensing technology used (e.g., sensing range of RFID or Bluetooth).

Visualization of available devices becomes important in ubicomp environments, as an increasing number of diverse devices are present. Their presence, location, and ability to connect (or not) are rarely easily visible to a user. A few systems began exploring methods to inform a person about surrounding devices and possible connections. Most commonly, a map visualizes devices located in the environment (e.g., Sentient Computing by Addlesee et al., 2001) or the same room (e.g., *ARIS by* Biehl and Bailey, 2004). Gellersen et al.'s *RELATE* Gateways provide a similar visualization, but make use of sophisticated tracking systems to dynamically update the positions of all devices (Gellersen et al. 2009). In an alternative view, icons at the border of a mobile device screen represent the type and location of surrounding devices (Gellersen et al. 2009; Rekimoto et al. 2003). Kray's *group coordination negotiation* introduced spatial regions for interaction around mobile phones (Kray et al. 2008). Their scenario used these regions to

negotiate exchange of information with others. Feedback about a phone's presence in any of the regions was visualized on a tabletop.

Explicit Connections. Various systems allow people to manually associate two devices from a distance. This is usually done via pointing one device at the other. Swindells et al. (2002) uses an infrared-emitting pen to point at a device to control it. *Semantic snarfing* (Myers et al. 2001) also uses pointing to allow someone to take over temporary control of remote interfaces. Similarly, others have suggested ways to manually associate nearby devices that are all within reach. With Smart-its friends (Holmquist et al. 2001), a connection is established when two devices are shaken simultaneously and sense similar accelerometer values. In *Synchronous Gestures* people can bump devices (Hinckley 2003) – including phones and interactive tabletops (Schmidt et al. 2010) – together to initiate a connection. In *Stitching*, users couple devices by drawing a stroke that begins on one display and ends on another (Hinckley et al. 2004). Overall, Chong et al. confirmed that proximity is one of the 'big five' categories of how users associate devices (Chong and Gellersen 2011).

8.2.2 Revealing Exchangeable Content.

Several systems visualized exchangeable content. *Hello.Wall* introduced the notion of 'distance-dependent semantics', where the distance (here: close, far, out of range) of a person's device from the wall screen defined the kind of information shown on the mobile display (Prante et al. 2003). The aforementioned *ARIS* shows applications running on devices located in the same room in a world-of-miniature fashion (Biehl and Bailey 2004). In *Drag-and-Pick*, content that is located in the direction of an initial drag operation appears close to the point of interaction – even on other devices in that direction (Baudisch et al. 2003).

8.2.3 Transferring Digital Content

Once connected, diverse techniques allow information transfer. For example, Want's RFID-based technique allows detecting nearby objects and devices and associating/retrieving digital information (Want et al. 1999). In *Pick-and-Drop*, users pick

up content on one display and place it on another with a digital pen (Rekimoto 1997). *Touch & Interact* temporarily shift the interaction focus and content from a large display onto a mobile device (Hardy and Rukzio 2008). Rekimoto combined near-field RFID and remote infrared communication for seamless information transfer (Rekimoto et al. 2003). Further examples for cross-device information transfer from a distance are: throwing gestures performed with a phone (Dachselt and Buchholz 2009), touch and pointing gesture combinations (Bragdon et al. 2011), chucking motions towards the other device (Hassan et al. 2009), or *corresponding gestures* through cursor selections in multi-screen environments (Nacenta et al. 2005).

In summary, various techniques exist – most suited for particular discrete distances between devices – that fit into particular stages (but rarely all stages) of the gradual engagement pattern (see Figure 8.4). In the next three sections of this chapter, we use our design pattern to build on these earlier works. In particular, we illustrate interaction techniques that allow a person to move seamlessly from awareness at a larger distance, to gradually revealing more detail about devices and content when approaching, to direct interaction for transferring digital information between devices when standing in either close proximity or at a distance. By extending earlier work, we also consider how particular device types can influence this interaction, e.g., personal handhelds *vs.* semi-public stationary devices.

8.3 Stage 1. Awareness of Device Presence and Connectivity

While ubicomp ecologies may contain many devices, only some of them – for a variety of reasons – are likely able to connect with a user's personal device to the point that the person can do useful work between them (such as transferring content). While these devices may sense this information (e.g., via service discovery protocols), the user is often left in the dark about these opportunities for inter-device interaction.

Consequently, we implemented methods that make a person aware of whether his personal device and other nearby devices can detect each other's presence and if they are

able to connect in a meaningful way. Building upon earlier work (Addlesee et al. 2001; Biehl and Bailey 2004; Gellersen et al. 2009; Rekimoto et al. 2003), the basic idea is that a person sees a visual indicator – a subtle notification – that shows which devices in the surrounding environment can possibly interact with his handheld device as well as their relative location (e.g., icons in Figure 8.7). People can then subsequently move toward a particular device to either establish that connection or to reveal further information and interaction possibilities (which would occur in stages 2 & 3, discussed shortly). This is particularly important in dynamically changing or unfamiliar environments: some devices may be hidden or disguised as a non-digital device (e.g., a digital picture frame appliance), or only some of the surrounding devices may allow meaningful connections to them (e.g., while two devices may see each other over the network, they may not support the same core application of interest to the user, so there is little point in revealing their connectivity). Information about these possible connections as well as simple ways to actually establish the connection is crucial if seamless interaction across devices is to occur.

8.3.1 Proxemics-dependent Awareness

We use rules to determine when to trigger awareness of device presence and connectivity. By connection, we mean whether one device should connect to another device based on human dynamics *vs.* whether a device is technically capable of connecting to another. We exploit the five proxemic dimensions (§3.2) as sensed factors: combinations of them allow us to create nuanced rules of connection behaviour.

Location informs devices if they (and the people using them) are in the same room. In almost all cases, devices present in the same room are far more relevant for interaction than the ones nearby but in another room. For example, when a person with a tablet enters a new room through the door, notifications can be triggered about other devices available in that particular room. Other devices in close proximity but in adjacent rooms – such as behind walls – are not shown. Figure 8.5a illustrates this situation, where two people and their tablets are in similar distance to the large display, but only the two devices that are in the same room connect to each other. In proxemics terms, doorways,

walls and other boundaries are fixed features that further demark people's sense of social distance (§3.1.3); we believe such fixed features are applicable to how devices determine possible candidates for cross-device connections. Location also informs context; some locations (e.g., public vs. home spaces) would afford quite different connectivity semantics.

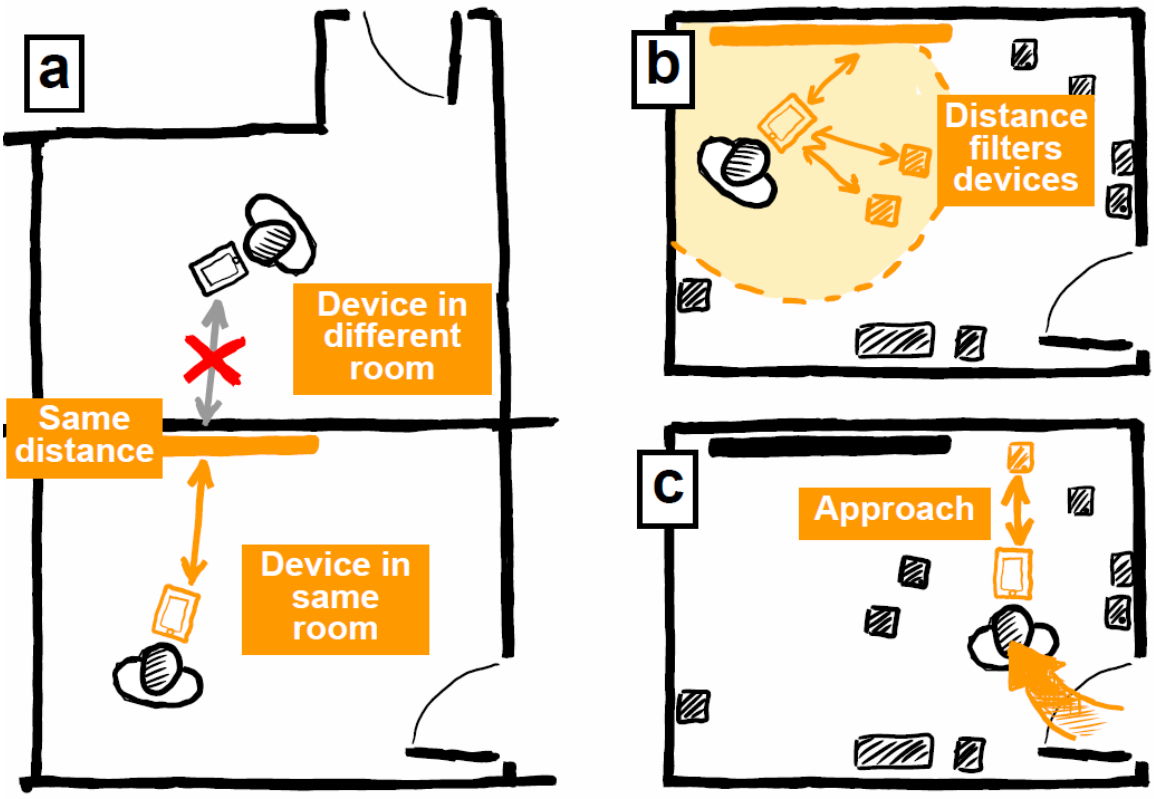


Figure 8.5 Proxemics-dependent awareness considering device's (a) location, (b) distance, and (c) movements.

Physical distance between devices is an essential factor we exploit for determining device connection and triggering notifications. Proxemic theory states that people naturally stand close to other people they are interested in and want to communicate with. Similarly, we believe that the distance between the user's personal device and other devices in the ecology is a natural indicator of whether a connection between the two should be signaled to the user and subsequently established. Distance measurements can also be applied as a filter that prevents too many notifications in environments with a large number of digital devices. In that case, awareness information is only shown of a

limited number of devices that have the smallest distance (e.g., the five closest devices). For example, in Figure 8.5b only the devices below a certain threshold connect to the tablet device a person is holding – all other devices in that room are ignored.

Movement – the change of distance over time – is an indicator of increasing or decreasing interest. When we are interested in something we move closer to it, while we move away when we are less interested. We can apply this to device-device connectivity. For example, if a person holding a tablet is approaching another device, we can interpret this as increasing interest of that person to connect and ultimately interact with both devices in tandem (Figure 8.5c).

Orientation of one device towards another is an additional indicator that the person wants to connect the two. This again mimics interpersonal interaction: when people interact, they orient themselves to either face the other person or stand side by side. Orientation between devices could simply be whether one device is facing towards or away from another device, or a finer measure that considers and acts upon the angle between the two (at the extreme, this becomes pointing (Myers et al. 2001; Swindells et al. 2002)). For determining cross-device connections, we focus on all devices that are either located in front or at the sides of the device (Figure 8.6a). We assume that if a person wants to interact with a device located behind them, they turn around to face this device, and if they are uninterested, they face away. For example, the visual feedback shown in Figure 8.1 a and b would appear or fade away as the person turns towards or faces away from the display.

Identity of devices functions as a filter for possible connections. Known devices can trigger the connection notification from a larger distance, while unknown devices need to be located next to each other to establish a successful (and more socially secure) connection. This technique follows the principle that "distance implies distrust" (Fishkin et al. 2005), and similarly that closer proximity between devices implies trust (although this depends on location context). Identity also distinguishes classes of devices, where (for example) connectivity to another person's personal device may be dealt with differently than to a semi-public device, as each suggests different social

expectations. For example, a person's tablet computer shown on the left side of Figure 8.6b only connects to the semi-public large display (shown at the top of Figure 8.6b) but not the second person's personal tablet computer (shown at the bottom of Figure 8.6b) – even though both devices are at the same distance.

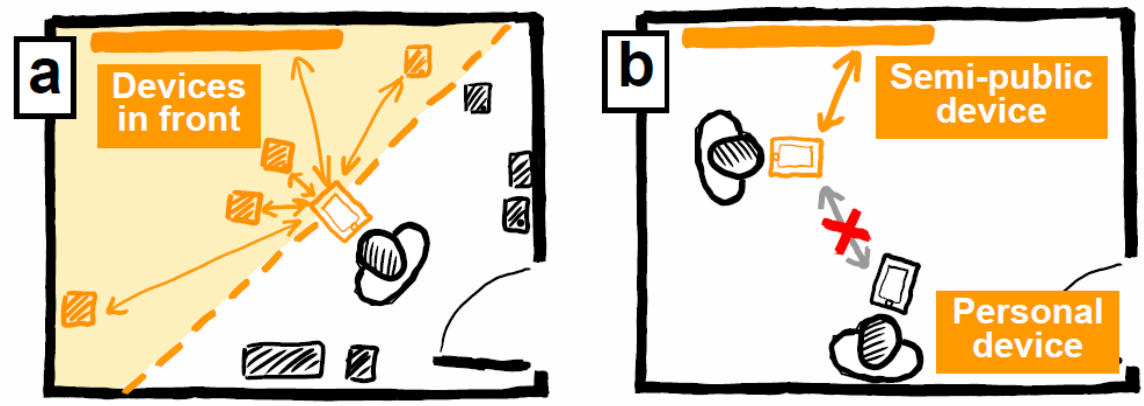


Figure 8.6 Proxemics-dependent awareness considering device's (a) orientation and (b) identity.

The combination of the five proxemics factors informs the decision about device connectivity, and the corresponding visual/auditory/tactile feedback provided, that eventually allows a user to leverage this knowledge of device presence and connectivity for further interaction.

8.3.2 Dynamic Notifications about Device Presence and Location

Given the above, a broad variety of notification mechanisms can inform a person about the presence of other nearby devices and opportunities for interaction: audible signals, vibro-tactile haptic feedback, visual notifications, etc. Yet, given the increasing number of devices in a ubicomp ecology, we opted for a visual approach, as such notifications can be displayed in a more ambient and distinguishable manner. Visuals can portray device identity and location, and – as we will shortly see – can also serve as containers showing content (stage 2) and act as portals for information exchange (stage 3).

In general, all device screens in close proximity display graphical icons representing the location of surrounding connectable devices (for example see circled areas in Figure 8.1a, Figure 8.7, Figure 8.8). Each icon informs the user: where the device represented by the

icon is physically located relative to the device displaying the icon; that there is a potential connection between those devices; and that devices can interact with one other (e.g., allowing information transfer). Icon appearance can be informative, such as a graphic that represents the nearby tablet. They can also be augmented with other information, such as the name of that device and/or its owner.

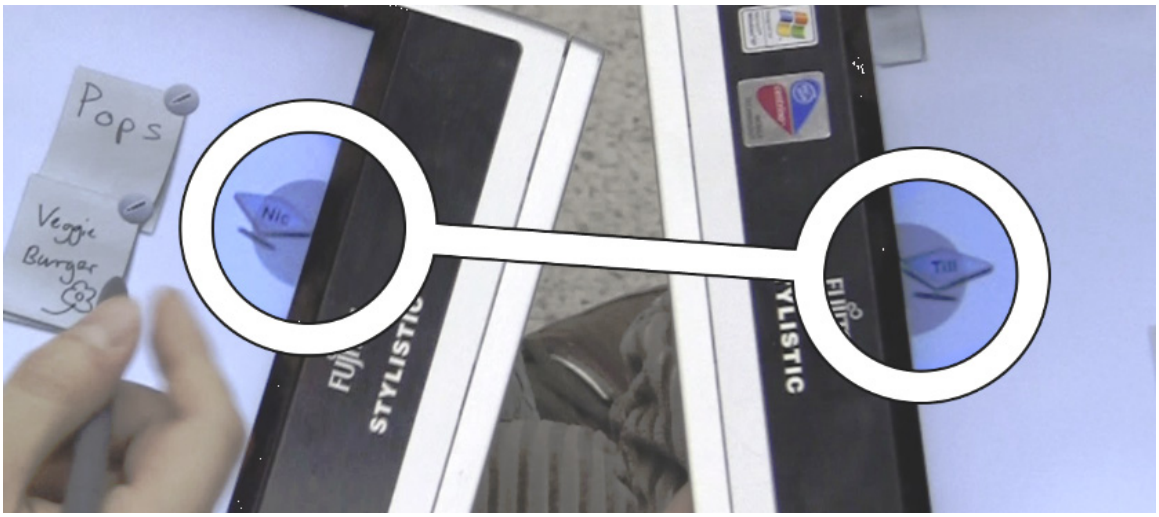


Figure 8.7 Icons at the edge of the screen indicate the presence and location of other devices in close proximity (2 tablet computers).

Figure 8.7 exemplifies this in Proxemic Brainstorming: as the two people move their tablets towards each other, icons at the edge of both screens show the other devices and the name of the device's owner. Extending earlier work (Gellersen et al. 2009; Rekimoto et al. 2003), icon locations are continuously animated around the edge to represent the relative directional location of the corresponding device. In Figure 8.7, we see how both displays' icon locations illustrate their physical spatial relationship. Figure 8.8 is similar, except it shows how several locations are indicated in a multi-device environment: in this case, two handhelds and a large display. Again, this helps reducing ambiguity of which icon corresponds to which device in the environment.

Because icon location is dynamic, people can further identify the mapping of device icons to actual physical devices by changing their own device's distance and orientation and observing icon changes. If multiple devices are shown on a tablet's edge, for example, a person can move and/or rotate the screen and see the icons' positions updated in real-

time. Naturally, the same continuous feedback applies when a person is moving closer to a cluster of devices. While approaching those devices, their corresponding icons on the tablet continuously change to reflect the new relationship between the tablet and each device. Thus, a person can move seamlessly towards and gradually engage the particular device desired for interaction.

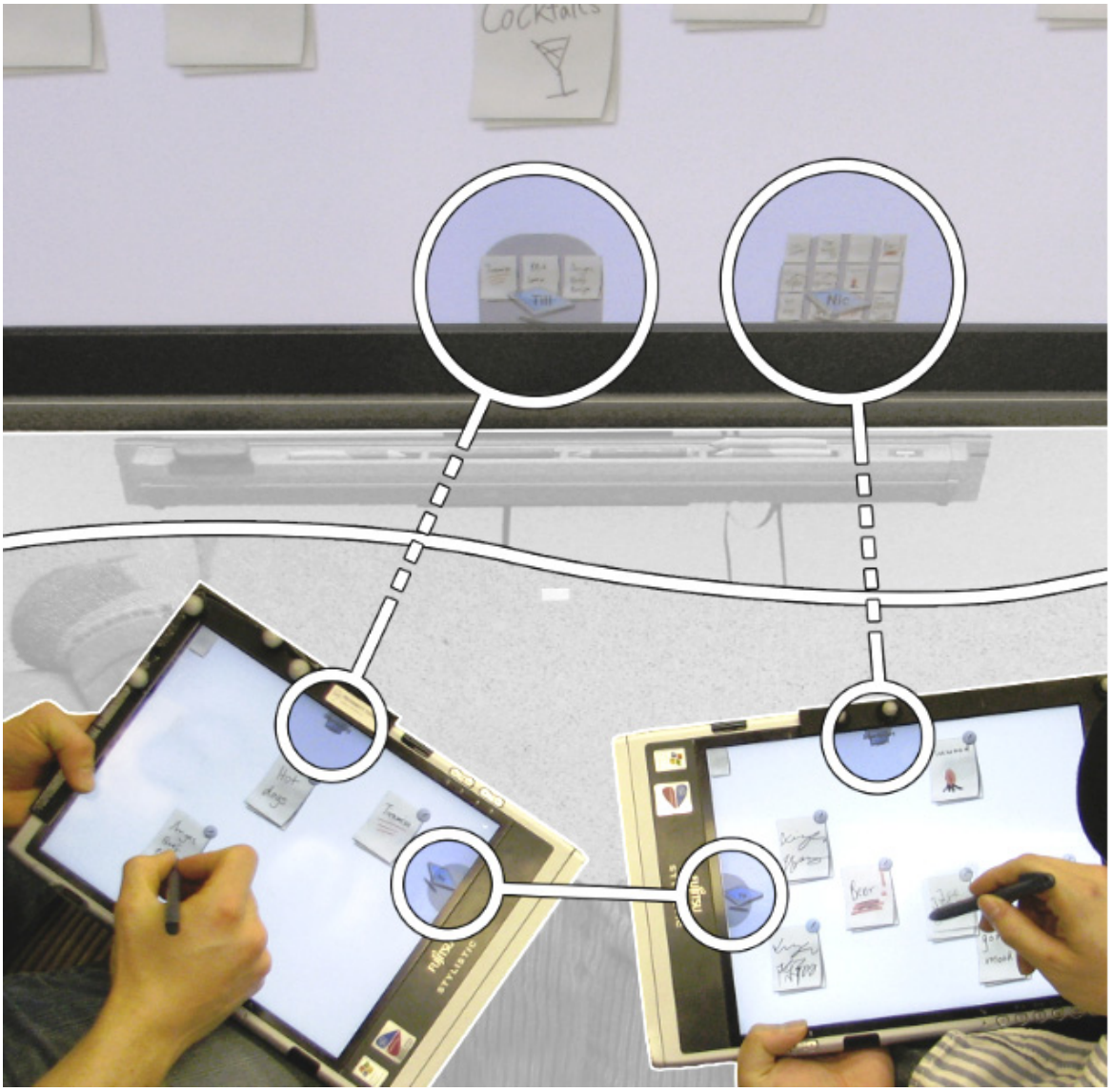


Figure 8.8 Content awareness: Proxy icons indicate the presence of nearby tablets and a large interactive display. Available content of the tablets is displayed as thumbnails atop the icons on the semi-public large display (top), but not on other people's personal devices (i.e., the two tablets at the bottom).

8.4 Stage 2. Reveal of Exchangeable Content

As proximity increases, the gradual engagement pattern suggests that devices should reveal *content* available for exchange. Knowing what content a device offers for transfer is important information for a person to decide on further interactions. In fact, revealing content available for interaction or transfer to another device creates opportunities that invite a person to discover more about this content, eventually leading to more in-depth interactions.

8.4.1 Proximity-dependent Progressive Reveal

Importantly, revealing content is not all or none. Rather, the *distance* and *orientation* between two devices can directly affect the amount and level of detail of content awareness information shown on other devices. Our *proximity-dependent progressive reveal* technique maps the distance between devices to the amount of information shared between them. The closer two devices are, the more information is shared between them. The level of detail shown (i.e., the amount of information shared) can change either at *discrete* distance levels, *continuously* with changes in distance, or at both *discrete and continuous* levels. As well, the level of detail can change depending on the orientation between devices. Again, this can happen at discrete angles (e.g., facing to or away from another), or through continuous changes of the orientation (e.g., from 0 to 180 degrees). Progressive reveal is important for three reasons. First, it presents people with opportunities as they approach another device; as with ambient displays, this could mediate the move from background peripheral awareness to foreground interaction (Vogel and Balakrishnan 2004). Second, it gives them the chance to pull away, for example, if they see content about to be revealed that they would rather not make public. Third, it provides implicit security: in public contexts, fine details may appear in small size, and only when a person is (say) directly in front of other device, thus masking it from passersby.

The following three example scenarios illustrate both discrete and continuous approaches for progressive reveal of device content when approaching another device.

For our first example, Figure 8.9 shows how content of the *proxemic photo canvas* is progressively revealed at *discrete* distances. Figure 8.9a reflects Stage 1: a person holding a digital camera first sees the camera icon on the large display from afar (for illustrative purposes, the icon is shown magnified as an inset in Figure 8.9a+b). Figure 8.9b,c reflect Stage 2. When the person approaches the wall display and crosses the next distance threshold (here 2m), the most recently taken photo stored in the digital camera is shown next to the camera icon (Figure 8.9b). When he moves directly in front of the wall display while holding the camera close to the screen (~30cm distance), multiple photos on the camera are revealed and shown in a spiral around the camera icon (Figure 8.9c).

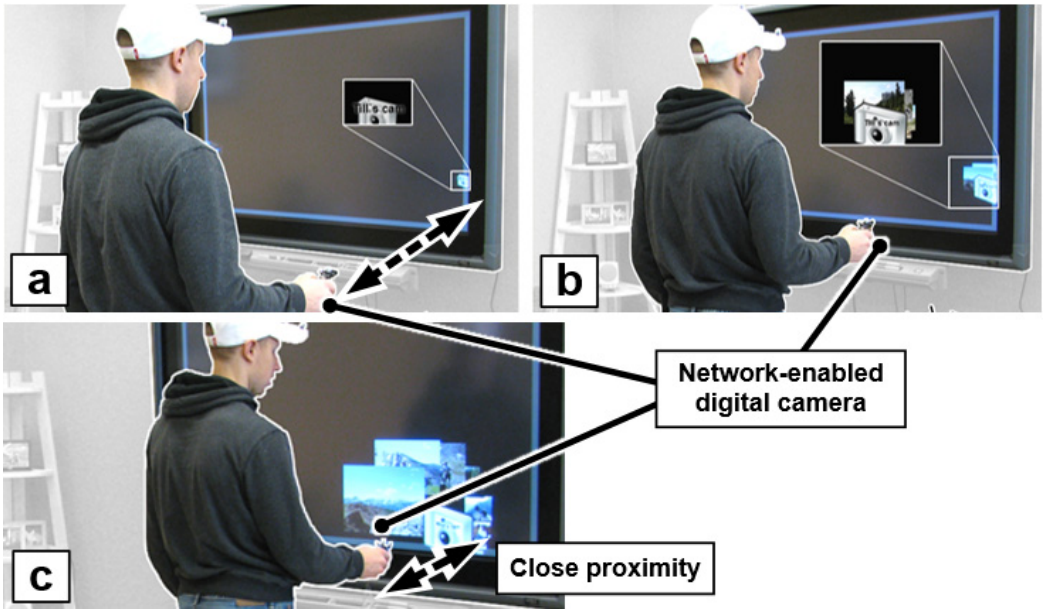


Figure 8.9 Proximity-dependent progressive reveal at discrete levels with the photo canvas: As a person approaches: (a) the large display first shows the presence of his digital camera, (b) then displays the last photo taken with the camera, and (c) a detailed view of multiple photos is revealed, all as a function of proximity.

In our second example, Figure 8.10 illustrates how Proxemic Brainstorming *continuously* reveals content during Stage 2 – in this case multiple sticky notes located on people's tablets – as they move closer to the large display. The wall display shows thumbnails of all sticky notes located on the tablets above the awareness icons (Figure 8.10, right side). For the person sitting at a distance, the actual text on these notes is not yet readable, but the number of available notes is already visible. For the second person moving closer to

the wall display, the thumbnails increase in size *continuously* (Figure 8.10b). For the third person standing directly in front of the display, the sticky notes are shown at full size (Figure 8.10c), allowing the person to read the text of all notes stored on the tablet and to pursue Stage 3 interactions, explained shortly.

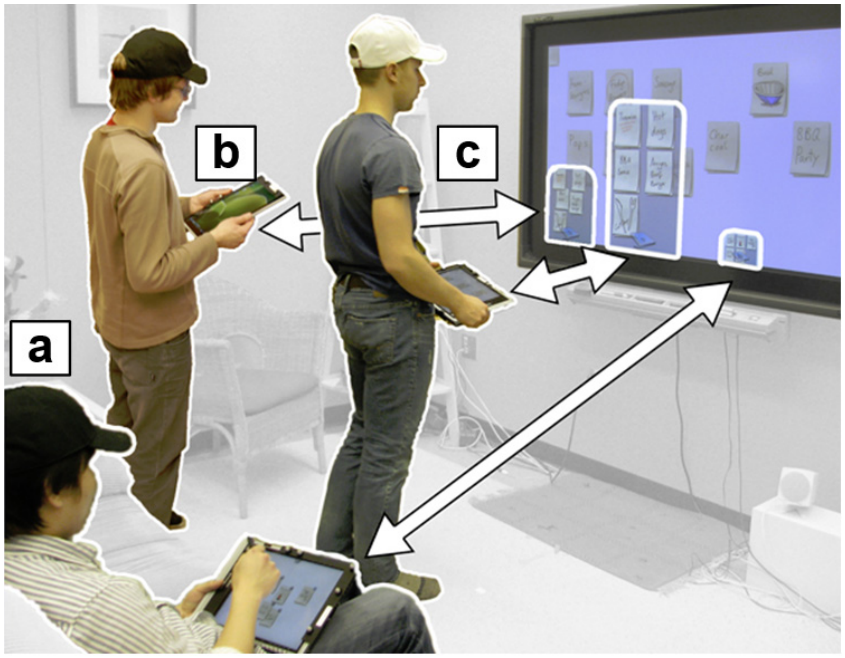


Figure 8.10 Proximity-dependent progressive reveal of personal device data of multiple users at different distances to the display: (a) minimal awareness of a person sitting further away, (b) larger, visible content of a person moving closer, and (c) large awareness icons of person standing in front of the display.

For our third example, we illustrate how we use *discrete zones* and *continuous movements* to progressively reveal available content. Figure 8.11 shows how content of the *proxemic media player* (our running example from Chapter 7) is progressively revealed when other devices around it are present. When a person takes a portable media player out of their pocket while sitting at a distance, in stage 1 the system recognizes the device and provides awareness information through a visual icon at the border of the display (Figure 8.11a). This icon represents the portable device, where it indicates the opportunity to share content between the large surface and the portable device. If the person then orients this portable device towards the large screen, more detailed information about that device (Figure 8.11b shows the device icon) and its contents become visible (Figure

8.11c shows small thumbnail images of the video content on the device). All these events so far happened at discrete distance levels. Depending on the orientation between the device and the large surface, the icons continuously and instantly update their position at the border of the interactive wall screen, so that they always face the direction of the portable device. As the person moves the personal device closer to the large display, the size of the icon and the video thumbnails shown grow continuously to a large area of the screen. This is visible in Figure 8.11d which shows the now large preview images of the videos. Finally, when the person is very close to the display, the actual titles of the portable media player's videos are shown on the screen (also visible in Figure 8.11d). When the person puts the device back in his pocket, the visualization immediately disappears and the media player continues its playback.

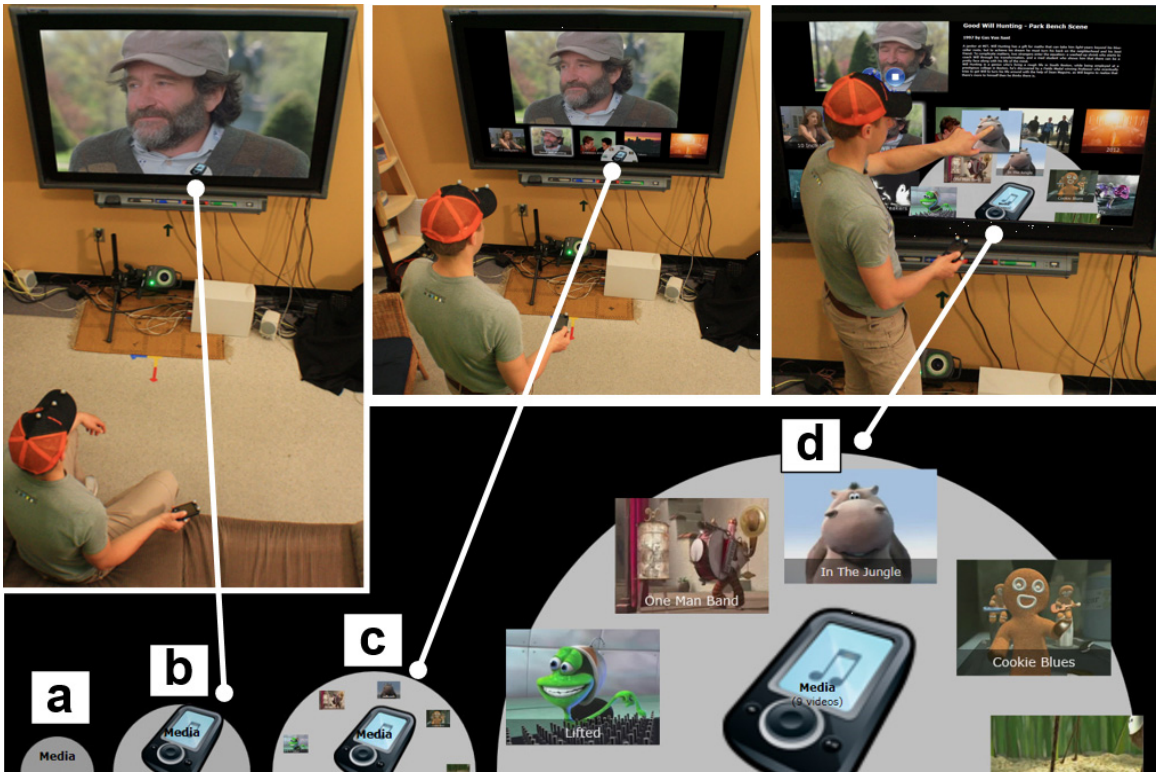


Figure 8.11 Proximity-dependent progressive reveal with proxemic media player, with increasing information available as a function of proximity: (a) awareness icon from afar, (b) more details about the device, (c) revealing content, and (d) showing large video thumbnails and titles.

8.4.2 Implicit vs. Explicit Reveal

The above method illustrates how content is revealed via a person's implicit actions. However, reveal can be complemented by explicit methods as well to fine-tune what is revealed. To illustrate a combination of implicit and explicit progressive reveal, we implemented a tilt-scrolling technique (a variation of (Dachselt and Buchholz 2009)) in Proxemic Photo Canvas (Figure 8.12). During Stage 2, a person sees a few of his camera's latest photos – organized in a spiral – progressively revealed as an implicit consequence of moving towards the display. To see more content (and while still distant from the display), the person can now explicitly tilt the camera leftwards or rightwards to browse through the timeline of photos. Of course, alternative forms of explicit input (e.g., hand gestures) could be considered to cause similar explicit reveal behaviours.

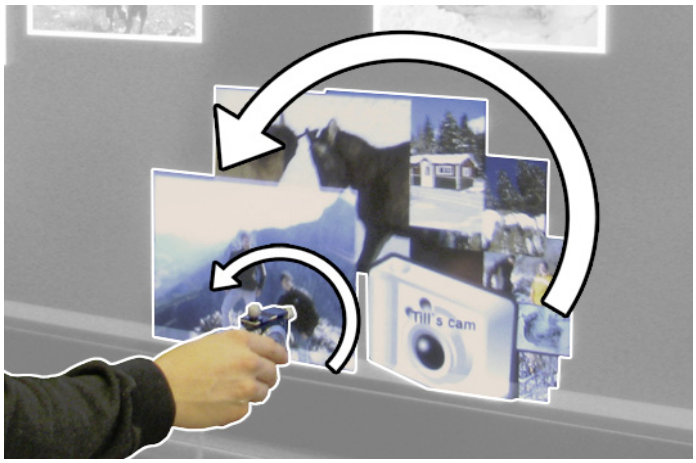


Figure 8.12 Explicit reveal: tilt-scrolling reveals content on digital camera.

8.4.3 Revealing Content on Personal vs. Public Devices

The information revealed about available content on the display of other devices should differ between *personal* and *semi-public* devices. For personal devices, we currently only provide an awareness icon of surrounding devices, but not their content. This is partially due to privacy reasons, but also size constraints: showing content on the small screens of *personal devices* may interfere with other content the user is viewing or interacting with. As we will see, we use other stage 3 methods to reveal content on personal devices during explicit information exchange.

Semi-Public devices (e.g., a wall-mounted display in a meeting room), however, reveal content located on one's *personal devices* as one approaches the display. For example, the wall display in Figure 8.8 shows both tablets' awareness icons at its lower edge, where each icon now contains small thumbnail images of all Proxemic Brainstorming notes on the corresponding tablet (i.e., 3 notes on the left tablet, 12 notes on the right one). Even though these thumbnails are too small to allow for full readability, they provide awareness information about the number of notes available for sharing on each of the tablets.

8.5 Stage 3: Techniques for Information Transfer between Devices

Stage 1 and 2 indicate device presence, connectivity and available content, eventually leading to Stage 3 of the gradual engagement pattern, where a person can interact with progressively revealed content. We now present a series of novel interaction techniques (and others from related work) that allow for sharing and transferring content between devices.

We stress that the power of these Stage 3 techniques is that they are used in conjunction with the Stage 1 and 2 methods *vs.* as stand-alone techniques similar to those found in the literature. Importantly, these techniques consider proxemic relationships between devices to drive the interaction, and come into play at particular points during Stages 1 and 2. We are particularly interested in two contexts:

- whether information exchange is a single person activity (based on the proximity of a handheld to a semi-public display) or a cooperative multi-person activity (based on the proximity of at least two handheld devices);
- how they allow people to interact at different levels of proximity e.g., from a distance *vs.* within reach.

8.5.1 Single Person Transfer: from Personal to Public Device

First, we present a series of techniques that primarily allow a *single person* to share content from their personal device to a public display. We begin with distance-based interactions that could be performed in the early periods of progressive reveal, to within reach interactions at later periods.

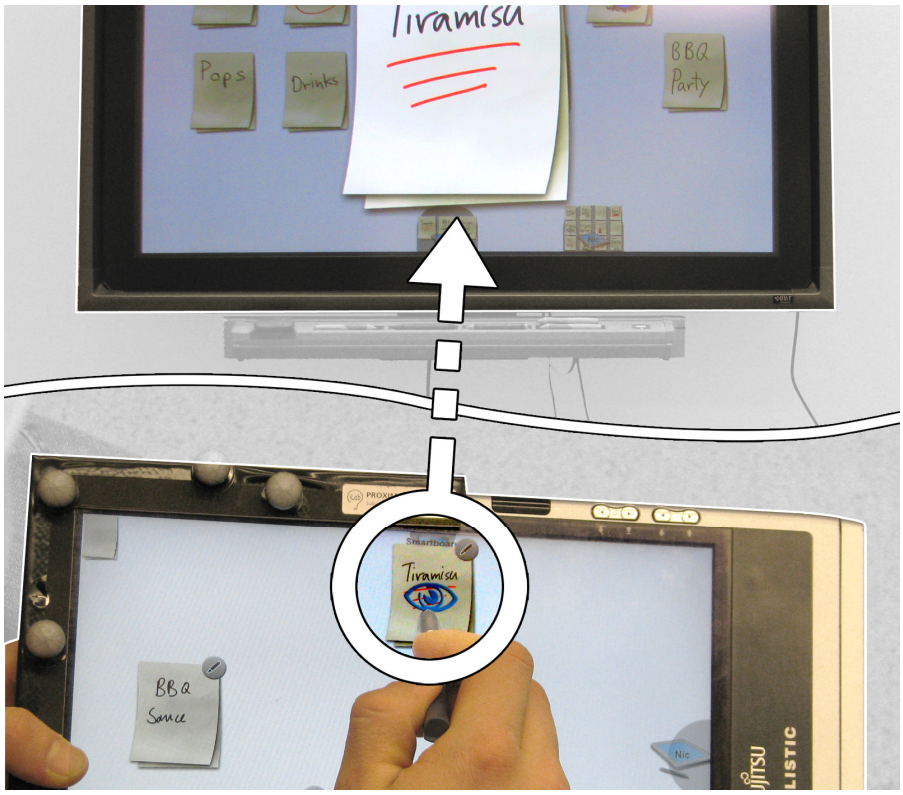


Figure 8.13 Large display drag and back: (bottom) dragging content on the wall display's awareness icon on the tablet, (top) content appears full screen on the large display.

8.5.1.1. Large Display Drag and Back (From a Distance)

Large display drag and back allows a person to temporarily show digital content from their personal device on a large *public* display. The idea is that the person owns the information, but is making it more convenient for others to view it. To share content temporarily on a large display, a person can drag content onto the awareness icon representing a nearby large screen.

For example, Figure 8.13 bottom shows a person dragging a note onto that icon. As he does so, a *viewing icon* appears atop the content (here, the 'eye' icon shown inside the circle of Figure 8.13) indicating that one is about to share the note on that particular public display. As the person releases the note, the content appears in full screen view on the wall display (Figure 8.13 top). To remove shared content, a person simply drags the content back from the device's awareness icon onto the tablet's canvas. Sharing also works for multiple people simultaneously: if others do similar actions, all shared content is shown side by side on the large display.

8.5.1.2. Cross-device Portal Drag to Transfer (From a Distance)

We can also exploit the awareness icons of Stage 1 as portals to transfer information between them via drag and drop. This extension of the *portals* concept (Voelker et al. 2011) supports transfer methods across *multiple* devices.

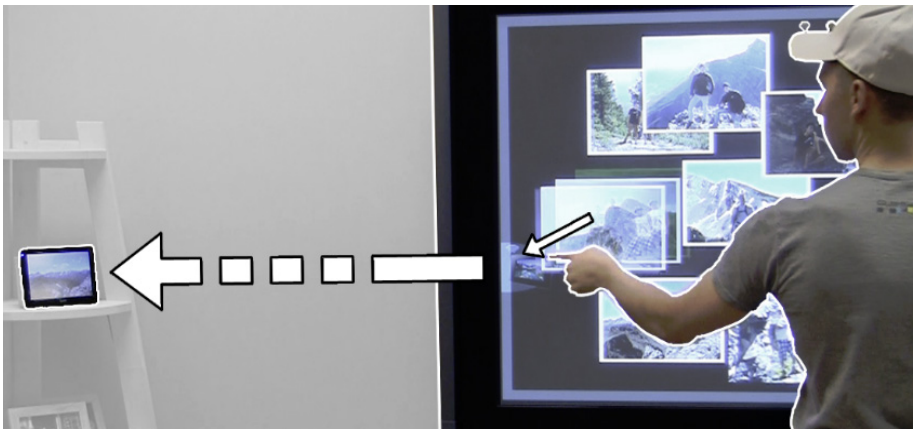


Figure 8.14 Cross-device portal drag.

Figure 8.14 illustrates the Proxemic Photo Canvas on a large display and a picture frame appliance; their awareness icons are visible at their borders. A person is transferring content from the large display to the picture frame simply by dragging a photo onto the picture frame portal, which then shows that image in full size in the frame. To foreshadow what is to come, in Chapter 9 we will extend this technique to the transfer between multiple personal tablet computers that are in close proximity to another.

8.5.1.3. Integrating and Refining Pointing Techniques: Point & Pin and Point & Edit (From a Distance)

Considering all stages of the gradual engagement pattern also helps us integrate and refine existing gestural interaction techniques for information transfer.

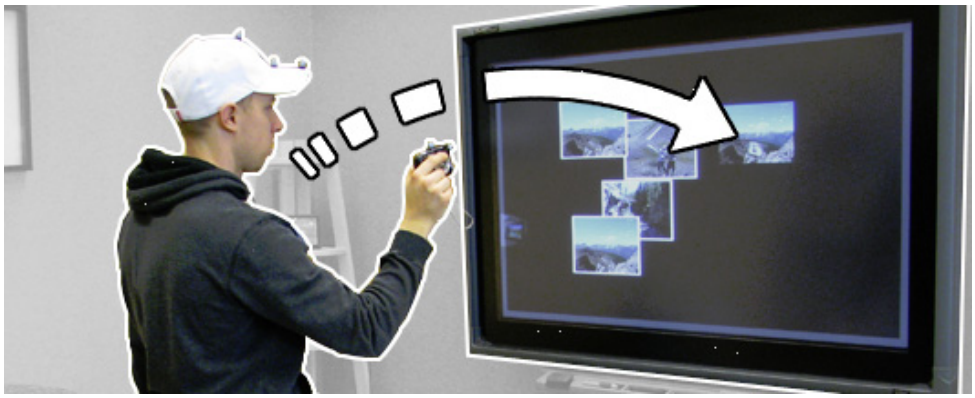


Figure 8.15 Point & pin technique to transfer content from a distance.

For example, our *point & pin* technique lets a person copy content from the camera onto a distant public display by pointing at it and subsequently performing a throwing gesture to pin it there. This technique refines throw & tilt (Dachselt and Buchholz 2009) and *chucking* (Hassan et al. 2009): the pointing ray is the extension of the stretched out arm holding the camera relative to the body, proxemic relationships of device-to-device determine recipient selection, and a preview of the most recently taken photo is shown where that ray meets the large wall display (i.e., an explicit Stage 2 action). People can 'throw' photos by forward-accelerating the hand holding the camera, permanently copying that photo onto the screen at that location (Figure 8.15). The technique also works on a digital picture frame, where the photo is then shown full screen in the frame.

In a similar fashion, we integrated the *point & edit* technique as a refinement of semantic snarfing (Myers et al. 2001) and touch+air pointing gestures (Bragdon et al. 2011). While content on a large display is convenient for viewing, editing may be more efficient on one's portable device. To select content for transferring back to the tablet, the tablet itself can function as a distant pointing device. As shown in Figure 8.16, a person holds the tablet away from his body and points it towards the display, where the specific

proxemic relationship of person and device is triggering the pointing mode (this is a variation of a technique we introduced in Section §7.4). The system calculates the intersection of the pointing ray with the large display's surface. This action highlights the note (with a colored border) that is closest to that intersection point. To transfer the note to the tablet temporarily for editing purposes, the person taps on the tablet's screen. To place back the note on the large display, the person points at a location on the display and again taps the tablet's screen to confirm.

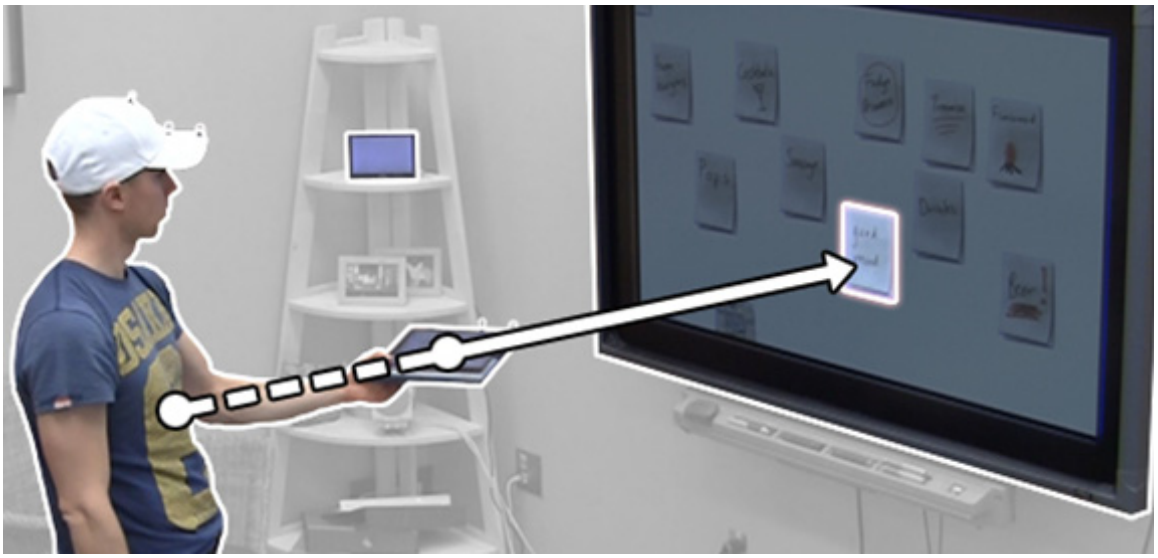


Figure 8.16 Point & edit technique to select and edit content from a distance.

8.5.1.4. Direct Touch: Drag In and Out (Close Proximity)

In this technique, illustrated in Figure 8.17, the tablet's content is progressively revealed in Stage 2 by growing it in size directly in front of the approaching person. (The area also follows the person's side-by-side movements). When within direct touch distance to the large display, this content becomes interactive, i.e., it allows that person to access his tablet's content by directly touching the large display. In particular, a person transfers content between the two devices (tablet & large display) by dragging items into or out of their personal area.

Figure 8.17a and Figure 8.17b illustrate how Proxemic Brainstorming and Photo Canvas allow one to drag notes and photos to an empty region on the screen, which transfers

them across devices. While both progressively revealed their contents in visually different ways, the transfer operation is identical.

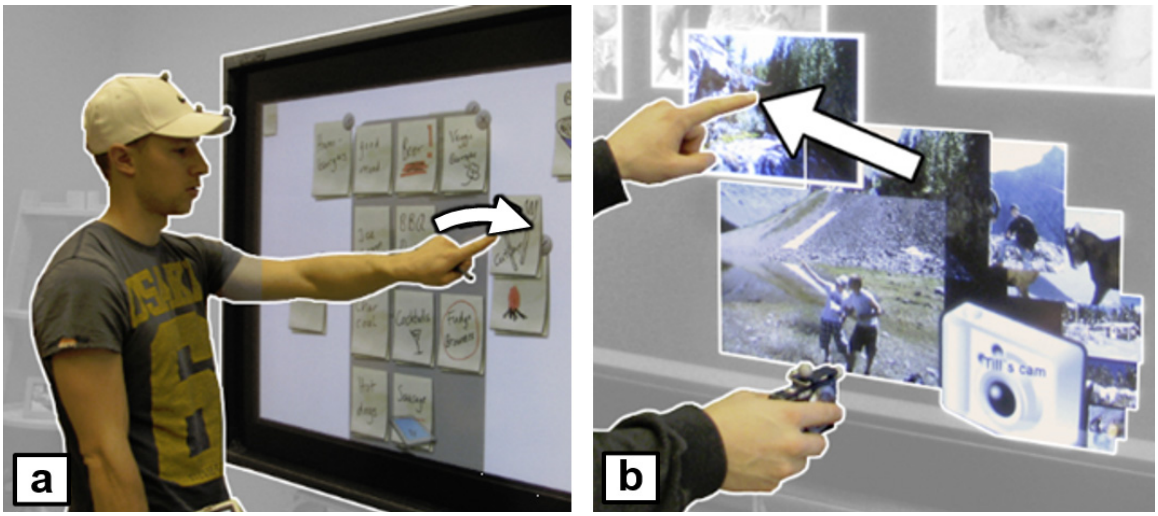


Figure 8.17 Drag in and out in close proximity in both applications.

Again, we can integrate refinements of existing techniques at this stage. For example, inspired by *PhoneTouch* (Schmidt et al. 2010), the tablet itself can now be used as a physical pointing and selection device. Touching the device on the large screen will pick up or drop off information (Figure 8.18). Considering proxemics refines this technique: the pointing function of the tablet becomes active when a person stands within touch distance, and holds the tablet in a way that one of its corners points at content on the large display. As the device moves towards the display, a projected pointer highlights the currently selected note (thus providing continuous feedback before touching). When the person touches a note with a corner of the tablet, the note is picked up and temporarily transferred to the tablet device for editing. After editing, a person can quickly place that note back to a given location on the large display by touching that location with a corner of the tablet.

Figure 8.18 Touching the device on the large screen will pick up or drop off information.

8.5.2 Collaborative Transfer

The next suite of techniques is tailored to multiple people collaboratively sharing content with each other through their *personal* devices, possibly including a large display. Unlike the single user techniques, these include coordination protocols that influence how handoffs are achieved.

8.5.2.1. Collaborative Handoff (From a Distance)

In collaborative work scenarios, people may want to pass on digital information to another person. Often, this requires tedious sequences of tasks such as sending files by email or copying and retrieving content using portable media. Our notion of a *proxemic-aware collaborative handoff* (inspired by collaborative stitching by Hinckley et al., 2004), represents a simpler method for transferring content between devices. The idea is that one person starts the gesture on his personal device, and a second person continues this gesture on his personal device to complete the handover process. That is, one person cannot transfer information without cooperation from the other person. Both must also be in close proximity before these techniques are activated. We expect people to monitor each other's actions in a way that mediates their social protocols.

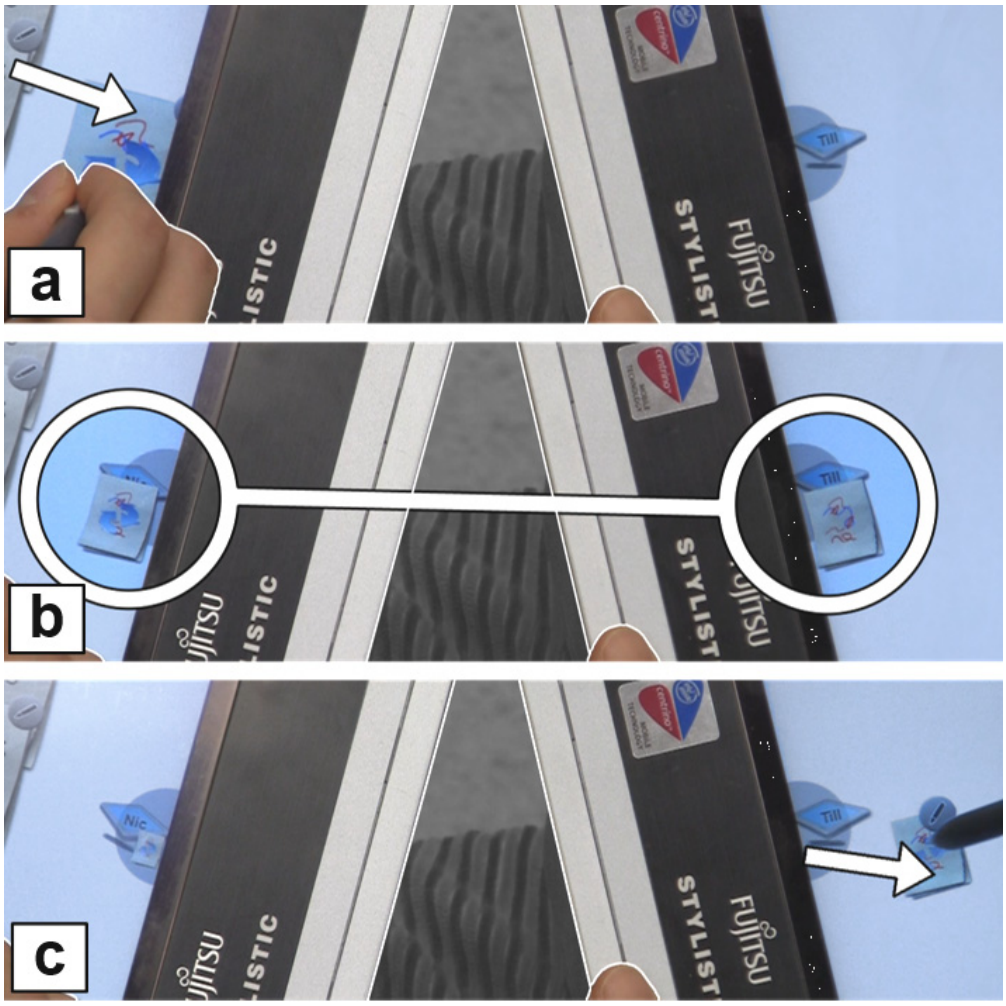


Figure 8.19 Collaborative handoff: (a) dragging content onto awareness icon representing the other tablet, (b) content appears on 2nd tablet, and (c) dragging content off the icon transfers it.

Figure 8.19 illustrates an example of content exchange in the Proxemic Brainstorming application between two people who have moved their tablets besides each other. As before, both are aware of connection availability via progressive reveal, where in this case the awareness icon size is larger as people move closer. Similar to our previously described 'portal drag to transfer', a person can initiate content sharing by dragging a sticky note onto the awareness icon of the second person's tablet (Figure 8.19a). What is different is that a thumbnail of the content then appears on the second tablet, so that it is temporarily visible on both screens (Figure 8.19b). If the second person drags the thumbnail image from the awareness icon onto his screen (thus *continuing* the first

person's drag operation), the thumbnail on the first person's tablet disappears and the content is now permanently stored on the second person's device (Figure 8.19c). Through this continuation of the gesture that was started by the first person, the second person '*accepts*' the content transfer action. If the person does not accept, the transfer is not performed. As well, if the transfer has not yet been accepted (i.e., phase 2; Figure 8.19b), the first person can cancel the transfer by dragging the content back onto his or her screen.

8.5.2.2. Drag between a Public Intermediary (Close Proximity)

Two people can use the shared screen area of the large public display as a way to hand off content. The idea is that because information on that display is public, it implicitly gives permission to both actors to exchange information between their devices.

Figure 8.20 illustrates this. Two people are standing in direct touch distance in front of a large wall display with their tablet device in hand. Via progressive reveal, the personal content of both their devices are visible on the wall display as two interaction areas – one per person – in positions that reflect the side-by-side locations of both people (see the rectangular grey boxes containing sticky notes on the screen in Figure 8.20). The large interaction areas on the screen make it easy to view and modify content.

Two different versions illustrate different ways of performing the transfer. In the handoff version, a person can drag a note to the shared public area (i.e., the regions not covered by individual interaction areas) on the large display (Figure 8.20 a,b), but not into the other person's area. The second person accepts that transfer by picking up this note and drag it to his own interaction area (Figure 8.20 c,d).

The second version does not require this handoff, relying instead on social protocol as augmented by the high visibility of all actions. Here, a person can move (or take) a note directly from one tablet to another by dragging it from one interaction area straight to the other.

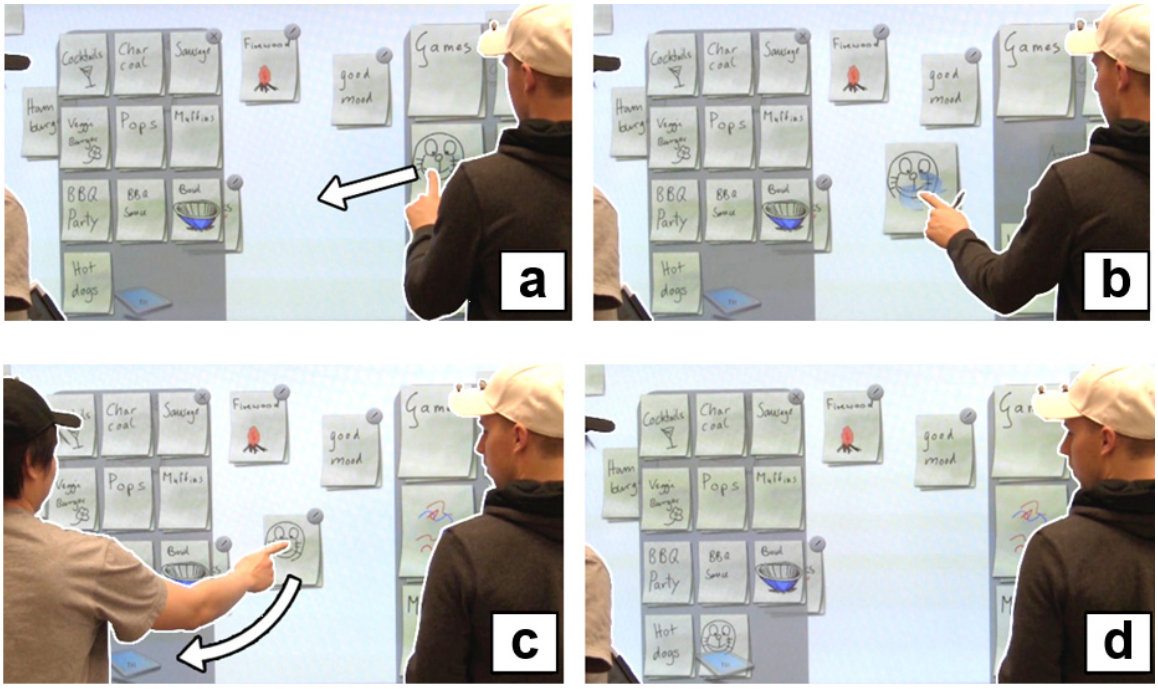


Figure 8.20 Drag between a public intermediary: (a) person drags note out of his personal interaction area, (b) using the empty space between the interaction areas as a clipboard. (c) Second person drags note into his interaction space of the tablet, and (d) the note is now moved to his tablet.

8.6 Implementation and Proxemic-Aware Widgets

As with our first case study, we implemented the techniques and applications described in this Chapter using the Proximity Toolkit. This allowed us to rapidly explore various interaction techniques that consider fine-grained proxemic measurements in the context of mitigating cross-device operations. We briefly explain a few aspects of our implementation that extend the provided functionality of the Proximity Toolkit, which in turn functioned as additional building blocks for our prototyping purposes.

First, many of our techniques monitored people's or devices' presence in one or multiple discrete zones around other devices. The system then triggers notifications about entities entering or leaving one of these zones. We extended the Proximity Toolkit's circular notifications zones (Figure 8.21a) to support a wider variety of zones: rectangular, elliptical, and other arbitrary (defined as a polygon mesh) shaped zones

(Figure 8.21b). The applications then subscribe to changes (i.e., entities entering or leaving) in these zones.

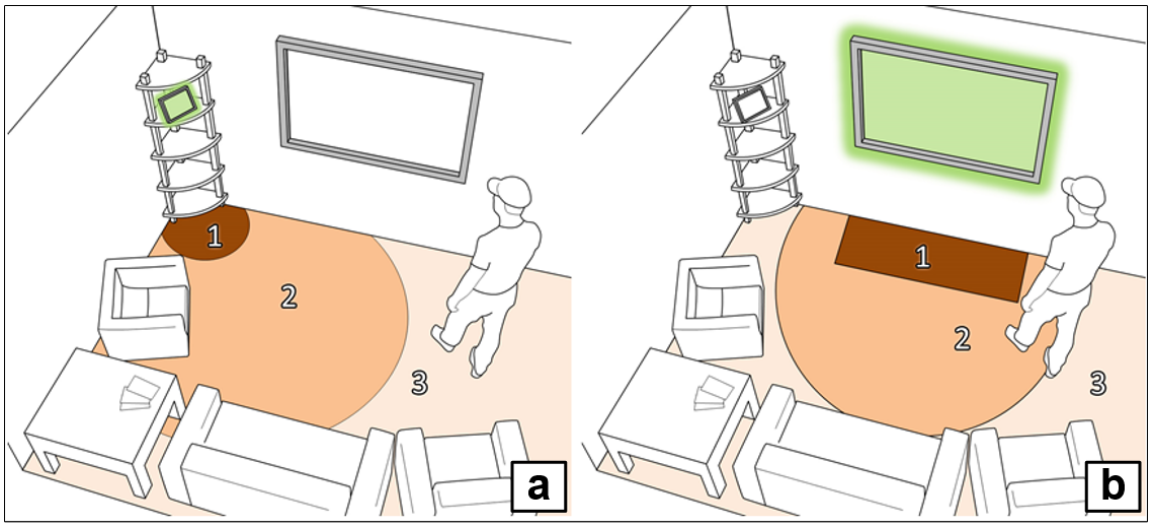


Figure 8.21 Extensions to the Proximity Toolkit's notification zones.

Second, our techniques frequently use visual content (on tablets or wall display) that reacts to changes in a person's or devices' proxemic relationship; e.g., the awareness icons that change their size and displayed content depending on distance, orientation, and so on. We designed reusable proxemic-aware GUI widgets that facilitate making content on the screen react to one or multiple of the proxemic dimensions. These widgets are designed as containers to contain other widgets as content. The widget's proxemic-aware features are a follows.

(a) **Automatically follow the direction/orientation of any entity tracked in the space around the display.** For example, the awareness icons are always displayed on the edge of the screen closest to the entity they represent.

(b) **Directly map the size of the widget container to any proxemic measures**, e.g., the distance or orientation angles between two entities. Once the monitored entities move, the size is adjusted automatically; e.g., it grows larger when entities move closer.

(c) **React to discrete events for changing the visual appearance of the widget and its content**. Examples are: show content (e.g., when person with device enters a room), hide content (when leaving), or show different content with different level of detail (when person moves closer).

Overall, we see these widgets as a starting point of an extensible programing library of generic proxemics-aware widgets. These widgets extend the behaviour of the Proximity Toolkit, and encapsulate the common and reused behaviours of software reacting to proxemic changes of people's and devices' relationships.

8.7 Discussion

In this section we discuss how the gradual engagement pattern applies to larger ecologies of people and devices, how we can address privacy concerns, and how to apply the pattern with different available tracking hardware.

8.7.1 Large Ecologies of People and Devices

We believe the gradual engagement pattern will be increasingly relevant as ubicomp ecologies emerge with larger numbers of personal and public devices featuring different form factors and capabilities. As shown, the pattern and the techniques we derived support a variety of 1-to-1 (e.g., *collaborative handoff*, *cross-portal drag*) and 1-to-many (e.g., *progressive reveal*, *large display drag and back*) collaborative settings. A major advantage of using the stages of gradual engagement is that it leads to an implicit *filtering* of choices presented to a user. For example, following the pattern prevents a system showing an overwhelming number of icons of all present devices in a large ecology, but instead fosters a design that only reveals the device presence between neighbours. Nevertheless, future work may extend our techniques to offer alternative 1-to-many and many-to-many sharing possibilities, e.g., by allowing dragging content on multiple device icons to share transfer content to all devices of a group.

8.7.2 Gradual Engagement and Privacy

We recognize that the gradual engagement pattern for cross-device transfer can introduce privacy concerns in some situations. Thus, designs must incorporate safeguards to guarantee privacy of people's personal information. First, as mentioned in Stage 1, location information is essential to drive sharing behaviour. For example, while it is likely to have loose sharing between devices at home, only restricted sharing might be desired in the office, and no sharing (but maybe only device awareness) in public settings. Second, implicit protection rules can be applied to prevent sharing. For example, a device in a person's pocket stays invisible, but shares content when the person takes it out and points it towards other devices. For example, Marquardt et al. (2010) followed this principle with their light-sensitive RFID tags (that are often integrated in credit cards or transit passes) that are deactivated when inside of a person's pocket and are only active when taken outside of the pocket. Third, explicit actions and commands on the device can allow a person to manually stop sharing and close inter-device connections at any time (e.g., by pressing a button). Fourth, designs should leverage existing social practices by allowing people to predict and control what they reveal to others. For example, Stage 1 reveals opportunities for information sharing. No actual information is revealed until Stage 2 – and even that is gradually done, where the reveal grows as the proxemic distance lessens. Using this feedback, a person can decide at any point to stop sensitive information display.

Of course, privacy in proxemic interactions is a rich and fertile area for future work. While beyond the scope of this dissertation, it is vital that it is addressed.

8.7.3 Pattern Applied to Different Tracking Hardware

The principles of the gradual engagement design pattern can be applied to interactive systems using diverse high- and low-fidelity tracking systems. While the technology can limit or enhance how the design pattern is applied in a particular situation, the pattern itself goes beyond any specific technology. At the high-fidelity end of the spectrum of possible hardware, our implementation uses an infrared-based motion capturing system provided via our Proximity Toolkit. The precise tracking information it provides of

people and devices' distance or orientation allowed us to explore a large part of possible kinds of interactions. Such a system, however, is not yet suitable for wide deployments.

At the opposite side of the spectrum, a possible low-fidelity system using sensor fusion of depth-camera streams and wireless-radio signals for distance and orientation measurements can similarly integrate gradual engagement methods. Despite its lower tracking fidelity, it would allow for applying diverse methods across all three stages of the pattern, such as progressive reveal, cross-device portals, and/or collaborative handoff. In Chapter 9, we discuss one approach to achieve this with low-fidelity tracking hardware.

Many other tracking systems would support gradual engagement as well. For example, eye-tracker based systems can provide interaction awareness, reveal information, and offer interaction methods depending on people's attention through eye contact. Likewise, a system using GPS based positioning and digital compass data could apply the pattern in a larger-scale outdoor deployment. Overall, the gradual engagement pattern has the potential of being applied to many other proxemic-aware systems with diverse tracking capabilities along this low-/high-fidelity spectrum.

8.8 Conclusion

We believe that proxemic-aware interaction techniques following the gradual engagement pattern can help us design future ubicomp systems to drive the information transfer process in a way that more appropriately reacts to people's social understanding of personal space. We argued that gradual engagement is an essential prerequisite of such systems. We believe this pattern will be increasingly relevant as ubicomp ecologies emerge with an increasing number of personal and public devices of all different form factors and capabilities. Besides the application to cross-device transfers discussed in this chapter, the generalized gradual engagement pattern can be applied to other areas, e.g., interactive advertisements or games. In a similar way, these applications could benefit from (1) giving awareness notifications about presence, (2) reveal of content or possible

interactions, and (3) providing a range of interaction techniques appropriate to the particular contexts as defined by distance, orientation, and group engagement.

The gradual engagement pattern and our derived set of techniques for mediating cross-device operations we introduced is not a complete or exhaustive set, nor do they handle all issues that will likely emerge (such as privacy). Rather, they are a starting point suggesting further interaction techniques that consider the gradual engagement pattern during ubicomp system design.

Overall, this chapter demonstrated how to consider device-to-device proxemics in interaction design. Our design of particular interaction techniques was informed by the proxemic theories reviewed in Section §3.1, the identified proxemic dimensions in Section §3.2, and the gradual engagement pattern in Section §7.6. In our next chapter, we drive this research further, where we consider both the *proxemics of people* and the *proxemics of devices* in order to facilitate small group collaboration.

Chapter 9. Considering Person-to-Person and Device-to-Device Proxemics

In this chapter[36] we now combine strategies introduced in the last two chapters, where we consider *person-to-person* proxemics (through small group F-formations) and *device-to-device* proxemic relationships, both which will be used to facilitate sharing of digital content between digital devices. In particular, we introduce GROUPTOGETHER as a system that explores cross-device interaction combining the two sociological constructs of proxemics (§3.1.2 – 3.1.7) and F-formations (§3.1.8), refined through a third construct named *micro-mobility* that describes how people orient and tilt devices towards one another to promote fine-grained sharing during co-present collaboration. Our work was also informed by an observational study we conducted of people working together with foam-core mock-ups of mobile displays. We explain how we designed a number of cross-device interaction techniques that support nuanced gradations of sharing, from the subtle to the overt, with the goal of minimizing the transaction costs—and social disruption—of sharing information across a small-group ecology of digital devices and situated displays.

---

[36] Portions of this chapter are published as:

Marquardt, N., Hinckley, K. and Greenberg, S. (2012) Cross-Device Interaction via Micro-mobility and F-formations. In *Proceedings of the ACM Symposium on User Interface Software and Technology – ACM UIST 2012.* (Cambridge, MA), ACM, October 7-10, pages 13-22.

(Filed as patent by Microsoft Research in 2013. Ken Hinckley and Nicolai Marquardt)

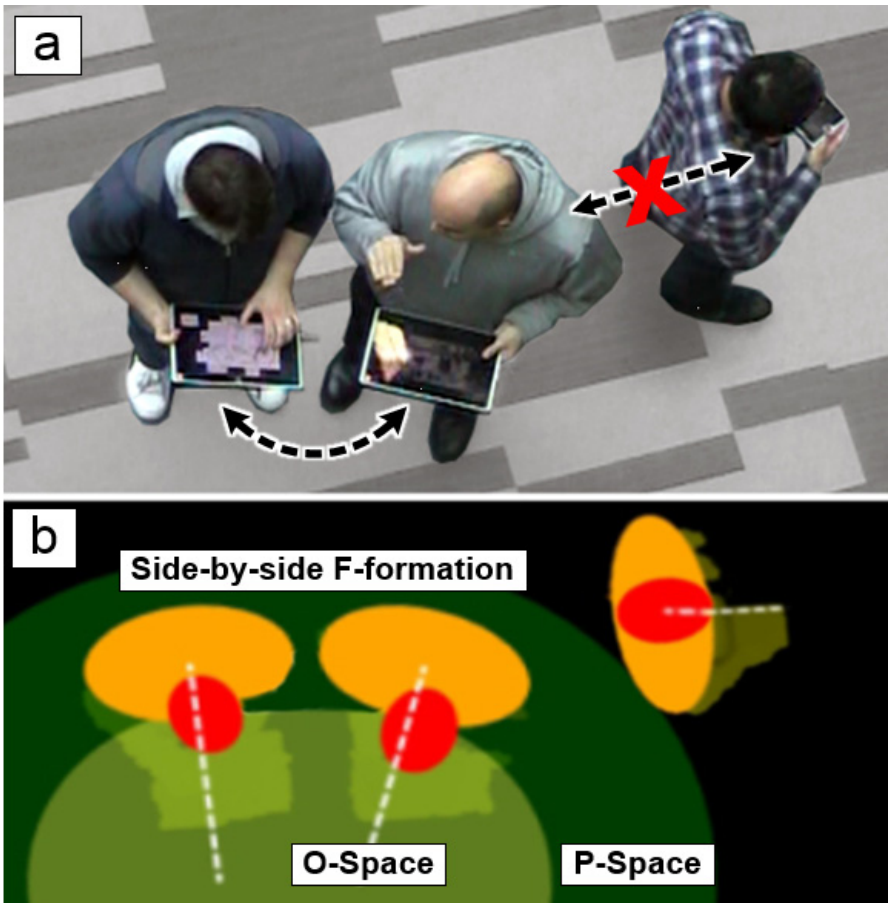


Figure 9.1 (a) F-Formation between multiple people as they collaborate, (b) grouped formations as tracked by the GROUPTOGETHER system.

The techniques are unique in that they consider device-to-device relationships as well as human-to-human relationships; that is, the context of the social group to mediate interaction. For example, in Figure 9.1a the two people on the left standing side by side engage in a collaborative discussion and share content on their tablet devices. The third person on the right – although standing very close by – is outside this social group and thus excluded from the lightweight federation of devices.

The chapter first briefly revisits related work and the theory of micro-mobility that motivate our work (§9.1). Next, we present an observational study which informs the behaviors that serve as the building blocks of our interaction techniques (§9.2). We then explain the GroupTogether system (§9.3) and detail the interaction techniques (§9.3), focusing initially on the micro-mobility aspects (§9.4.1 – §9.4.4). We then describe how

we extend these techniques to federated devices held by users in F-formations consisting of two or more persons (§9.4.5), and how we consider the case where the F-formation encompasses a digital whiteboard (§9.4.6). Next, we describe our system implementation and the hybrid sensing approach (§9.5). We close with an informal evaluation of the techniques (§9.6) as well as a discussion of salient issues raised by this research (§9.7). A video figure[37] illustrates the applications and the dynamics of the GROUPTOGETHER system and its interaction techniques described in this chapter.

9.1 Using Theory to Motivate Group Interaction Techniques

Despite the ongoing proliferation of devices and form-factors such as slates and electronic whiteboards, technology often hinders (rather than helps) informal small-group interactions. Whereas natural human conversation is fluid and dynamic, discussions that rely on digital content—slides, documents, clippings—often remain stilted due to the awkwardness of manipulating, sharing, and displaying information on and across multiple devices. To address these shortcomings, we leverage three sociological constructs that have been observed in human-human social activity to inform our designs of novel cross-device interaction techniques: (1) *proxemics*, (2) *F-formations*, and (3) *micro-mobility*.

First, we consider the interpersonal *proxemic* relationships between people as they engage, and between devices as they are used by people. We argued throughout this dissertation that computing systems—which are carried about as well as embedded in the world around us—could be enhanced by sensing proxemic relationships (via distance, orientation, movement, identity, and location) between all entities in a ubicomp ecology. In Chapter 7 we focused on possible interactions between a person and a large interactive surface, where the system responds to movement, proximity, and

[37] Video available for download at: http://grouplab.cpsc.ucalgary.ca/Publications/2012-GroupTogether.UIST

orientation of one person or multiple people in relation to the environment (cf. techniques introduced in Sections §7.3 and §7.9). In Chapter 8 we applied proxemic interaction concepts to device-to-device interaction, where we used the information about entities' orientation, distance, motion, location, and identity as a way to gradually engage people with opportunities for information exchange across devices (cf. techniques introduced in Sections §8.3, §8.4, and §8.5). In this chapter we focus on new device interactions afforded by micro-mobility (defined shortly) as mediated by the proxemics between people and devices and the group's social context of the F-formation (described next).

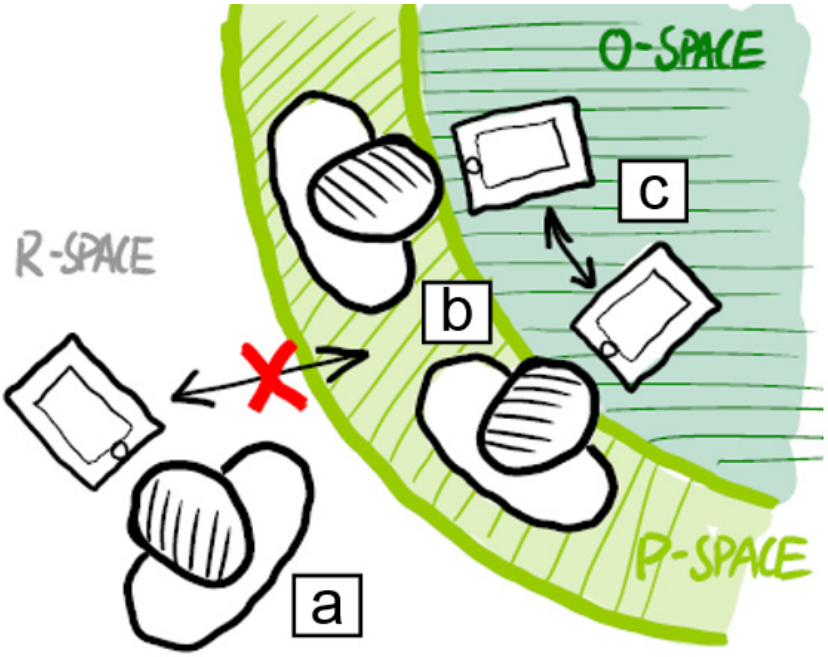
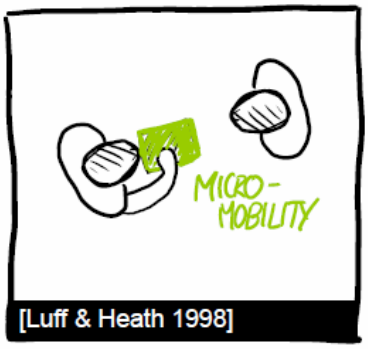


Figure 9.2 Sensing F-formations of both the people and the devices they carry to automatically federate devices and mediate cross-device interaction.

Second, we consider the dynamics of small-group gatherings as defined by *F-formations*. In particular, our system observes proxemic relationships and F-formations of both the people and the devices they carry to automatically federate devices and mediate cross-device interaction in a way that corresponds to real-time social groupings. Physically nearby people in the R-space (Figure 9.2a), especially those facing away, are recognized as socially distant from the current assemblage of users that are present in the P-space

(Figure 9.2b) and therefore can be excluded from the exchange. If people within the F-formation then orient their devices towards (say) the central O-space shown in Figure 9.2c (i.e., using micro-mobility), the system can offer lightweight ways to share device contents across the group.



Third, we consider how people employ *micro-mobility* of physical artifacts to afford nuanced collaboration. The *micro-mobility* of physical artifacts is the fine-grained orientation and repositioning of objects so that they may be fully viewed, partially viewed, or concealed from other persons. For example, Luff and Heath (1998) reported on patient-doctor consultations via paper records and observed how individuals rely on the micro-mobility of these physical artifacts to facilitate communication. The doctor might gesture at a record, orient it such that it invites the patient to view material, or pull it back to give the doctor time to read information. These subtle manipulations of the paper record afford the shifting demands of an activity. Thus micro-mobility posits that orienting an artifact towards other persons, moving it closer to them, or even subtly tilting it towards or away from that person affords powerful and nuanced ways for individuals to share (or not share) information, as well as to fluidly manage the focus of conversation and make their intentions clear to others.

Luff and Heath (1998) introduced micro-mobility and its implications for the design of groupware systems mostly as a cautionary tale, i.e., that new technologies may disrupt the natural movement of artefacts necessary for effective communication. Micro-mobility has since been applied to other areas, particularly the spacing and orientation of digital objects on tabletop systems to denote personal territoriality as well as groupings of objects (Everitt et al. 2006). Other related techniques include adjusting the posture of dual-screen devices (Hinckley et al. 2009), bumping and pouring (Hinckley 2003), stitching (Ramos et al. 2009), and "chucking" content from one device to another (Hassan et al. 2009) (also see our earlier discussion in Sections §2.2.1 and §2.2.2).

The work in this chapter differs from this prior work and extends our interaction techniques from Chapter 7 and 8 in several ways. First, we apply the above concepts of proxemics, F-formations, and micro-mobility in unison. By explicitly incorporating sensing mechanisms for both interpersonal proxemics (via F-formations) as well as device-to-device orientation and identification (via micro-mobility), our system embraces these approaches in a hybrid design. Second, our focus is on cross-device interaction methods, and in particular how to decide when devices should be federated (by sensing F-formations) and how that information should be shared (by sensing micro-mobility).

Our belief is that ubiquitous computing environments can sense social proximity of people in the form of F-formations, the device-to-device assemblages, as well as the micro-mobility of physical devices used by the group of people. Previous efforts have explored techniques for mediating cross-device interaction (we reviewed techniques in Sections §2.2.1 and §2.2.2), sometimes informed by proxemics or a more general notion of proximity sensing, but past work did not apply the multiple lenses of proxemics, F-formations, and micro-mobility in unison. Our design study – described next – helped us to further narrow down the important aspects of proxemics, F-formations, and micro-mobility that we can then leverage in our interaction designs.

9.2 Design Study: Proxemics of People & Devices

We conducted an exploratory design study to gain insight into: (a) how people vary proxemic variables, such as distance and orientation, while interacting with one another; (b) how the less well-studied proxemic relationships of device to people, and device to device vary, and (c) how micro-mobility of devices occurs during people's interactions. Our goal was to inform the behaviors we should look for with our sensing system, as well as to observe naturally occurring gestures and device movements that we might leverage—or need to disregard—for our interaction techniques.

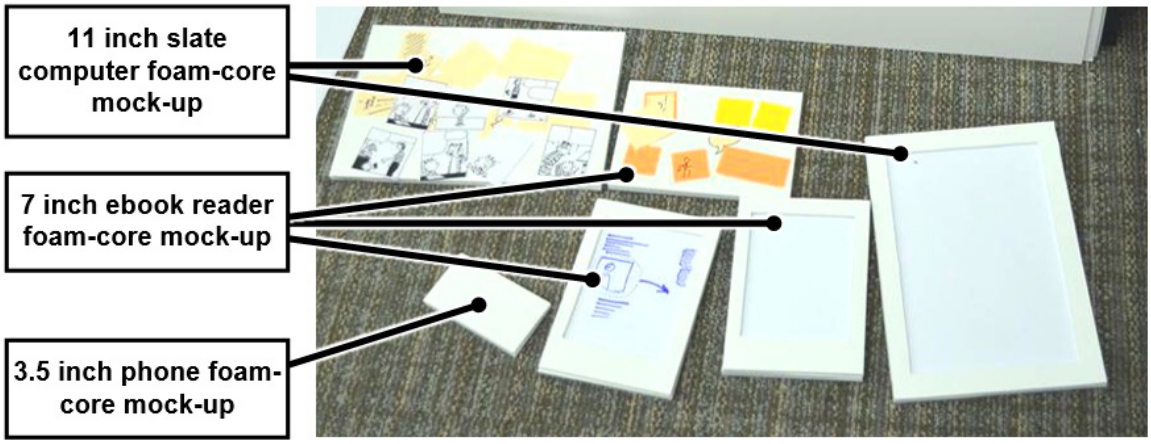


Figure 9.3 Foam-core mock-ups of slates, readers, and phones.

Participants performed joint activities, consisting of both cooperative and competitive tasks, using handheld displays of different sizes (slate, reader, and phone; 30.5 x 21 cm, 15 x 21cm, and 6 x 12cm, respectively). Tasks included:

- *Collaborative activities*: discuss and compare info graphics divided across displays;
- *Competitive activities*: find named locations on a map as quickly as possible;
- *Co-present but individual activities*: memorize a list of items, and then enumerate them while the other participant checks the list; and
- *Confidential activities:* participants read pre-printed "private" emails or financial information.

We videotaped ten paired participants (2 pairs male-female, 3 pairs male-male) as they solved these tasks. Three cameras filmed the participants from different perspectives: two cameras filmed from 45 degree downward angles and were mounted at the ceiling, and one camera filmed from the side and was placed on a table.

The participants worked with 1cm thick foam-core mock-ups of slates, readers, and phones fitted with paper "displays" (Figure 9.3). We used foam-core models so that we could focus on the human behaviors of micro-mobility and F-formations, rather than limitations induced by the physical affordances (e.g., weight, screen glare) of current devices.

From these sessions we observed the following behaviors (B1-B8) that illustrate interpersonal postures, movements, and gestures that people naturally exhibit in such situations:

Figure 9.4 Behaviors from study: use of (a) side-by-side, (b) L-shaped, and (c) face-to-face formations depends on task.

B1. Device as extension of person. Participants treated the devices as part of their person (Figure 9.4a). They were reluctant to bring displays into direct contact (for example noticeable in the distance between the two displays in Figure 9.4a) or to touch one another's displays (though they sometimes did so, albeit briefly). Users clearly exhibited a notion of "personal space" surrounding their devices, but the dimensions of the socially acceptable *device-space* was compressed as compared to normal interpersonal distances.

B2. Distance and shape of F-formations vary by task. In the presence of hand-held devices, the task influenced the choice of formation, reinforcing related findings from the proxemics literature, such as Sommer's (1969) observations of task-dependent preferred seating arrangements. For collaborative tasks that required synthesis of information across displays, users moved close together and adopted side-by-side formations (Figure 9.4a). For parallel individual work, users moved apart, often in L-shaped formations (Figure 9.4b). For competitive or private tasks, users often moved face-to-face (Figure 9.4c), as well as further apart.

Figure 9.5 Behaviors from study: devices shift in and out of the shared o-space as users attend to them.

B3. *Movement of devices in and out of focal zone.* Users extended displays into the o-space defined by their joint gaze when discussing content (Figure 9.5a, right). Displays that are not the topic of communication are moved into p-space at the user's side, with the display often facing inward (Figure 9.5a left, and also visible in the highlighted areas in Figure 9.6 a-d). We observed fluent changes of device position and tilting angles that participants performed to bring devices into the focal zone of the o-space, as shown in Figure 9.5 b and c.

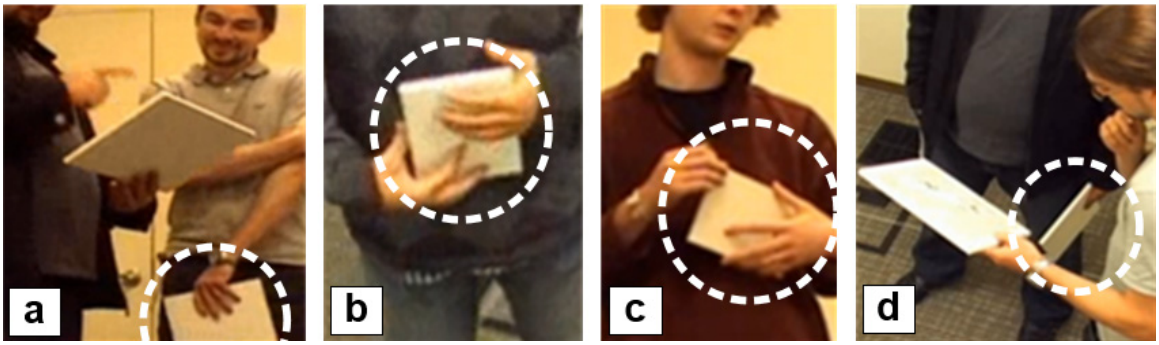


Figure 9.6 Behaviors from study: incidental tilting of devices when not focus of interaction.

B4. *Incidental tilting.* As long as a display was not oriented towards the other person, users seemed largely oblivious to the particular tilt angles of their device (Figure 9.6 a-d). That is, any particular tilt angle is not a definitive cue in and of itself, but rather has to be interpreted in the context of the F-formation.

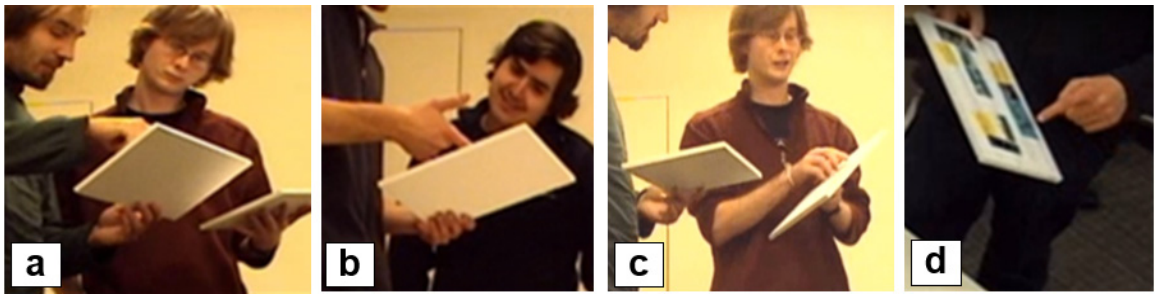


Figure 9.7 Behaviors from study: tilting while pointing to an item on the display.

B5. *Pointing while tilting within the o-space.* People often pointed to an item while tilting the display towards the other person (Figure 9.7 a-d). That is, people often discuss *specific pieces of content* rather than the entire display.

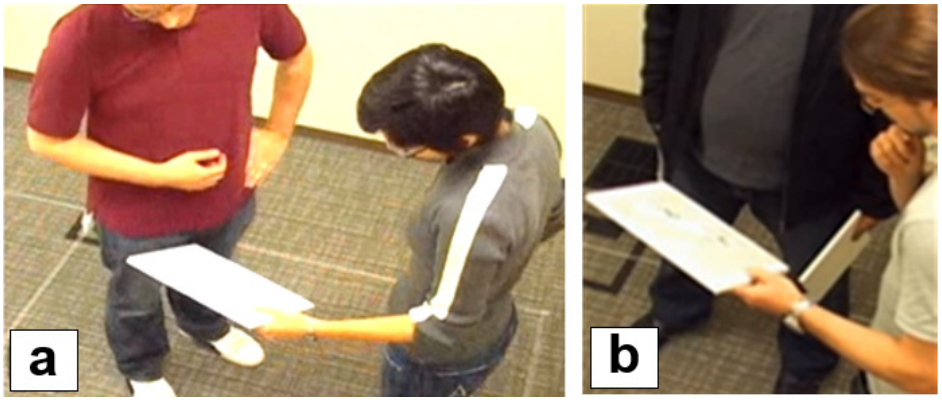


Figure 9.8 Behaviors from study: moving a display into o-space and orienting to a compromise viewpoint.

B6. *Reorientation with gradation in response.* As with micro-mobility (Luff and Heath 1998), we observed subtle tilting and reorientation to nonverbally indicate when people wanted to say something about their display. Larger, more overt movement of the display to a compromise viewpoint was a powerful cue to redirect the other person's attention (Figure 9.8 a,b). That is, people employed a gradation in response, from the subtle to the overt, to suit the current communicative need.

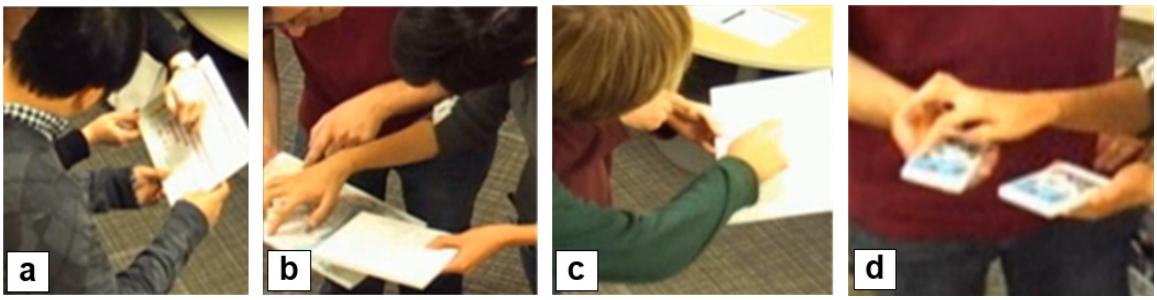


Figure 9.9 Behaviors from study: avoid persistent spatial invasion.

B7. *Avoid persistent spatial invasion.* In the few cases when users gestured at each other's display (Figure 9.9 a-d), such intrusions always remained brief, typically just a second or two. Users often want to indicate referents on the other's display, yet doing so may be uncomfortable. This suggests cross-device interactions should avoid direct interaction with another user's display (such as those proposed by Ramos et al., 2009).

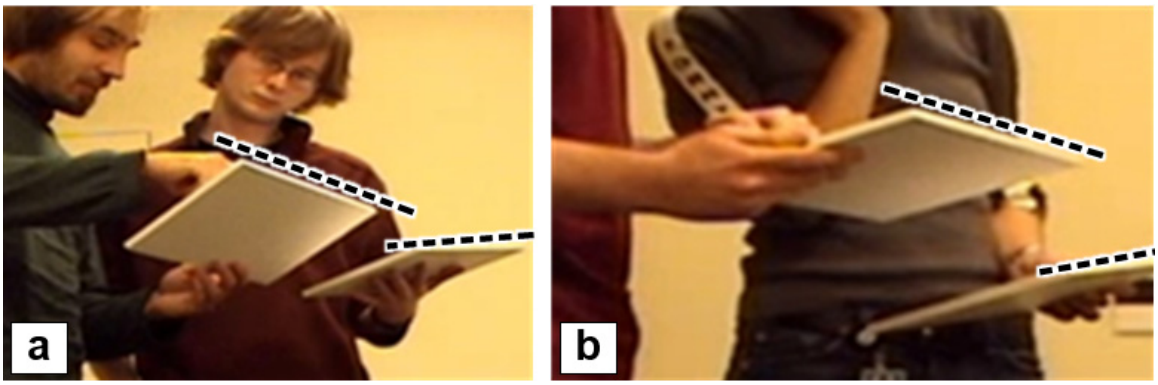


Figure 9.10 Behaviors from study: users tended to mirror tilt angle of devices when collaborating.

B8. *Matching device pose while side-by-side.* When actively working on a joint task, users tended to mirror the tilt angle of one another's slates (e.g. Figure 9.10 a,b), suggesting that equal tilt angles offer a context where it might be useful to reduce the barriers to sharing across devices.

Our intent is not to mimic B1-B8 in our designs per se, but rather to use them as building-blocks, or points of departure for our explorations. Furthermore, some of the above points—while perhaps obvious in retrospect—did not strike us as important

observations until after we had completed our initial implementation and preliminary usability study. We will return to later to these observed behaviours as we discuss the strengths, weaknesses, and possible extensions to our techniques.

9.3 GROUPTOGETHER System

We built the GROUPTOGETHER system to explore our envisioned interaction concepts in action. We implemented a prototype informal information workspace where users can freely arrange collected content (photos, sketches, clippings, etc.) on an interactive canvas. We implemented the application for both tablets and digital whiteboards, and each instance of the application may connect to other devices nearby, as mediated by our sensing of F-formations and device micro-mobility. We chose this application context because collecting and loosely organizing content is a common need of information workers, such as during active reading (e.g., Hinckley et al., 2009).

The GROUPTOGETHER system automatically realizes the federation of devices (i.e., connecting to the devices nearby) by tracking people and devices through a hybrid of on-device sensors as well as extrinsic sensors in the environment. This hybrid sensing demonstrates the tracking of proxemics by means of *low-cost sensing technologies*, which is in contrast to the proxemic-aware systems described in Chapters 7 and 8: there, the Proximity Toolkit used high-end motion capturing cameras for tracking all entities. As we detail later in this chapter, we sense proxemic relationships, people's F-formations, and devices' micro-mobility using: (1) a pair of overhead Kinect depth cameras to visually track and thus sense small groups of people as seen in Figure 9.1b (i.e., their proxemic relationships and F-formations), (2) low-power 8GHz band radio modules to establish the identity, presence, and coarse-grained relative locations of devices, and (3) accelerometers to detect the micro-mobility (e.g., tilting) of slate devices. The resulting hybrid sensing system then supports fluid, minimally disruptive techniques for co-located collaboration by leveraging the proxemics of people as well as the proxemics of devices.

The interaction techniques we designed for the GROUPTOGETHER system facilitate transient sharing, copying, transfer, and reference to digital information across federated devices. These actions suit our application context and allow us to explore various possibilities in order to generate design insights. In particular, the system offers multiple ways to support co-located collaborative activity, ranging from the subtle to the more overt, with various and nuanced semantics of what it means to *share* content. For clarity, in the following we refer to a two-user F-formation involving handheld devices: the user initiating the interaction is the *sender*; and the other person is the *recipient*. We later describe how our techniques work with more than two people, and also with a large display.

9.4 Interaction Techniques

We developed four interaction techniques facilitating the sharing of digital information between the devices of people standing in an F-formation. These techniques are directly inspired by behaviors B1-B8 noted in our design study above, and as we discuss each technique we will refer back to these behaviors to reinforce our rationale, design considerations, and behavioral constraints. Note, however, that B1 and B2 are more meta-observations underlying all our techniques: they validate our preconception that there exists a proxemics of devices (B1), reinforce our design approach (devices must be proximal, but do not have to touch any other device; again B1), and furthermore that a variety of F-formation structures must be properly sensed and supported in a consistent manner by the interaction techniques (B2). Furthermore, while in the following descriptions of specific techniques we primarily focus on micro-mobility gestures, keep in mind that these gestures are only active when the user is currently sensed as standing in an F-formation. We note that we do not claim that these techniques are the only ones possible, or even that they are the best techniques. Their purpose is to illustrate how our study findings can inform and inspire several reasonable interaction methods. Other methods are certainly possible.

9.4.1 Tilt-to-Preview Selected Content

The *Tilt-to-Preview* technique provides an extremely lightweight way to *transiently share* selected digital content across devices. The receiving user can then elect to *grab a copy* of the transiently shared information.

Following the example of behavior B5, *Pointing while tilting within the o-space*, the sender shares a selected piece of content by holding his finger on said content while simultaneously tipping the slate slightly (by a minimum of 10 degrees, Figure 9.11a). This gesture is only active when the tablet is held up within o-space. When triggered it causes a transient semi-transparent representation of the selected item to appear on the display of all devices in the current F-formation (on the right tablet in Figure 9.11a). To make it easy for recipients to identify who is offering an item, an animation slides-in the shared item from the side of the screen where the sender is standing.

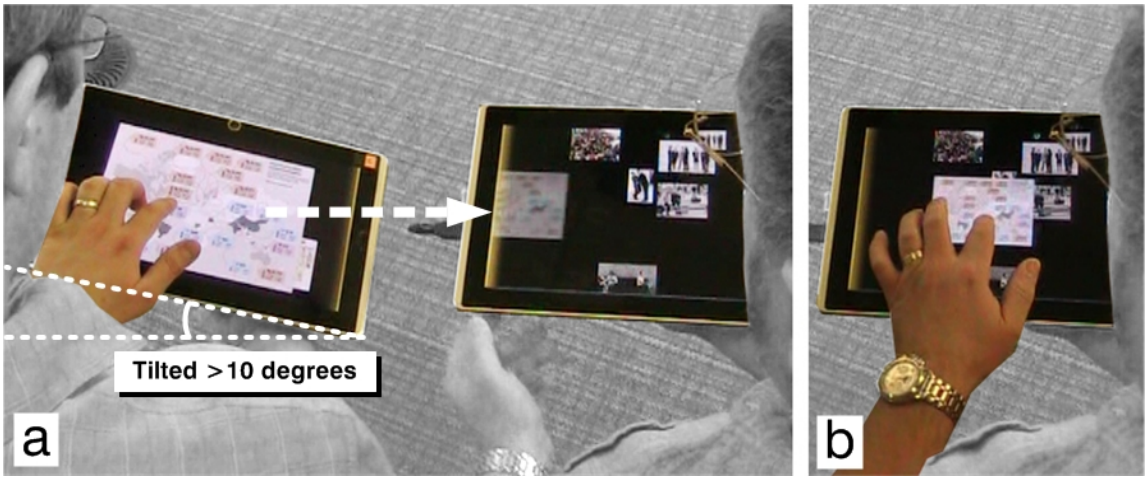


Figure 9.11 Tilting tablet and touching content to transfer temporary copy (a), touch copy on second tablet to keep permanent copy (b).

We employ a tilt threshold of 10 degrees: during pilot testing we found this angle well beyond the *incidental tilting* (B4) that users typically exhibit while holding a slate. Tipping a slate beyond this threshold serves both as a gesture to trigger the behavior as well as a social cue observable to the recipient (and any other people nearby) that the sender wishes to share something. The gesture therefore also leverages observation B6, *Reorientation with gradation in response*, with a fairly subtle overture. Note also that the

recipient can ignore such an overture merely by leaving his tablet down, in p-space (as in B3, *Movement in/out of focal zone*), or accept it by holding the tablet up.

When the sender lets go of the transiently shared content, it disappears from the recipient's screen. However, the recipient can choose to keep a *copy* of the transiently shared content by touching a finger down and grabbing it while it remains visible (Figure 9.11b). This precludes any need for either user to reach onto the other's display, in accordance with observation B7 (*Avoid persistent spatial invasion*).

In dyadic F-formations the sender just tips the slate towards the other user. Alternatively, or if there are more than two people in the F-formation, the user can also tip the slate towards the o-space (i.e., the center of the formation). Again, all participants of the current F-formation can keep the shared content on their own devices by touching the semi-transparent preview on their screen.

9.4.2 Face-to-Mirror the Full Screen

As noted in B6, *Reorientation with gradation in response*, we observed both subtle and overt tipping gestures in the course of our design study. As we observed there, a user can employ a larger tilt as a more demanding nonverbal request to interrupt the current thread of conversation and introduce something else. We also noticed that users often employed larger tilts to show content to their more distant partner in face-to-face formations (B2, but not pictured).

Face-to-Mirror explores how we might provide digital affordances based on these observed behaviors. Using this technique, a person can share the full screen view of the primary digital content displayed on their screen to the other tablets in the social group of an F-formation.

When a person holds their tablet vertically (at an angle larger than 70 degrees), the interactive canvas is mirrored, at full-screen scale, to the display of all other tablets of the group (Figure 9.12). Note that unlike Tilt-to-Preview, this is a pure tilting gesture; the user does not have to touch the screen to explicitly select content. Thus, while the tilting motion is larger, the transaction cost of sharing is potentially lower because the required

action is simply "show your screen to the others." Again, the tilting motion is large enough that incidental tilting (B4) is not an issue with this technique, and furthermore we only observed the *Tilting while pointing* behavior (B5) in the context of more subtle tilting motions, so requiring pointing during Face-to-Mirror would run counter to our design study observations.

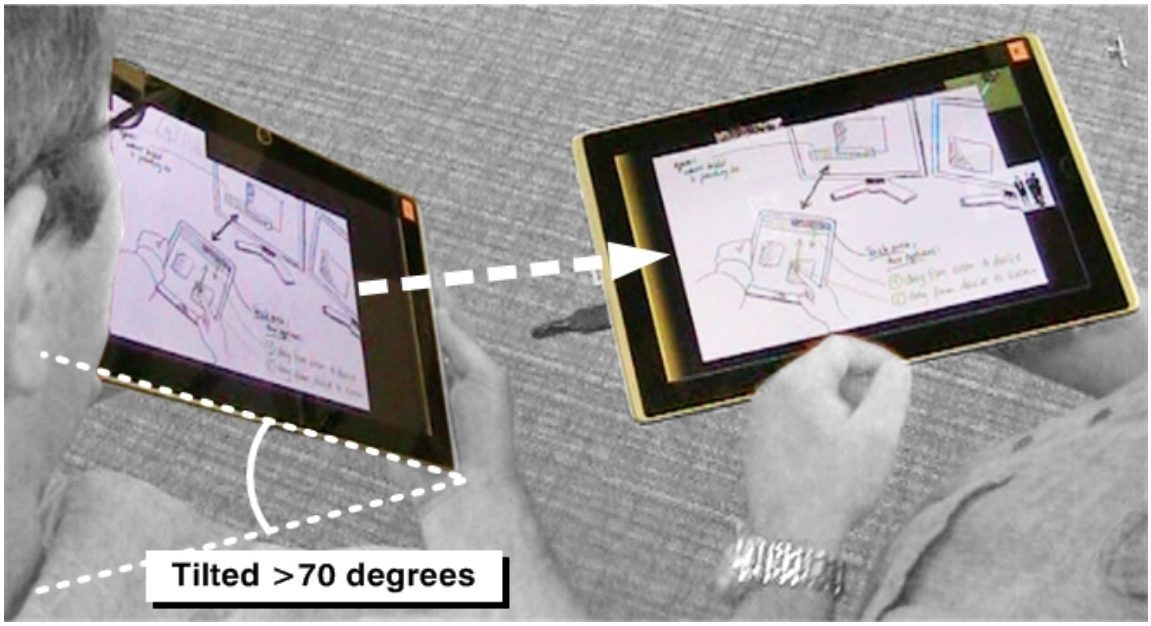


Figure 9.12 Holding tablet vertically (angle larger than 70 degrees) shows a full screen copy on the other tablet.

As with Tilt-To-Preview, Face-to-Mirror begins as a transient sharing technique that ends when the sender moves his slate away from the vertical posture, but where recipients can retain a copy by grabbing the mirrored item.

9.4.3 Portals

The above two techniques both share either a transient representation of an item, or a permanent copy if the recipient touches down and grabs it. To explore an alternative semantic of *transferring* content from one device to another (that is, moving rather than copying content), we implemented the *Portals* technique.

When tilting a tablet (more than 10 degrees) towards the device of any other group member, a tinted edge appears along the shared screen edge of the two slates (Figure 9.13a). By dragging an item through this edge and releasing the touch, the item is (permanently) transferred to the other device (Figure 9.13b). A continuous cross-display animation reinforces the metaphor of the gesture: the content slides off the sender's screen, and slides into the recipient's screen. The recipient can then drag, resize, and otherwise manipulate the content that was transferred to their tablet (Figure 9.13c). As with Tilt-to-Preview, the recipient will only receive items sent through a Portal if his tablet is held up in o-space (as opposed to moving it down to p-space), as observed in B3.

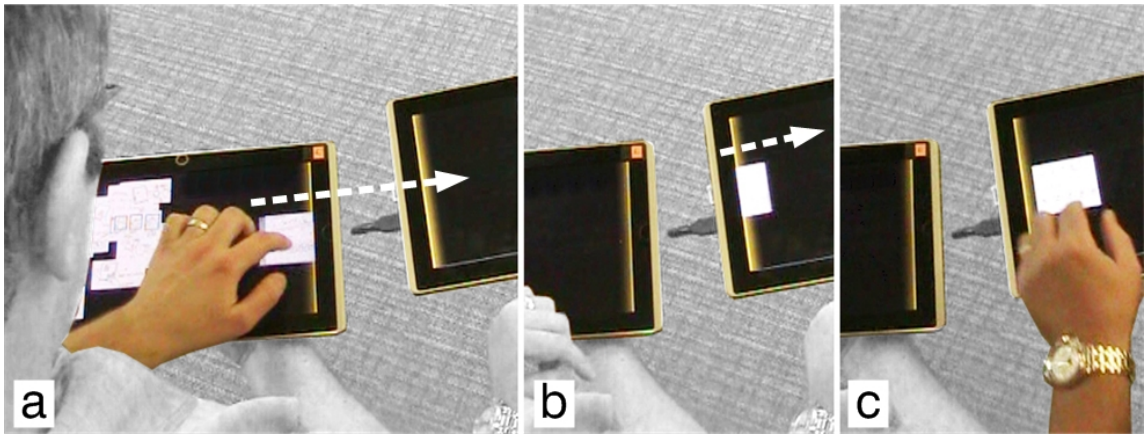


Figure 9.13 Moving content from one slate to another: (a) dragging content through the tinted edge of screen,(b) item moves onto the other slate, and (c) the recipient repositions the item.

Note that in one sense the gesture for Portals is a hybrid of Tilt-to-Preview and Face-to-Mirror: the user performs a fairly subtle (>10 degree) tilting motion (like Tilt-to-Preview) to create the portal, but does not have to touch the screen while doing so. On the one hand this means that Portals may be more prone to incidental tilting (B4). On the other hand, the feedback for Portals (a visually unobtrusive tinting along the matching edge of the devices) as well as the semantics of using the Portal (a transfer only occurs if the user explicitly passes an item through the shared edge of the Portal) means that there is very little impact if accidental activation of a Portal does occur.

9.4.4 Cross-Device Pinch-to-Zoom

Cross-Device Pinch-to-Zoom was motivated in part by B8, *Matching pose while side-by-side*. Here, we explore ways that users can explicitly share items when the slates are *not* tilted (e.g. as shown in Figure 9.14a), but just held together side-by-side in o-space (B3 and B7) and at the same relatively flat angle (B8).

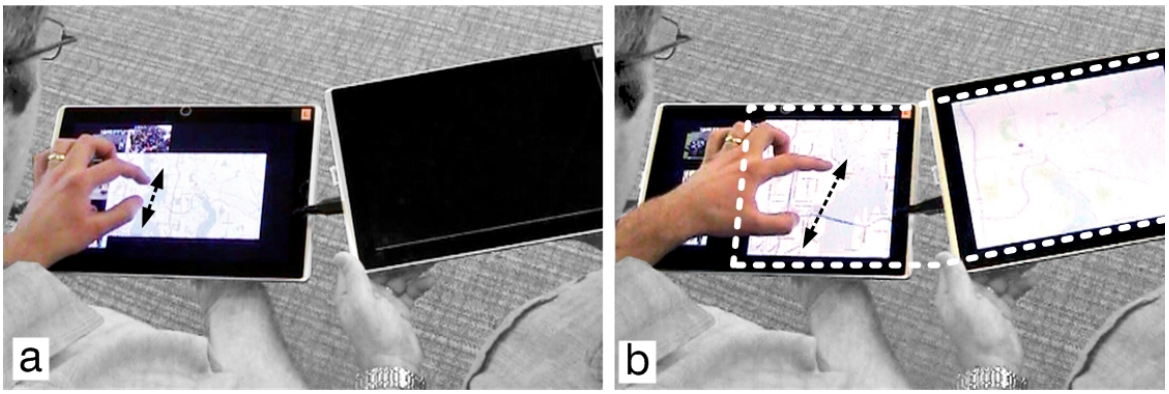


Figure 9.14 Cross-Device Pinch-to-Zoom: (a) the sender begins pinch-to-zoom on the first slate; (b) the zoomed content is displayed on surrounding slates that are part of the F-formation.

This technique allows viewing content across multiple tablet devices when using a pinch-to-zoom gesture. As typical of freeform canvas interfaces, a person can use a two finger pinch gesture to enlarge any content on the screen (Figure 9.14a). But since our technique leverages the GroupTogether system's knowledge of F-formations and the pose of nearby devices, when the sender enlarges the zoomed content beyond the visible area of the slate's display, the remaining content expands onto the surrounding tablets in o-space (Figure 9.14b). That is, while the person zooms in, the content is displayed on the combined screen area of the tablets that form a particular f-formation (i.e., a larger cross-device virtual canvas is created).

9.4.5 Propagation through F-formations

While the above interactions illustrate how our ideas apply to two-person F-formations, the techniques also apply to larger groups (Figure 9.15). For *Tilt-to-Preview* and *Face-to-Mirror*, for example, a person can share content with the entire group by tilting their

tablet towards the center of the formation (i.e., towards o-space) rather than just tilting towards a single person.

Furthermore, we implemented the techniques described above for all three types of F-formations (side-by-side, face-to-face, and corner-to-corner). While it would be possible to support assemblage-specific gestures, we felt that this might not be intuitive to users. However, we do adapt the feedback on the screen (e.g. placement of the tinting indicating an active Portal) to match the spatial arrangement of users.

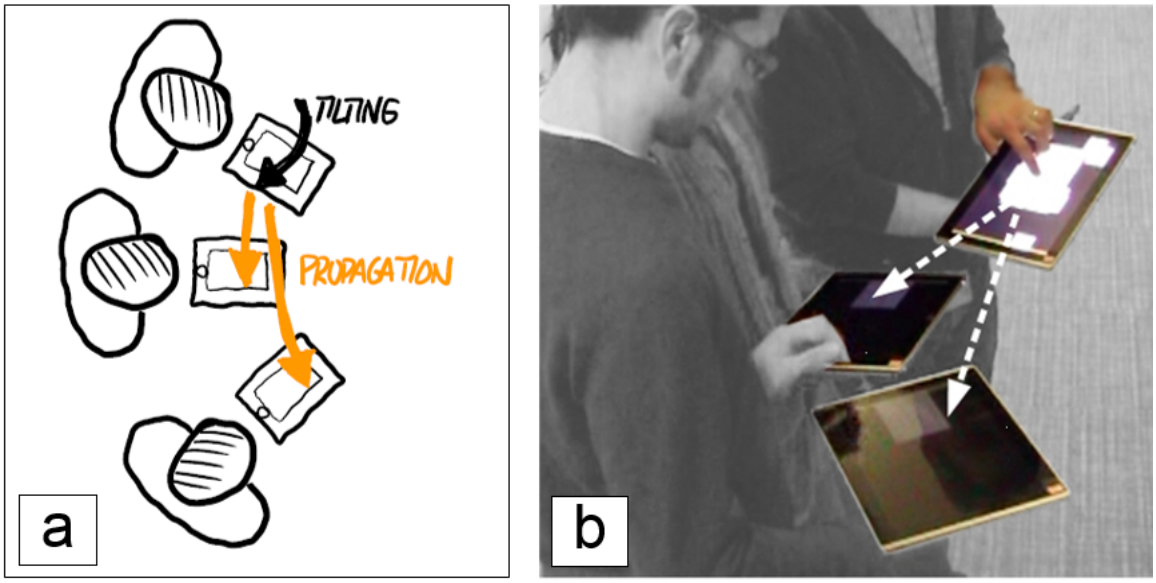


Figure 9.15 Shared content propagates to devices of all members of the current F-formation.

Likewise, users who are sensed as external to the F-formation cannot participate in group interactions, unless of course they move to stand within the group. While it might be interesting to explore various techniques for beyond-arms-reach interaction (Bragdon et al. 2011)—for example, to allow distant persons to send items to an F-formation, or to the digital whiteboard— by the same token in the context of GROUPTOGETHER this would go against the simplicity of the natural human behaviors that we observed in our design study, so we chose not to pursue such mechanisms here.

9.4.6 A Digital Whiteboard as Part of an F-formation

Because F-formations can implicitly encompass fixed or semi-fixed features of the environment (§3.1.8), we included a large screen digital whiteboard in our system as an exemplar.

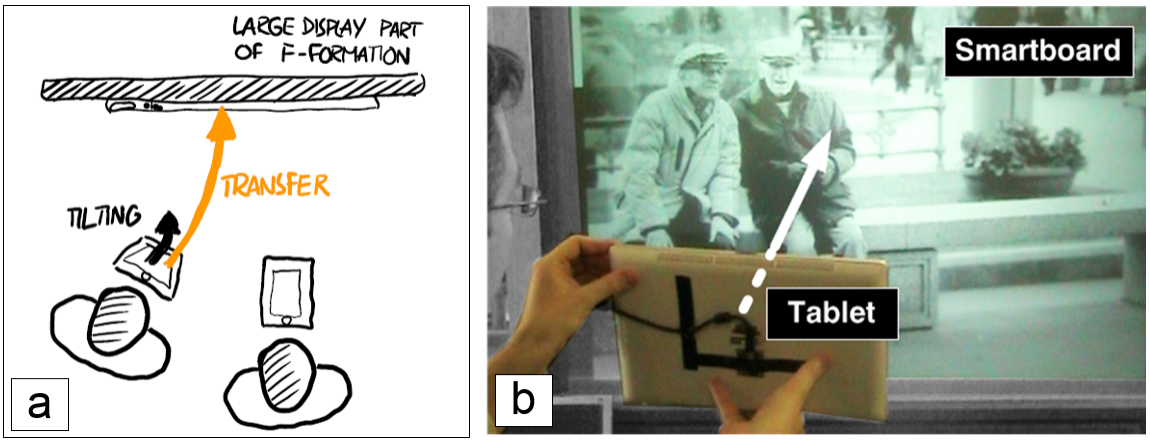


Figure 9.16 Using the Face-to-Mirror technique with a large digital whiteboard display sensed as part of an F-formation.

Users within a sensed F-formation can share content with the digital whiteboard in a manner analogous to sharing content to slates held by other participants. For example, consider the *Face-to-Mirror* technique. A person can hold their tablet vertically towards the large display, and a temporary copy of the tablet's content appears on the large screen (Figure 9.16). Similarly, a person standing next to the whiteboard can use the Portals technique to move content to the large display by first tilting their tablet towards the large display and then begin dragging content onto the edge of the slate facing towards the whiteboard.

Our implementation considers the digital whiteboard in a manner similar to the human participants in an F-formation; that is, it has a particular location and a facing vector. When it falls along the perimeter of p-space it is treated as part of the F-formation, but if it falls outside the huddle, in r-space, it is not. For example, if a circle of users stands in front of the digital whiteboard, with one user's back to the display, and another user performs a Face-to-Mirror gesture, the content will be mirrored to the F-formation but

not to the whiteboard. But if the same circle expands to encompass the whiteboard, then the content would be sent to the whiteboard as well.

9.5 Implementation and Hybrid Sensing Approach

The implementation of the GROUPTOGETHER system using low-cost tracking hardware stands in contrast to our earlier systems from Chapter 7 and 8 that we prototyped leveraging the Proximity Toolkit with high-end capturing technology. In this section we describe the major system components of GROUPTOGETHER that allow us to track the required proxemic measures with our hybrid sensing approach. This includes the hardware and sensors, how we employ 8GHz band radios to associate devices to people, and how we process the raw Kinect depth data to sense F-formations.

9.5.1 Hardware System Components and System Network Architecture

We implemented all techniques on Asus EP121 slates (1.16 kg, 312 x 207 x 17 mm), with two-point capacitive multi-touch screens running Windows 7. To the back of each slate, we attached a Phidgets Spatial 3/3/3 motion sensor and a Qualcomm Short Range Communication Technology (QSRCT) 8GHz band radio module.

We mounted a pair of Kinect depth cameras in the ceiling above our prototyping environment. This enabled us to sense users in a roughly L-shaped active area (Figure 9.17). The wall display was a fifty inch diagonal SmartBoard with single-point optically-sensed touch. We mounted fixed QSRCT radio modules on either side of the SmartBoard to enable triangulation of the slate-mounted radio modules. This enabled identification of specific devices (by their radio identification) as well as coarser-grained proximity sensing of devices that were nearby, but not held by a user within the viewing area of the Kinect cameras.

All the GroupTogether system components were connected via TCP over wireless Ethernet. We implemented a message protocol using Windows Communication Framework (WCF) to transmit all application and sensor state. A central server

maintained a global model of all spatial relationships, and notified clients of state changes.

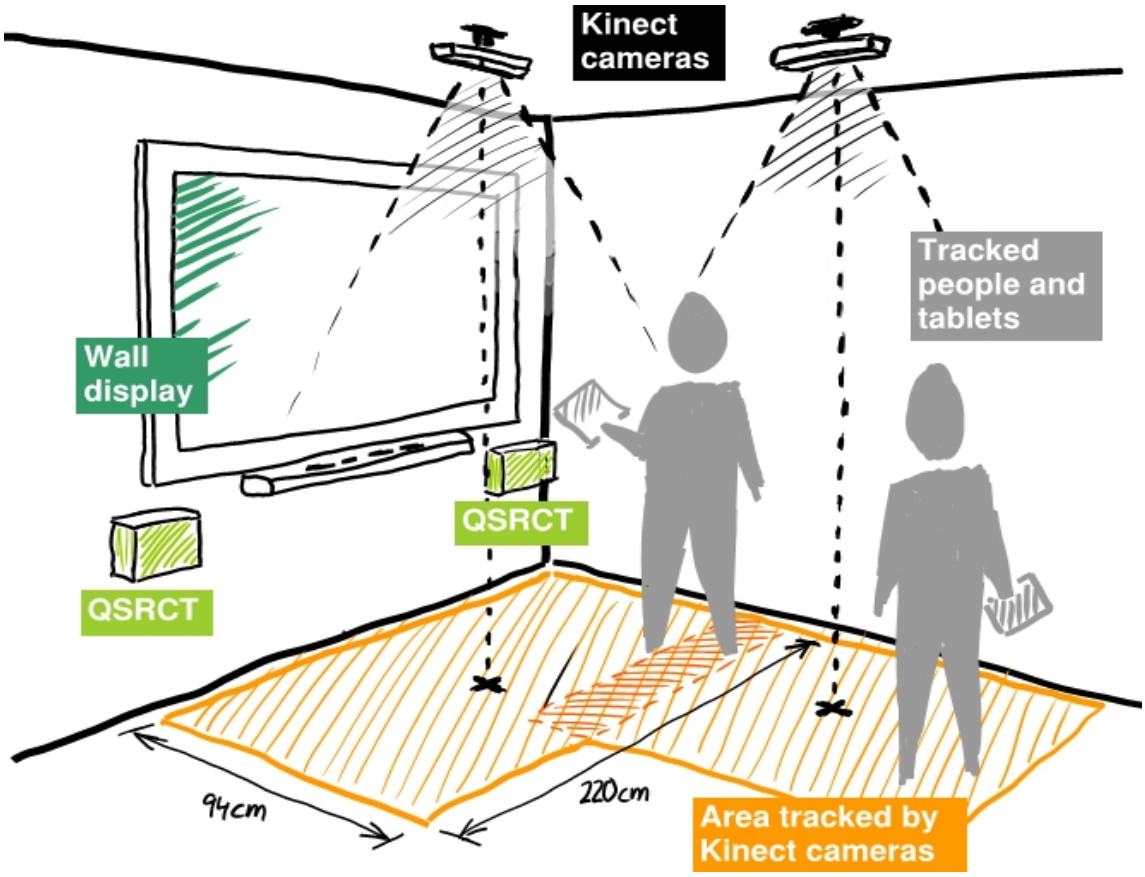


Figure 9.17 Schematic of prototype environment with two overlapping Kinect cameras, and wall display with 8GHz band QSRCT radios (a radio was also attached to the back of the tablets, not shown).

9.5.2 Associating Devices to People via 8GHz Band Radios

The overhead Kinect cameras allow us to track moving blobs that we recognize as people, but this tells us nothing about the *devices* that they carry. We employ the QSRCT radios via *wireless radio signal trilateration* to associate devices to sensed persons.

The Qualcomm Short Range Communication Technology (QSRCT) radio modules transmit signals in the 8 GHz band, which avoids interference with the popular 2.4 GHz WLAN band, but also restricts QSRCT to line-of-sight communication. An advantage of this property is that the signals tend to stay within the social boundaries of meeting

spaces delineated by walls, partitions, furniture, etc. The QSRCT radios have a maximum range of 15m and also *sense approximate distance* between radio modules (10 cm accuracy at 90% confidence). We employ *three point location trilateration* where we put three QSRCT radio nodes at fixed locations in the space around the edges of the area tracked by the depth cameras. The mobile radio sends range-finding requests to each fixed node and we Kalman-filter the resulting measurements to reduce noise. The device location is then the intersection of the three circles.

The server matches the device to a person by assigning the device to the person who is standing closest (as sensed by the Kinect cameras) to the triangulated location. If the distance between the device and all currently tracked people is above a tolerance threshold (currently 1m), the device remains unassigned. To assign all other devices, the process is repeated with the remaining QSRCT radios attached to the tablets.

Trilateration with the QSRCT radios—and matching these radio ID's to the devices held by people tracked via the Kinect depth cameras—allows us to not only associate each device to a particular person, but also to identify a person based on the device they carry (under the assumption that each person carries their own device). Furthermore, when people come and go from the field-of-view of the Kinect camera, the QSRCT radios enable us to keep track of *who* is entering or leaving, as well as to maintain some awareness of whether the person has walked away, or still remains nearby but out of depth-camera view.

9.5.3 Kinect Tracking of F-Formations

We use overhead Kinect depth cameras, mounted in the ceiling and looking downwards, to track spatial relationships between people, including the distance, viewing direction, and relative body orientation between persons. Our system then aggregates these sensed persons into F-formations and classifies the type of assemblage (that is, face-to-face, side-by-side, corner-to-corner, or as singletons not a member of any formation). We note that our implementation assumes users are *standing*, and does not yet handle seated participants.

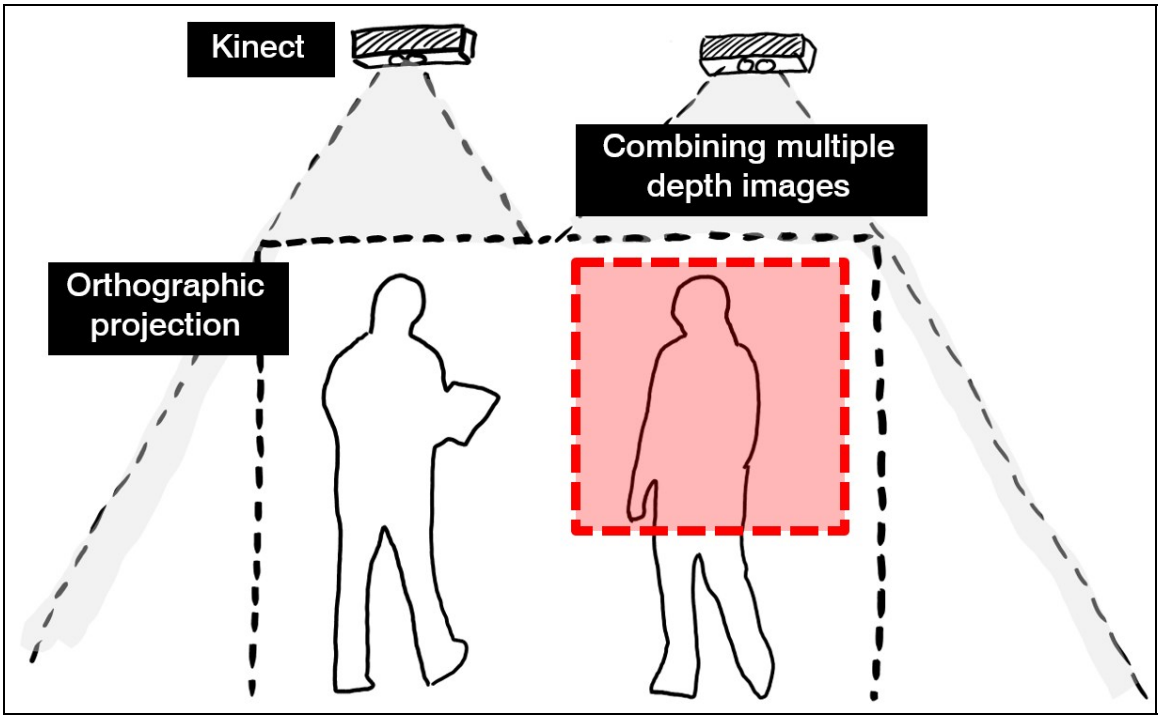


Figure 9.18 GroupTogether tracking with overhead depth cameras, calculating the orthographic projection, and combining the sensing input of multiple cameras.

The Kinect field-of-view of 58.5 degrees, with the cameras mounted 200cm above the floor, produces a 220x94cm tracking area per camera (after clipping the raw image). We arrange the Kinects in an L-shaped sensing region (Figure 9.17). For each depth image we calculate the orthographic projection of the camera view, and then use filters and linear interpolation to remove noise resulting from the partial overlap of the Kinects' structured light patterns (Figure 9.18). After a one-time calibration step to align the separate Kinect depth images we compose the data into a combined depth image.

9.5.4 Processing Pipeline for F-Formation Detection

We process the combined depth image in multiple steps to identify the location, orientation, and formations of people.

1. Filter. We first filter out all connected components too small or too large to be an actual person, leaving just those components most likely to represent a person.

2. Normalize height. Next, the height of all detected people is normalized to match. The algorithm finds the highest point of all detected people identified in the previous step, and shifts the depth values of all remaining outlines of people by the difference in height.

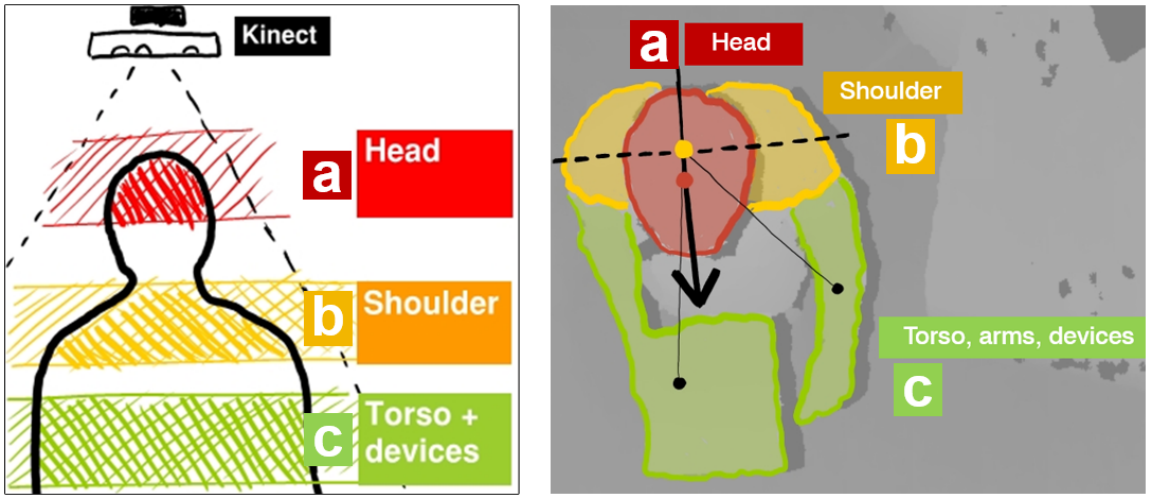


Figure 9.19 GroupTogether tracking and segmentation: standing people as perceived by overhead Kinect depth cameras, with (left) schematic and (right) depth image labeled with recognized (a) head, (b) shoulders, and (c) torso/arms/devices.

3. Detect heads. We assume the topmost depth band represents people's heads (Figure 9.19a). Our algorithm identifies all connected components in this separated 2D image. The results are ellipsoidal outlines of people's heads. We take the major axis of the ellipse as the orientation of the head.

4. Calculate body orientation. The second depth band (Figure 9.19b) includes all regions belonging to people's shoulders. We assign each detected shoulder region to the person it is closest to. We then take the convex hull (since the shoulder is not necessarily a single connected component) to get an ellipsoidal outline of a person's shoulders. The major axis of that ellipse gives us the orientation of the person's body.

5. Determine which way people face. The orientations calculated in steps 3-4 still have a 180-degree ambiguity. To determine which way the user is facing, we take a third depth band that corresponds to the user's torso, including their arms and hands (Figure 9.19c). We then identify which side of the major body axis a person's arms and hands appear on,

as well as which side the head is shifted towards, and take this as the front. We further apply hysteresis to prevent momentary flips in the facing vector due to noise.

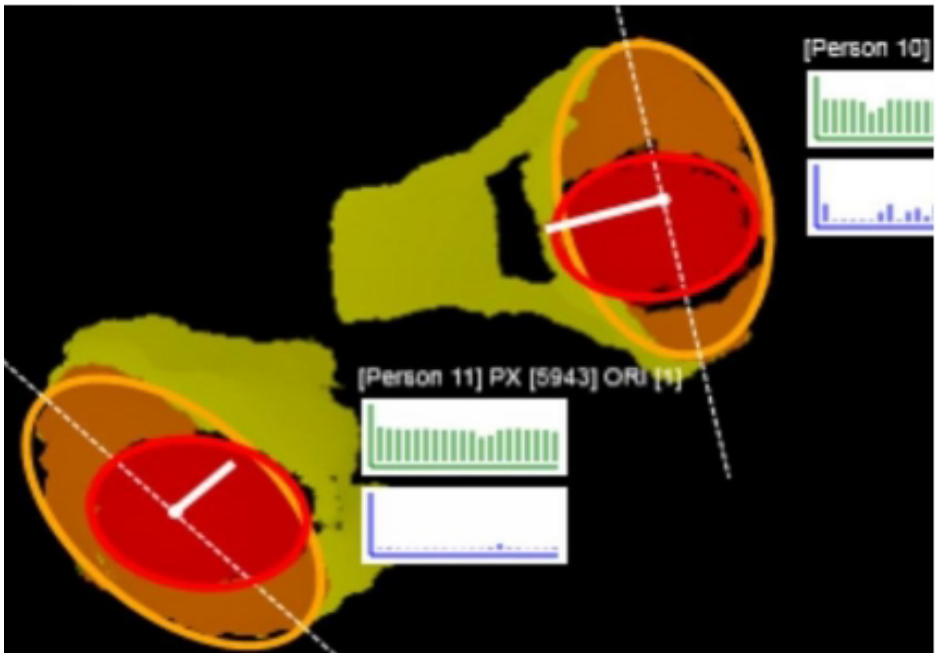


Figure 9.20 Processed raw depth image as segmented by the GroupTogether tracking algorithm. Note the facing vector (white line segments) and principal body axis (dotted lines).

The right side of Figure 9.20 shows the outcome of these five processing steps, where the system recognizes a person's head, shoulders, and hands – and derives the major body axis (dashed white lines in Figure 9.20) and body orientation (thick solid white lines in Figure 9.20)

6. Detect F-Formations. Finally, our algorithm identifies F-formations. Two people can be in an F-formation if: (a) they are not standing behind each other, (b) the angle between their orientation vectors is smaller than 180 degrees (otherwise they would face away from each other), and (c) the distance between individuals is small enough so they can comfortably communicate and their o-spaces overlap. Our algorithm iterates over all pairs of people, calculates the distance and angle between them, and assigns an F-formation type (i.e., side-by-side, L-shaped, face-to-face, or none) based on tolerance

thresholds. Hysteresis prevents frequent switching of detected formations if measurements lie close to a threshold.

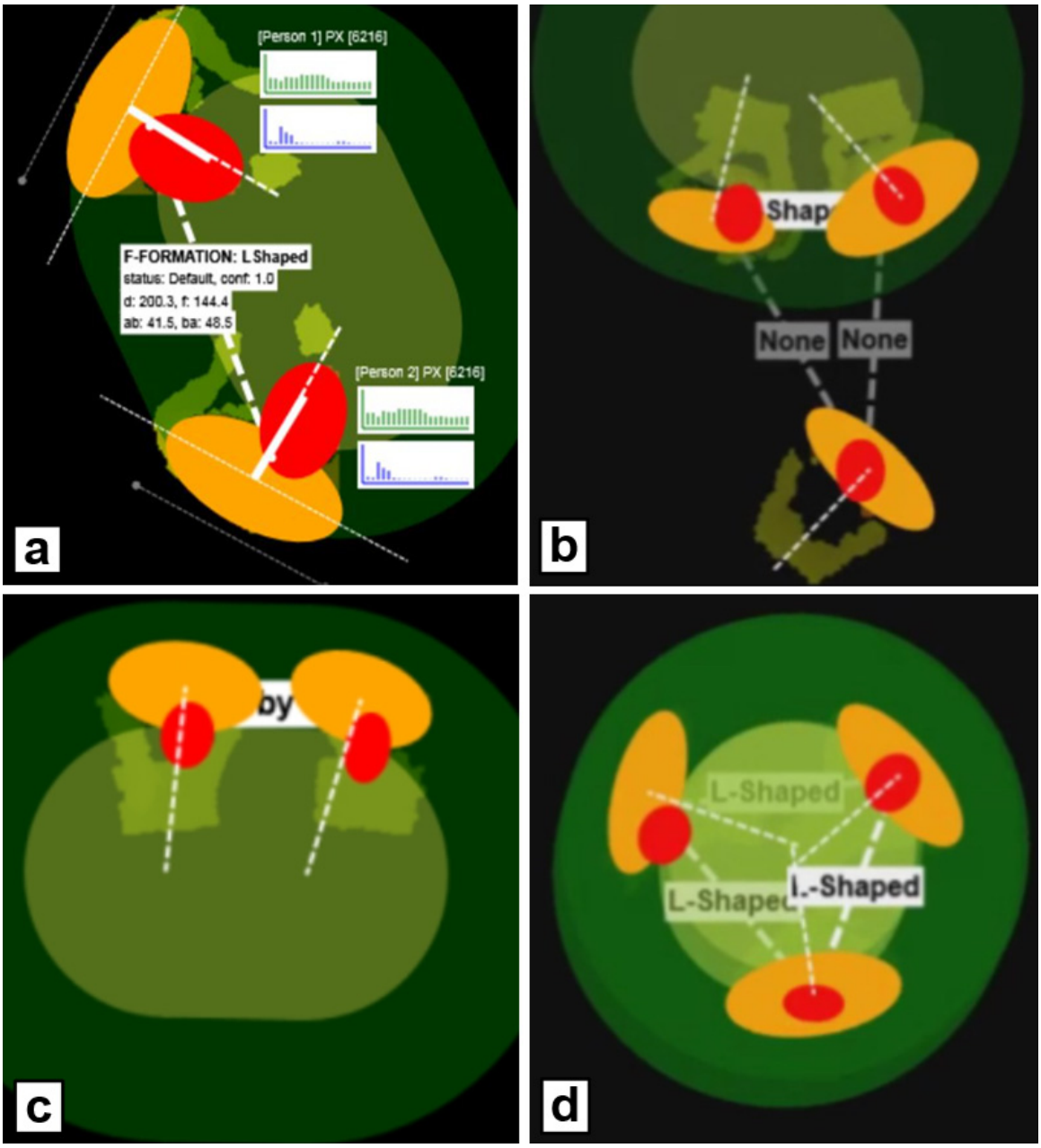


Figure 9.21 F-formations tracked by the GROUPTOGETHER system: (a) two people in L-shape formation, (b) F-formation with a nearby person not part of the formation, (c) two people in side-by-side formation, and (d) group of three people in a circular F-formation (composed out of multiple L-shape formations).

Four F-formations detected by our system are shown in Figure 9.21 (these are screenshots of the graphical debugging output of GROUPTOGETHER). In Figure 9.21a two people stand in an L-shape formation, where they also currently share an artefact in the o-space in front of them. Figure 9.21b shows a similar formation of the two people, with a third person nearby but standing separate from the group. In Figure 9.21c, two people stand side by side, forming an o-space area in front of them where they also positioned two tablet devices. Last, Figure 9.21d shows a group of three people in a circular formation – recognized by the system as composed out of multiple L-shaped formation.

9.6 Informal Evaluation and Reactions from Users

We conducted an informal evaluation in the above sensing environment to gather feedback of people using our techniques in practice. We recruited 6 participants for our experiments. All participants were male, with an age of 29-50, and were part of the same organization but not members of our research group. We paired the participants into groups of two. Each was given a slate running our software. The experimenter explained the four main cross-device interaction techniques. We asked participants to use the technique for basic information-sharing and viewing tasks. They then responded to several 7-point Likert scale questions, discussed what the best and worst thing was about each technique and ranked them for overall preference.

We observed that participants were quickly able to perform each of the four techniques, and that they generally found them intuitive, quick, and easy to perform. Likert responses confirmed their overall positive feedback regarding ease-of-use, where they particularly liked the notion of lightweight sharing across devices. They ranked *Portals* as their favorite technique (and also gave it the highest average rating on a "comfortable to perform" Likert scale question), followed by *Tilt-to-Preview*, *Cross-Device Pinch-to-Zoom*, and *Face-to-Mirror*. However, a number of concerns were raised by our participants or from our own observations, as summarized below.

Physical effort and tilting. Participants felt comfortable performing the smaller tilting gestures, such as that used for *Tilt-to-Preview*. They usually quickly moved the tablet back

to the horizontal position after performing one of the sharing interactions. Yet the size, weight, and fragility of the sensor-augmented slates were an issue, as some participants found it difficult to hold the tablet in one hand while touching an item with the other hand. However, this points more to the limitations of these particular tablets and attached sensors than any particular failing of the micro-mobility idea: recall that people had no problem with this using the lighter foam-core models during the design study.

Participants commented on the "intuitive" use of *Face-to-Mirror*: "this is like… I show it to you" or that there was "even less work to do" than the other techniques (i.e., just hold the slate vertically, with no touch required). Yet participants did raise concerns about the large tilting gesture (› 70 degrees) required for *Face-to-Mirror*, particularly for side-by-side interaction. One user stated "you have to tilt too much" and another reported that the technique "required extreme tilt." Observations of people using the technique also revealed difficulties repeatedly holding the slate in the vertical position; users often changed grips to avoid fatigue and maintain a secure grip on the slate with both hands. This issue can be mitigated simply by relaxing the required angle.

Copy vs. move. The semantics of copy vs. transfer actions adopted by our techniques sometimes caused confusion for participants ("not sure when it copies vs. moves the image"). Our feedback did not make it sufficiently clear which type of "sharing" a particular gesture enabled.

Ownership, control, and handover. For all but the Cross-Device Pinch-to-Zoom technique (for the reasons described above), participants felt in control of what content was shared, transferred and kept between devices and the minimal effort required to do so. Participants commented positively: "it felt natural to either 'take it' or 'leave it'," or that the "receiving user decides to keep [it]." However, a few difficulties arose when users attempted to simultaneously share items (such as by dragging through Portals): "when both of us were trying to transfer images at the same time, it became confusing / difficult."

Structure of F-formations. Most participants agreed that the physical distance between themselves and the second person felt appropriate to enable sharing of data. Participants

mostly stood in a side-by-side formation during the experiment, usually faced slightly inwards towards the o-space. Users also clearly expected the techniques to extend to more than two users ("Will this work in groups?"). Groups of more than two people were implemented in our system, but not tested in this study.

Overall preference. Participants ranked Portals as their favorite technique (and also gave it the highest average rating on a "comfortable to perform" Likert scale question), followed by *Tilt-to-Preview*, *Cross-Device Pinch-to-Zoom*, and *Face-to-Mirror*.

In future studies, we would like to evaluate our techniques more systematically for a variety of F-formation types as well as for larger groups of users (3-5 people). It would also be interesting to explore further variations of the techniques, such as *Face-to-Mirror* with a less extreme requirement for the tilt angle.

9.7 Discussion and Future Work

While our system explored a number of novel ideas, it is clear that much more could be done. For example, we explored a pair of techniques—*Tilt-to-Preview* and *Face-to-Mirror*—that looked at two extremes in gradation of response (B6). But from our observational studies, as well as from the commentary of Luff & Heath (1998), it is clear that people's use of micro-mobility is more nuanced still. Therefore, it would be interesting to explore similarly nuanced techniques that explore this continuum further. These might include both implicit ways of using device tilt and motion for context sensing, as well as explicit gestures or posturing of devices (as explored with the techniques in this chapter) to support finely delineated notions of sharing content.

The observations in our design study, as well as those resulting from our implementation and informal evaluation, strongly suggest that people treat handheld objects as extensions of their person (B1). Yet by the same token the sociological rules governing "object territoriality" are clearly not the same as those governing our physical bodies. To our knowledge the notion of object territoriality has never been systematically studied, which suggests the need for further experimental and observational studies to better

understand behavioral principles that might inform interaction design for micro-mobility and F-formations.

Of course, other sensing platforms currently exist that we could exploit for sensing the presence and location of people and devices – such as high-fidelity Vicon-based tracking we used for the majority of explorations using the Proximity Toolkit. In this chapter, we deliberately wanted to design a practical, low-cost sensing system to illustrate that such systems could realistically be deployed in the near–rather than the far–future. While the implementation we pursued still requires equipment installation within a room (e.g., the ceiling-mounted depth cameras, or the nodes of the QSRCT radios that are fixed in the environment), it could be practically and affordably done within (say) a small dedicated meeting room context such as a breakout room.

More generally, we believe that further investigation of lower-cost, coarser-grained, or more easily deployable sensing modalities to detect micro-mobility and F-formations is important. Exploring the trade-offs inherent in this liminal zone of imperfect sensing—rather than assuming high-fidelity optical sensing of fully tagged persons and devices, for example—is a worthy area of continuing research.

9.8 Conclusion

In this chapter we have described GROUPTOGETHER, a sensing system that jointly brings into play the sociological constructs of micro-mobility and F-formations. We described a design study that enumerates a number of behaviors illustrating how these combined notions of the *proxemics of people* and the *proxemics of devices* surface during joint activity. We then demonstrated how the attributes sensed by our system can leverage these behaviors to enable a number of novel interactions that leverage both the micro-mobility of devices and the social structure of F-formations. We have described how the system senses these interactions by using a combination of motion sensors, low-power radio modules with coarse-grained range finding capability, and overhead Kinect-based depth camera tracking. And finally, we have presented an informal study illustrating user reactions to the techniques.

We believe we have only just scratched the surface of a potentially rich space of such techniques. As mobile devices become thinner, lighter, and more flexible, the opportunities to explore these types of socially-situated interactions should continue to expand. Likewise, systems and techniques that leverage ubiquitous proxemics should benefit from ongoing developments in spatially-aware technologies such as the Kinect depth cameras and QSRCT radio modules utilized in this research. These should be welcome developments for the many contexts where users need to share and discuss digital artifacts while remaining fully engaged in the seamless flow of social exchange with friends and colleagues.

Chapter 10. The Proximity Toolkit in Action

In this chapter[38] we review selected proxemic applications built by students and other researchers. What is important in these examples is how our proxemic interaction framework and the Proximity Toolkit lowered the threshold for students and other researchers to begin their exploration of proxemics within ubiquitous computing applications. As we will see, the easy access to proxemic information through the toolkit and API allowed them to rapidly prototype alternative system designs, leading towards explorations of a diverse design space of proxemic-aware ubicomp systems.

In the following, we first introduce the context and scope of the various projects as a whole (§10.1). Next, we detail each proxemic-aware ubicomp applications built by students, categorized by how they mediate people's interactions with: large displays (§10.2.1), public ambient displays (§10.2.2), non-digital objects (§10.2.3), other devices (§10.2.4), and cross-device operations (§10.2.5). We then review how our framework and

[38] Portions of this chapter are published as:

Marquardt, N., Diaz-Marino, R., Boring, S. and Greenberg, S. (2011) The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the ACM Symposium on User Interface Software and Technology – UIST'2011.* (Santa Barbara, CA, USA), ACM, October 16-18, pages 315-326.

Greenberg, S., Marquardt, N., Ballendat, T., Diaz-Marino, R. and Wang, M. (2011) Proxemic Interactions: The New Ubicomp? In *ACM Interactions, 18(1):42-50.* ACM, January-February. Invited cover story.

Many projects surveyed in this chapter are the result of students' projects in two ubicomp HCI classes in 2010 and 2012. Students gave permission to summarize their work, with attribution. Thus we acknowledge the students' names when introducing each of their projects.

the toolkit functionality fit within these applications (§10.3). Finally, we evaluate the Proximity Toolkit within this context (§10.4).

10.1 Introduction and Overview

Our work in proxemic interactions allowed many developers – graduate students, interns, and colleagues, most from our department – to rapidly design a large variety of proxemic-aware ubicomp systems. Depending on the person and project, they drew from three sources. First, our theoretical foundation (Chapter 3 and 4) operationalizes proxemics for ubicomp interaction design and guides developers exploring this research domain. Second, our three cases studies of person-to-device and device-to-device proxemic interactions (Chapters 7,8,9) – including the gradual engagement design pattern – functions as inspiration for novel interaction designs developers are envisioning. And third, our Proximity Toolkit (Chapter 5 and 6) was invaluable for these explorations: instead of struggling with the underlying low-level implementation details, developers focused on the design of novel interaction techniques and applications that considered people's use of space.

We survey 18 diverse examples throughout this chapter. Our primary goal is to demonstrate and emphasize how our proxemic interactions framework and the Proximity Toolkit:

(a) foster students' and researchers' creative potential by simplifying programming, and facilitates exploration and prototyping of proxemic-aware ubicomp systems (**"low threshold"**, §5.1.2);

(b) enable the application of proxemic interaction concepts to a breadth of diverse application domains, with a large variety of tracked entities – people, devices, other objects – that developers were considering for their system designs (**"wide walls"**, §5.1.2); and

(c) allow the development of comprehensive systems and in-depth explorations of proxemic interactions in ubicomp environments (**"high ceiling"**, §5.1.2).

Similar to what Resnick's et al. (2005) identified as a key aspect of evaluating the use of creativity support tools (of which prototyping toolkits are a part of), we also "*consider the diversity of outcomes as an indicator of success*" (Resnick et al. 2005). In our closing discussion, we will refer back to the three essential properties of effective toolkits (*low threshold*, *high ceiling*, and *wide walls*) (Myers et al. 2000; Resnick et al. 2005) as well as Olsen's suggested criteria for evaluating user interface tools (Olsen 2007) to evaluate our Proximity Toolkit.

Table 10.1 summarizes the 18 projects that will be presented over next three sections. The first column of Table 10.1 indicates two broad types of projects.

Class Projects (CP) and Semester Projects (SP) are those built by students in a graduate ubicomp class in fall 2010 and fall 2012. All students received at least a one-hour tutorial presentation, which included demonstrations of how to program two different examples. Projects ranged in complexity and duration. The students' initial assignment was to create a simple proxemic interface of their choosing, where they had to demonstrate it in the next class. The primary purpose was for students to demonstrate basic competency with the toolkit by building a rudimentary proxemic-aware system. Thus these initial prototypes were built and demonstrated by the students within a week of the tutorial (all CP and SP). The **Semester Projects (SP)** were more fully developed course deliverables, where students continued project development throughout the semester and submitted these as their final course projects. This later case saw students evolve their original project ideas (that they presented as a CP) through a development process that included exploring alternatives designs and refining designs across many iterative cycles.

Thesis Project (TP) or Internship Projects (IP). These projects are the most elaborate examples we discuss, where students worked over a longer period of time (from 6 months for an internship project up to 2-3 years for a MSc thesis) on a particular project.

Project type and number	Application	Monitored relationships (between the following people, devices and objects)	Continuous dist./orient.	Discrete zones	Gradual engagement	Context and application domain
CP1	**Proxemic presenter**	1 person, 1 wall display		X		Office
SP2	**Proxemic visualizations in 3D**	1 remote control, 1 wall display	X	X	X	Science
IP3	**Proxemic-aware pong game**	2 people, 1 wall display	X	X		Games
CP4	**Attention demanding advertisements**	2 people, 1 wall display, 1 tablet		X	X	Retail
TP5	**Proxemic peddler framework**	1 person, 1 wall display		X	X	Retail
SP6	**Ambient alert display**	1 person, 1 wall display		X	X	Office
SP7	**Spalendar**	1 person, 1 wall display	X	X	X	Office
CP8	**Spatial music experience**	2 people, 4 objects/instruments	X			Music
SP9	**Tip-me-lens**	1 mobile phone, 1 wall display	X	X	X	Retail
SP10	**Attentive transparent displays for museums**	1 person, 1 semi-transparent display, 3 objects	X	X	X	Museum
SP11	**Proxemic-aware robot**	1 person, 1 robot		X	X	Robotics
SP12	**ProxemiCanvas workspaces**	2 notebook computer	X		X	Art
CP13	**Proxemic-aware bulletin board**	1 person, 1 wall display, 1 mobile phone	X	X	X	Office
SP14	**Multi-device medical image viewer**	1 tablet, 1 interactive tabletop, 1 wall display	X			Science
SP15	**Proxemic control of ubicomp appliances**	1 mobile phone, appliances (e.g., lights)		X	X	Home
IP16	**ProjectorKit toolkit for mobile projectors**	2 mobile projectors, 2 physical objects, 1 wall display	X	X	X	*(toolkit extension)*

Project type and number	Application	Monitored relationships (between the following people, devices and objects)	Continuous dist./orient.	Discrete zones	Gradual engagement	Context and application domain
IP17	**Focus+Context mobile projectors**	2 mobile projectors, 1 wall display	**X**	**X**	**X**	Office
TP18	**OptiTrack plugin** (University of Konstanz)	-	**-**	**-**	**-**	*(toolkit extension)*

Table 10.1 Overview of built projects and prototypes: project type (CP = course project, SP = semester project, TP = thesis project, IP = internship project) and sequential number, application name, the proxemic relationships they monitor, the kinds of proxemic variable(s) and/or pattern they consider, and the context and application domain.

The next two columns of Table 10.1 list: (2) the name of the application, and then (3) the monitored relationships between people/devices/objects. The subsequent three columns indicate whether the application makes use of (4) continuous distance and orientation information, (5) discrete zones, or (6) the gradual engagement pattern. Column 7 describes the context and domain where the application was designed for and where it would be typically used (e.g., office, museum, retail). We discuss details of this table in Section §10.4.

10.2 Explorations of Proxemic-Aware Ubicomp Systems

In the following five sections we survey these applications, where we focus on explaining the introduced interaction concepts, as well as the applications' use of proxemic dimensions, the gradual engagement design pattern, and the toolkit API.

10.2.1 People's Interaction with Large Displays

This first section describes projects considering proxemic relationships between a person and a large display to mediate the interaction with ubiquitous technology.

Related to our proxemic-media player and the demonstrated interaction concepts in Chapter 7, some students built prototypes exploring people's interaction with large displays. What is in particular interesting is how all these projects – while identical from a technological point of view in regards of the devices they use (large display + tracked person) – apply the proxemic interaction concepts in novel ways to a diverse set of applications and activities: *giving presentations*, *exploring scientific data sets*, and *playing games*.

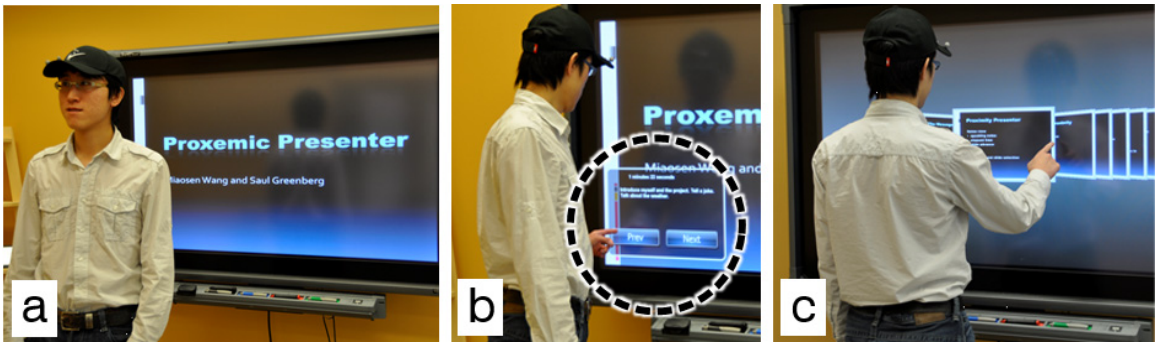


Figure 10.1 Proxemic Presenter (Miaosen Wang).

CP1 - Proxemic Presenter (Miaosen Wang) is a presentation tool (e.g., akin to Microsoft PowerPoint slides) that reacts to the presenter's position relative to a large display. The focus of this rapidly-created project was on two specific capabilities: making it easier for a speaker to access their speaking notes; and making it easier for them to move through slides, and to jump over slides by selecting one from a set of thumbnails. The Proxemic Presenter exploits distance, orientation, and identity (to discriminate the speaker from others). The sequence in Figure 10.1 shows how it works. When a speaker is facing the audience, the presentation fills the screen as expected (Figure 10.1a). When the speaker stands at the side of the screen and faces towards it, a small but readable pane containing speaker notes, timing information, and next/previous controls fades into view next to the speaker (Figure 10.1b). As he looks back towards the audience, the notes pane fades away. The notes pane also follows the speaker: if the speaker moves to the other side of the display and looks towards it, the pane appear at that side. If the speaker moves away from the display and then looks towards it, the notes pane does not appear. This is because the speaker is too far to read them, and showing large notes would be distracting to the audience. If the speaker shields the

display from the audience by standing near and at the center of the surface, a scrollable deck of slide thumbnails appear, allowing the speaker to rapidly switch to any slide (Figure 10.1c).

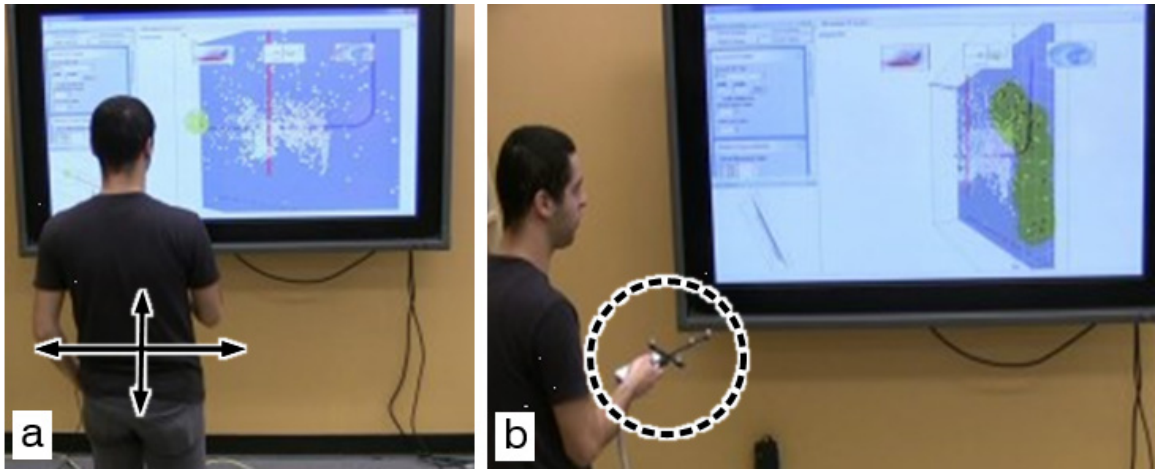


Figure 10.2 Proxemic Interactions with a 3D visualization system (Ahmed E. Mostafa)

SP2 - Proxemic Interactions with a 3D Visualization System (Ahmed E. Mostafa) allows a person to interact with graphical visualizations displayed on a large display (Mostafa et al. 2013) (video figure demonstrates the interactions[39]). It includes a proximity-based immersive navigation technique: it tracks a person's location (via the proxy of a remote control the person is holding in his hand) that directly controls how one navigates through the 3D data set shown on the display (Figure 10.2a). The system uses the proxemic properties of distance and orientation between a person and the large display to let the person coarsely navigate the 3D data set. Furthermore, as an example of the progressive reveal technique, distance and orientation also affect the details shown (i.e., more details are shown when the person orients towards the display and moves closer). This method mimics how people can walk around a 3D object to view it from different angles, and how they can see greater detail as they move closer. A person can also perform explicit gestures – as tracked with the Proximity Toolkit – with the remote control to manipulate the currently displayed data set (Figure 10.2b).

[39] http://grouplab.cpsc.ucalgary.ca/Publications/2013-Microseismic-CHIVideo

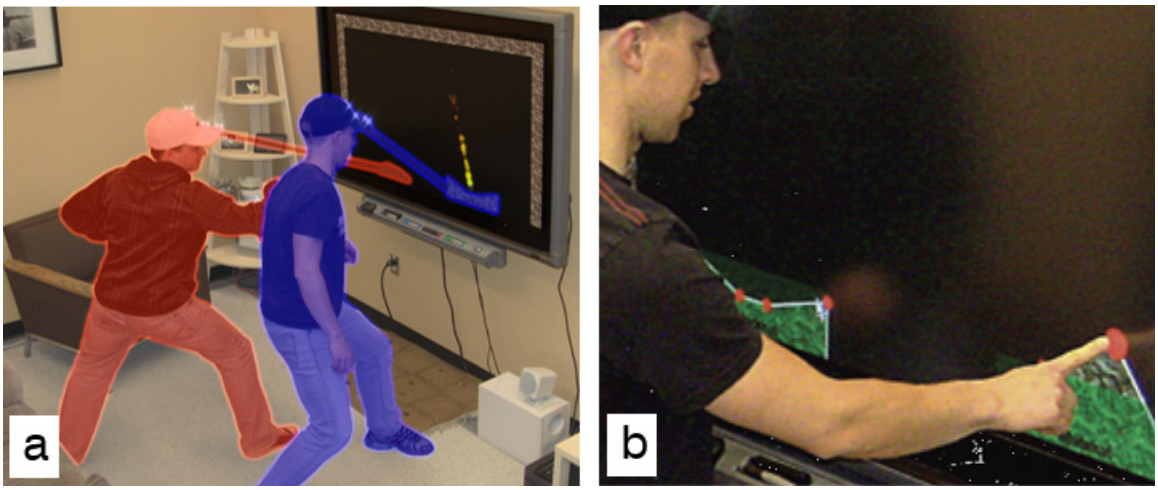


Figure 10.3 Proxemic pong game: (a) two players playing the game and (b) configuring game and changing paddle size when in close proximity to the screen (Till Ballendat).

IP3 - Proxemic-aware Pong (Till Ballendat) is inspired by Atari's Pong game. It illustrates how proxemic interaction concepts can serve as a guiding strategy for the design of interactive computer games, where the proxemic relationships between the game's visual screen and one or multiple players can affect the gaming experience. Proxemic-aware Pong reacts to distance, orientation, motion and identity, where identity just distinguishes between different players. In standby mode (which displays a splash screen showing the game title), Proxemic Pong recognizes when a person enters and stand in front of the screen: it creates a paddle for that person, and starts the game. That person then controls the paddle for bouncing the ball with their body by facing forward and physically moving left and right in front of a large screen. When a second person enters the space and stands in front of the display, a second paddle is created and the game continues via turn-taking (as seen in Figure 10.3a). To penalize the player who interferes with the active player by standing in their way, Proxemic Pong grows the active player's paddle to make it easier to hit the ball.

Like Wii games, Proxemic Pong introduces exertion element into computer game play. Initially, the player's motion matches the paddle's motion. As game play continues, the system increases the ratio of the physical distance that needs to be covered to move the

paddle, while also increasing the speed of the ball. This means that people have to move further and faster to hit the ball.

Proxemic Pong also exploits front to back motion. If a player moves very close to the display, the game automatically pauses; control points appear on the paddle allowing that person to adjust the paddle shape by direct touch (Figure 10.3b). If a player moves backwards and sits on the couch (i.e., the player becomes an observer), their paddle disappears and the game continues in single player mode. When both sit down on the couch or move away, the game pauses.

10.2.2 People's Interaction with Public Ambient Displays

As a special case of people's interaction with large interactive displays, some projects explored in particular novel interaction strategies for public ambient displays. Such ambient displays provide (often non-critical) information to a person while remaining in the periphery of their attention (Mankoff et al. 2003). The following three applications demonstrate how proxemic interactions can drive the interaction of a person with *advertisements*, *event notifications*, and *calendars* displayed on such public ambient displays.

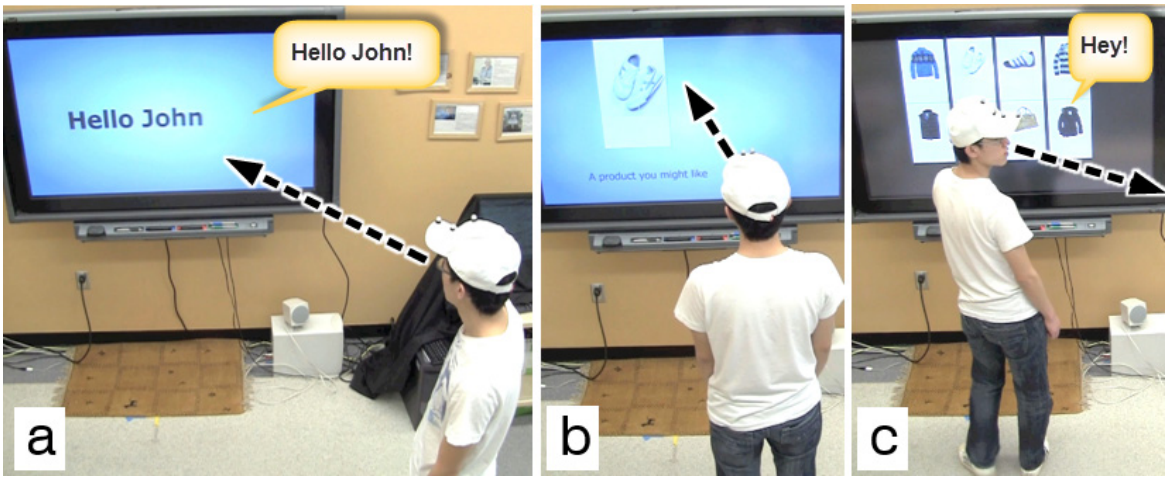


Figure 10.4 Attention-Demanding Advertisements (Miaosen Wang).

CP4 - Attention-Demanding Advertisements (Miaosen Wang) explores how future advertisement displays might try to grab and keep a person's attention. A digital advertisement board (implemented with a Smartboard interactive wall display) reacts to

the presence, distance, identity, orientation, and movements of a nearby person. The advertisement board is designed to attract the attention of a passer-by by welcoming them by calling out their name (via visual and audible feedback, see Figure 10.4a). The display then shows items of interest to them as the passer-by faces the board and looks at the screen (Figure 10.4b). The board persistently tries to regain the attention of that person if they look or move away by playing sounds and flashing the background color (Figure 10.4c).

TP5 – Proxemic peddler framework (Miaosen Wang). Miaosen Wang's class project described above turned into the MSc thesis project (Wang 2012)that further explored attention-demanding advertisements. This required significant changes over the original designed prototype, resulting in iterative refinement of a new prototype – all facilitated by both the theoretical foundations of proxemic interactions and the technical building blocks of the Proximity Toolkit. His **Proxemic Peddler framework** describes the various attentional states of a passer-by in relation to a public advertisement display (Figure 10.5), and explains strategies to capture and preserve the attention of that person (Wang et al. 2012).

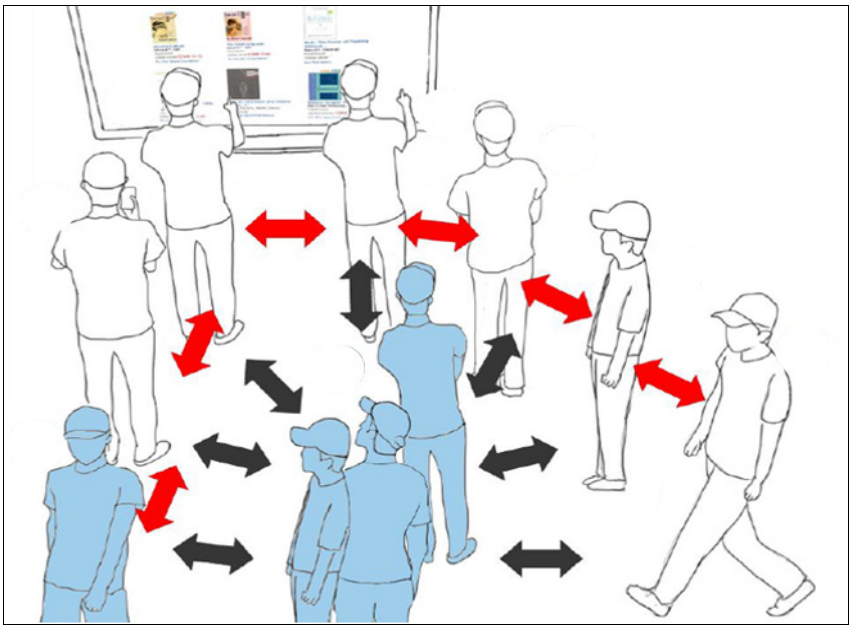


Figure 10.5 Proxemic Peddler framework and a passerby in different attentional states with respect to a public display (Wang et al. 2012).

SP6 - Ambient Alert Display (Fereshteh Amini) is a proxemics-based visualization of sensor data (e.g., the structural sensor readings of a building). As one example of applying the gradual engagement pattern in practice, the system changes the content of the sensor data visualization depending on people's distance to the display. A person in the ambient zone sees coloured circles on the screen indicating past events, where different colors (green, yellow, red) represent the urgency of each event (Figure 10.6a). A person can read more information about each event when moving closer to the display, as the display then shows labels describing the events and the sensor readings (Figure 10.6b). Finally, the person sees detailed graphs and statistics when directly in front of the screen (Figure 10.6c).

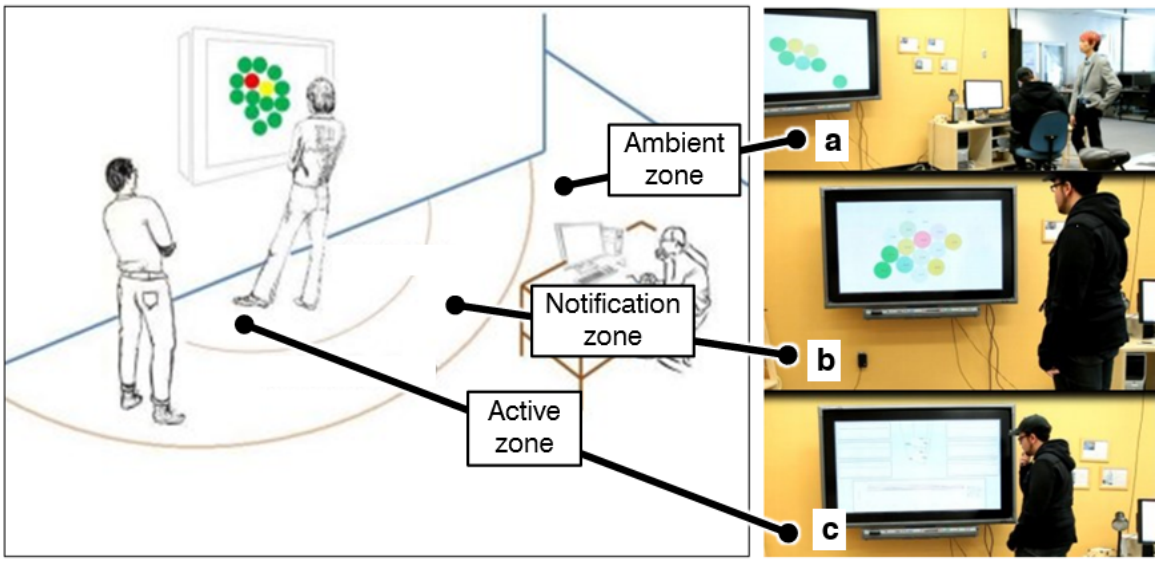


Figure 10.6 Three zones of interaction with the ambient alert display (Fereshteh Amini).

SP7 - SPALENDAR (Xiang Anthony Chen) is our final example of a public ambient display. It shows spatial visualizations of scheduled meeting locations of one's collaborators overlaid on a geographical map (Chen et al. 2012). Part of SPALENDAR 's interaction sequence also follows the stages of the gradual engagement pattern. A person starts by watching the display from afar (Figure 10.7a') where it shows simple flows of one's collaborators as they move between their events (Figure 10.7a). When the person approaches the display, more information about the collaborators' identities is revealed

(Figure 10.7b). The identity of the person approaching the display directly affects the amount of information shown. Finally, when directly in front of the display (Figure 10.7c'), a person can use direct touch gestures to reveal detailed calendar information on demand (Figure 10.7c). A person can use hand gestures to, for example, adjust the time/date of the current information shown, or add new events to the calendar.

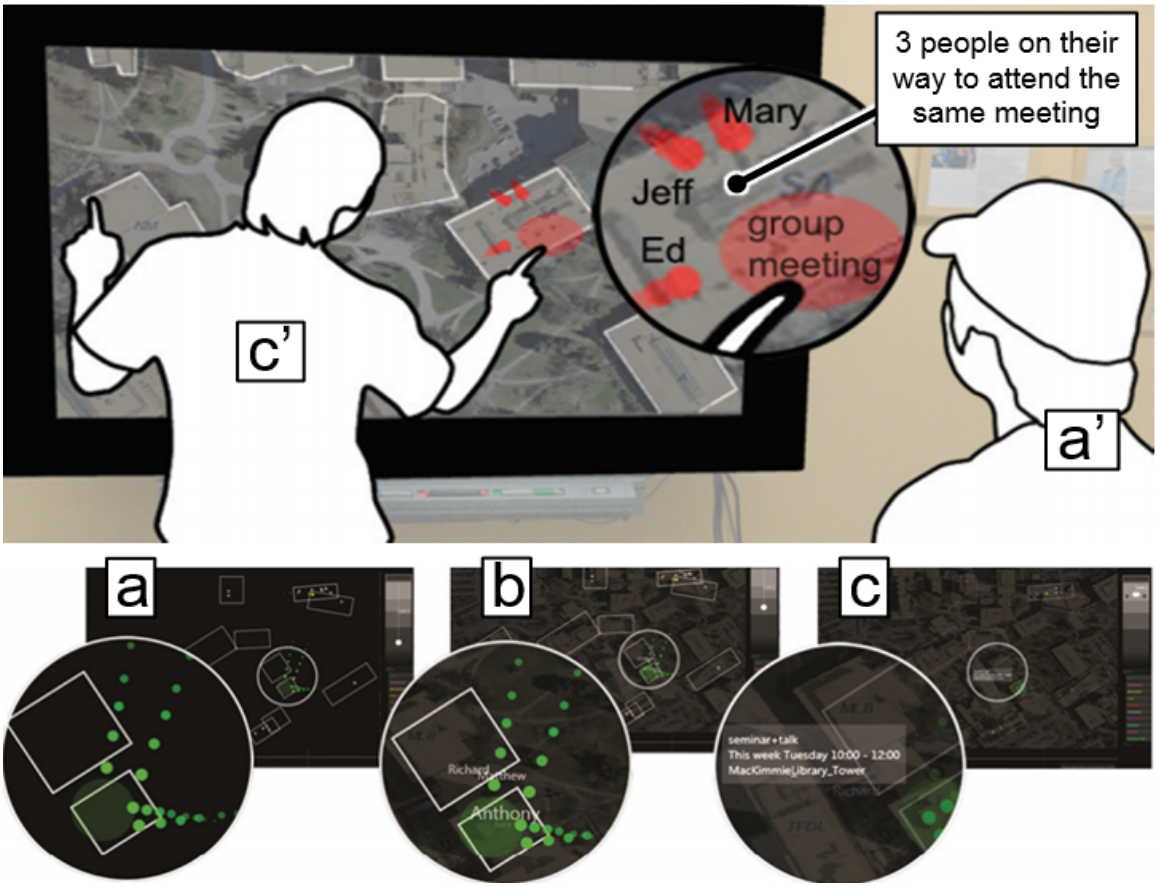


Figure 10.7 Interaction with SPALENDAR (modified from Chen et al., 2012).

10.2.3 Interaction with Non-Digital Objects

Another set of projects explored the design of systems that consider people's or devices' proxemic relationship to surrounding non-digital objects, as shown with the following two applications built by the students in one of the ubicomp classes: an *interactive music installation*, and a *mobile shopping recommender system*.

CP8 - Spatial Music Experience (Matthew Dunlap) is an interactive music installation. The kinds of synthesizer sounds generated by the computer and their volume is determined by the proxemic relationships of people and physical objects in the space (Figure 10.8a). Generated sounds react fluently as people move and perform gestures in the space, and when they grab and move physical objects (e.g., the drums in Figure 10.8b).

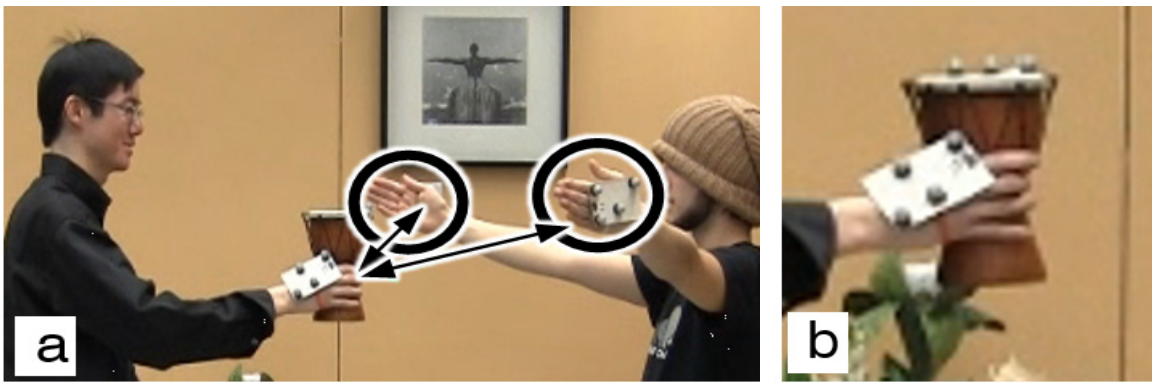


Figure 10.8 Spatial Music Experience (Matthew Dunlap)

SP9 - Tip-me-lens (Bon Adriel Aseniero) is a mobile recommender system used to inform a customer's buying decisions while browsing products on a shelf (Aseniero et al. 2013) (video available[40]). It uses augmented reality techniques to superimpose additional information on the phone's display (Figure 10.9a) about the store items on the shelf in front of the person (Figure 10.9b). Depending on a person's distance to the shelf, the information displayed on the screen varies. Following the gradual engagement pattern, more detailed information about the products (e.g., nutritional information of food items, Figure 10.9c) is revealed once a person approaches and moves closer to the display. The built prototype used both the Proximity Toolkit to track devices, and an augmented reality toolkit to capture the exact position of the products on the shelf (via printed 2D markers as shown in Figure 10.9).

[40] http://grouplab.cpsc.ucalgary.ca/Publications/2013-TipMeLens-Report2013-1040-07

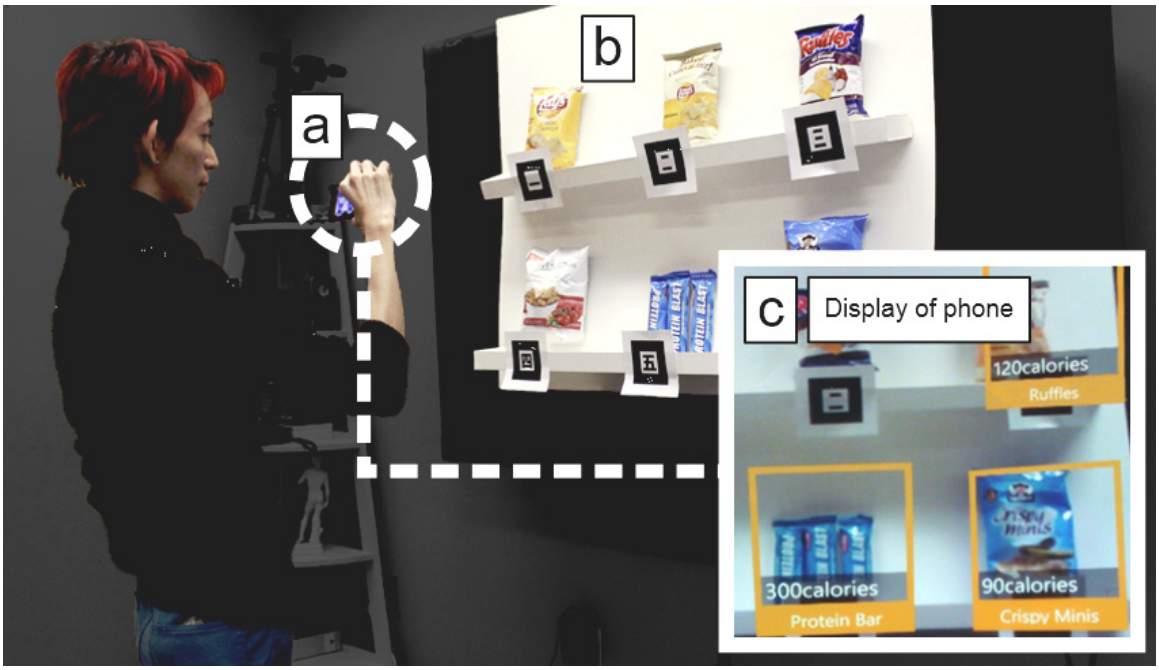


Figure 10.9 Tip-me-lens mobile recommender system (Aseniero et al. 2013)

10.2.4 Interaction with Other Devices

Students also applied proxemic interaction concepts to the design of interactive systems using hardware and devices going beyond the large screens, mobile phones, or tablets. Two examples illustrate this: the *attentive transparent displays for museums*, and a *proxemic-aware humanoid robot*.

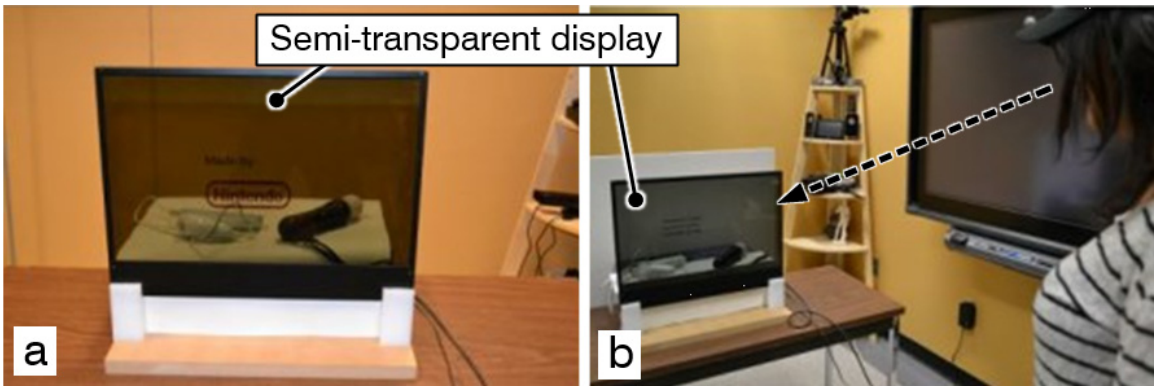


Figure 10.10 Attentive transparent display for museums (Jiannan Li).

SP10 - Attentive Transparent Display for Museums (Jiannan Li) is a semi-transparent computer display (Figure 10.10a) positioned in front of museum artefacts that can

display further information about the shown physical artefacts. Depending on a person's position and their distance to the physical artefacts behind the semi-transparent display, information about these artefacts is progressively revealed and shown on the screen. The position of the displayed information is determined by ray casting from the viewer's eyes towards the physical artefact, where the ray's intersection with the transparent display sets the position for the shown information (Figure 10.10b). That is, the position the displayed information tracks the viewing angle of the person. A person can perform simple hand gestures (tracked through gloves with attached IR markers) to override actions performed by the system.

SP11 - The proxemic-aware humanoid robot (Setareh Aghel Manesh) applied the gradual engagement pattern to the behaviour design of a humanoid robot. The robot crudely mimics human to human proxemics, where it changes its behaviour and determines its actions according to the distance of a nearby person. The robot tries to maintain a certain distance to the person, walks after them when they turn around and the person walks away, and waves its hand when a person looks at the robot. The system divides the space around the robot into three discrete zones (near, medium, and far; see Figure 10.11) that trigger certain behaviours of the robot once a person enters any of these zones.

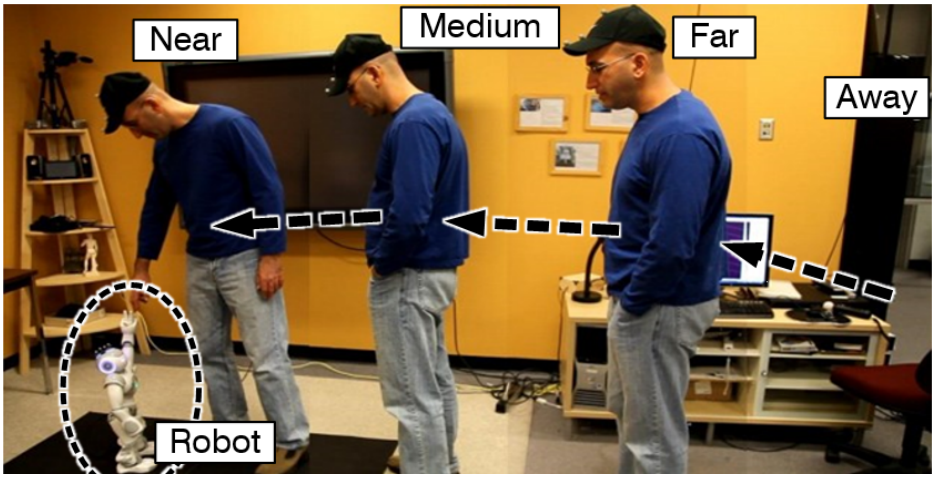


Figure 10.11 Four discrete distance zones determining the interaction with the proxemic-aware humanoid robot (Setareh Aghel Manesh).

10.2.5 Cross-Device Interactions

The final four applications consider the proxemics of devices (related to our interaction concepts of Chapter 8 and 9) to mediate people's interaction with *sketching applications*, *digital bulletin boards*, *volumetric data sets*, and *appliances in the home*.

CP12 - ProxemiCanvas (Xiang Anthony Chen) is an interactive drawing application in which drawing canvases displayed on people's portable computers gradually merge as a function of proxemic relationships between people and devices. For instance, from close to far distance, this ranges from: merged workspaces when very close (i.e., a merged canvas in Figure 10.12a); awareness of other people's work when sitting nearby (i.e., seeing another person's drawing cursor, Figure 10.12b); gradually less shared information when moving away (i.e., the edge of the other person's canvas is visible in Figure 10.12c); and finally no shared information when turning away (e.g., when people are sitting far apart or back to back). Technically, the system tracks the position, identity, movement, and orientation of both people's notebook computers and maps those values to various degrees of coupled workspaces.

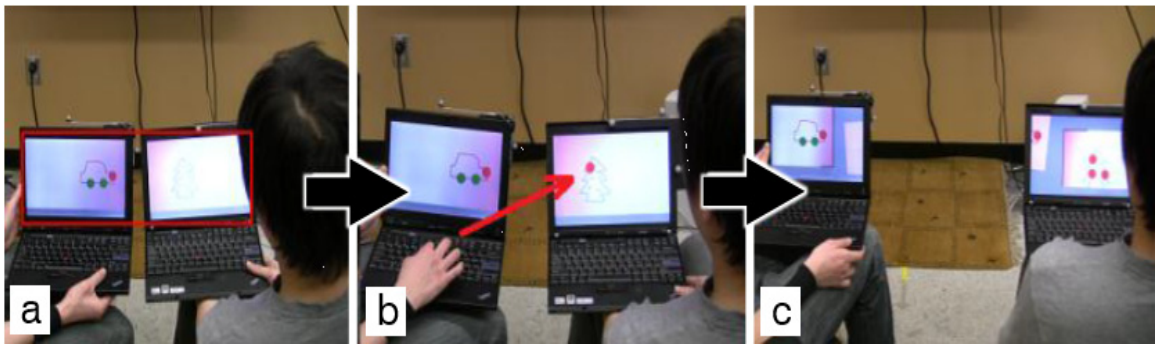


Figure 10.12 ProxemiCanvas facilitates collaborative drawing (Xiang Anthony Chen).

CP13 - Proxemic-aware Bulletin Board (Richard Fung) illustrates how an interactive bulletin board can be designed to react to people around them (as with the projects described in Section §10.2.1) and the personal devices they carry. A person passing by the interactive bulletin board (displayed on a large interactive screen) can see the enlarged icons and headlines of the postings (Figure 10.13a). When moving closer, the detailed content of the posting is revealed (inspired by the gradual engagement pattern). Since

the interactive board recognizes people's identity, it also highlights new postings since the last time the person passed by the board. Icons on the screen indicate interaction possibilities, such as entering one's email address to receive more information about a posting on the board (Figure 10.13b), or ways to download further information about a posting onto a mobile device the person is carrying (Figure 10.13c). Bringing the mobile device close to the large display links the two devices to enable transfer of digital information.

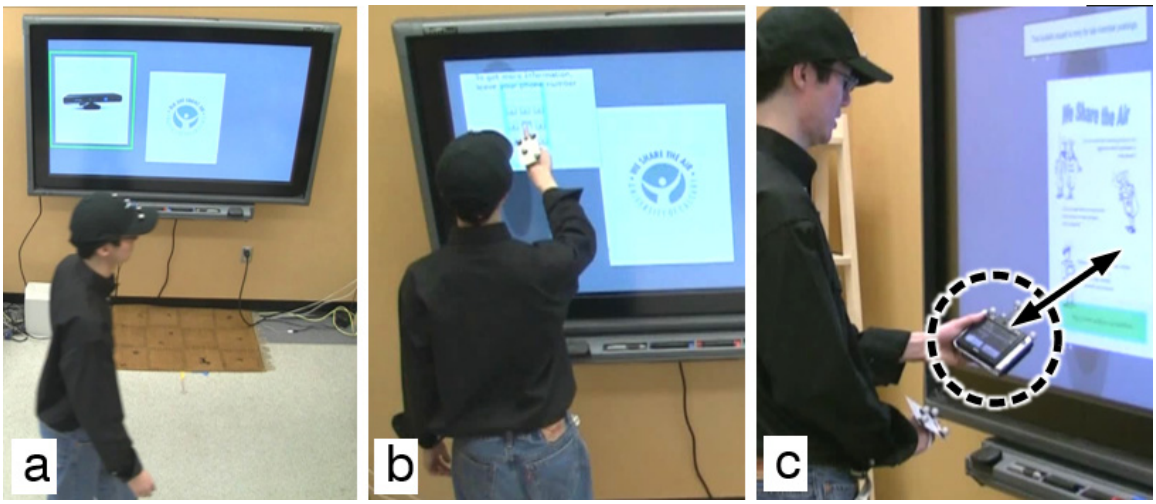


Figure 10.13 Proxemic-aware Bulletin Board (Richard Fung).

SP14 - Multi-device Viewer for Medical Images (Francisco Marinho Moreira Rodrigues) lets a person browse through 3D volumetric medical data sets by moving a tablet in 3D space above a tabletop. While the tabletop displays a static top-down projection of the medical dataset (e.g., a person's torso as shown in Figure 10.14), the tablet PC shows a visualization of a 2D slice cut through the 3D data set. The tablet's visualization is continuously updated when moving the tablet through the volume above the tabletop.

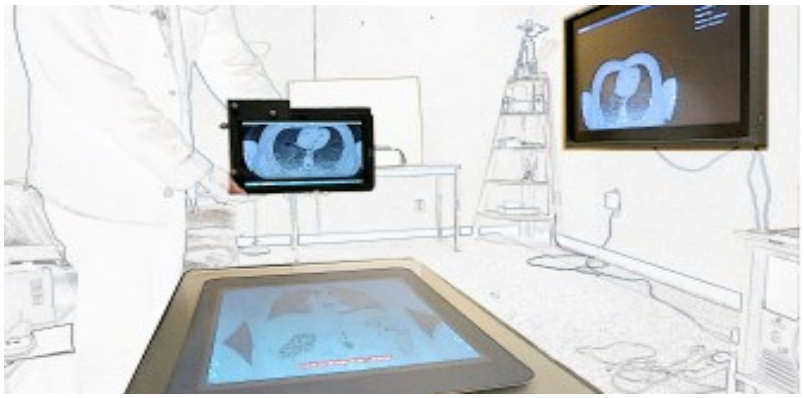


Figure 10.14 Multi-device viewer for medical images (Francisco Marinho Moreira Rodrigues).

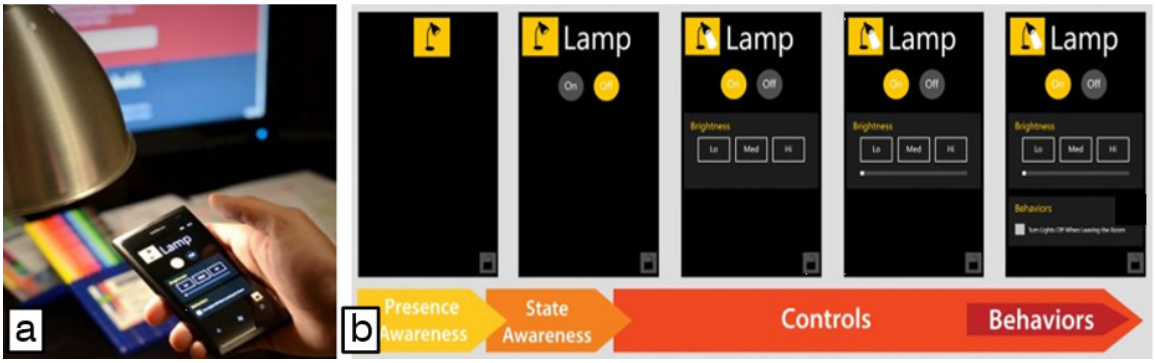


Figure 10.15 Proxemic Awareness and Control: (a) using a mobile device to control a lamp with (b) a series of progressively revealed interaction controls on the mobile phone's screen (David Ledo).

SP15 - Mobile Proxemic Awareness and Control (David Ledo) leverages proxemic interaction strategies to facilitate the control of appliances via a mobile phone (Ledo and Greenberg 2013). The phone's distance to an appliance (currently a lamp or an alarm clock) determines the status of the control interface on the mobile screen (Figure 10.15a). As a variation of the gradual engagement pattern, information about the presence, state, content, and control possibilities of surrounding appliances are progressively revealed on the screen of the mobile phone when the person orients towards and moves closer to the appliance. The progression of the interface elements progresses through a series of stages (Figure 10.15b): from device awareness (e.g., showing which devices are in the environment around a person), to state awareness (e.g., giving information whether a

lamp is on or off), to displaying control elements (e.g., allowing to set a timer of when to turn a lamp on or off during the day).

10.3 Toolkit Extensions

The Proximity Toolkit was built with the goal in mind to make it easily extensible with additional features supporting different kinds of input or output hardware. Two examples illustrate this extensible architecture.

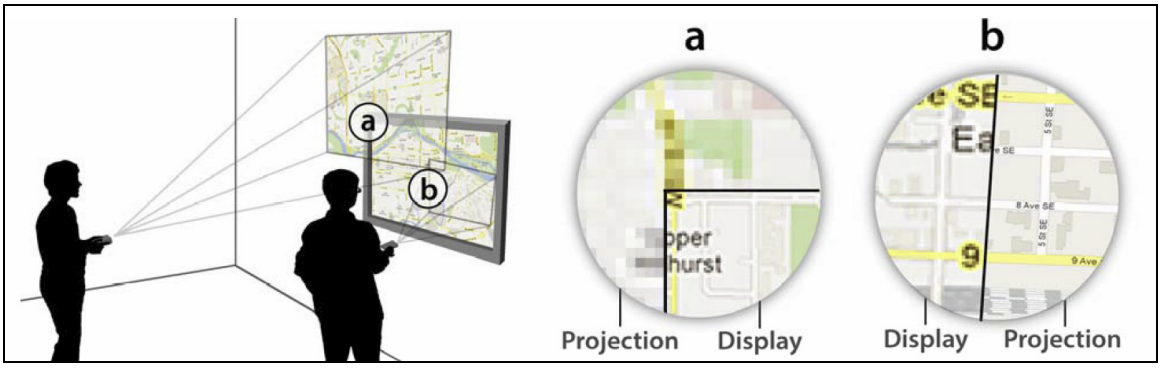


Figure 10.16 Using mobile projections to add (a) context and (b) focus areas on a large stationary display (Weigel et al. 2013).

IP16 - ProjectorKit (Martin Weigel) facilitates the development of interactive applications using one or multiple mobile projectors (Weigel et al. 2013). This toolkit directly builds upon the foundation of the Proximity Toolkit, and extends the functionality through five essential building blocks for prototyping applications with mobile projectors. First, it improves projection stability through a keystone and hand jitter correction. Second, the automated projection mapping allows binding graphical textures to physical objects (e.g., wall, books) so they become visible once any projector covers that particular area. Third, the targeting and hotspot mechanisms facilitate the capturing of pointing events. Fourth, a gesture recognizer captures device movements and triggers gesture events. Fifth, the support for dealing with multiple overlapping projections to – for example – monitor the overlapping area between projections and trigger event notifications when changes occur. Together, these additional building blocks enable the prototyping of, for example, the **IP 17 – Focus + Context application**

that is using mobile projections to add focus and context areas on a stationary display (Weigel et al. 2012). In this application, the projector of a person further away from the display provides a low resolution view of the context beyond the display's borders (Figure 10.16a). A person standing closer to the display can use the mobile projector to provide a high resolution view of the stationary display's content (Figure 10.16b).

TP18 - OptiTrack plugin (described earlier in Chapter 6) is another example for the extensible architecture of the Proximity Toolkit. This plugin adds support for the OptiTrack system – a lower cost tracking hardware alternative – as a provider of raw data for the toolkit. While the OptiTrack is somewhat similar in functionality to the Vicon hardware, its API and some of its details differ considerably. The OptiTrack plugin is directly built based on the plugin architecture that we described earlier in Section §6.2.3, which requires only minimal input to stream tracking data into the toolkit, as all higher-level calculations are done by the toolkit (see Section §6.2.3 and other sub-sections of §6.2). Most importantly, the API used by developers of proxemic-aware applications remains unchanged, no matter which tracking plugin currently provides the raw data into the Proximity Toolkit server. Thus the tracking system can be substituted with no changes to the code.

These projects are only a partial list. Besides the explorations of the proxemic interaction design space, other students and researchers used the toolkit in novel ways applied to other areas of research. For example, the toolkit's API was also a fundamental building block for other projects not directly related to proxemic interactions, such as the explorations of body-centric interaction techniques with mobile devices (Chen 2012) or explorations of a continuous interaction space on and around interactive tabletops (Marquardt et al. 2011).

10.4 Evaluation of the Proximity Toolkit

We now take a step back and evaluate the Proximity Toolkit in terms of how well it serves as a rapid prototyping platform. As argued by Olsen (2007), many traditional usability testing methods are not adequate for evaluating user interface toolkits. This is

because many usability experiments rely on three key assumptions that do not hold when evaluating toolkits: (1) that anyone with a shared expertise can walk up and use the system, (2) that there is a standardized task to perform, and (3) that the experiment can be performed in a rather short and limited time frame (often 1-2 hours). Toolkit research often does not meet any of these assumptions: (to 1) toolkits require a person to gain knowledge in using the tool, (to 2) problems solved with a toolkit are often more complex and allow many possible paths to solve them, and (to 3) the use of toolkits often continues over a longer period of time.

As an alternative method for evaluating user interface toolkits, Olsen (2007) offers a set of criteria to judge the results of the toolkit design. We now review selected criteria from this set for evaluating the value of the Proximity Toolkit for facilitating explorations of proxemic-aware ubicomp systems (Olsen 2007). As we will see, these criteria directly correlate to the three desired characteristics for successful toolkit designs (Myers et al. 2000; Resnick et al. 2005) discussed earlier in Sections §5.1.2 and §10.1: *low threshold* (i.e., moving the entry barrier as low as possible), *high ceiling* (i.e., facilitating to build sophisticated and powerful applications), and *wide walls* (i.e., allowing diverse projects of a wide design space to be built with the toolkit).

1. **Empowering new design participants.** Myers et al.'s (2000) describe this criteria as *low threshold*. As shown with the student class and semester projects (CP and SP in Table 10.1), the Proximity Toolkit successfully lowered the entry barrier for undergraduate and graduate computer science students – who had no or only minimal prior knowledge about spatial ubicomp tracking systems – to explore their own envisioned proxemic-aware ubicomp systems. The toolkit fostered students' and researchers' creative potential by providing a set of higher-level building blocks (e.g., Table 5.1) that encapsulate our identified important measures of proxemics (§3.2.1) that can be leveraged in design. The visual monitoring tool's view of tracked entities (§5.3.2) and the proxemic relationships between (§5.3.3), and the templates/examples were important parts for lowering that initial entry barrier for new developers. Maybe most importantly, all students were able to build a first fully functional prototype application with the toolkit in their first week after being

introduced to the toolkit. These "hello world"-like applications were often the starting point for iterative refinements of the project throughout the course (SP in Table 10.1).

2. **Generality** (of a toolkit to apply to diverse situation, task and user contexts). Resnick et al. (2005) describe this criteria as *wide walls*: "*tools should support and suggest a wide range of explorations*". The generality of the Proximity Toolkit is best demonstrated with the large variety of contexts of the proxemic-aware applications that students were able to address (contexts summarized in column 7 of Table 10.1): art and music (CP8, SP12), office applications (CP1, SP6, SP7, CP13, IP17), scientific tools (SP2, SP14), museum exhibits (SP10), retail environments (CP4, TP5, SP9), games (IP3), and tools at home (SP15). These prototypes also illustrate the application of our proxemic interaction concepts in general to a breadth of diverse application domains; including various implementations of the gradual engagement pattern (column 6 of Table 10.1) or the use of discrete distance zones vs. continuous measures (column 4 and 5 of Table 10.1). The spectrum of different tracked entities considered in the design of interaction techniques was also diverse, including: people, phones, tablets, large wall displays, interactive tabletops, transparent displays, mobile projectors, physical instruments, lamps, radios, and robots (see column 2 of Table 10.1).

3. **Reduce solution viscosity.** As suggested by Olsen (2007), a reduced solution viscosity through a toolkit can be demonstrated with the following three characteristics:

    a. **Flexibility** describes a tool's ability to let the developer make rapid design changes. The Proximity Toolkit's design provides flexibility in multiple ways. First, changing or adding tracked entities usually requires only one to three lines of code. Similarly, changing or adding the monitoring for another proxemic relationship requires a single line of code. Second, as the possible design alternatives and available tools should self-revealing (Resnick et al. 2005), our visual monitoring tool provides a 1-to-1 visual match of what the system sees and what proxemic measures are available via the event driven-API. Third, the

decoupling of tracking systems from the API and the design of our plugin architecture allows easy experimentation with different tracking hardware (replacing one another or mix-and-match).

b. **Expressive Leverage** is provided when a "designer can accomplish more by expressing less" (Olsen 2007). The Proximity Toolkit provides expressive leverage by managing the underlying tracking hardware, setup and calibration, and hiding this process from the application developer. Thus ubicomp developers do more with less as they can concentrate on the high-level interaction design of their applications rather than low level plumbing. The toolkit API components for monitoring relationships between two entities encapsulate many behaviours (e.g., the use of discrete distance zones, or updates about orientation changes) commonly used in earlier proxemic-aware applications (see survey in Chapter 2). Reusing these components minimizes the need for repetitive code across applications, so that the code developers write is primarily for defining the actual interactive behaviour of their application.

c. **Expressive Match** is an "estimate of how close the means for expressing design choices are to the problem being solved" (Olsen 2007). We designed our API vocabulary based on the identified five proxemic dimensions (§3.2.1), which in turn are derived from our analysis of proxemic theories (§3.1). Due to this design, our provided building blocks (e.g., the methods/events in Table 5.1) are a closer expressive match to the nuanced theory of proxemics than low-level programming frameworks for a particular tracking hardware. Our API allows developers to program against the actual *entities* in space and their *relationships* instead of dealing with vectors, matrices, filters, data streams, calibrations or affine transformations. A potential limitation of this higher expressive match is that it adds another abstraction layer. This might make some low-level (usually advanced) operations – if required by the developer – more difficult to do. To mitigate this problem, we decided to make it possible to always access underlying raw data in case a developer needs this functionality (e.g., by directly

accessing shared dictionary values), but guaranteeing ease of use by making the higher-level access methods the primary/default method for accessing the data.

4. **Power in combination** comes from several means. First, the unified API provides access to all tracked entities, independent of the underlying hardware. Thus different tracking systems can be used or combined, giving different expressive powers (e.g., to fit the needs and costs of a particular deployment). Second, many different entities (people, devices, objects) can be tracked and combined in a single application. Third, the toolkit is also extensible in various ways, which is important to provide additional powers for cases that are not covered (Olsen 2007). In particular, the toolkit allows extensions via new plugins (§6.2.3), new decorators (§6.2.2), or additional API components (§10.1.6), each which can be combined with (and leverage) existing facilities.

5. **Scalability** concerns whether the toolkit scales up to realistic problems in the research domain it is designed for. While some applications reviewed in Section §10.1 focus on the relationships between only two entities (see projects 1,2,5,6,7,9,11,12), other projects considered a larger number of people, devices, and objects present in ubicomp ecologies (3 to 6 entities in projects 3,4,8,10,13,14,15,16,17). In addition, our in-depth explorations of proxemic interactions in Chapters 7 and 8 illustrate how the Proximity Toolkit scales to larger ubicomp ecologies (the "high ceiling", §5.1.2 and Myers et al., 2000). Scalability, however, is currently limited and challenging, both from a technical point of view of providing adequate hardware to track more entities and processing that information in real time, and also from a conceptual point of view considering the increasing complexity when simultaneously monitoring higher numbers of relationships between entities (in particular in 1-to-n and n-to-n relationships).

10.5 Conclusion

In this chapter we surveyed a diverse set of proxemic-aware ubicomp systems built by students and researchers, all built by using the Proximity Toolkit API. We then

evaluated the Proximity Toolkit against a set of toolkit evaluation criteria provided by others. What we believe is most important in these examples is how the Proximity Toolkit and our theoretical framework of proxemic interaction lowered the threshold for developers to not only begin their exploration of proxemics in the ubicomp context, but to develop quite sophisticated and varied applications. Easy access to proxemic information through the toolkit and API allowed them to rapidly prototype alternative system designs, all leading towards exploring the design space of future proxemic-aware ubicomp systems.

Chapter 11. Conclusion

As I explained in the beginning of this dissertation in Chapter 1, my thesis is that:

> *we can leverage information about people's and devices' fine-grained proxemic relationships for the design of novel interaction techniques in ubicomp ecologies.*

With this thesis in mind, the overarching goal of this dissertation was to inform the design of future proxemic-aware devices that – similar to people's natural expectations and use of proxemics – allow increasing connectivity and interaction possibilities when in proximity to people, other devices, or objects. Towards this goal, I explored how the fine-grained knowledge of proxemic relationships between the entities in small-space ubicomp ecologies (people, devices, objects) can be exploited in interaction design. In particular, I identified three specific goals that I addressed with the work of this dissertation.

In this conclusion chapter, I first review my progress towards addressing the research problems described in Chapter 1 (§11.1). Next, I summarize the major and minor contributions (§11.2). Last, I provide pointers to future research (§11.3).

11.1 Progress towards Addressing Dissertation Problems and Goals

I begin by reviewing the progress I have made towards my dissertation goals addressing the problems identified in Chapter 1.

11.1.1 Problem #1: Framework of Proxemic Interaction

Problem #1: We do not know how proxemic theories and measures of proxemic relationships can be applied to interaction design in ubicomp ecologies.

Goal #1: I will operationalize proxemics for ubicomp interaction design. I will describe a framework of Proxemic Interactions that informs the design proxemic-aware interactive systems.

Status: **Completed.** To achieve this goal, in Chapter 3 I provided a comprehensive analytical survey of relevant proxemic literature and synthesized key theories about people's use and understanding of personal space and proxemics. I operationalized proxemics for ubicomp interactions in form of the Proxemic Interaction framework. The framework informs ubicomp developers' design process to let them create, invent, or discover novel interaction techniques considering proxemics. As part of the framework, I identified five key dimensions of measured proxemic relationships that can be considered to mediate people's interactions with ubicomp systems: distance, orientation, movement, identity, and location. I explained how to use information from these five dimensions to guide the design of novel interaction techniques. For a better understanding of the kinds of possible proxemic interactions and for demonstrating how to use Proxemic Interactions as a lens to categorize prior work, I identified a series of proxemic interaction themes that address six ubicomp interaction design challenges. As part of thesis goal #3, Chapters 7-9 later demonstrated the design of novel interaction techniques in proxemic-aware systems.

11.1.2 Problem #2: Rapidly Prototyping Proxemic Interactions

Problem #2: We do not have adequate developer tools for rapidly prototyping and exploring proxemics aware interfaces in ubicomp ecologies.

Goal #2: I will design and implement rapid prototyping tools that make these proxemic measurements between people/devices/objects part of ubicomp ecologies easily accessible for developers.

Status: **Completed.** As described in Chapter 5, I designed and implemented the *Proximity Toolkit* allowing ubicomp developers to rapidly prototype proxemic-aware ubicomp systems. The toolkit simplifies the development process by supplying higher-level information about proxemic relationships (along the five dimensions part of thesis goal #1) between the entities in ubicomp ecologies through an event-driven API and visual inspection tools. As explained in Chapter 6, the toolkit architecture allows the integration of diverse tracking hardware, allowing ubicomp developers to mix and match different tracking technologies when prototyping their applications. What is important is how the Proximity Toolkit lowered the threshold for students and other researchers to begin their exploration of proxemics within ubiquitous computing applications (Chapter 10). The easy access to proxemic information through the toolkit and API allowed others to rapidly prototype alternative system designs, leading towards explorations of a diverse design space of proxemic-aware ubicomp systems.

11.1.3 Problem #3: Applying Proxemics to Interactions in Small-Space Ubiquitous Computing Ecologies

Problem #3: We do not know how people's interactions with devices in small space ubicomp ecologies can be mediated by the system's knowledge of proxemic relationships.

Goal #3: I will design and implement proxemic-aware ubicomp systems that integrate concepts of proxemic interactions and explore the design space of proxemic interactions in ubicomp ecologies.

Status: **Completed.** As described in Chapters 7, 8, and 9, I demonstrated the design of novel Proxemic Interaction techniques for people's interaction with devices in ubicomp ecologies. For the design of these techniques I exploited the knowledge of distance, orientation, location, movement, and identity as part of the introduced dimensions of proxemic interactions (§3.2) to drive the possible interactions. To achieve goal #3, I decided to focus on three particular parts of the ubicomp ecology design space.

First, I focused on proxemic interactions of one person or multiple people with a large interactive display situated in a ubicomp ecology. I introduced a set of novel interaction techniques for people's interaction with large interactive screens. For the design of these techniques I considered relationships of different aspects of a small space ubicomp ecology: the relationships of a single person to a device, of multiple people to a device, and of non-digital objects to people and devices. The interaction concepts incorporated the knowledge of the fixed and semi-fixed feature space, interpreted a person's directed attention extending attentive user interfaces, supported fine-grained explicit interactions, and demonstrate how proxemic information can regulate interaction based either on continuous movement or by movement in and out of discrete proxemic zones. I introduced the gradual engagement design pattern, and described how to apply the stages of the gradual engagement pattern to person-device interaction. Finally, I explained how the techniques extend beyond pairwise interaction.

Second, I focused on applying concepts of proxemic interactions to mediate device-to-device operations – both between personal (e.g., tablets) and semi-public devices (e.g., digital whiteboards). In particular, I refined the gradual engagement pattern from Section §7.6 to ease the cross-device information transfer task. As people move and orient their personal device towards other surrounding devices, I

demonstrate how the interface progressively moves through the three stages affording gradual engagement: (1) awareness of device presence and connectivity, (2) reveal of exchangeable digital content, and (3) interaction methods for transferring digital content between devices tuned to particular distances and device capabilities. I describe novel interaction techniques, where their design fits to the three stages of the gradual engagement design pattern.

Third, I considered both person-to-person proxemics (through small group F-formations) and device-to-device proxemic relationships, both which were used to facilitate sharing of digital content between digital devices. Informed by micro-mobility theory and by an observational study of people working together with foam-core mock-ups of mobile displays, I introduced a number of cross-device interaction techniques that support nuanced gradations of sharing, from the subtle to the overt, with the goal of minimizing the transaction costs—and social disruption—of sharing information across a small-group ecology of digital devices and situated displays.

11.2 Thesis Contributions

With this dissertation I have informed the design of future proxemic-aware devices that – similar to people's natural expectations and use of proxemics – allow increasing connectivity and interaction possibilities when in proximity to people, other devices, or objects. In particular, I made the following major and minor contributions:

11.2.1 Major Contributions

1. **The operationalization of proxemics for interactions in ubiquitous computing ecologies in form of the Proxemic Interactions framework.** It synthesizes fundamental concepts of proxemics and identifies five key

dimensions relevant for proxemics in ubicomp interactions: distance, orientation, movement, identity, and location.

2. **The design and implementation of the Proximity Toolkit** which simplifies access to proxemic information for developers of ubicomp interfaces. The toolkit provides representations of higher-level proxemic concepts (e.g., relationships, fixed/semi-fixed features) derived from low-level information. The toolkit also includes complementary visual tools that allow developers to explore proxemic relationships between entities in space without coding.

3. **Three in-depth explorations of designing proxemic-aware ubicomp systems**, where the designs of a series of novel interaction techniques were guided by the Proxemic Interactions framework. The case studies address three areas of the ubicomp interaction design space: person/people-to-device interactions, device-to-device interactions, and interactions considering both proxemics of people and proxemics of devices. These three case studies and the designed proxemic interactions can serve as inspirational examples for others exploring proxemics in ubicomp design.

11.2.2 Minor Contributions

- The identification and definition of the **gradual engagement design pattern**.
- The **application of the gradual engagement pattern to cross-device information transfer**.
- A **novel algorithm for detecting F-formations** describing spatial relationships between people by interpreting depth information images captured with ceiling-mounted structured light cameras.
- A **novel hybrid sensing approach** for tracking people's and devices' proxemic relationships by combining wireless radio signals estimating inter-radio distance, processed visual images captured by depth-sensing cameras, and data from accelerometer/gyroscope/compass sensors attached directly to the devices.

- Insights of an explorative **study about people's variations in proxemic relationships, type of f-formations, and devices' micro-mobility** in co-located settings while working on collaborative, competitive, and individual tasks.
- A **survey of 18 diverse proxemic-aware ubicomp applications** built by others using the Proximity Toolkit, covering many different application domains and uses of technology.

11.3 Potential Directions for Future Work

In this section, I describe possible directions for research extending the work presented in this dissertation.

11.3.1 Defining Rules of Behaviour

One of the unsolved issues in proxemic interaction is how one can configure the *rules of behaviour*, i.e., how the system should react to the proxemic information it gathers. While computers can take action based on its inference of the proxemic dynamics, it will sometimes get it wrong. Creating meaningful behaviours and repairing mistakes will, I believe, become a central issue in the design of such systems. Even with this caveat, I believe that proxemic interactions will become a powerful way to realize embodied interaction, where – ideally – the system naturally responds to people's social expectations and practices in their everyday environments, and where mistakes are easily repaired or of little consequence.

As one approach to address the difficult challenge for defining appropriate rules of behaviour in interactive ubiquitous computing systems, future work could also investigate end-user programming methods for reconfiguring existing or creating new behaviours. Balancing ease-of-use vs. potential expressiveness, and finding the right building blocks for such a non-expert tool interface, would be key challenges for these explorations.

11.3.2 Other Factors Influencing Proxemic Behaviour

While going beyond the scope of this dissertation, people's perception of proxemic relationships is also influenced by other factors such as gender, cultures, age, work hierarchies, and other factors (Hall 1966). These differences also affect the design of proxemic interactions. For example, we can imagine a system that requires people to stand in very close proximity to each other to collaboratively interact with an interactive surface, e.g., to exchange digital documents. This close proximity might be perceived as adequate by some, but as too intimate by others. Therefore, the design of proxemic interactions has to consider these variations in proxemic perception. In this regard, the presented explorations in this dissertation serve just as examples that illustrate design possibilities for proxemic interactions. I do not suggest that the rules of behaviour I described are ideal, nor that they achieve the perfect balance between adjudicating proxemic information and implicit or explicit interaction. Future research should further investigate the impact of these other factors influencing the proxemic behaviour, and explore how to address this in ubicomp system designs.

11.3.3 Pattern Language of Proxemic Interactions

Our gradual engagement pattern is just a starting point suggesting further exploration of proxemic patterns in interaction design. These patterns would then also unify prior work, synthesize essential, generalizable interaction strategies, and provide a common vocabulary for discussing design solutions. Most importantly, the patterns inform and inspire future designs, but also allows for variations of the pattern applied to different domains. A possible outcome could be a Proxemic Interaction Pattern Language – a collection of proxemic patterns that not only describe the essential patterns themselves, but also the inter-relations between – similar to Christopher Alexander's seminal architectural pattern language (1977).

11.3.4 Violating Proxemic Expectations

Despite our belief of the importance to consider proxemic theories and people expectations of personal space in interaction design, as a contradicting possibility one

could imagine interaction designs that deliberately violate expectations of proxemics. Violations of a person's personal space as defined by proxemics do not always cause negative reactions (Burgoon and Hale 1988). Thus, depending on the context and design of a ubicomp applications, deliberate violations of personal space (such as requiring people standing in close/intimate distance) might be an integral part of the user experience itself (for example, in games or public interactive art installations). Violations of interpersonal space as described by proxemics (e.g., requiring people to stand close to each other) do not necessarily cause negative reactions. This project explores how such deliberate violations of personal space might be an integral part of the user experience, e.g., in games or public art installations.

11.3.5 Interactions in Large-scale (cluttered) Ubicomp Ecologies

The research of this dissertation concentrated on interactions in small space environments. An interesting question to answer in future work would be how we can apply proxemics to mitigate challenges of people's interactions in cluttered environments with 10's or 100's of devices present simultaneously. This research challenge would address multiple challenges in ubicomp interaction, such as handling scalability, providing meaningful feedback through visualizations, and exploring people's interactions in these ubicomp device ecologies.

11.3.6 Other Concerns

The suggested future research above touches upon many possible directions. As often the case with new interaction paradigms in the early stages, the area is wide open. There are many other possibilities to explore proxemics in interaction design: application-specific issues and opportunities, qualitative studies about people's use and perception of proxemic-aware systems, or new hardware developments – which again both restrict and expand what can be done.

11.4 Closing Remarks

With an increasing number of digital devices available around us to facilitate our everyday tasks, it also becomes increasingly important to find interaction strategies that let us more naturally and easier connect to and interact with these devices. I believe that proxemic interactions has great potential for designing such interactions, as it can exploit people's expectations of how they and their devices should interact within particular ecologies as they move towards each other. I hope the work presented in this dissertation will inspire researchers to consider proxemic measures between all entities in ubicomp ecologies – the people, devices, and other objects – to design interactive systems that match closer to our natural understanding and expectations of proxemics.

References

ABOWD, G.D., ATKESON, C.G., HONG, J., LONG, S., KOOPER, R. AND PINKERTON, M., 1997. Cyberguide: A Mobile Context-Aware Tour Guide. *Wireless Networking*, 3(5), pp.421–433.

ABOWD, G.D. AND MYNATT, E.D., 2000. Charting Past, Present, and Future Research in Ubiquitous Computing. *ACM Transactions on Computer-Human Interaction*, 7(1), pp.29–58.

ABOWD, G.D., MYNATT, E.D. AND RODDEN, T., 2002. The Human Experience [of Ubiquitous Computing]. *IEEE Pervasive Computing*, 1(1), pp.48–57.

ADAMS, L. AND ZUCKERMAN, D., 1991. The Effect of Lighting Conditions on Personal Space Requirements. *The Journal of General Psychology*, 118(4), pp.335–340.

ADDLESEE, M., CURWEN, R., HODGES, S., NEWMAN, J., STEGGLES, P., WARD, A. AND HOPPER, A., 2001. Implementing a Sentient Computing System. *Computer*, 34(8), pp.50–56.

AIELLO, J.R., 1987. Human Spatial Behaviour. In D. Stokols & I. Altman, eds. *Handbook of environmental psychology*. New York: John Wiley & Sons, pp. 359–504.

AIELLO, J.R. AND AIELLO, T.D.C., 1974. The Development of Personal Space: Proxemic Behavior of Children 6 through 16. *Human Ecology*, 2(3), pp.177–189.

ALEXANDER, C., ISHIKAWA, S. AND SILVERSTEIN, M., 1977. *A Pattern Language: Towns, Buildings, Construction*, Oxford University Press.

ALTMAN, I., 1975. *The Environment and Social Behavior: Privacy, Personal Space, Territory, and Crowding*, Brooks/Cole Publishing Company, Monterey, California.

ANNETT, M., GROSSMAN, T., WIGDOR, D. AND FITZMAURICE, G., 2011. Medusa: A Proximity-Aware Multi-Touch Tabletop. In *Proceedings of the 24th Annual ACM*

Symposium on User Interface Software and Technology. UIST '11. New York, NY, USA: ACM, pp. 337–346.

ANTIFAKOS, S. AND SCHIELE, B., 2002. Beyond Position Awareness. *Personal Ubiquitous Computing*, 6(5-6), pp.313–317.

ARGYLE, M. AND DEAN, J., 1965. Eye-Contact, Distance and Affiliation. *Sociometry*, 28(3), pp.289–304.

ASENIERO, B.A., TANG, A., CARPENDALE, S. AND GREENBERG, S., 2013. *Showing Real-time Recommendations to explore the stages of Reflection and Action*, Calgary, Alberta, Canada: Technical Report #2013-1040-07, Department of Computer Science, University of Calgary.

BACK, M., COHEN, J., GOLD, R., HARRISON, S. AND MINNEMAN, S., 2001. Listen Reader: An Electronically Augmented Paper-Based Book. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '01. New York, NY, USA: ACM, pp. 23–29.

BALDASSARE, M., 1978. Human Spatial Behavior. *Annual Review of Sociology*, 4(1), pp.29–56.

BALLAGAS, R., RINGEL, M., STONE, M. AND BORCHERS, J., 2003. iStuff: A Physical User Interface Toolkit for Ubiquitous Computing Environments. In *Proceedings of the 21th ACM Conference on Human Factors in Computing Systems*. CHI '03. New York, NY, USA: ACM, pp. 537–544.

BARDRAM, E., 2005. The Trouble with Login: On Usability and Computer Security in Ubiquitous Computing. *Personal and Ubiquitous Computing*, 9(6), pp.357–367.

BARDRAM, J. AND FRIDAY, A., 2010. Ubiquitous Computing Systems. In J. Krumm, ed. *Ubiquitous Computing Fundamentals*. Boca Raton, Florida, USA: CRC Press, pp. 37–94.

BAUDISCH, P., CUTRELL, E., ROBBINS, D., CZERWINSKI, M., TANDLER, P., T, P., BEDERSON, B. AND ZIERLINGER, A., 2003. Drag-and-Pop and Drag-and-Pick: techniques for accessing remote screen content on touch- and pen-operated systems. In *In Proceedings of IFIP TC13 International Conference on Human-Computer Interaction*. INTERACT '03. IOS Press, pp. 57–64.

BEAUDOUIN-LAFON, M., 2004. Designing interaction, not interfaces. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '04. New York, NY, USA: ACM, pp. 15–22.

BECHTEL, R.B. AND CHURCHMAN, A., 2002. *Handbook of Environmental Psychology*, John Wiley and Sons.

BELL, P.A., GREENE, T., FISHER, J. AND BAUM, A.S., 2005. *Environmental Psychology* 5th ed., Psychology Press.

BELLOTTI, V., BACK, M., EDWARDS, W.K., GRINTER, R.E., HENDERSON, A. AND LOPES, C., 2002. Making sense of sensing systems: five questions for designers and researchers. In *Proceedings of the 20th ACM Conference on Human Factors in Computing Systems*. CHI '02. New York, NY, USA: ACM, pp. 415–422.

BENNETT, F., RICHARDSON, T. AND HARTER, A., 1994. Teleporting-Making Applications Mobile. In *Proceedings of the Workshop on Mobile Computing Systems and Applications*. IEEE Computer Society, pp. 82–84.

BIEHL, J.T. AND BAILEY, B.P., 2004. ARIS: an interface for application relocation in an interactive space. In *Proceedings of Graphics Interface*. GI '04. London, Ontario, Canada: Canadian Human-Computer Communications Society, pp. 107–116.

BORCHERS, J., 2001. *A Pattern Approach to Interaction Design*, New York, NY, USA: Wiley.

BORING, S., BAUR, D., BUTZ, A., GUSTAFSON, S. AND BAUDISCH, P., 2010. Touch projector: mobile interaction through video. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems*. CHI '10. New York, NY, USA: ACM, pp. 2287–2296.

BOYLE, M. AND GREENBERG, S., 2005. Rapidly prototyping multimedia groupware. In *Proceedings of the 11th International Conference on Distributed Multimedia Systems - DMS 2005*. Knowledge Systems Institute, pp. 178–183.

BRAGDON, A., DELINE, R., HINCKLEY, K. AND MORRIS, M.R., 2011. Code space: touch + air gesture hybrid interactions for supporting developer meetings. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS '11. New York, NY, USA: ACM, pp. 212–221.

BRAVE, S., ISHII, H. AND DAHLEY, A., 1998. Tangible Interfaces for Remote Collaboration and Communication. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. CSCW '98. New York, NY, USA: ACM, pp. 169–178.

BRUMITT, B., KRUMM, J., MEYERS, B. AND SHAFER, S., 2000. Ubiquitous computing and the role of geometry. *Personal Communications, IEEE*, 7(5), pp.41–43.

BRUMITT, B., MEYERS, B., KRUMM, J., KERN, A. AND SHAFER, S., 2000. Easy Living: Technologies for Intelligent Environments. In *Proceedings of the Second International Symposium on Handheld and Ubiquitous Computing*. HUC '00. Springer, pp. 12–27.

BURGOON, J.K. AND HALE, J.L., 1988. Nonverbal Expectancy Violations: Model Elaboration and Application to Immediacy Behaviors. *Communication Monographs*, 55(1), pp.58–79.

BUXTON, W.A.S., 1995. Integrating the Periphery and Context: A New Model of Telematics. In *Proceedings of Graphics Interface*. GI '95. Canadian Human-Computer Communications Society, pp. 239–246.

BUXTON, W.A.S., 1997. Living in Augmented Reality: Ubiquitous Media and Reactive Environments. In K. Finn, A. Sellen, & S. Wilber, eds. *Video Mediated Communication*. Hillsdale, N.J., USA: Lawrence Erlbaum Associates, Inc., pp. 363–384.

BUXTON, W.A.S., 2013. Multi-Touch Systems that I Have Known and Loved. *Bill Buxton Personal Website, http://www.billbuxton.com/multitouchOverview.html (accessed April 10, 2013)*.

BUXTON, W.A.S., 2007. *Sketching User Experience : Getting the Design Right and the Right Design*, San Francisco, CA, USA: Morgan Kaufmann Publishers.

CHEN, X., BORING, S., CARPENDALE, S., TANG, A. AND GREENBERG, S., 2012. SPALENDAR: Visualizing a Group's Calendar Events over a Geographic Space on a Public Display. In *Proceedings of the 11th International Working Conference on Advanced Visual Interfaces*. AVI '12. ACM.

CHEN, X.A., 2012. Body-centric interaction with mobile devices. In *Extended Abstracts of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*. TEI '12. New York, NY, USA: ACM, pp. 385–386.

CHEN, Y. AND SINCLAIR, M., 2008. Tangible security for mobile devices. In *Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services - MobiQuitous '08*. Dublin, Ireland: ICST, pp. 1–4.

CHONG, M.K. AND GELLERSEN, H., 2011. How users associate wireless devices. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems*. CHI '11. New York, NY, USA: ACM, pp. 1909–1918.

CIOLEK, T.M., 1983. The proxemics lexicon: A first approximation. *Journal of Nonverbal Behavior*, 8(1), pp.55–79.

CIOLEK, T.M. AND KENDON, A., 1980. Environment and the Spatial Arrangement of Conversational Encounters. *Sociological Inquiry*, 50(3-4), pp.237–271.

CLARK, H.H., 2003. Pointing and Placing. In S. Kita, ed. *Pointing. Where language, culture, and cognition meet*. Hillsdale, N.J., USA: Erlbaum, pp. 243–268.

COOPERSTOCK, J.R., FELS, S.S., BUXTON, W.A.S. AND SMITH, K.C., 1997. Reactive Environments: Throwing Away Your Keyboard and Mouse. *Communications of the ACM*, 40, pp.65–73.

COULOURIS, G., DOLLIMORE, J., KINDBERG, T. AND BLAIR, G., 2011. *Distributed Systems: Concepts and Design* 5th ed., Addison-Wesley.

DACHSELT, R. AND BUCHHOLZ, R., 2009. Natural throw and tilt interaction between mobile phones and distant displays. In *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '09. New York, NY, USA: ACM, pp. 3253–3258.

DEY, A.K., 2010. Context-Aware Computing. In J. Krumm, ed. *Ubiquitous Computing Fundamentals, Edited by J. Krumm.* Boca Raton, Florida, USA: CRC Press, pp. 285 – 319.

DEY, A.K., ABOWD, G.D. AND SALBER, D., 2001. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2), pp.97–166.

DIAZ-MARINO, R. AND GREENBERG, S., 2010. The proximity toolkit and ViconFace: the video. In *Extended Abstracts of the 28th International Conference on Human Factors in Computing Systems*. CHI EA '10. New York, NY, USA: ACM, pp. 4793–4798.

DOURISH, P., 2001a. Seeking a foundation for context-aware computing. *Human-Computer Interaction*, 16(2), pp.229–241.

DOURISH, P., 2001b. *Where the Action Is: The Foundations of Embodied Interaction*, The MIT Press.

DUKE, M.P. AND NOWICKI, S., 1972. A New Measure and Social-Learning Model for Interpersonal Distance. *Journal of Experimental Research in Personality*, 6(2-3), pp.119–132.

DUNBAR, R.I.M., DUNCAN, N.D.C. AND NETTLE, D., 1995. Size and structure of freely forming conversational groups. *Human Nature*, 6(1), pp.67–78.

ERICKSON, T., 2002. Some problems with the notion of context-aware computing. *Communications of the ACM*, 45(2), pp.102–104.

EVANS, G.W., LEPORE, S.J. AND SCHROEDER, A., 1996. The Role of Interior Design Elements in Human Responses to Crowding. *Journal of Personality and Social Psychology*, 70(1), pp.41–46.

EVERITT, K., SHEN, C., RYALL, K. AND FORLINES, C., 2006. MultiSpace: enabling electronic document micro-mobility in table-centric, multi-device environments. In C. Shen, ed. *Proceedings of First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*. TABLETOP '06. IEEE, pp. 27–34.

FEINER, S., MACINTYRE, B., HÖLLERER, T. AND WEBSTER, A., 1997. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. *Personal Technologies*, 1(4), pp.208–217.

FISHKIN, K.P., ROY, S. AND JIANG, B., 2005. Some Methods for Privacy in RFID Communication. In *Security in Ad-hoc and Sensor Networks*. Lecture Notes in Computer Science. Springer, pp. 42–53.

FITZMAURICE, G.W., 1993. Situated information spaces and spatially aware palmtop computers. *Communications of the ACM*, 36(7), pp.39–49.

GAINES, B.R., 1991. Modeling and forecasting the information sciences. *Information Sciences*, 57-58, pp.3–22.

GAMMA, E., HELM, R., JOHNSON, R. AND VLISSIDES, J.M., 1994. *Design Patterns: Elements of Reusable Object-Oriented Software* 1st ed., Addison-Wesley Professional.

GELLERSEN, H., FISCHER, C., GUINARD, D., GOSTNER, R., KORTUEM, G., KRAY, C., RUKZIO, E. AND STRENG, S., 2009. Supporting device discovery and spontaneous interaction with spatial references. *Personal Ubiquitous Computing*, 13(4), pp.255–264.

GIBSON, J., 1977. The Theory of Affordances. In R. Shaw & University of Minnesota., eds. *Perceiving, acting, and knowing: toward an ecological psychology*. Hillsdale, N.J., USA: Lawrence Erlbaum Associates.

GIFFORD, R., 1997. *Environmental Psychology: Principles and Practice* 2nd ed., Prentice-Hall.

GREENBERG, S., 2001. Context as a dynamic construct. *Human-Computer Interaction*, 16(2), pp.257–268.

GREENBERG, S., 2007. Toolkits and Interface Creativity. *Journal of Multimedia Tools and Applications (JMTA)*, 32(2), pp.139–159.

GREENBERG, S. AND FITCHETT, C., 2001. Phidgets: Easy Development of Physical Interfaces Through Physical Widgets. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*. UIST '01. New York, NY, USA: ACM, pp. 209–218.

GREENBERG, S. AND KUZUOKA, H., 2001. Using Digital but Physical Surrogates to Mediate Awareness, Communication and Privacy in Media Spaces. *Personal Technologies*, 3(4), pp.182–198.

GREENBERG, S., MARQUARDT, N., BALLENDAT, T., DIAZ-MARINO, R. AND WANG, M., 2011. Proxemic Interactions: The New Ubicomp? *ACM Interactions*, 18(1), pp.42–50.

GREENBERG, S. AND ROSEMAN, M., 1999. Groupware Toolkits for Synchronous Work. In M. Beaudouin-Lafon, ed. *Computer-Supported Cooperative Work (Trends in Software 7)*. Wiley, pp. 135–168.

GROUPLAB, 2013. Proximity Toolkit, download and documentation. *GroupLab, University of Calgary, http://grouplab.cpsc.ucalgary.ca/Projects/ProjectProximityToolkit (accessed May 10, 2013)*.

HALL, E.T., 1963. A system for the notation of proxemic behavior. *American Anthropologist*, 65(5), pp.1003–1026.

HALL, E.T., 1968. Proxemics. *Current Anthropology*, 9(2/3), pp.83–108.

HALL, E.T., 1966. *The Hidden Dimension* 1st ed., Garden City, N.Y: Doubleday.

HARDY, R. AND RUKZIO, E., 2008. Touch & interact: touch-based interaction of mobile phones with displays. In *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services*. MobileHCI '08. ACM, pp. 245–254.

HARRISON, C. AND DEY, A.K., 2008. Lean and zoom: proximity-aware user interface and content magnification. In *Proceeding of the 26th SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. New York, NY, USA: ACM, pp. 507–510.

HARRISON, S. AND DOURISH, P., 1996. Re-place-ing space: the roles of place and space in collaborative systems. In *Proceedings of the 1996 ACM Conference on Computer Supported Cooperative Work*. CSCW '96. New York, NY, USA: ACM Press, pp. 67–76.

HARTMANN, B., KLEMMER, S.R., BERNSTEIN, M., ABDULLA, L., BURR, B., ROBINSON-MOSHER, A. AND GEE, J., 2006. Reflective Physical Prototyping Through Integrated Design, Test, and Analysis. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. UIST '06. New York, NY, USA: ACM, pp. 299–308.

HASSAN, N., RAHMAN, M.M., IRANI, P. AND GRAHAM, P., 2009. Chucking: A One-Handed Document Sharing Technique. In *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction*. INTERACT '09. Berlin, Heidelberg: Springer, pp. 264–278.

HAYDUK, L.A., 1985. Personal space: The conceptual and measurement implications of structural equation models. *Canadian Journal of Behavioural Science/Revue canadienne des sciences du comportement*, 17(2), pp.140–149.

HE, H.A., 2010. *One Size Does Not Fit All: Extending the Transtheoretical Model to Energy FeedbackTechnology Design*. Calgary, Alberta, Canada: MSc Thesis, Department of Computer Science, University of Calgary.

HEDIGER, H., 1950. *Wild animals in captivity*, Butterworths Scientific Publications.

HIGHTOWER, J. AND BORRIELLO, G., 2001. Location Systems for Ubiquitous Computing. *Computer*, 34(8), pp.57–66.

HIGHTOWER, J., BRUMITT, B. AND BORRIELLO, G., 2002. The location stack: A layered model for location in ubiquitous computing. In *Proceedings of the 4th IEEE Workshop on Mobile Computing Systems & Applications*. WMCSA '02. IEEE, pp. 22–28.

HINCKLEY, K., 2003. Synchronous gestures for multiple persons and computers. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*. UIST '03. New York, NY, USA: ACM, pp. 149–158.

HINCKLEY, K., DIXON, M., SARIN, R., GUIMBRETIERE, F. AND BALAKRISHNAN, R., 2009. Codex: A Dual Screen Tablet Computer. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. CHI '09. New York, NY, USA: ACM, pp. 1933–1942.

HINCKLEY, K., PIERCE, J., SINCLAIR, M. AND HORVITZ, E., 2000. Sensing techniques for mobile interaction. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*. UIST '00. New York, NY, USA: ACM, pp. 91–100.

HINCKLEY, K., RAMOS, G., GUIMBRETIERE, F., BAUDISCH, P. AND SMITH, M., 2004. Stitching: pen gestures that span multiple displays. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '04. New York, NY, USA: ACM, pp. 23–31.

HOLMQUIST, L., MATTERN, F., SCHIELE, B., ALAHUHTA, P., BEIGL, M. AND GELLERSEN, H.-W., 2001. Smart-Its Friends: A Technique for Users to Easily Establish Connections between Smart Artefacts. In *Proceedings of Third International Conference on Ubiquitous Computing*. UbiComp '01. Springer, p. 116.

ISHII, H., KOBAYASHI, M. AND ARITA, K., 1994. Iterative design of seamless collaboration media. *Communications of the ACM*, 37(8), pp.83–97.

ISHII, H. AND ULLMER, B., 1997. Tangible Bits: Towards Seamless Interfaces Between People, Bits and Atoms. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. CHI '97. New York, NY, USA: ACM, pp. 234–241.

JU, W., LEE, B.A. AND KLEMMER, S.R., 2008. Range: exploring implicit interaction through electronic whiteboard design. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. CSCW '08. New York, NY, USA: ACM, pp. 17–26.

KATZ, D., 1937. *Animals and Men: Studies in Comparative Psychology*, Longmans, Green.

KENDON, A., 1990. *Conducting Interaction: Patterns of Behavior in Focused Encounters*, Cambridge University Press.

KENDON, A., 2010. Spacing and orientation in co-present interaction. In *Proceedings of Development of Multimodal Interfaces: Active Listening and Synchrony*. Lecture Notes in Computer Science. Springer, pp. 1–15.

KLEMMER, S.R., LI, J., LIN, J. AND LANDAY, J.A., 2004. Papier-Mache: Toolkit Support for Tangible Input. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. CHI '04. New York, NY, USA: ACM, pp. 399–406.

KNOWLES, E.S., 1989. An affiliative conflict theory of personal and group spatial behaviour. In P. B. Paulus, ed. *Psychology of group influence*. Hillsdale, NJ: L. Erlbaum.

KORTUEM, G., KRAY, C. AND GELLERSEN, H., 2005. Sensing and Visualizing Spatial Relations of Mobile Devices. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*. UIST '05. New York, NY, USA: ACM, pp. 93–102.

KRAUT, R., EGIDO, C. AND GALEGHER, J., 1988. Patterns of contact and communication in scientific research collaboration. In *Proceedings of the 1988 ACM Conference on Computer-Supported Cooperative Work*. CSCW '88. New York, NY, USA: ACM, pp. 1–12.

KRAY, C., ROHS, M., HOOK, J. AND KRATZ, S., 2008. Group Coordination and Negotiation through Spatial Proximity Regions around Mobile Devices on Augmented Tabletops. In *3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems*. TABLETOP '08. IEEE, pp. 1–8.

KRUEGER, M.W., GIONFRIDDO, T. AND HINRICHSEN, K., 1985. VIDEOPLACE - An Artificial Reality. *SIGCHI Bull.*, 16(4), pp.35–40.

KRUMM, J. AND HINCKLEY, K., 2004. The NearMe wireless proximity server. In *Lecture notes in computer science*. UbiComp 2004: Ubiquitous Computing. Springer, pp. 283–300.

LANGHEINRICH, M., 2010. Privacy in Ubiquitous Computing. In J. Krumm, ed. *Ubiquitous Computing Fundamentals*. Boca Raton, Florida, USA: CRC Press, pp. 37–94.

LEAHU, L., SENGERS, P. AND MATEAS, M., 2008. Interactionist AI and the promise of ubicomp, or, how to put your box in the world without putting the world in your box. In *Proceedings of the 10th international conference on Ubiquitous computing*. UbiComp '08. New York, NY, USA: ACM, pp. 134–143.

LEDO, D. AND GREENBERG, S., 2013. Mobile Proxemic Awareness and Control: Exploring the Design Space for Interaction with a Single Appliance. In *Proceedings ACM CHI 2013 Video Program*. CHI EA '13. New York, NY, USA: ACM, pp. 2831–2832.

LEE, J.C., AVRAHAMI, D., HUDSON, S.E., FORLIZZI, J., DIETZ, P.H. AND LEIGH, D., 2004. The Calder Toolkit: Wired and Wireless Components for Rapidly Prototyping Interactive Devices. In *Proceedings of the 5th ACM Conference on Designing Interactive Systems*. DIS '04. New York, NY, USA: ACM, pp. 167–175.

LI, F.C.Y., DEARMAN, D. AND TRUONG, K.N., 2009. Virtual shelves: interactions with orientation aware devices. In *Proceedings of the 22nd annual ACM Symposium on User Interface Software and Technology*. UIST '09. New York, NY, USA: ACM, pp. 125–128.

LI, Y., HONG, J.I. AND LANDAY, J.A., 2004. Topiary: a tool for prototyping location-enhanced applications. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. UIST '04. New York, NY, USA: ACM, pp. 217–226.

LUFF, P. AND HEATH, C., 1998. Mobility in collaboration. In *Proceedings of the ACM conference on Computer Supported Cooperative Work*. CSCW '98. ACM, pp. 305–314.

MACINTYRE, B., GANDY, M., DOW, S. AND BOLTER, J.D., 2004. DART: a toolkit for rapid design exploration of augmented reality experiences. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. UIST '04. New York, NY, USA: ACM, pp. 197–206.

MANKOFF, J., DEY, A.K., HSIEH, G., KIENTZ, J., LEDERER, S. AND AMES, M., 2003. Heuristic Evaluation of Ambient Displays. In *Proceedings of the 21st ACM Conference on Human Factors in Computing Systems*. CHI '03. New York, NY, USA: ACM, pp. 169–176.

MARQUARDT, N. AND GREENBERG, S., 2007. Distributed Physical Interfaces with Shared Phidgets. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. TEI '07. New York, NY, USA: ACM, pp. 13–20.

MARQUARDT, N., JOTA, R., GREENBERG, S. AND JORGE, J.A., 2011. The continuous interaction space: interaction techniques unifying touch and gesture on and above a digital surface. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction*. INTERACT '11. Berlin, Heidelberg: Springer, pp. 461–476.

MARQUARDT, N., TAYLOR, A., VILLAR, N. AND GREENBERG, S., 2010. Rethinking RFID: Awareness and Control for Interaction with RFID Systems. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. CHI '10. New York, NY, USA: ACM, pp. 2307–2316.

MARSHALL, P., ROGERS, Y. AND PANTIDI, N., 2011. Using F-formations to analyse spatial patterns of interaction in physical environments. In *Proceedings of the ACM 2011 conference on Computer Supported Cooperative Work*. CSCW '11. New York, NY, USA: ACM, p. 445.

MATTHEWS, T., DEY, A.K., MANKOFF, J., CARTER, S. AND RATTENBURY, T., 2004. A toolkit for managing user attention in peripheral displays. In *Proceedings of the 17th annual ACM Symposium on User Interface Software and Technology*. UIST '04. New York, NY, USA: ACM, pp. 247–256.

MAYRHOFER, R., GELLERSEN, H. AND HAZAS, M., 2007. Security by Spatial Reference: Using Relative Positioning to Authenticate Devices for Spontaneous Interaction. In *Proceedings of the International Conference on Ubiquitous Computing*. UbiComp '07. Springer, pp. 199–216.

MENTIS, H.M., O'HARA, K., SELLEN, A. AND TRIVEDI, R., 2012. Interaction proxemics and image use in neurosurgery. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. New York, NY, USA: ACM, pp. 927–936.

MERRILL, D., KALANITHI, J. AND MAES, P., 2007. Siftables: towards sensor network user interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. TEI '07. New York, NY, USA: ACM, pp. 75–78.

MICROSOFT, 2013. Kinect for Windows. *Developer SDK, Toolkit & Documentation, http://www.microsoft.com/en-us/kinectforwindows/develop/ (accessed May 28, 2013)*.

MICROSOFT MSDN, 2013. Microsoft .NET Framework and Visual Studio. *http://msdn.microsoft.com/en-us/vstudio/ (accessed April 22, 2013)*.

MOSTAFA, A., GREENBERG, S., BRAZIL, E., SHARLIN, E. AND SOUSA, M., 2013. Interacting with Microseismic Visualizations. In *Proc. ACM CHI 2013 Extended Abstracts*. CHI EA '13. New York, NY, USA: ACM, pp. 1749–1754.

MYERS, B.A., HUDSON, S.E. AND PAUSCH, R., 2000. Past, Present, and Future of User Interface Software Tools. *ACM Transactions on Computer-Human Interaction*, 7(1), pp.3–28.

MYERS, B.A., PECK, C.H., NICHOLS, J., KONG, D. AND MILLER, R., 2001. Interacting at a Distance Using Semantic Snarfing. In *Proceedings of the 3rd international conference on Ubiquitous Computing*. UbiComp '01. London, UK, UK: Springer, pp. 305–314.

NACENTA, M.A., ALIAKSEYEU, D., SUBRAMANIAN, S. AND GUTWIN, C., 2005. A comparison of techniques for multi-display reaching. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '05. New York, NY, USA: ACM, pp. 371–380.

NACENTA, M.A., GUTWIN, C., ALIAKSEYEU, D. AND SUBRAMANIAN, S., 2009. There and Back Again: Cross-Display Object Movement in Multi-Display Environments. *Human-Computer Interaction*, 24(1), pp.170–229.

NACENTA, M.A., SALLAM, S., CHAMPOUX, B., SUBRAMANIAN, S. AND GUTWIN, C., 2006. Perspective cursor: perspective-based interaction for multi-display environments.

In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '06. New York, NY, USA: ACM, pp. 289–298.

NATURALPOINT, 2013. Motion Capture Systems by OptiTrack. *http://www.naturalpoint.com/optitrack/ (accessed April 2nd 2013)*.

NORMAN, D., 1988. *The psychology of everyday things*, New York, NY, USA: Basic Books.

NÖTH, W., 1995. *Handbook of semiotics*, Indiana University Press.

OLSEN, J., 2007. Evaluating user interface systems research. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*. UIST '07. New York, NY, USA: ACM, pp. 251–258.

OLWAL, A. AND FEINER, S., 2009. Spatially aware handhelds for high-precision tangible interaction with large displays. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*. TEI '09. New York, NY, USA: ACM, pp. 181–188.

OSMOND, H., 1957. Function as the basis of psychiatric ward design. *Mental Hospitals (Architectural Supplement)*, 8(4), pp.23–29.

OULASVIRTA, A. AND SALOVAARA, A., 2009. Ubiquitous Computing and the Concept of Context. In C. S. Ang & P. Zaphiris, eds. *Human Computer Interaction: Concepts, Methodologies, Tools, and Applications*. pp. 630–634.

PATTERSON, M.L., 1975. Personal Space: Time to burst the bubble? *Man-Environment Systems*, 67(5).

PETRI, H.L., HUGGINS, R.G., MILLS, C.J. AND BARRY, L.S., 1974. Variables Influencing the Shape of Personal Space. *Personality and Social Psychology Bulletin*, 1(1), pp.360–361.

PRANTE, T., RÖCKER, C., STREITZ, N., STENZEL, R., MAGERKURTH, C., VAN ALPHEN, D. AND PLEWE, D.A., 2003. Hello. Wall–Beyond Ambient Displays. In *Video and Adjunct Proceedings of Ubicomp Conference*.

PRICE, G.H. AND DABBS, J.M., 1974. Sex, Setting, and Personal Space: Changes as Children Grow Older. *Personality and Social Psychology Bulletin*, 1(1), pp.362–363.

PRIMESENSE, 2013. OpenNI SDK. *http://www.openni.org (accessed March 27, 2013)*.

RAMOS, G., HINCKLEY, K., WILSON, A. AND SARIN, R., 2009. Synchronous Gestures in Multi-Display Environments. *Human-Computer Interaction*, 24(1), pp.117–169.

REITMAYR, G. AND SCHMALSTIEG, D., 2005. OpenTracker: A flexible software design for three-dimensional interaction. *Virtual Reality*, 9(1), pp.79–92.

REKIMOTO, J., 1997. Pick-and-drop: A Direct Manipulation Technique For Multiple Computer Environments. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. UIST '97. New York, NY, USA: ACM, pp. 31–39.

REKIMOTO, J., AYATSUKA, Y., KOHNO, M. AND OBA, H., 2003. Proximal Interactions: A Direct Manipulation Technique for Wireless Networking. In *Proceedings of the IFIP International Conference on Human-Computer Interaction*. INTERACT '03. IFIP, pp. 511–518.

REKIMOTO, J. AND SAITOH, M., 1999. Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '99. New York, NY, USA: ACM, pp. 378–385.

RESNICK, M., MYERS, B., NAKAKOJI, K., SHNEIDERMAN, B., PAUSCH, R., SELKER, T. AND EISENBERG, M., 2005. Design Principles for Tools to Support Creative Thinking. In *Report of Workshop on Creativity Support Tools, http://repository.cmu.edu/isr/816*. Institute for Software Research, Paper 816.

ROGERS, Y., 2012. *HCI Theory: Classical, Modern, and Contemporary* 1st ed., Morgan & Claypool Publishers.

ROGERS, Y., 2006. Moving on from Weiser's Vision of Calm Computing: Engaging UbiComp Experiences. In *Proceedings of the Eighth International Conference on Ubiquitous Computing*. UbiComp '06. Springer, pp. 404–421.

ROGERS, Y., 2004. New theoretical approaches for human-computer interaction. *Annual Review of Information Science and Technology*, 38(1), pp.87–143.

RUDD, J., STERN, K. AND ISENSEE, S., 1996. Low vs. High-Fidelity Prototyping Debate. *ACM interactions*, 3(1), pp.76–85.

SAKURAI, S., ITOH, Y., KITAMURA, Y., NACENTA, M.A., YAMAGUCHI, T., SUBRAMANIAN, S. AND KISHINO, F., 2008. Interactive Systems. Design, Specification, and Verification. In T. C. Graham & P. Palanque, eds. Berlin, Heidelberg: Springer, pp. 252–266.

SALBER, D., DEY, A.K. AND ABOWD, G.D., 1999. The Context Toolkit: Aiding the Development of Context-Enabled Applications. In *Proceedings of the ACM Conference*

on Human Factors in Computing Systems. CHI '99. New York, NY, USA: ACM, pp. 434–441.

SANDOR, C. AND KLINKER, G., 2005. A rapid prototyping software infrastructure for user interfaces in ubiquitous augmented reality. *Personal and Ubiquitous Computing*, 9(3), pp.169–185.

SCHILIT, B., ADAMS, N. AND WANT, R., 1994. Context-Aware Computing Applications. In *IEEE Workshop on Mobile Computing Systems and Applications*. Los Alamitos, CA, USA: IEEE, pp. 85–90.

SCHMIDT, D., CHEHIMI, F., RUKZIO, E. AND GELLERSEN, H., 2010. PhoneTouch: a technique for direct phone interaction on surfaces. In *Proceedings of the 23nd annual ACM Symposium on User Interface Software and Technology*. UIST '10. New York, NY, USA: ACM, pp. 13–16.

SCHWARZ, J., HUDSON, S., MANKOFF, J. AND WILSON, A.D., 2010. A framework for robust and flexible handling of inputs with uncertainty. In *Proceedings of the 23nd annual ACM Symposium on User Interface Software and Technology*. UIST '10. New York, New York, USA: ACM, pp. 47–56.

SHNEIDERMAN, B., 2006. A Second Path to HCI Innovation: Generative Theories Tied to User Needs. In *ACM CHI 2006 Workshop Position Paper: What is the Next Generation of Human-Computer Interaction?*

SHOEMAKER, G., TANG, A. AND BOOTH, K.S., 2007. 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, pp. 53–56.

SNIBBE, S.S. AND RAFFLE, H.S., 2009. Social immersive media: pursuing best practices for multi-user interactive camera/projector exhibits. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*. CHI '09. New York, NY, USA: ACM, pp. 1447–1456.

SOMMER, R., 2002. Personal Space in a Digital Age. In R. B. Bechtel & A. Churchman, eds. *Handbook of environmental psychology*. John Wiley and Sons, pp. 647–660.

SOMMER, R., 1969. *Personal space: the behavioral basis of design.*, Englewood Cliffs, N.J: Prentice-Hall.

SOMMER, R., 1959. Studies in Personal Space. *Sociometry*, 22(3), pp.247–260.

STREITZ, N., PRANTE, T., MÜLLER-TOMFELDE, C., TANDLER, P. AND MAGERKURTH, C., 2002. Roomware©: the second generation. In *Extended Abstracts on Human Factors in Computing Systems*. CHI EA '02. ACM New York, NY, USA, pp. 506–507.

STREITZ, N., PRANTE, T., RÖCKER, C., ALPHEN, D. VAN, MAGERKURTH, C., STENZEL, R. AND PLEWE, D., 2003. Ambient displays and mobile devices for the creation of social architectural spaces. In *Public and Situated Displays - Social and Interactional Aspects of Shared Display Technologies*. The Kluwer International series on Computer Supported Cooperative Work. Dordrecht: Kluwer, pp. 387–409.

STREITZ, N.A., GEISLER, J., HOLMER, T., MÜLLER-TOMFELDE, C., REISCHL, W., REXROTH, P., SEITZ, P. AND STEINMETZ, R., 1999. i-LAND: an interactive landscape for creativity and innovation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '99. New York, NY, USA: ACM, pp. 120–127.

SUNDSTROM, E. AND ALTMAN, I., 1976. Interpersonal relationships and personal space: Research review and theoretical model. *Human Ecology*, 4(1), pp.47–67.

SWINDELLS, C., INKPEN, K.M., DILL, J.C. AND TORY, M., 2002. That one there! Pointing to establish device identity. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*. UIST '02. New York, NY, USA: ACM, pp. 151–160.

TANDLER, P., PRANTE, T., MÜLLER-TOMFELDE, C., STREITZ, N. AND STEINMETZ, R., 2001. Connectables: dynamic coupling of displays for the flexible creation of shared workspaces. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*. UIST '01. New York, NY, USA: ACM, pp. 11–20.

TERRENGHI, L., QUIGLEY, A. AND DIX, A., 2009. A taxonomy for and analysis of multi-person-display ecosystems. *Personal and Ubiquitous Computing*, 13(8), pp.583–598.

TIDWELL, J., 2005. *Designing Interfaces: Patterns for Effective Interaction Design*, O'Reilly Media, Inc.

ULLMER, B., ISHII, H. AND GLAS, D., 1998. mediaBlocks: physical containers, transports, and controls for online media. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '98. New York, NY, USA: ACM, pp. 379–386.

UNDERKOFFLER, J. AND ISHII, H., 1999. Urp: a luminous-tangible workbench for urban planning and design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '99. New York, NY, USA: ACM, pp. 386–393.

VARSHAVSKY, A. AND PATEL, S., 2010. Location in Ubiquitous Computing. In J. Krumm, ed. *Ubiquitous Computing Fundamentals*. Boca Raton, Florida, USA: CRC Press, pp. 285 – 319.

VERTEGAAL, R. AND SHELL, J.S., 2008. Attentive user interfaces: the surveillance and sousveillance of gaze-aware objects. *Social Science Information*, 47(3), pp.275–298.

VICON MOTION SYSTEMS, 2013. Nexus software. *http://www.vicon.com/products/nexus.html (accessed March 22nd, 2013)*.

VILLAR, N. AND GELLERSEN, H., 2007. A Malleable Control Structure for Softwired User Interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction*. TEI '07. New York, NY, USA: ACM, pp. 49–56.

VOELKER, S., WEISS, M., WACHARAMANOTHAM, C. AND BORCHERS, J., 2011. Dynamic portals: a lightweight metaphor for fast object transfer on interactive surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ITS '11. New York, NY, USA: ACM, pp. 158–161.

VOGEL, D. AND BALAKRISHNAN, R., 2004. Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*. UIST '04. New York, NY, USA: ACM, pp. 137–146.

WANG, M., 2012. *The Proxemic Peddler Framework: Designing a Public Display that Captures and Preserves the Attention of a Passerby*. Calgary, Alberta, Canada: MSc Thesis, Department of Computer Science, University of Calgary.

WANG, M., BORING, S. AND GREENBERG, S., 2012. A Public Advertising Display that Captures and Preserves the Attention of a Passerby. In *Proceedings of the 2012 International Symposium on Pervasive Displays*. PerDis '12. New York, NY, USA: ACM.

WANT, R., FISHKIN, K.P., GUJAR, A. AND HARRISON, B.L., 1999. Bridging Physical and Virtual Worlds with Electronic Tags. In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. CHI '99. New York, NY, USA: ACM, pp. 370–377.

WANT, R., HOPPER, A., FALCÃO, V. AND GIBBONS, J., 1992. The Active Badge Location System. *ACM Transactions on Information Systems*, 10(1), pp.91–102.

WEIGEL, M., BORING, S., STEIMLE, J., MARQUARDT, N., GREENBERG, S. AND TANG, A., 2013. *ProjectorKit: Easing the Development of Interactive Applications for Mobile Projectors.*,

Calgary, Alberta, Canada: Technical Report #2013-1041-08, Department of Computer Science, University of Calgary.

WEIGEL, M., TANG, A., BORING, S., MARQUARDT, N. AND GREENBERG, S., 2012. *From Focus to Context and Back: Combining Mobile Projectors and Stationary Displays*, Calgary, Alberta, Canada: Technical Report #2012-1031-14, Department of Computer Science, University of Calgary.

WEISER, M., 1991. The Computer for the 21st Century. *Scientific American*, 265(3), pp.94–104.

WEISER, M., 1994. The world is not a desktop. *ACM interactions*, 1(1), pp.7–8.

WEISER, M. AND BROWN, J.S., 1996. Designing Calm Technology. *PowerGrid Journal*, 1, p.1.

WIGDOR, D., JIANG, H., FORLINES, C., BORKIN, M. AND SHEN, C., 2009. WeSpace: the design development and deployment of a walk-up and share multi-surface visual collaboration system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '09. New York, NY, USA: ACM, pp. 1237–1246.

WIGDOR, D., SHEN, C., FORLINES, C. AND BALAKRISHNAN, R., 2006. Table-centric interactive spaces for real-time collaboration. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '06. New York, NY, USA: ACM, pp. 103–107.

WILSON, A.D. AND BENKO, H., 2010. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM Symposium on User Interface Software and Technology*. UIST '10. New York, NY, USA: ACM, pp. 273–282.

ZHAO, T., AGGARWAL, M., KUMAR, R. AND SAWHNEY, H., 2005. Real-Time Wide Area Multi-Camera Stereo Tracking. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. CVPR '05. IEEE Computer Society, pp. 976–983.