LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
Department “Institut für Informatik”
Chair of Human-Machine-Interaction
Prof. Dr. Andreas Butz

Diploma Thesis

Visualization of and Interaction with Digital Devices around Large Surfaces as a Function of Proximity

Till K. Ballendat
till@ballendat.info

Time needed for completion:	15.08.2010 to 15.02.2011
Supervisors:	Nicolai Marquardt M.Sc. and Prof. Dr. Saul Greenberg
Professor in charge:	Prof. Dr. Andreas Butz

Abstract

People increasingly use a variety of personal and shared electronic devices in everyday situations. Many of them are connected over a network, but still interacting in between them can be very tedious, as one has to know which devices can intercommunicate, how devices can be addressed, what information they contain and how information can be exchanged. Why could digital interaction not be as easy as physical interaction? *Proxemic Interaction* [1] is an approach that exploits spatial relationships to ubiquitous computing environments. In this work I employ and further explore this approach under the umbrella of information exchange between devices. I contribute concepts for inter-device proxemic interaction. These range from how proxemics can create awareness of other devices and their content to how information exchange can be facilitated and how interaction techniques can profit from knowledge about fine grained proxemic dimensions. I illustrate them with applications for small space digital environments that enable information exchange between different devices, such as digital cameras, large displays and personal tablet computers.

Zusammenfassung

Eine Vielzahl digitaler Endgeräte bereichert unser tägliches Leben. Es gibt Sie in unterschiedlichsten Größen und Formen, ob mobil oder stationär, ob persönlich oder zur gemeinschaftlichen Nutzung. Sie sind allgegenwärtig und können über Datennetzwerken miteinander verbunden werden. Dies gestaltet sich aber oft schwierig: Woher weiß man ihren Namen im Netzwerk? Wie kann man ein Gerät im Netzwerk finden? Woher weiß man welche Daten übertragen werden können und wie initialisiert man Datentransfers? Warum kann das nicht so einfach sein wie in unserer physikalischen Welt, in der man einfach Gegenstände nehmen und Anderen geben kann? Mit der Frage, ob man physikalische Gesetzmäßigkeiten nicht auch für die Welt der Computer verwenden kann, um deren Interaktion zu vereinfachen, haben sich Forscher im Bereich "Proxemic Interaction" beschäftigt [1]. Meine Arbeit baut auf dieser Forschung auf. Sie beschäftigt sich aber im Speziellen damit, wie der Informationsaustausch zwischen verschiedenen digitalen Endgeräten er- leichtert werden kann, wenn Geräte räumliche Informationen über andere Geräte und ihre Nutzer haben. Ich stelle Interaktionskonzepte vor, die es erleichtern Geräte in der näheren Umgebung aufzufinden und die erkennen lassen, welche Informationen mit Anderen ausgetauscht werden können und präsentiere Techniken die Datenaustausch mit Aktionen im Raum anstoßen. Diese Konzepte verdeutliche ich mit konkreten Beispielanwendungen, die die genaue Positionen anderer Geräte im Raum kennen.

Task definition

People increasingly use personal and shared electronic devices in a variety of everyday situations. Examples for personal devices include mobile phones and media players, digital cameras, and tablet computers, whereas shared devices are for instance large interactive displays or TV sets at home. Currently, even though these devices are often connected over a network, almost all are unaware of other devices that are nearby. A few research systems do recognize nearby devices in either very specific or somewhat crude ways [32, 18]. As with that work, I want to explore interaction techniques and visualizations that illustrate how one device can connect, reveal information and allow information exchange with other devices as a function of proximity. Unlike this existing work, I will investigate fine-grained proximity information and use that to continually change what the interface shows and how users can explore and trigger information exchange between devices.
More precisely I consider an extended notion of proximity that takes four dimensions into account. Those are position and orientation between different devices and people in a room and identity and movement of entities. I will look at proximal relations of devices like a media player, digital camera or tablet computer and people around a large vertical display.
One focus of this work will be to explore the different proximity dependent visualization possibilities of entities and their content on large vertical display and if applicable on a device screen. How can the visualization create awareness and reveal additional content? Does it therefore change continuously or in discrete steps and what content is shown on which screen? A second focus is to examine the interaction concept when using devices around a large display. How does the extended notion of Proximity determine different functions of a device and in which way can a person interact on the device or on the display. How do interaction possibilities change depending on their proximity.
As a possible extension of this work I might explore how multiple people with multiple devices can use the system, exchange information and work collaboratively. How can proximal cues be interpreted and how can the system react to better support those tasks?
An example application that demonstrates these concepts will probably start with having picture managing abilities. This might extend to a more sophisticated application design, which is capable of managing multiple types of media. So the application can support people‘s everyday interaction with the digital world both in a home environment setting or in a computer supported collaborative work scenario. The technical implementation of this application will use the proximity toolkit framework [11]. Tracking of people and devices will mainly be done by Vicon motion tracking cameras but with the open distributed architecture of the toolkit it might be possible to use technologies like RFID or Infrared sensors in simple examples. The network communication between the large display and other devices will use the .NetwokingGT toolkit [10].

Herewith I declare that this document and the accompanying code has been composed by myself and with only the help of the mentioned references, unless otherwise acknowledged in the text.

München, April 21, 2011

. . . . . .

Contents

1 Introduction

In everyday live we use the physical space around us - we interpret and perceive spatial relationships between objects and other people. These relationships have meanings in a social context and define how people communicate and interact. Edward Hall‘s studies describe how people perceive and use distance, posture and orientation as a non-verbal form of communication when interacting with others in everyday situations [22]. In this work I explored how proxemics can be exploited when interacting with a variety of digital devices - how they can facilitate the design of interfaces and trigger meaningful behaviour. I will revisit concepts of *Proxemic Interaction* [1] and present how I applied and further explored these concepts in two applications that focus on interaction between digital devices.

1.1 Motivation

When looking at peoples everyday environment one can see that it is more and more sculpted by a variety of digital devices in many form factors (see figure 1.1). Mark Weiser called this *Ubiquitous Computing* and envisioned in his influential work ’The Computer for the 21st Century’ [70] what in many cases has become reality by now. He wrote: *’Ubiquitous computers will also come in different sizes, each suited to a particular task. These machines and more will be interconnected in an ubiquitous network’*. It is true, that today many devices are connected over a network. Mobile phones, tablet computers, laptop computers, TV‘s and others are often connected to the internet or a local network using wireless technologies and can thereby communicate with each other. But Weiser continues his thoughts: *’Therefore we are trying to conceive a new way of thinking about computers in the world, one that takes into account the natural human environment and allows the computers themselves to vanish into the background’*. By now hardly any devices take into account their immediate natural physical environment. They mostly don’t know where objects, people and other devices are, how far things are apart and how they are oriented towards each other. When interacting with surrounding devices, the technology itself comes to the foreground of attention - meaning that people have to know a lot about the underlying functionality, how to operate interfaces and how to set up connections to interoperate between devices.
Spatial relationships play an important role in how we physically interact, communicate and engage with other people and objects in our everyday environment. *Proxemics* is Edward Hall‘s theory of interpersonal relationships [22]. It describes how people perceive, interpret and use distance, posture and orientation to mediate relations to other people, and to their environment.
Proxemic Interaction [1] which is an approach based on proxemic theory and imagines a world of devices that has fine-grained knowledge of nearby people and other devices - how they move in range, their precise distance and their orientation - and how such knowledge can be exploited in interaction design. The motivation of this work is the believe, that detailed knowledge about spatial relations can help to design systems that let people focus on their task rather than the technology. Why can sending a digital document not be as easy and accessible for everyone as handing a piece of paper?
I want to find out how proxemics can facilitate interaction between devices and explore the design space of such interfaces.

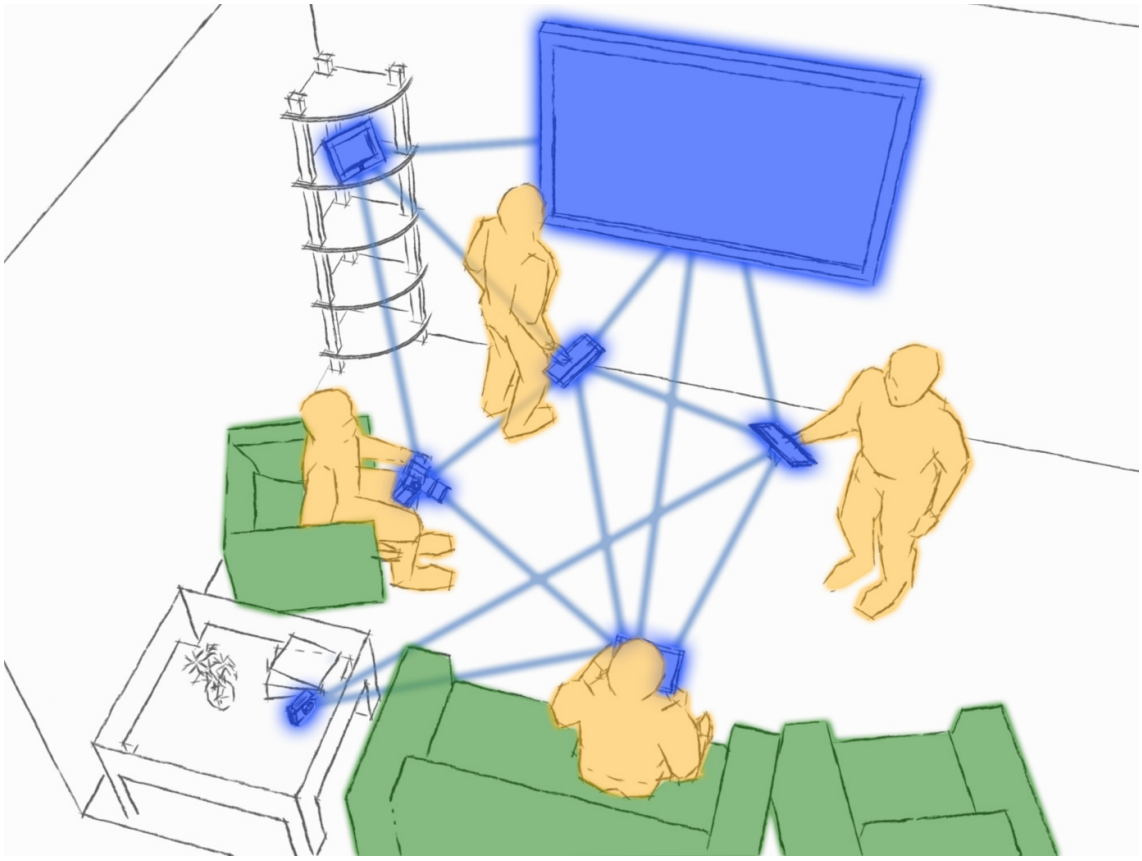

Figure 1.1: An ubiquitous computing scenario including devices of different sizes and form factors - all connected over a network - and people interacting with them in a room sized environment.

1.2 Background

Although this work is applied in the context of ubiquitous computing [70], it is rooted in concepts from sociology and anthropology. Namely Edward T. Hall who defined the term *Proxemics* [21]. It describes how people use interpersonal distance to understand and mediate their interactions with others. One relevant finding is the definition of *proxemic zones* which are based on these distances and each provide an implication of the social meaning attached to it: intimate (less than 50cm), personal (0.5 to 1m), social (1 to 4m), and public (greater than 4m). As the terms suggest, the distances lend themselves to a progression of interactions ranging from highly intimate to personal, to social and then to public.

Based on Hall‘s proxemic zones, other researchers in the field of Human Computer Interaction consider distances between people and displays. Hello.Wall [57] introduced the concept of distance dependent semantics. Their system could discriminate peoples distances to a wall by three zones. Each zone triggered different visualizations or offered interaction possibilities. Also closely related to Hall‘s zones, Vogel and Balakrishnan defined four interaction zones that affect how a large digital vertical surface reacts to one or more approaching people [65]. It supports ambient display of notification in the outermost zone, then implicit and subtle information and finally offering explicit personal interaction with calender or email messages in the innermost zone. Similarly Ju et. al presented a system of an interactive whiteboard with distance dependent zones around it [30]. In-

teraction from afar was considered implicit and triggered ambient awareness by showing recent drawings. When people moved closer to the board, interaction became more explicit and automatically cleared space on the whiteboard for people to draw or sketch by using a pen. My applications also make use of proxemic zones around a large display, not only changing the visualization but offering appropriate input modalities in each zone. While these approaches exploit only *discrete* distance regions, I combine them with visualizations that change *continuously* depending on distance.

Harrison presented a technique called *Lean and Zoom* which reacts to continuous distance changes of a person's hat towards a computer screen [23]. The content on the display changes its zoom factor or the level of detail depending on the distance.

In these cases distance is a discrete or continuous measure between two entities. In the simplest case, this is just binary - telling whether one entity is or is not in the same room as another entity or whether two entities are placed next to each other or are apart. A simple example is the *Smart Light Switch* [8] which reacts to the presence of people in a room. The *ActiveBadge* [68] system provided both identity and binary presence in rooms of a larger building. This information was used to forward phone calls appropriately when a person was not at his desk.
Later Want introduced the technique of detecting nearby objects and devices through attached RFID tags [67]. He showed how a tablet computer could trigger certain actions when bringing it close to other objects. Rekimoto [47] combined RFID and Infrared to detect other devices and establish a connection between them. Holmquist establishes a connection between devices by simultaneously shaking them [27]. The approach was called *context proximity* where similarities in sensor values determine close proximity. Another approach by Hinckley that uses accelerometer sensors is to bump devices into each other [26]. His technique could even identify the side, where the bump of the other device came from. For a survey on techniques to transfer media between displays (both based and not-based on their spatial relationships) see [38].
These techniques are powerful for connecting devices that are in very close proximity or - like in many cases - are even directly touching one another. Swindells [59] introduced a similar technique that worked from a larger distance, where he applied Infrared technology to the *GesturePen* for initiating remote pointing for device selection. Later projects [48, 3] used a PDA like device to point at other devices or appliances to initiate interaction. I extend on this prior work, by using interaction techniques that go beyond a binary device connection state - they move from awareness at a larger distance, to gradually revealing of more and more detail, to direct interaction for transferring digital information between devices.

Spatial relations have also been used to mediate the information exchanged between devices. For example, Kray's group coordination negotiation [12] introduced spatial regions around mobile phones. These regions were visualized in a tabletop and could be used to negotiate exchange of information with others. Depending on how devices were moved in and out of three discrete regions, the transfer of media data between the devices is initiated. I extend their approach to interaction around large surfaces, where the degree of shared information between devices depends not only on their relative distance, but also orientation.

Gellersen's *RELATE Gateways* [18] provided a spatial-aware visualization of nearby devices. A graphical map showed the spatial room layout, and icons indicated the position of other nearby devices. Similar to Rekimoto's approach [47], alternatively icons at the

border of a mobile device screen represented the type and indicated the location of surrounding devices, as the icon was oriented towards the physical device. My interfaces are inspired by this idea, but I extend their notion with visualizations that include proximity dependent level of detail, and with techniques that move from awareness to direct interaction depending on distance and orientation of devices to the display.

1.3 Problem Statement

With hardware components like microprocessors or display getting cheaper, more powerful and smaller year by year, the number of digital devices in peoples everyday environment rises significantly. Therefore we are surrounded by devices of different form factors. Ranging from small and mobile devices like cameras, smart phones and tablet computers to information appliances like digital picture frames to large interactive displays. While these devices are often connected over a network, a huge amount of work and knowledge is required, and it can be very tedious, to interact between multiple digital devices. Hence, my higher level goal is to design applications and explore interaction concepts that facilitate the interaction between multiple devices in small Ubicomp environments by exploiting spatial relationships.In particular there are three important questions that arise from this problem and are very relevant for this work:

1. **How does a person know which devices can intercommunicate in his immediate surrounding?**
   With the multitude of network technologies, communication protocols and types of data that devices can support or not support, often it is not obvious which devices can be interconnected and exchange information.

2. **What kind of information do these devices contain and which interactions do they support?**
   For one to find out which devices he wants to interact with, it is essential to be aware of the information devices currently contain and in which ways this content can be accessed or shared.

3. **How can a person address a particular digital device in his nearby environment?**
   In many cases a piece of digital technology can be easily identified and addressed by physical means. This however in most cases does not solve the problem of identifying it in a digital network interface. So are there means of proximity that better support this task?

4. **How can a person share information between devices?**
   Once a device is addressed, what are the interaction possibilities? Do they build upon natural expectations and are they easy to use?

For this thesis I designed applications for multi-device ubiquitous environments that try to address these problems by applying concepts of Proxemic Interaction [1].

1.4 Goals and Methods

The goal of this thesis is to explore how proxemics can facilitate interaction between devices in small space Ubicomp environments. prior work by Marquardt, Greenberg and Ballendat on Proxemic Interactions [1, 19] extracted five *Proxemic Dimensions* for ubiquitous computing environments - Distance, Orientation, Movement, Identity and Location - and also provided a set of general *Concepts of Proxemic Interaction*. While they were

illustrated with an example of a proxemic aware media player, I want to apply these concepts to other application domains - focusing on the interaction between different devices. To do so I developed applications that include a variety of different digital devices. Applications should allow to explore the relation of people to one or many vertical interactive surfaces and people using mobile digital devices to interact with those shared surfaces and in between their devices.
The goal is to apply the concepts in a way that makes interaction more natural and easier because it exploits people's behaviour in the spatial environment of everyday life. The interfaces should react in a meaningful way building upon people's expectations and visualizations should create awareness of other people, devices and interaction possibilities - leading to seamless and easy to accomplish exchange of information facilitated through proxemics.
To achieve this I employ technology that senses fine grained proxemic values in a room sized environment. This includes precise and real time tracking of people, objects and mobile digital devices as well as knowledge about fixed and semi fixed features like walls, furniture and interactive vertical displays.

1.5 Results and Contributions

I applied concepts of Proxemic Interaction [1] in two applications - thereby illustrating how these techniques can work in Ubicomp scenarios that include a variety of mobile or fixed, shared or personal digital devices of different form factors.
In detail I apply the five proxemic dimensions [1] to specific interaction techniques illustrated by the Proxemic Photo Canvas and Proxemic Brainstorming application. With this I build on prior work by further exploring how information about movements and spatial relations between digital devices can facilitate interaction. The main contribution of this thesis is the application and further exploration of the following concepts of proxemic interaction:

- **Creating Awareness through Continuous Visualization of Spatial Relations between Devices:** Visualizing the proxemic dimensions between devices helps to identify surrounding devices by spatial relation - it can create awareness of other devices, their content and interaction possibilities - while the meaning associated to closeness and orientation can be exploited in the level of information shown and detail of interaction offered.
  I apply this concept by showing spatially related representations of other devices on a large vertical display or personal tablet computer screens. These representations continuously change their position, their size and the level of detail about the representing device, revealing content like images of a camera or sticky notes of a tablet while devices approach another.

- **Using Discrete Proxemic Zones for Appropriate Interaction Modalities:** Proxemic zones are defined by the relative distance to a device or the location in the environment. People can use different modalities of interaction depending on the zone in which they are located. Interaction modalities should consider the notion of providing more explicit and private interaction with detail control in a closer zone ranging to more implicit and public interaction from afar.
  I illustrate this concept in several examples: being within arms reach of an interactive surface reveals controls for touch interaction. From a further distance interaction is possible through pointing, gestures, movements or remote control with personal devices. sitting down on a couch or chair triggers an ambient but controllable presentation.

- **Explicit Interaction through Physical Devices:** People can use their physical devices in the surrounding space in a meaningful way to interact with a system. My examples range from executing a throw gesture to pointing or tilting a device to using a device to touch other surfaces.

1.6 Thesis Overview

I shortly introduce the contents of the remaining seven chapters of this thesis:

2. **Related Work:**
   In this chapter I provide an overview of the research in Ubiqutos Computing, Human Computer Interaction and sociology, that relates to my thesis topic. I start by introducing the basic idea of Ubicomp and the research about proximity in a social context. Next I present work where HCI researchers employ spatial information about people in their environment. I will revisit interaction and visualization on large vertical displays in general and then focus on projects that consider proximity when interacting with those surfaces. Finally I focus on device to device interaction using proximity.

3. **Introduction to Proxemic Interaction:**
   In this chapter I introduce our earlier work about *Proxemic Interaction* [1]. First I explain how Ubicomp relates to proximity in a social context. Next I explain the five *Proxemic Dimensions* and then present our concepts of Proxemic Interaction. I will illustrate these concepts with two applications - the Proxemic aware Media Player and a game called Proxemic Pong.

4. **Proxemic Applications:**
   The main part of my thesis project is the development of two applications. They both integrate a large interactive display and a number of interconnected digital devices in a small space environment. Their design allows the exploration of concepts of proxemic interaction and how they apply to interaction between digital devices. The *Proxemic Photo Canvas* let people explore and exchange information in between their digital camera, the large surface and a digital picture Frame. The *Proxemic Brainstorming* application incorporates mobile tablet computers in combination with a large display to create, present and exchange ideas on digital sticky notes.

5. **Concepts of Proxemic Interaction between Digital Devices:**
   In this chapter I discuss how concepts of *Proxemic Interaction* apply to interaction between digital devices of different kinds. I explain how I applied existing concepts and how I extended and build upon them, therefore presenting a selection of the most important ones in terms of interaction between devices. All of these concepts are illustrated by example applications that I implemented.

6. **Implementation:**
   In this chapter I describe the challenges and important techniques for the implementation of the proxemic aware applications. I start by introducing tracking technology and how to access it, then present a library of modules and widgets that I built to use in our own applications and to provide higher level access to common tasks related to proxemic applications. Finally I explain details about networking, user interfaces and other design challenges in our three applications - Proxemic Pong, Proxemic Photo Canvas and Proxemic Brainstorming.

7. **Discussion:**
   In this chapter I discuss chances and benefits, but also challenges and issues of Proxemic Interaction. This is for example how to configure the *rules of behaviour* for a system that reacts to certain spatial relations, which problems can arise, and how a system can repair mistakes it made.

8. **Conclusion and Future Work:**
   I sum up the work of this thesis and explain how I addressed the problems stated earlier and which contributions I made. Finally I identify topics that might be beneficial to further explore in future research.

1.7 Terms and Definitions

- **HCI:** will be used as an abbreviation for Human Computer Interaction throughout the thesis.
- **Ubicomp:** will be used as an abbreviation for ubiquitous computing throughout the thesis.
- **Proxemics:** with proxemics I mean the five proxemics dimensions - distance, orientation, movement, identity and location - that describe the relation between entities.
- **Visualization:** I will use this term for visual representations in our applications that are not interactive - thereby not meaning the research field of Information Visualization.

2 Related Work

2.1 Proximity in Ubiquitous Computing Systems

This work considers the modern digital environment which is more and more sculpted by a variety of digital devices in many form factors. Mark Weiser called it the world of *Ubiquitous Computing* and envisioned in his influential work 'The Computer for the 21st Century' [70] what in many cases has become reality by now. He said: 'Ubiquitous computers will also come in different sizes, each suited to a particular task. These machines and more will be interconnected in an ubiquitous network'. It is true, that many devices are connected over a network. Mobile phones, tablet computers, laptop computers and others often include technologies like Bluetooth or Wi-Fi together with the necessary network protocols like TCP/IP and HTTP that allow them to communicate with each other. But Weiser continues his thoughts: 'Therefore we are trying to conceive a new way of thinking about computers in the world, one that takes into account the natural human environment and allows the computers themselves to vanish into the background.' By now hardly any devices take into account their natural physical environment. They mostly don't know where other devices and appliances are, where people are, how far things are apart and how they are oriented towards each other.

Concerning Human interaction - these relations have been studied extensively in the field of anthropology, environmental psychology and sociology. In 1968 Edward T. Hall defined the term *Proxemics* as the theory of interpersonal spatial relationships [22]. He studies how people perceive, interpret and use distance, posture and orientation to mediate relations to other people. Therefore he correlates physical distance with social distance and defines four proxemic distance zones ranging from intimate 0-50 cm, personal 0.5-1 m, social 1-4 m to public >4 m (illustrated in figure 2.1). He also considers relations to the natural environment which he calls the *fixed and semi-fixed feature space* and how it influences social behavior. The theory emphasizes the role of proxemic relationships on people's implicit communication.
Robert Sommer [55] defined and studied personal space and spatial invasion in general and focused on the use of space in small group ecologies. He discussed the influence of personality, task, environment and cultural differences on spatial arrangements in small groups.

2.2 Proxemic Relationships between People and Devices

But also computer scientists - mainly researchers in the field Human Computer Interaction - consider proxemics in the design of their systems. A very simple example is the Smart Light Switch [8] which only receives binary information about motion in a room and accordingly turns on or off a light, still providing the opportunity to use a soft touch switch for manually overriding the automatic behavior. In the same work Cooperstock et al. present the Reactive Room where one feature is, that a person can control the camera of a remote meeting location. To do so, a person can just lean to the left or right and the motorized camera pans until the remote screen shows the desired field of view.
Like these projects, which consider a very low level of binary presence sensing, my systems do also react to binary presence of people entering or leaving a room. The Proxemic Pong game for example starts when a person enters and switches into two player mode if a second person joins.

Active Badge [68] is a system that acquires binary room presence and the identity of

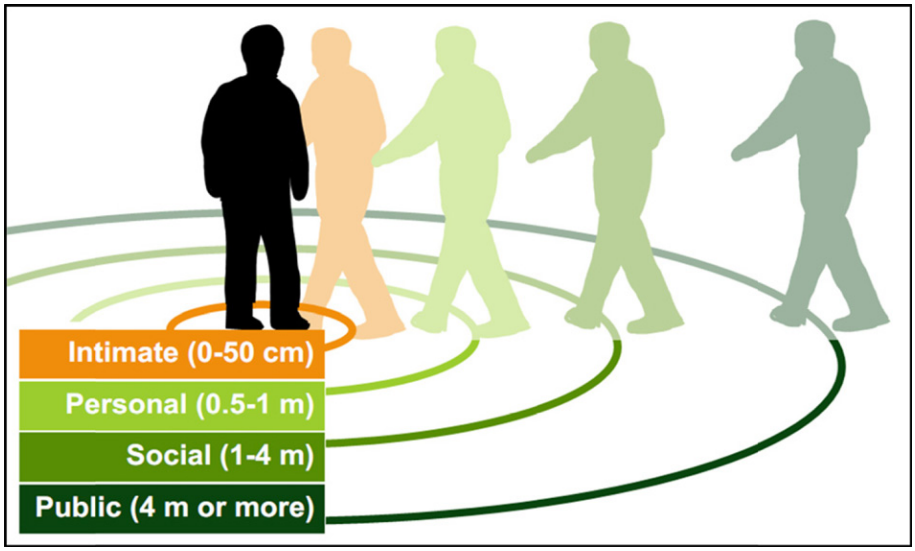

Figure 2.1: Edward Hall's proxemic zones.

people in larger buildings through an active tag that has to be carried around and continually emits an unique ID. The signal is picked up by a network of sensors distributed throughout a building. A master station processes the data and makes it available to client applications. One application is a table containing the current location of each person in a group of employees and the nearest telephone extension to that person. It supports a receptionist who has to forward phone calls.
Schilits Proximate Selection [52] uses rough location information of a mobile personal digital assistant (PDA) computer to adapt the interface (see figure 2.2). A list of possible devices emphasizes the ones that are nearby. This in particular makes sense for devices that require co-location in use like printers, displays, speakers or thermostats. My applications also consider the binary presence of devices and trigger appropriate actions. For example the Proxemic Photo Canvas recognizes when a camera is in the room. However it does not only consider binary presence of digital devices, but exploit information about fine grained spatial location.
There are a lot more projects that make use of knowledge about the environment around people in the research field of context- and location-awareness [52] in ubicomp systems. Most of them consider very coarse and large scale location information, which drifts away from this research focus - I try to make sense of fine grained small scale proximity information and go far beyond binary presence information.

2.3 Interaction with Large Displays

Large displays are increasingly deployed in a variety of public and semi public places and the HCI literature describes a huge number of use cases. They are conceptually different to conventional desktop screens, because of the social aspect and context of their deployment. This is why interaction design has to consider different aspects than traditional desktop applications. People can move around the display, stand in different distances to it and use their personal devices around them. So they might not actively work with or attend the display, but just use it for ambient information. Also several people might want to use it simultaneously or collaboratively. I want to give a short glance at the variety of approaches in the HCI research area for these situations.

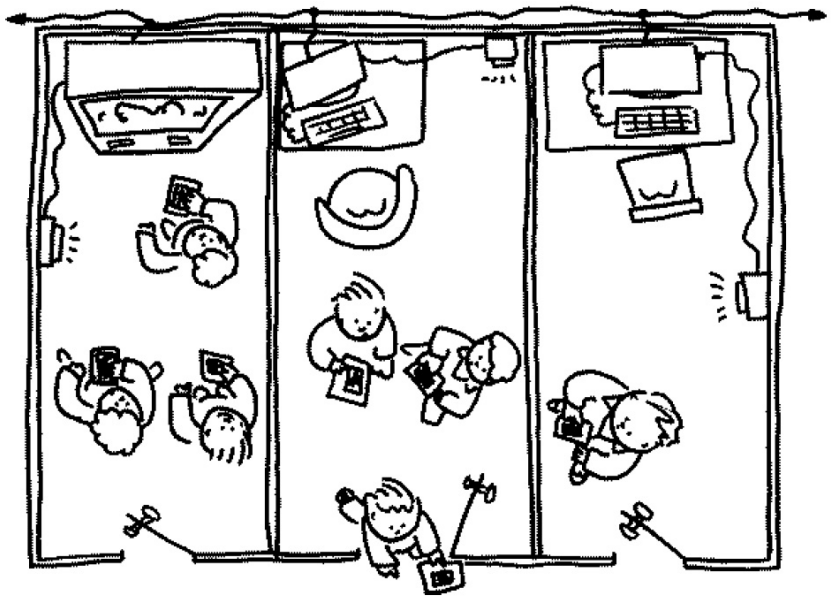

Figure 2.2: A context-aware computing system (PARCTAB) [52].

Concerning interaction, there are three primary approaches that let people manipulate work artifacts on a vertical display: people can walk up to the display and directly touch it, they can interact with the display at a distance using freehand pointing or gestures, or they can use indirect devices such as mice or PDA's to manipulate items on a large display. I will present an overview of research, that explores how people can interact with a large vertical display. Techniques range from direct to indirect interaction, in close to far vicinity to the display, single use to simultaneous collaboration and from simple pointing to file transfer capabilities. Input can be the bare hand, the whole body or a tangible - digital or non digital - device.

Developed in 1992, Liveboard by Elrod is one of the early approaches towards interactive whiteboard design [14]. It supports direct single point interaction on a large vertical surface using a pen. A whiteboard application allows a person to annotate charts or take short notes and a slideshow application lets people navigate back and forth between slides using simple swipe gestures.
In order to support multiple people interacting on a large wall surface, the Dynawall project tiled several single touch Smart Boards next to each other [58]. Each person could only interact simultaneously on separate tiles, which limits collaboration in terms of physical distance.
Interaction from a distance might be preferred or required in certain situations when using large displays. Therefore Vogt et al designed a large vertical screen environment where people can use laser pointers for distant interaction [66]. It is even possible to distinguish between pointers by using different blinking patterns and they where equipped with buttons for selection.
Dynamo is a large multi-user interactive surface that allows people to share, display and exchange media with others [29]. Every collaborator is provided with a mouse and keyboard and can interact with the system in a desktop computer fashion. Also it is possible to connect personal devices and access their data.
BlueBoard [49] employs a large interactive surface with a resistive touch screen, where people can access personal information by walking up to the display and identify them

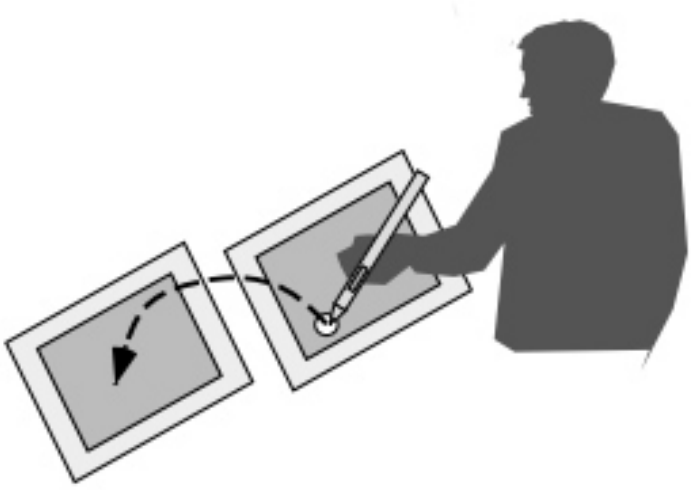

Figure 2.3: Rekimoto‘s Pick and Drop technique [44].

self with their RFID card. It is also possible to use the system collaboratively and exchange personal data.
Coming from large touch-enabled walls, an approach by Alexander Schick extends direct touch in a fluent manner to pointing with the whole arm from a larger distance [51]. This is accomplished by creating a 3D model of the people in front of the display by triangulating the video images of several webcams arranged around the area. This enables easy access of unreachable areas without explicitly switching the interaction mode.
Greenberg‘s Notification Collage [20] focuses on ambient visualization on public displays. A large screen is placed in an office environment and people can use their desktop computers to post a variety of different media elements to it - sticky notes, a series of photos, web pages or even live videos. This creates awareness and ambient information for other people coming by the screen and entices collaboration.
Pick and Drop [44] is a technique that lets people use a digital pen to grab data from one surface and transfer it to another one (see figure 2.3). In one example it is possible to touch images on the palmtop display and then touch the whiteboard to display them on the large surface. Similarly Rekimoto‘s implementation shows a tool palette on a palmtop computer. By touching a tool with the pen its function is assigned to the pen‘s ID and can be used on the whiteboard.
While these techniques provide solutions for interaction in different situations, they ignore information about proximity. Even if some techniques require the user to be at a certain distance from the large display, this is just a function of where people have to stand for the technique to work.

2.4 Large Proximity-aware Interactive Surfaces

Others however do consider spatial information about people or devices around large displays. This research topic is very relevant to this work, as many of the presented applications visualize spatial relationships of people and devices on a large display and also use the large display for interaction. An early approach is Chameleon [17] - a palmtop computer aware of its position and orientation. When used relative to a large display, Cameleon‘s content would vary depending on its spatial orientation to that surface. One example application for the large display shows a map and by bringing the Chameleon close to a city, it shows detailed information on the mobile device. Similar Rekimotos

Figure 2.4: Text can be modified with the M-Pad device while holding it in close proximity [45].

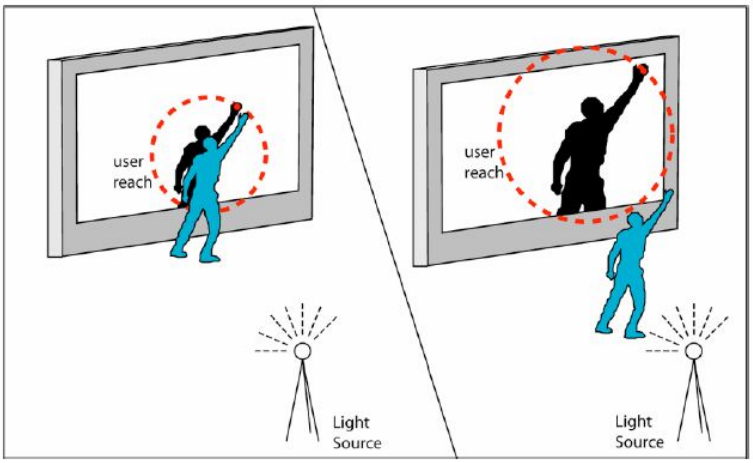

Figure 2.5: Shadow Reaching - a technique that lets people interact on a large display through the metaphor of their shadow [53].

system [45] knew about the rough position of a palmtop computer - the M-Pad - when holding it close to a whiteboard. People could drop data from the M-Pad to the screen - just next to where the device was held - by pressing a button. More interestingly he offered the possibility to edit a particular text from the whiteboard by holding the M-Pad in close proximity to it (see figure 2.4).
Shoemaker [53] developed a novel interaction concept for large wall-size displays that employs the metaphor of a light source located behind a person that stands in front of the screen and projects a shadow of the person‘s body. The person can resize the shadow by moving back and forth and control it with his movements in space as illustrated in figure 2.5. The shadow’s hands are used for interaction on the screen and functions like tools or storage are assigned to certain body regions. The concept overcomes the reaching problem for large displays and provides an easy interpretable technique where the person‘s movements in space and regions on his body become part of the interaction.

Harry Brignull and Yvonne Rogers created a system called Opinionizer for a large public

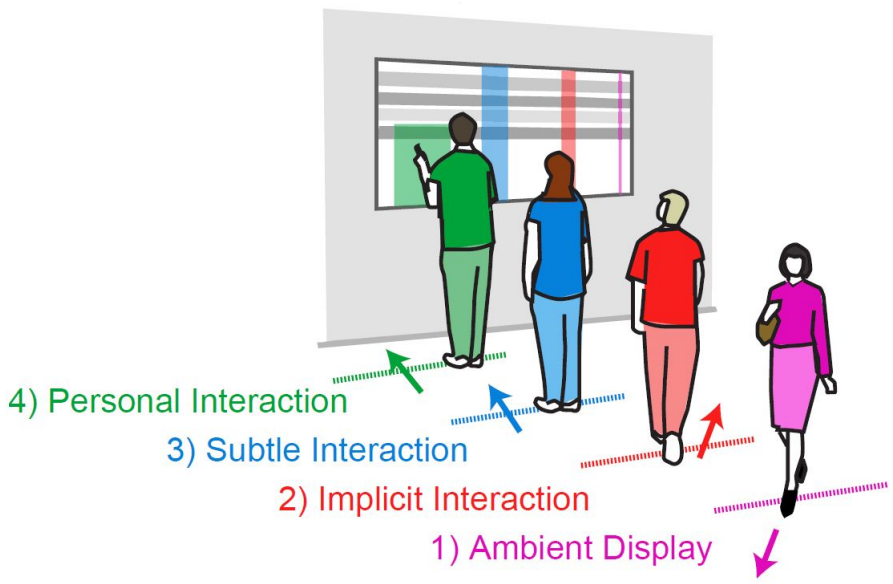

Figure 2.6: The four interaction phases in Vogel's work on Interactive Public Ambient Displays [65].

display and studied peoples interaction and behavior around it [5]. The screen shows a contextually relevant provocative phrase and allows people to share their opinion about it by using the provided keyboard close by. It was deployed during public events and they qualitatively observed people's socialization and interaction with it. They considered three activity spaces around the display: peripheral awareness, focal awareness and direct interaction activities. The main finding was, that the most critical part in public interaction is to encourage people's transition between these activity spaces. The paper presents some design recommendations that try to overcome this; mainly placing the display in a traffic flow, having an attractive and easily perceivable visualization and providing a lightweight way of interaction. While these findings and the categorization of the three activity spaces are very interesting, they do not apply them to change interaction possibilities or the presented content depending on information about proximity.

Hello.Wall [57] introduced the notion of 'distance-dependent semantics' where the distance of an individual from the wall defines the interactions offered and the kind of information shown. The system uses a combination of different far field radio-frequency identification (RFID) readers which enables detection of people and an interactive handheld-sized device called 'Viewport' in the surrounding area of the wall. It discriminates three zones of interaction - the *ambient zone* where the wall shows ambient messages through matrix signaling light patterns; in the *notification zone* the wall reacts to identified individuals or groups showing specific light patterns and the *cell interaction zone* where people could use the Viewport device to get detailed information.

Daniel Vogel et al. take this concept even further and designed a system that is built around a public large display and tracks people and their gestures in the space in front of it [65]. People can access information like the weather forecast, office activity, their personal calender and messages. This information is presented differently and interaction possibilities with it change depending on the distance to the display. Therefore they consider a framework with four interaction phases as shown in figure 2.6: the *Am-*

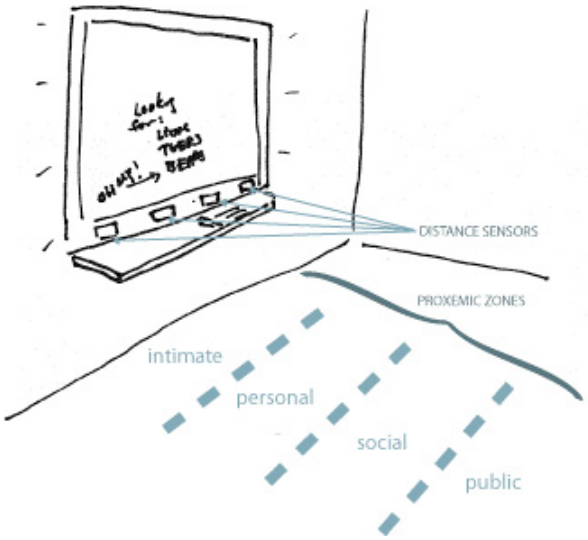

Figure 2.7: Diagram of interaction zones in Ju's electronic whiteboard system Range [30].

bient Display phase is the default phase when a person is far away from the screen and can only roughly see that there is something going on. The *Implicit Interaction phase* is entered when a person focuses on the display and the visualization shows an abstract representation of the user, trying to create awareness of further interaction possibilities. In the *Subtle Interaction phase* the person further approaches the display and the prototype application presents public information. The person can now use explicit gesture interaction to explore the information. The *Personal Interaction phase* is entered when standing close to the screen and it shows personal information which can be further explored by using touch. The main idea of the concept of proxemic zones is to provide public and implicit interaction from afar, becoming more private and explicit when coming closer.

Wendy Ju elaborated on this concept and contributed a framework for implicit interaction [30]. She defined four types of interactions - reactive/foreground, reactive/background, proactive/foreground, proactive/background - in a matrix that helps to illuminate transitions between interaction sequences from implicit to explicit and to identify successful strategies. They applied the framework to the design of a public interactive whiteboard - Range (see figure 2.7). It has proximity sensing capabilities and proactively changes between ambient display mode when people sit far away and authoring mode when someone comes close to the board. One feature is automatic clearing and moving of content to provide screen space for people approaching the whiteboard. The presented framework, if applied correctly, keeps a system design from being very risky in terms of proactive actions and makes sure to provide awareness and manual override at any point. I will also discuss these points in chapter 7.
These concepts of discrete distance regions are inspired by Hall's definition of proxemic zones [21] and as described above, researchers applied them in variations to interaction with large displays. My projects also use such zones around an interactive large display, reacting to people and devices approaching them - they change the interface and offer appropriate interaction modalities. Atop I consider zones between multiple devices and combine discrete distance zones with continuous information about distance and orientation to create awareness and facilitate interaction.

Figure 2.8: Lean and Zoom: magnifying the content of an internet browser when leaning closer to the display [23].

Chris Harrison's *Lean and Zoom* [23] project is a rare example that reacts to continuous change of distance instead of discrete zones: It observes the distance between a person's head and a computer display. When getting closer to the display the content grows up and vice versa the content shrinks when leaning away from the display. As shown in figure 2.8 he illustrates this with a zoom effect for an internet browser. In another example he shows a semantic zoom effect - providing additional labels for a technical drawing when moving closer. This is a rare example that shows how a display can react to continuous distance changes. I apply it to large interactive surfaces and in between devices and my interfaces react continuously to distance and orientation.

2.5 Device to Device Interaction using Proximity

A major problem in Upiquitos Computing is how to identify and connect devices to exchange information between them. In small space environments, an obvious idea is to consider spatial relations to facilitate device connections. Many researchers followed this approach and mostly defined a single discrete spatial region - often based on the requirements of sensing technology - where a connection is triggered when the spatial regions between devices overlap.
With *Smart-its Friends* [27] such a connection can be established once two devices sense similar values through attached sensors (such as accelerometers). Holmquist at al. called this approach *context proximity* where measured similarities in sensor values determine close proximity of artifacts. Smart-its Friends are pieces of hardware sensors that can be attached to devices. A pair of devices can then establish a connection by shaking them simultaneously. SyncTap [46] is a closely related technique that connects two devices by pressing a button on each one at the very same time and then compares the press and release timestamps over a network.

Hinckley's *Synchronous Gestures for Multiple Persons and Computers* [26] uses a similar approach: Two accelerometer sensor-equipped devices can be bumped into each other to establish a connection. Again he compares the accelerometer data over a Wi-Fi network to identify the bumped devices. With his technique it is also possible to receive information about the rough orientation of the devices in relation to the paired one. He illustrates his technique with an application that merges the workspaces of two tablet computers in a spatially correct way and spans an image across their surfaces (see figure 2.9). Ramos et al. [42] further elaborate on this concept and suggest additional techniques: *Arms-Length Stitching* requires a person to draw a pen stroke across two

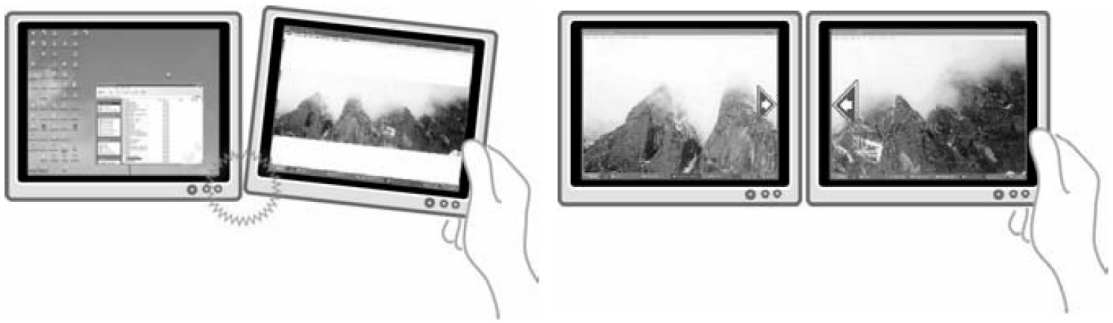

Figure 2.9: Pumping a tablet into another one merges their screens [42].

tablet screens while the devices are placed next to each other. Doing so identifies and connects the devices and provides information about relative orientation. As an example they implemented an application that lets people transfer and display an image from one tablet computer to the other by dragging it across their surfaces. This technique is different in terms of proximity. One person has to invade into another person‘s intimate space with his stylus, whereas the simple bumping can still be considered personal. *Cooperative Stitching* lets people draw each part of the stroke themselfs. So everyone keeps control over his own device but people have to precisely coordinate their gestures.
BlueTable is an approach that uses infrared light patterns to identify and connect a phone when placed on an interactive tabletop [71].
Tandler et al. designed the ConnecTable system [60] that interconnects two movable interactive tables depending on their spatial relation. Each person has his separated workspace until they are moved close together. Then the workspaces merge into a shared one that expands across the borders of both displays. Now users are able to exchange information by shuffling them over to the other display. Pulling the tables apart instantly separates the workspaces again.

Want introduced the technique of sensing nearby objects and devices using RFID tags [67]. A mobile tablet computer displays information or triggers certain actions when bringing it close to another object. Therefore the computer is equipped with an RFID reader and readable transponders are attached to everyday objects and devices in the environment. They show several examples that include books that can instantly be ordered when held close to the reader, printers that automatically print the currently opened document or how a tablet computer brings up a meeting agenda when a person enters a meeting room. Other research [31, 61] followed this concept and used long range RFID technology, barcodes or infrared sensors to identify the physical environment from a larger distance.
Several other researchers developed techniques that work from larger distances and use pointing to identify nearby devices. Swindels *GesturePen* [59] is a pointing device equipped with an infrared transceiver that can send an optical signal to compliant tags attached to other devices. In their implementation the GesturePen is coupled with a handheld computer and people can remotely point at devices like printers, desktop computers or tabletop displays to establish a data connection with them (see figure 2.10). Similar projects [48, 3] developed small PDA like devices that could be used to point at digital appliances or other devices in the environment. These appliances provide a simple interface through line of sight infrared light, so that the person can control them using his pointing device. Rekimoto [47] combined RFID and infrared (see figure 2.11) for estab-

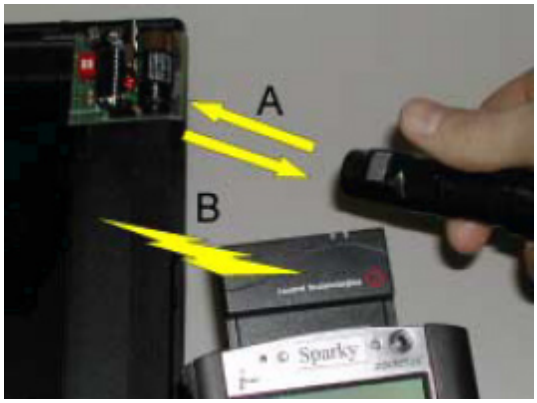

Figure 2.10: The GesturePen [59]: a pointing device with an infrared transceiver which can be used to select other devices.

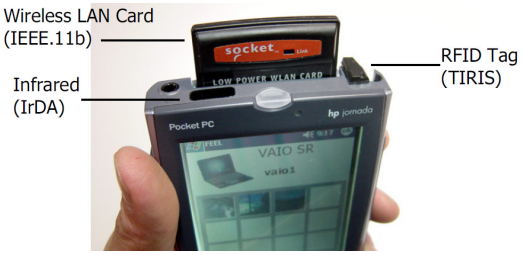

Figure 2.11: Rekimoto's M-pad is equipped with a RFID sensor for close interaction and an infrared sensor for distant pointing [47].

lishing secure device connectivity while enabling selection from a distance and in close proximity. When connecting for example a PDA to a large screen their system even visualized the spatial relation by showing an icon of the PDA on the screen close to the PDA's physical location.

While all these techniques provide a solution for identifying a device through spatial means instead of selecting them in a traditional user interface from a list of available network devices, they only consider a binary device connection state. I extend this prior work by contributing techniques that go beyond this binary connection state: I introduce techniques that go from *awareness* at a larger distance to *gradually revealing* of higher level detail to *direct interaction* for transferring rich digital information between devices.

Spatial relations have also been used to mediate the information exchanged between devices. For example, Kray's group coordination negotiation [32] introduced spatial regions around mobile phones. Their scenario uses these regions to negotiate exchange of information with others and to visualize the regions on a tabletop (as illustrated in figure 2.12). Depending on how devices were moved in and out of three discrete regions, the transfer of media data between the devices is initiated. The scenario shows how people

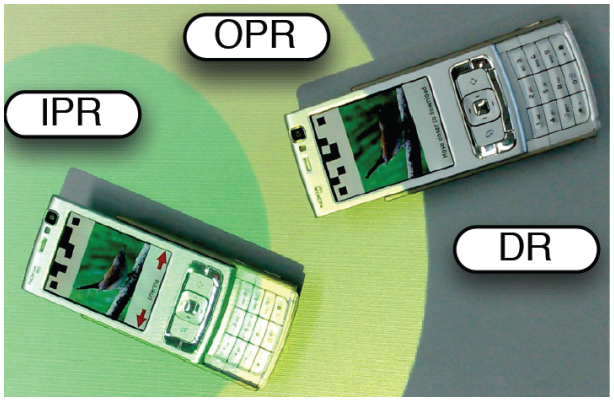

Figure 2.12: Spatial proximity regions around mobile devices on augmented tabletops [32].

Figure 2.13: Hans Gellersen‘s Relate Gateways [18]: an icon on the screen spatially relates to a physical device

can independently browse their images when holding their phone in the *distant region*. Moving it towards other devices into the *outer proximity region* lets them remotely explore images. A transfer of selected images is initiated by bringing the devices even closer to each other into the *inner proximity region*. I extend their approach to interaction around large vertical surfaces, where the degree of shared information between devices changes continuously and depends not only on their relative distance but also orientation.

Gellersen‘s RELATE Gateways [18] provided a spatial-aware visualization of nearby devices. A graphical map showed the spatial room layout, and icons indicated the position of other nearby devices. Alternatively, icons at the border of a mobile device screen represented the type and location of surrounding devices (similar to Rekimoto‘s earlier concept of a spatial related visualization of a PDA [47]). They called these icons *Relate Gateways* (see figure 2.13) as it was possible to identify and connect to the corresponding device by the spatial reference on the screen, rather than selecting them from a traditional list of available network devices. I extend this notion with: visualizations that include proximity dependent level of detail and with techniques that move from awareness to direct interaction depending on a person‘s distance and orientation to the display.

3 Introduction to Proxemic Interaction

In this chapter I present previous work published together with Nicolai Marquardt and Saul Greenberg - *Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment* [1]. It lays the foundation for this thesis as it operationalizes proxemics as five measurable dimensions and presents general concepts for proxemic interaction.

3.1 Dimensions of Proxemic Relationships

People use interpersonal proxemics while interacting with another. While they don‘t have to measure distances or angles to derive a social meaning from them, computing systems do. Hence, when we want to talk about proxemics in Ubicomp, we need to operationalize them. Hall and others [21, 22, 55] studied proxemics in a social context, but Ubicomp proxemics is somewhat different. It concerns inter-entity relationships, where entities can be a mix of people, digital devices, and non-digital objects. We identified five dimensions as essential if a system is to determine the basic proxemic relationships between entities: distance, orientation, movement, identity and location. These five dimensions describe our extended notion of proxemics for ubiquitous computing environments:

1. **Distance** describes the value that measures how far two entities are apart. It is a fundamental dimension when considering entities like people, devices and objects in space. While we normally think of distance as a continuous measure as e.g. a value between zero and four meters, we suggest to consider measures that vary by fidelity and the values they return - continuous or discrete. Based on Halls proxemic zones, others [65, 30, 57] have considered discrete regions for people around a digital vertical surface. What these regions have in common is the notion, that interaction in a far region is public and implicit and becomes more private and explicit when people move into closer regions to the display. In the simplest case discrete measurement of distance is just a binary value - returning whether two entities can see each other or if they are in the same room (as explored in [68, 52, 8]) or if entities touch [42] or are in close proximity to each other [47].

2. **Orientation** provides the information about which direction an entity is facing. This makes sense only if an entity has a well-defined ’front’ (e.g., a person‘s eyes, the point of a pencil). Again we can differentiate between continuous orientation of an entity (e.g., described through yaw, pitch, and roll) or discrete orientation (e.g., a quantitative description such as "this person is facing that object"). From continuous orientation values, like the exact yaw and pitch angle, we can determine where a beam from one entity would intersect with another entity (ray casting). Discrete

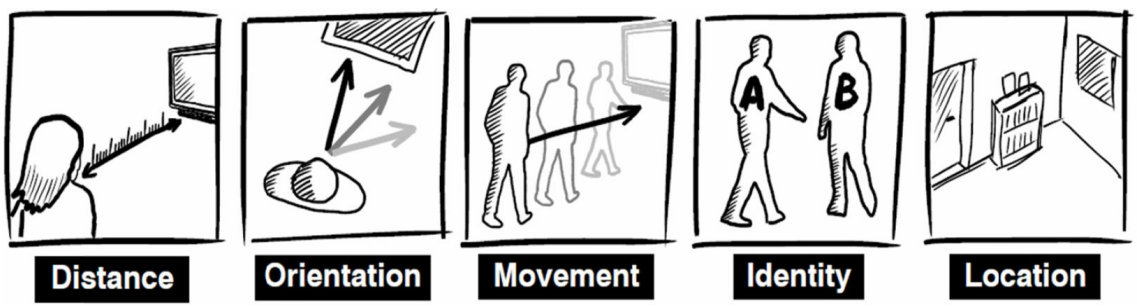

Figure 3.1: The five dimensions of proxemics in ubiquitous computing [19].

orientation as an input measure has been applied to attentive user interfaces [63] where a device takes actions when sensing the attention of a person.

3. **Movement** lets one understand the changes of distance and orientation of an entity over time. This also means we can calculate the velocity of these changes. Movements, for example, reveal how a person is approaching a particular device or object - determined by the speed of motion and whether he is moving and turning towards or away from an entity.

4. **Identity** uniquely describes the entities in the space - again varying in fidelity. The most detailed information would provide the exact identity of a person, device or object (e.g., "Fred", "Person A", "Fred‘s cell phone"). 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").

5. **Location** describes the physical context in which an entity resides. Location measures are related to Hall‘s theory of fixed- and semi-fixed features [22] and can vary from fine grained information like when an entity is close to piece of furniture (semi-fixed feature) to coarse measures that capture for example when an entity enters a room (fixed features). Location is important, as the meaning applied to the four other inter-entity dimensions may depend on the contextual location.

As stated throughout the descriptions, some of these dimensions have been applied to ubiquitous computing systems before, but only a few combine several measurements. We think that a combination of all these dimensions can lead to improved design of interactive Ubicomp systems. Hence we will show applications that illustrate the use of these dimensions and then describe interaction concepts that apply the proxemic dimensions in a meaningful way.

3.2 The Proxemic Media Player Application

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.

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 distance and orientation. 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. (see figure 3.2, left). A Vicon motion capture system tracks, via 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 3.2 (left) shows Fred approaching the display at four distances (a’ - d’), while the four scenes at the bottom shows 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’), 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 (see figure 3.2 a). As Fred moves closer to the display (b’), the video preview thumbnails and titles shrink continuously to a smaller size, thus showing an increasing number of available videos (3.2 b).

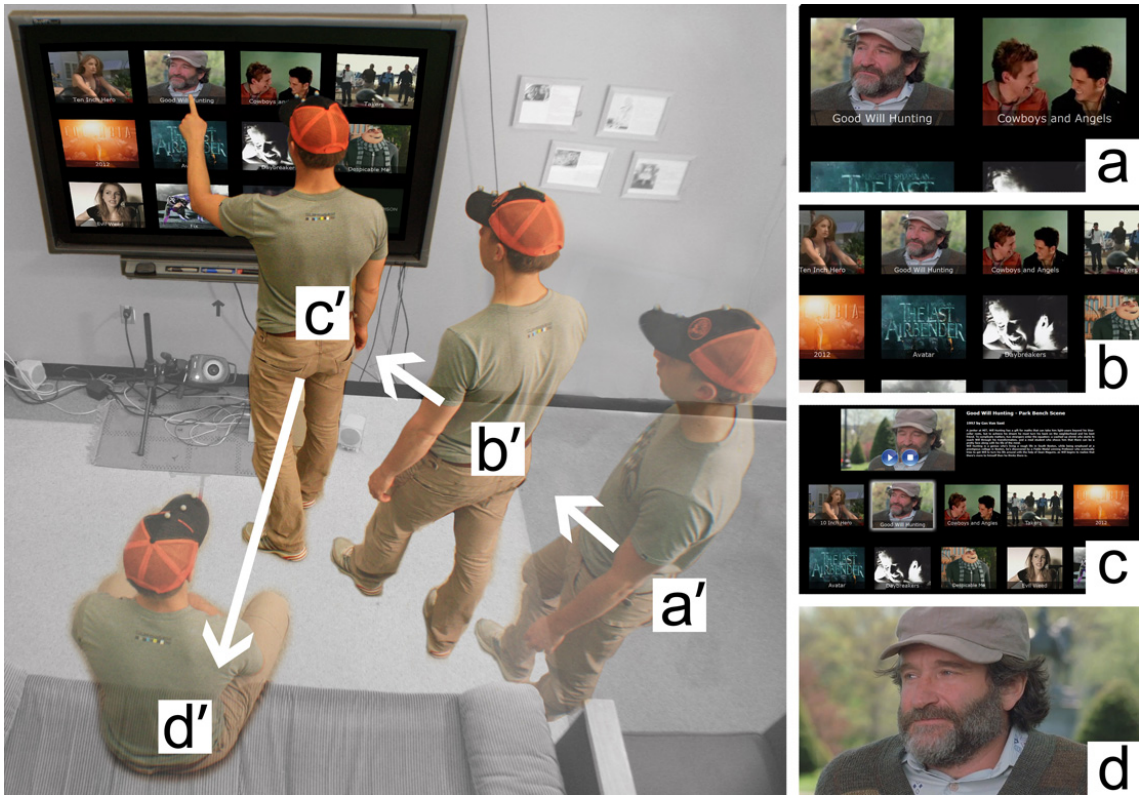

Figure 3.2: Proxemic Interaction - **a)** activating the system when the person enters the room, **b)** continuously revealing more content with decreasing distance of the person to the display, **c)** allowing explicit interaction through direct touch when the person is in close distance, and **d)** implicitly switching to full screen view when the person is taking a seat.

When Fred is very close to the surface (c'), he can select a video directly by touching its thumbnail on the Figure 3.2: Proxemic Interaction: a) activating the system when a person enters the room, b) continuously revealing of more content with decreasing distance of the person to the display, c) allowing explicit interaction through direct touch when person is in close distance, and d) implicitly switching to full screen view when person is taking a seat. screen. More detailed information about the selected video is then shown on the display (3.2c), which includes a preview playback that can be played and paused (3.2 c), as well as its title, authors, description and release date (3.2 c). When Fred moves away from the screen to sit on the couch (d'), his currently selected video track starts playing in fullscreen view (3.2 d). 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 (see figure 3.6). 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 (as in Figure 3.6). 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.

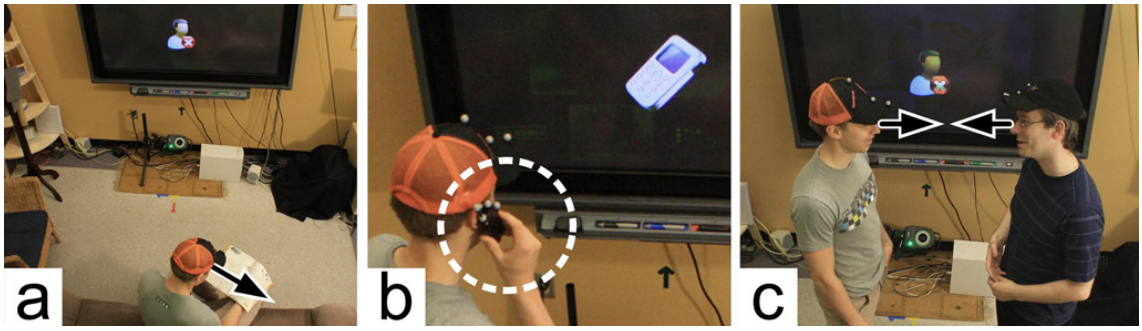

Figure 3.3: Integrating attentive interactive behaviour: pausing the video playback when the person is **a)** reading a magazine, **b)** answering a call, **c)** talking to another person.

Somewhat later, Fred receives a phone call. The video playback automatically pauses when he answers the phone (see figure 3.3 b), but resumes playback after he finishes the call. Similarly, if Fred turns away from the screen to (say) read a magazine (see figure 3.3 a), the video pauses, but then continues when Fred looks back at the screen.

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 (see figure 3.8 a). When George approaches the display, more detailed information about the current video becomes visible at the side of the screen where he is standing (see figure 3.8 b). When George moves directly in front of the screen (thus blocking Fred's view), the video playback pauses and the browsing screen is shown (see figure 3.8 c). George 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. If Fred and George start talking to each other, the video pauses until one of them looks back at the screen (see figure 3.3 c).

Fred takes out his personal portable media player from his pocket. A small graphic representing the mobile device appears on the border of the large display, which indicates that media content can be shared between the surface and portable device (see figure 3.7 a). Fred moves closer to the surface while pointing his device towards it; the graphic on the surface responds by progressively and continuously revealing more information about the content held on the media device (see figure 3.7 b). When Fred moves directly in front of the surface while holding the device, he sees large preview images of the device's video content, and can then transfer videos to and from the surface and portable device by dragging and dropping their preview images (see figure 3.7 c). The video playback on the large screen resumes as Fred puts his portable device back in his pocket and sits down on the couch. When all people leave the room, the application stops the video playback and turns off the display.

While this media player is a simple application domain, it provided a fertile setting to develop and explore concepts of proxemic interaction. We will discuss the details of proxemic interaction concepts associated with a single person or multiple people interacting with a large digital surface.

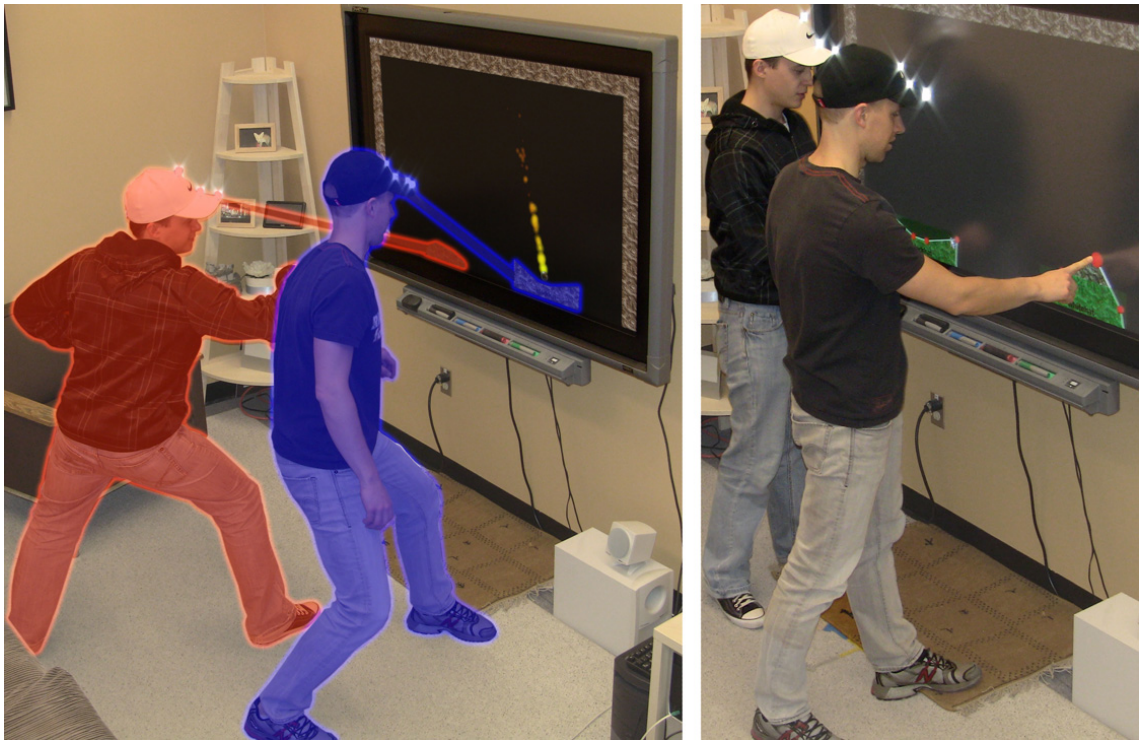

Figure 3.4: **left)** Two players controlling pong paddles with side to side body movements in front of the display, **right)** modifying the paddle-shape by using direct touch when standing close to the display.

3.3 The Proxemic Pong Game

The original Pong game was created by Atari Inc. in 1972. It was their first game and one of the earliest arcade video games. Two players have to compete against each other by controlling paddles that hit a ball back and forth until someone misses. Proxemic Pong is a variation of this game, but instead of using joysticks, buttons and menus as controls, it exploits proxemic dimensions - reacting to distance, motion, identity and location.

Identity distinguishes between players and location recognizes a person‘s presence in the room. When no one is in the room, the game sleeps and the screen is black. As a person enters the room, a splash screen shows up to introduce the game. When moving into the play area in front of the display, the game starts and a paddle is created for the player. He can control the paddle with his body while facing forward and moving side to side. A fire ball appears and falls down, while the player has to move his paddle, so that it bounces back up. This becomes more and more difficult and exhausting, because the ball increases its speed over time and the ratio between physical movements and paddle movements increases. While at first the amount of body motion match the paddle motion on the screen, over time the player has to cover greater physical distances in relation to the movements of his paddle.

When a second person enters the room and moves into the play area, the game interrupts, announces a new opponent and creates a paddle for him. Now game play continues by turn taking. Each paddle is controlled by a player‘s body (see figure 3.4 left), while only one paddle is active and the other one is slightly faded out and sized down a little. As one hits the ball, his paddle becomes inactive and the other paddle is activated

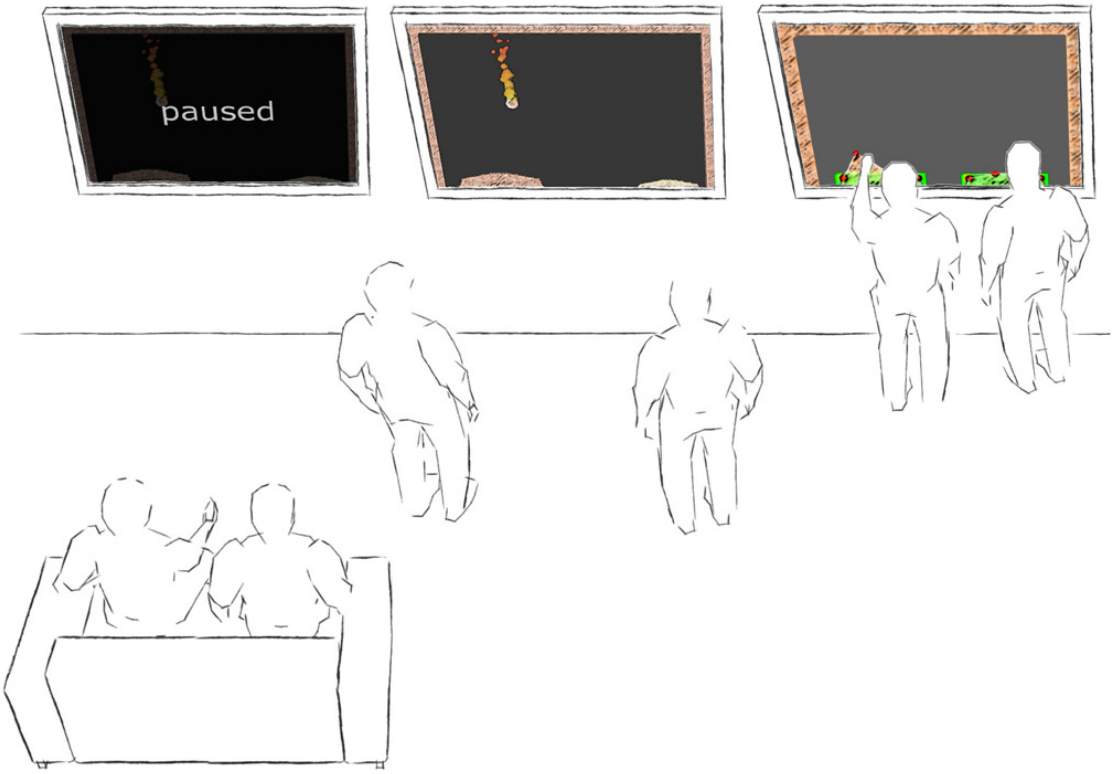

Figure 3.5: Three game states depending on distance - **left)** paused when sitting, **center)** playing the game in the play-area, **right)** modifying paddles when standing in close proximity to the display

to return the ball. If a player interferes physically with the other player by standing in his way, the game penalizes the passive player by enlarging the active player‘s paddle - making it easier for him to hit the ball. This is determined by the interpersonal distance of the player‘s bodies. People can independently join and leave the room at any time and the game switches between one and two player mode or turns itself on and off accordingly. Also when two people enter at roughly the same time it instantly starts in two player mode.

The game also considers back and forth movements (as illustrated in figure 3.5). Therefore the room is split into proxemic regions: A play-region in the center, a touch-region around the large interactive display and a sitting-region around the couch and chairs. When a person comes closer to the display and enters the touch region, the game play is interrupted and four points appear at each paddle. As sown in figure 3.4 on the right image, players can use direct touch functionality of the interactive display to move the control points and thereby modify the shape of their paddle. Moving back into the play area resumes the game. If for example the players are exhausted and sit down to rest on the couch, the game pauses.

3.4 Concepts of Proxemic Interaction

We will now describe concepts of applying the five proxemic input dimensions in meaningful ways to people’s interactions with Ubicomp systems. To ground the explanation, we highlight particular examples from the above described scenarios of - the media player

and the pong game - that illustrate how each concept can be applied.

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 and environment. Dourish framed this concept as embodied interactions [12]; technology that is seamlessly integrated into people’s everyday practices, rather than separated from them. Context-aware computing is one outcome of this, where some kind of context-aware sensing [52] provided devices with knowledge about the situation around them. This sensing usually involved measuring a coarser subset of our dimensions, e.g., very rough positions, and other factors such as noise, light, or tilting. We contribute to this by introducing the notion of having context-aware systems mediate embodied interaction by understanding the proxemic relationships (as defined by our dimensions) of people to the fixed- and semi-fixed feature space [21] surrounding them.

For an interactive system (such as the interactive wall display in our media player application), knowledge about the fixed feature space includes the layout of the fixed aspects of the room, such as existing walls, doors and windows. It also includes knowledge about fixed displays - such as a digital surface - located in this environment. For instance, the knowledge about the position of the fixed entrance doors allows our system to recognize a person entering the room from the doorway, and then take implicit action by awaking 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.

Semifixed features in the environment include all furniture, such as bookshelves, chairs, and tables whose position may change over time. While it is somewhat object-dependant, 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 will have to 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 3.2, position a’) and watching the video when seated at the semi-fixed couch (see figure 3.2 position d’). The actual distance of the person relative to the surface is similar in both situations, 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. (Of course, inferences may not always be correct. This will be discussed later).
Thus, information about distance and orientation of a person relative to the fixed and semi-fixed feature space provides cues that can mediate implicit interactions with the system.

Interpreting Directed Attention to People and Objects
Proxemic interactions can be used to extend the concept of attentive user interfaces (AUIs) that are designed to "support users‘ attentional capacities" [63]. In AUIs, 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) [63]. We take this AUI con-

cept one step further, where we also incorporate information about: what entity a person is attending, and the importance of distance and orientation 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 [63] in our media player application: the system plays the video as long as at least one person faces the large display, but pauses when that person looks away for a length of time. Attention to other surrounding objects and devices. We enrich the concept of AUIs by including how a person‘s directed attention to other surrounding objects of the semi-fixed feature space can trigger implicit system reactions. In our system, the fact that a person is holding and facing towards a newspaper (shown in Figure 3.3 a) 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. If Fred had a similar gaze to (say) 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, our system detects when Fred is holding his mobile phone close to his ear (as shown in Figure 3b). 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 relative distance of phone to the person‘s head, as well as their orientation towards each other, provided the necessary information for the system to implicitly react to this situation. Attention to other people. We can discriminate how one person attends other people as a means to trigger implicit system reactions. For example, consider Fred and George when they turned towards each other to converse (see Figure 3.3 c). Our scenario illustrated how the system implicitly reacts to this situation by pausing the video. However, by knowing that they are in conversation (rather than just knowing that they are looking away from the display), the system could have just turn down its volume.

Supporting Fine Grained Explicit Interaction
Instead of 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. 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 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. The way this works is that all mobile tracked objects are interpreted as mobile tokens. Three units of information caused our 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, we showed how two quite different devices can serve as similar tokens: the pen in Figure 3.6 a, and the mobile phone in Figure 4b. We emphasize that we are not using any of the digital capabilities of the mobile digital phone to make this inference. Rather (and as with the physical pen) we are using only the knowledge of its 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 [7], 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 [7] to control that particular device.

Figure 3.6: Explicit interaction triggered through distance and orientation between a person and a digital or non-digital physical artefact - **left)** a pen, **right)** a cell phone

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 that 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 it uses the distance and location of the objects relative to the person and other objects to infer a certain explicit interaction mode.
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 depended on its proxemic relation to its holder and to the display. The distance of the phone to a person‘s head indicated an ongoing phone conversation, while holding the same device in front and towards the display shifts its meaning to an interaction pointer. For the actual explicit interaction with the digital video content displayed on the large surface, the person can move the position of the mobile token. Changes of the orientation angle allow fine grained positioning of a pointer icon on the screen, while fast acceleration downwards can be used for selection.

Interpreting Continuous Movements or Discrete Proxemic Zones
Another concept is that the behaviours of proxemic interfaces can react to the position and distance of its 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. For example, as a person approaches a screen of the media player application, the number of visible video preview thumbnails shown continually increase with distance (see Figures 3.3 a,b). To do this, the system gradually resizes the preview images to a smaller size (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 [23]). Another example of continuous mapping of distance as an input regulator are the awareness icons of nearby digital devices (visible in Figure 3.7). These icons grow continuously, from a small circular icon indicating its presence, to a large area on the screen that displays rich content and allows direct touch interaction with it (as in the progression from Figure 3.7 a-c).

With discrete proxemic zones we can divide the space into discrete regions. 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 [21], and others have applied zones as a way to mediate interaction with public ambient displays [65] and digital whiteboards [30].

Our media player uses discrete zones in several ways. We use it to trigger an associated implicit action (e.g., we activate a display screen when entering the room). We 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 distance to the screen). A problem associated with discrete zones occurs when the interface rapidly

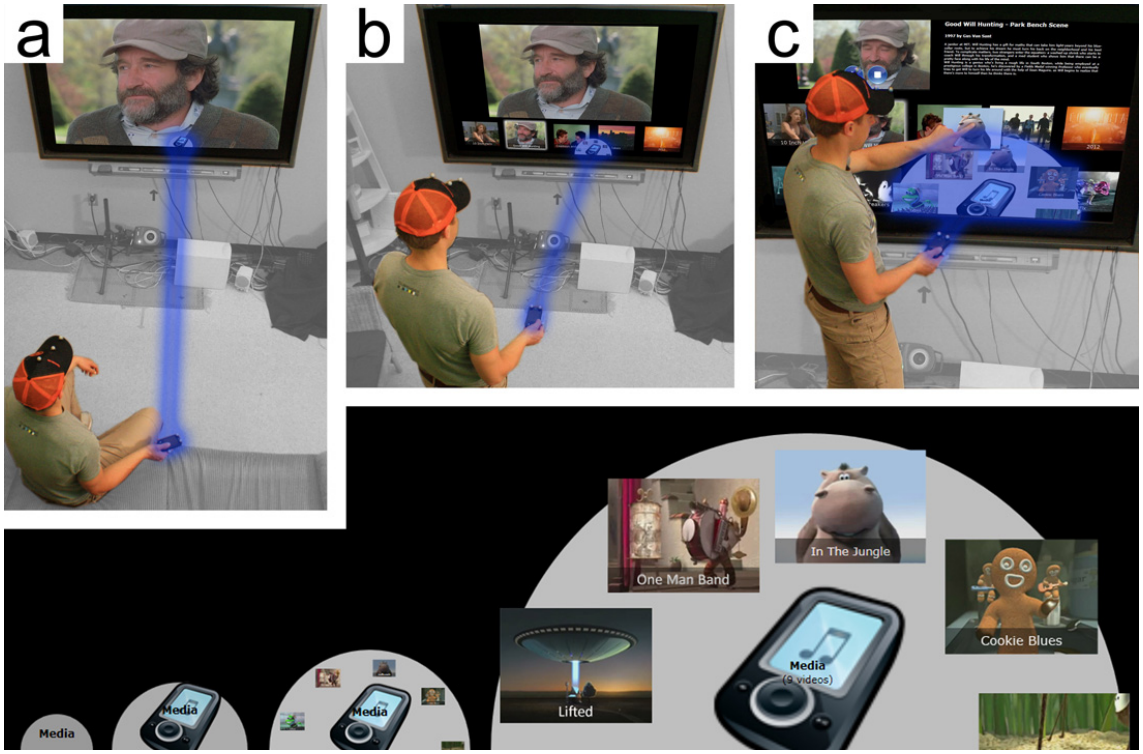

Figure 3.7: Proximity mediates device to device interaction - ranging from **a)** awareness information, to **b)** gradually revealing information, to **c)** direct interaction

switches back and forth between two states; this occurs when the person stands exactly at a border of one of the discrete zones. This 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, we use a 15-20% hysteresis tolerance for proxemic regions around the interactive wall display (percentage of the region dimension) to avoid this rapid switching.

Moving From Awareness to Direct Interaction

Next, we can combine both continuous movements and discrete proxemic zones to design system interfaces that move fluently from awareness to direct explicit interaction. Two examples illustrate this combination. Our 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, we used a discrete proxemic zone around the digital display that triggers this activation behaviour. At this point, if the person just walks pass the display, or does not face the display, the media player application would revert to sleep mode. If, however, the person does move closer, the system shows preview images of video content, where it gradually reveals more preview items on the screen as the person approaches the screen. Here, we use 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. So far, we have focused on implicit and explicit interactions

mediated through changes of a person‘s distance and orientation relative to the large digital surface. Interactions, however, increasingly take place in an environment comprising an ecology of devices - from shared large displays to portable personal devices. Using our four proximity dimensions, we can recognize nearby devices and thus facilitate using them in conjunction with one another. This opens new possibilities for interaction, communication, and information exchange. However, to make sense of device interaction, people require awareness of device interconnections and a means to move into direct interaction over them.

To explain, we illustrate device-to-device proxemic awareness and interactions with the interactive vertical display, where the surface reacts to nearby portable devices carried by a person. This time the system reacts to distance, continuous movement, and orientation of a person‘s portable digital device when approaching the media player displayed on the surface. Again, we illustrate how we use discrete zones and continuous movements to move from awareness to direct interaction.

When a person takes a portable media player out of their pocket while sitting at a distance, the system recognizes the device and indicates a possible interaction through a visual icon at the border of the display (visible in Figure 3.7 a). This icon represents the portable device, where it indicates to the person that there is now an opportunity to share content between the large surface and the portable device. While this icon visualization is inspired by earlier approaches (e.g., [18] [47]) for visualizing spatial relationship between devices, it differs in how it incorporates proxemic distance and orientation information leading from awareness to direct interaction. If the person then orients this portable device towards the large screen, more detailed information about that device and its contents becomes visible. 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, even more details about the content (e.g., titles) become visible and the preview thumbnails are shown at a larger size (see figure 3.7 b). When the person holding the device is within reach of the interactive screen (i.e., a discrete zone is entered), the size of the icon grows to a large area of the screen (visible in Figure 3.7 c). The icon not only provides detailed information about the content of the device, but also allows full direct touch interaction. The person can now drag and drop video items from the portable device to the large surface and vice versa. When putting the device back in the pocket the visualization immediately disappears and the media player continues its playback.

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 a last video that a person previously watched but did not finish. Personalization. The media player could save one‘s settings as a personal profile. This can include personal configurations, idiosyncrasies of how the system should respond to that particular person, and that person‘s media content. For example, when a particular person approaches the display, our media player would then display content out of that person‘s media library. Safeguards. Identifying the person interacting with the system can also function as a safeguard to restrict access. For

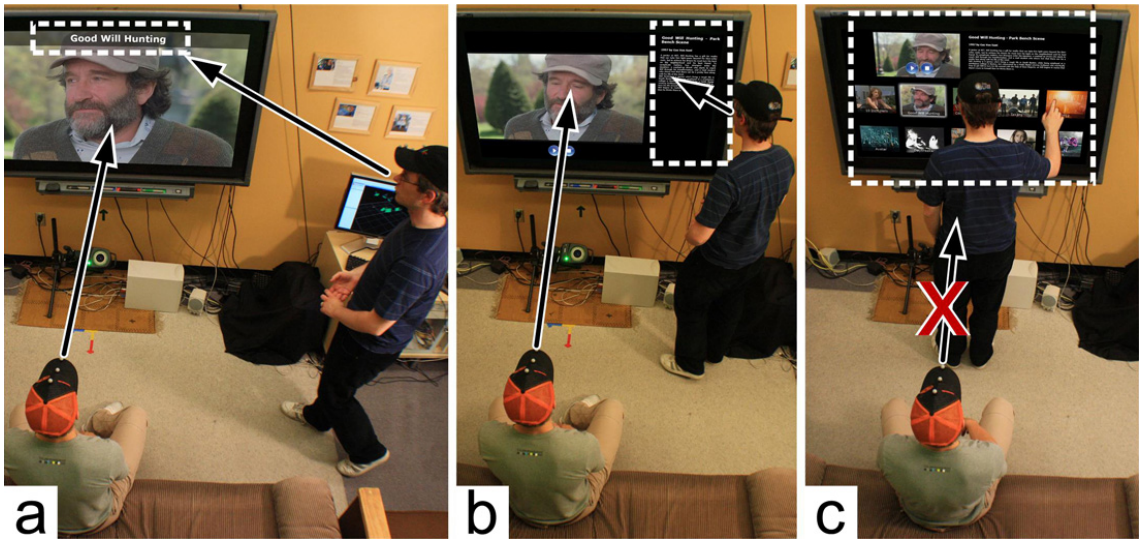

Figure 3.8: Mediating between multiple people: **a)** incoming person sees basic information such as video title, **b)** as one moves closer, the split view provides a more detailed description, **c)** when within reach of the display, the person gets full control.

instance, children may only be allowed to access the media player application during predefined time slots, or access to available media content could be restricted to movies rated for their age.

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

For example, we saw George enter the room while Fred was watching a video. George wants to know what was being played, while Fred wants to keep watching. To compromise between these needs, the system displayed 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 (see figure 3.8 a). 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 (see figure 3.8 b). The positioning of this region also depends on George's spatial relation to the display - if he moves between the left to right side, the information panel smoothly animates to that side of the display. 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.

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 3.8 c: 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 3.8 c, 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, where they would have 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.

4 Proxemic Applications

In the previous chapter I described concepts of *Proxemic Interaction* which consider the complete ecology or people, digital and non digital artifacts and fixed- and semi-fixed features of the environment (e.g. walls and furniture). As stated earlier, the main goal of this work is to further explore how these concepts apply to interaction between devices. Hence Nicolai Marquardt and me developed application scenarios that incorporate a variety of interconnected digital devices. Therewith I implemented two systems that integrate a particular set of devices (see figure 4.1) that differ in three aspects important to how people interact with them in a spatial environment:

- **Mobility:**
  Mobility means how easy it is to physically move a device. And whether the role of a device suggests to be in a fixed place or to be carried around. The device with the highest mobility is a small *digital camera*. People can take it anywhere in their pockets and it can easily be moved around in space. This means that interaction can include holding it at any position or even performing gestures with it. Next, I use *mobile tablet computers*. They can still be moved around arbitrary but a person might want to hold them in a comfortable position for most of the time. A *digital picture frame* could be moved around, but has limitations due to a power supply. Also its function does normally not require it to be moved frequently. Last, a large 52 inch *interactive display* in most cases stays in a fixed position as moving it can be very strenuous.

- **Collaboration:**
  The collaboration aspect distinguishes, whether a device is considered to be private and only used by one particular person or whether it is shared and can be used collaboratively. This makes a huge difference in terms of interaction and the design of interfaces. While a large display and a picture frame can be used collaboratively and information on them can be accessed by everyone nearby, a tablet computer will in most cases be operated by a single person, but in some situations others might look at it, too. The digital camera however will almost always be operated by a single person.

- **Capabilities:**
  With capabilities I mean mainly the input and output possibilities and the processing power of a device. In our applications the simplest device in this regard is the digital picture frame which can only show static information on a small display and has no

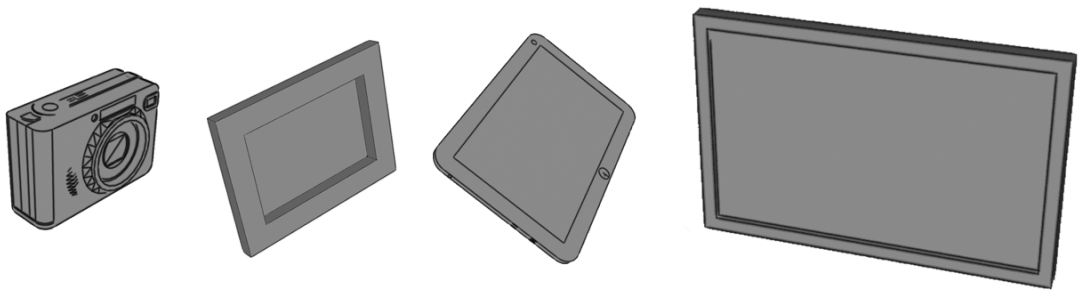

Figure 4.1: Devices used in our applications: digital camera, digital picture frame, tablet computer, large interactive display.

integrated method for input. The digital camera has a very limited screen for output and only some input possibilities like simple buttons. The tablet computers however have higher processing power and provide touch input with a pen on a 12 inch color display. So the device is well suited for interaction with and visualization of hardly any data, at least for a single person. The large display is connected to a computer with full processing power and its screen is big enough for people to see content from a larger distance as well as for simultaneous interaction through touch input.

In my applications all devices of these four types (see figure 4.1) are interconnected at any time using different technologies, as I will explain in more detail in chapter 6. While the intention of this work is to focus on inter-device interactions, of course people and the physical environment still play an important role. Hence I will heavily consider them in my applications, as spatial relations to and between them do significantly influence the meaning of how the devices are used.

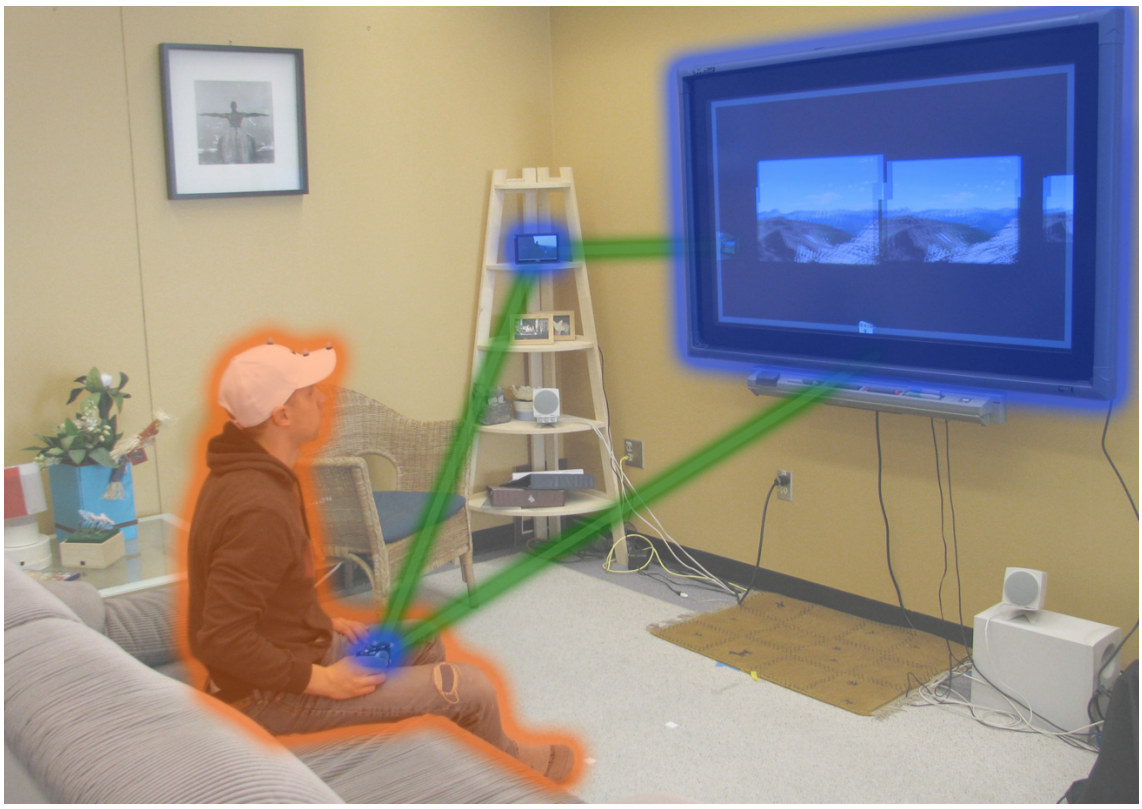

Figure 4.2: The room setup for the Proxemic Photo Canvas (devices highlighted in blue).

4.1 Proxemic Photo Canvas

This application is designed for a home environment, where people can walk in with their camera and transfer images to and explore them on other surfaces. As figure 4.2 shows, the Proxemic Photo Canvas integrates the following devices:

- **Digital Camera:** A small point-and-shoot camera can be used for capturing and storing pictures. Additionally I equipped the camera with Wi-Fi capabilities (see chapter 6.4 for more detailed information) which means it is connected to the other devices in the environment and can share its images over the network.
- **Vertical Surface:** A 52 inch vertical surface (Smartboard [54]) is mounted to the wall at eye level. It is used for displaying pictures and lets people interact through direct touch. By default it shows a *photo canvas* where people can arbitrarily move images around (see figure 4.3).
- **Digital Picture Frame:** This digital appliance is placed on a shelf in the corner of the room. It simply displays pictures that other devices send over the network.

The scenario is set up as a living room with furniture like a couch, some chairs, a couch table and a shelf. This room is equipped with a Vicon Infrared tracking system which provides information about the exact position and orientation of people and all devices. Atop the application knows about the spatial arrangement of the furniture and the dimensions of the room itself. I first explain how the application works by describing the scenario and then revisit the interaction techniques. For more details on the technology and the actual implementation see chapter 6.1.

4.1.1 Scenario

When a person enters the room with his digital camera, a subtle Icon (see figure 4.4 a) of the camera follows him on the border of the large screen, indicating that it is connected. While moving closer the icon reveals the latest pictures taken with the camera as a stack of images slowly growing from below the camera icon (see figure 4.4 b). This creates awareness of the content on the camera and makes it obvious which camera relates to

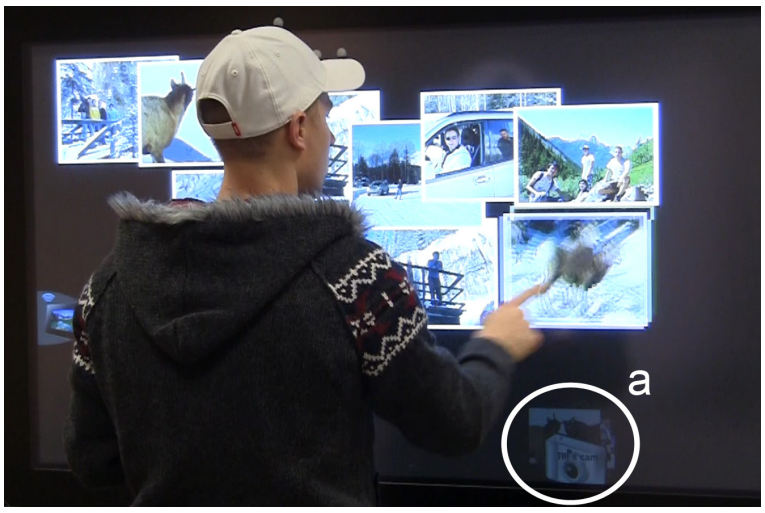

Figure 4.3: A person moves around pictures on the *photo canvas* using direct touch. **(a)** The camera icon fades out, when the camera is hidden in a pocket.

Figure 4.4: These three screenshots show how the camera gradually reveals its content while approaching the display from afar **(a)** to close **(c)**.

which icon and therefore how to address the specific device, because the icon and its content follows the physical position of the camera in the room at any time. When the person brings the camera close to the screen the pictures size up further and arrange like a fly around the position of camera. The last taken picture thereby appears as the largest one and up to ten more are following along a half circular path decreasing in size (see figure 4.4c). If the camera is hidden - e.g. in a person‘s pocket - the related icon on the screen fades out as shown in figure 4.3 a. It still provides the awareness of the device, but after about 15 seconds it disappears completely. As the camera icon on the screen should spatially relate to the digital camera, it is always the placed at the closest point on the screen to the camera. Hence it does not only stay at the border of the screen, but is projected to the inside when a person holds the camera in front of the display (see figure 4.5). In this case the representation changes from a *border mode* into a *projection mode*. This means when standing close to the screen and bringing the camera directly in front of the screen after a certain threshold the camera icon and the images move away from the border to the inside, projected straight onto the display. Also the fly visualization smoothly changes into a stack of images again and keeps following the physical device. This creates further awareness of possible interaction and better visualizes the relation between the physical camera and the camera icon on the screen.

By tilting the camera to the left or right, the person can navigate back and forth through all pictures on the camera. When standing close tilting causes the fly visualization to rotate, adding a new image from one side and removing it from the other one. This behaviour is similar to the ’lazy Susan’ or ’unlimited length half pie menu’ which has been designed for interaction on interactive tabletops [24]. Even when standing further away from the display, tilting reorders the images in the stack so that it shows the following or previous picture on top. The tilting technique itself is inspired by earlier research [6, 9, 43].

The application offers several ways of transferring pictures in between devices:

- **Touch interaction:** When standing close to the screen with the digital camera, one can use direct touch and drag images out of the fly visualization to copy them on the screen canvas. The copy is now associated to the display and can be freely moved around by direct touch interaction.
  Similar to the camera icon there is also an icon of the digital picture frame at the

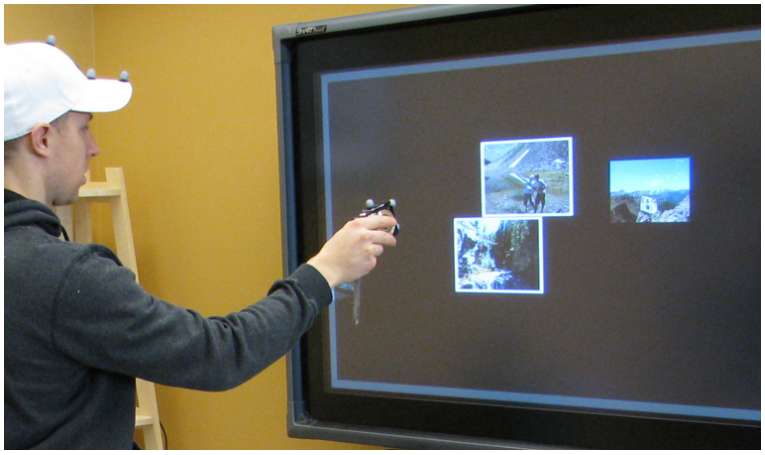

Figure 4.5: The camera icon with the last image beneath it is projected to the screen when holding the camera in front of it.

Figure 4.6: **left)** to show a picture on the picture frame a person drags a picture into the spatially related icon. **right)** While dragged over the picture-frame-icon blue arrows appear to indicate the transfer possibility.

side of the photo canvas. This icon is again placed in a way that it spatially relates to the picture frame‘s position in the room. It can be used as an "image tunnel" to the picture frame. This means a person can drag images onto this icon. Once dragged over it the picture indicates the transfer possibility by showing two rotating arrows (as shown in figure 4.6 right). When the drag is released the image animates into the tunnel and appears on the picture frame (see figure 4.6 left).

- **Touch with camera:** Also the camera itself can be used to touch the display. Doing so copies the current upper image of the stack directly onto the canvas on the screen, just where the camera touched it. The interaction is facilitated through a visualization of the image stack in *projection mode*. This means that it follows the camera away from the border to the inside of the screen and creates further awareness of possible interaction and of which image will be copied (see figure 5.5).
  The very same interaction can be used with the digital picture frame. When touching the picture frames‘s display with the camera, the top picture of the stack appears at the touch point and scales up until it fills the complete picture frame. The picture

frame however does not show a visualization of the camera, because in the scenario this device is seen as a simple information appliance.

- **Throw gesture:** While holding the camera in front and moving further away from the screen the camera icon stays projected to the inside. The images in the stack are continuously scaling up depending on the distance, so that the person can optimally identify them. When the person stands too far away to touch the screen he can use a throw gesture to pin an image to the photo canvas. The throw gesture is recognized with a certain acceleration of the digital camera towards the screen. This throw gesture can also be used to send images to the digital picture frame which is located in the shelf about one meter left of the Smartboard. For this to work, both the system and the user have to know which device he is attending to. As the natural bahaviour and the theory of attentive user interfaces [63] suggest, a person would orient with his camera towards the device he wants to interact with. Hence the application observes the position of the person in relation to the camera. Depending on the pointing direction of this relation the attended screen is chosen and a subtle highlighted selection border is shown on it, indicating the currently active screen. In this way it is possible to direct the throw gesture at either the Smartboard or the small picture frame (compare figure 4.7).

Once pictures are transferred onto the picture canvas, people can drag them around to change their visual order (z-index). A touched picture will always come to the front. When everyone sits down on the couch or a chair in front of the display an ambient slide show of pictures plays in a loop, according to the defined visual order on the screen (as shown in figure 4.2). When someone moves towards the screen or holds the camera up in front of the screen, the presentation is interrupted and the pictures on the canvas are restored to its original position and people can interact as described above.

The digital camera of course keeps its basic functionality: people can take pictures with it at any time and they will instantly appear in the current visualization of the camera‘s images on the large display.

4.1.2 Interaction

In the previous section I explained the general functionality of the Proxemic Picture Canvas. Now I revisit some interaction specifics with relevance to proxemics.

Visualization and awareness:
A main part of the application focuses on providing awareness of other devices, their contents and interaction possibilities through visualizing spatial relations.

One simple example is the icon of the digital picture frame which appears at the left side of the large screen (see figure 4.6 left), exactly where the physical entity of the picture frame is currently placed in the room. This helps the user of the system to easily identify the particular device by its spatial relation and he does not have to navigate through a traditional list-interface of devices in the network to select one. This example is inspired by Gellersen‘s Relate Gateways [18] and applied to a scenario with a vertical touch screen.

A more advanced visualization is the one of the digital camera. It takes into account continuous distance and orientation as well as discrete proxemic zones. When the camera is in the *distant zone* (see figure 4.8 c) it is visualized on the border of the Smartboard with a small icon of a camera. The icon continuously moves according to the position of

Figure 4.7: Directing the attention towards the digital picture frame enables interaction with it: a person can transfer pictures by performing the throw gesture.

the physical device and also changes its size depending on the exact distance. It grows when a person brings the camera closer, thereby creating awareness that further actions are possible and that further information will be revealed. When entering the *intermediate zone* (see figure 4.8 b) where a person has good sight to the display, a small stack of images appears from below the camera icon. This stack again continuously grows while approaching the display. When entering the *close zone* (see figure 4.8 a) the visualization spreads the stack apart into a circular fan of images around the icon. As the person is now close enough to touch the display, the visualization of the camera changes its behaviour and becomes interactive. It does not anymore react to subtle, but only significant changes of the camera‘s position. This is, to not interfere when the person drags images out of the visualization by using direct touch.

Interaction in discrete proxemic zones:
As mentioned already, the system uses discrete proxemic zones around devices to determine the most appropriate forms of interaction. Both the digital picture frame and the large display have three distant dependent zones, as shown in figure 4.8. These are inspired by the inter-personal distance zones defined by Hall [21].

The *close zone* (see figure 4.8 a) is defined by arms reach, so it is possible to interact with a device by touching it. The person can either interact with the images on the large display using direct touch or use the camera to touch a screen (this drops an images onto the picture frame or the large display).

The *intermediate zone* (see figure 4.8 b) provides a proper sight to the device and can be seen as an ’active usage zone’. Therefore the Proxemic Photo Canvas offers a remote interaction possibility: a throw gesture that pins images to the display or shows them on the picture frame.

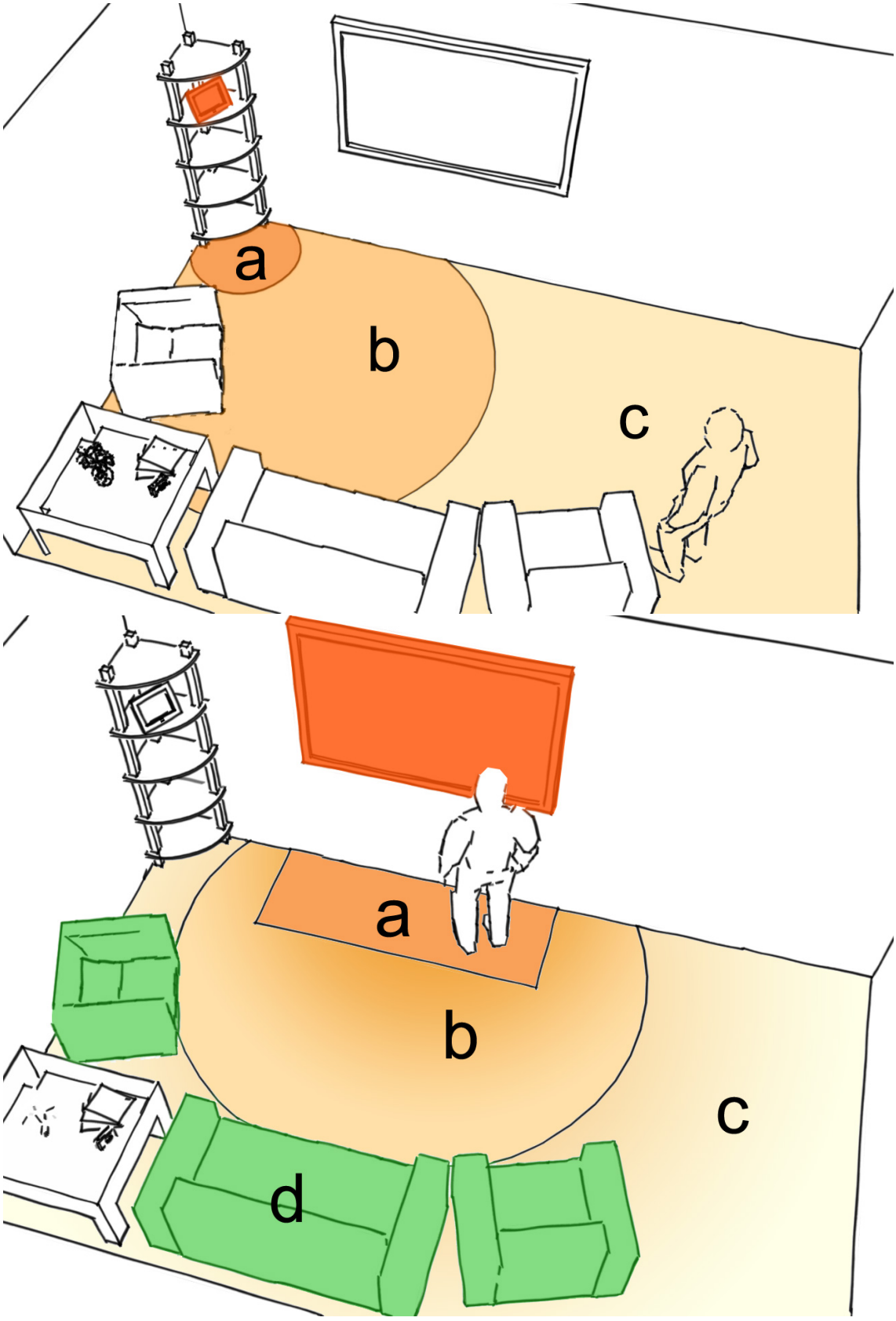

Figure 4.8: **top)** Discrete zones around the digital picture frame. **bottom)** Zones around the large display. **a)** close **b)** intermediate **c)** distant **d)** watching.

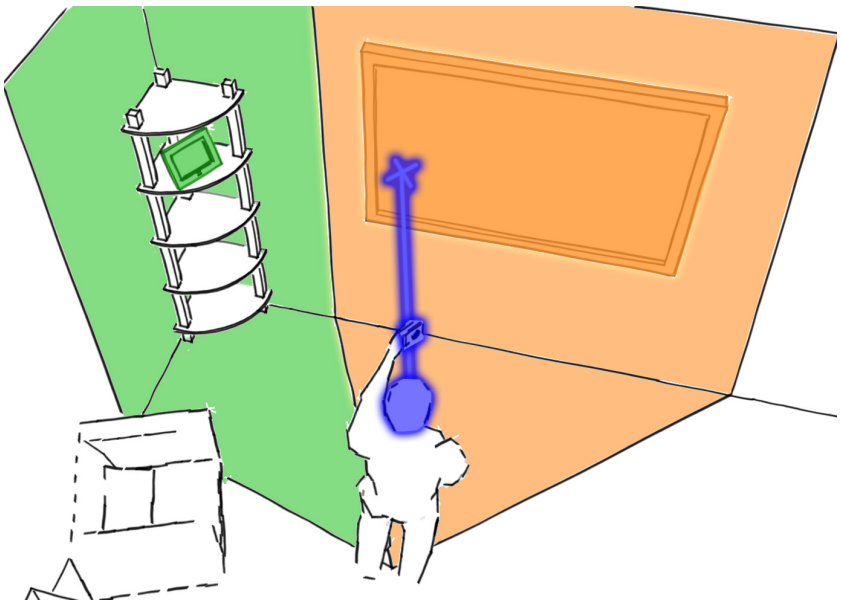

Figure 4.9: This figure illustrates how the application determines, if a person is attending the *picture frame* or the *large display*.

When a person is still far away, the system recognises him in the *distant zone* as marked in figure 4.8c. This is the area where the device can be recognized, but does not provide optimal view on its content. At this distance one would not actively engage with a system. Hence the application does not provide any interaction possibilities, but only creates ambient awareness through the small camera icon following the device.

While these zones depend on distance to the device, they are not just a circle with the device in its center. They are elliptical half circles or rectangles depending on the device itself and walls or furniture in the environment. So there is for example no zone at the back of the screen or behind a wall.

The Proxemic Photo Canvas has one more special type of zone which also depends on the position of the chairs and the couch. A person is in the *watching zone* (see figure 4.8 d) when he sits on the couch or a chair that is oriented towards the display. This is a place where a person or many people would passively watch content on the large display rather than actively engage with the system. Hence the application switches into a slide-show mode as soon as everyone sits on the couch or chair.

Interpreting directed attention:
Because the application can determine which device a person attends to, one can use the same throw gesture to either transfer an image to the picture frame or to the large display. In detail, the system looks at several proxemic dimensions. First it checks if the person is currently in the intermediate zone of one or more devices. In the case of standing in several overlapping intermediate zones, the system assigns a non overlapping area around every device and checks if the extension line from the persons head through the camera intersects with this area (see figure 4.9). In this case a visual border appears on the attended screen - the picture frame or the large display - and signals the attention to it. This indicates that a throw gesture will be directed at this device (when attending to the large display, also the tilt action can be performed). For example in figure 4.5 the large

Figure 4.10: The camera directly touches the display and lets an image from the camera‘s storage appear at the touch point.

screen is active and shows a white border around it. If the line does not intersect with a device area, none of the screens will be marked as attended.

Explicit interaction through the digital camera:
Rather than using virtual tools, menus or buttons, the physical camera device can be exploited to trigger certain actions.
As visualized on the large display, there is always a front image assigned to the camera. By default this is the latest picture taken and it shows on top of all others, either in the stack or in the fan visualization. This is the image that will be transfered when a person touches one of the screens with the camera. Figure 4.10 shows how a person pins the current image to the picture canvas. The same thing can be accomplished by the throw gesture when standing further away.
Another way to use the camera as a physical tool for interaction is to tilt it to the left or to the right. Doing so browses through the images and thus changes the front image. Only eight images will be shown in the stack or on the fan visualization at a time and they are ordered by date. Tilting to the left will load an earlier image from the camera and shift the others accordingly, so that the first one disappears. This way, one can navigate through all the images on the camera when standing in the close or intermediate zone of the Smartboard and holding the camera towards it.
While sitting on the couch and watching the slide-show the camera can be used to point at the Smartboard. This triggers a switch from the slide show to the default picture canvas mode and by performing a tilt action or a throw gesture it is again possible to browse and add images.

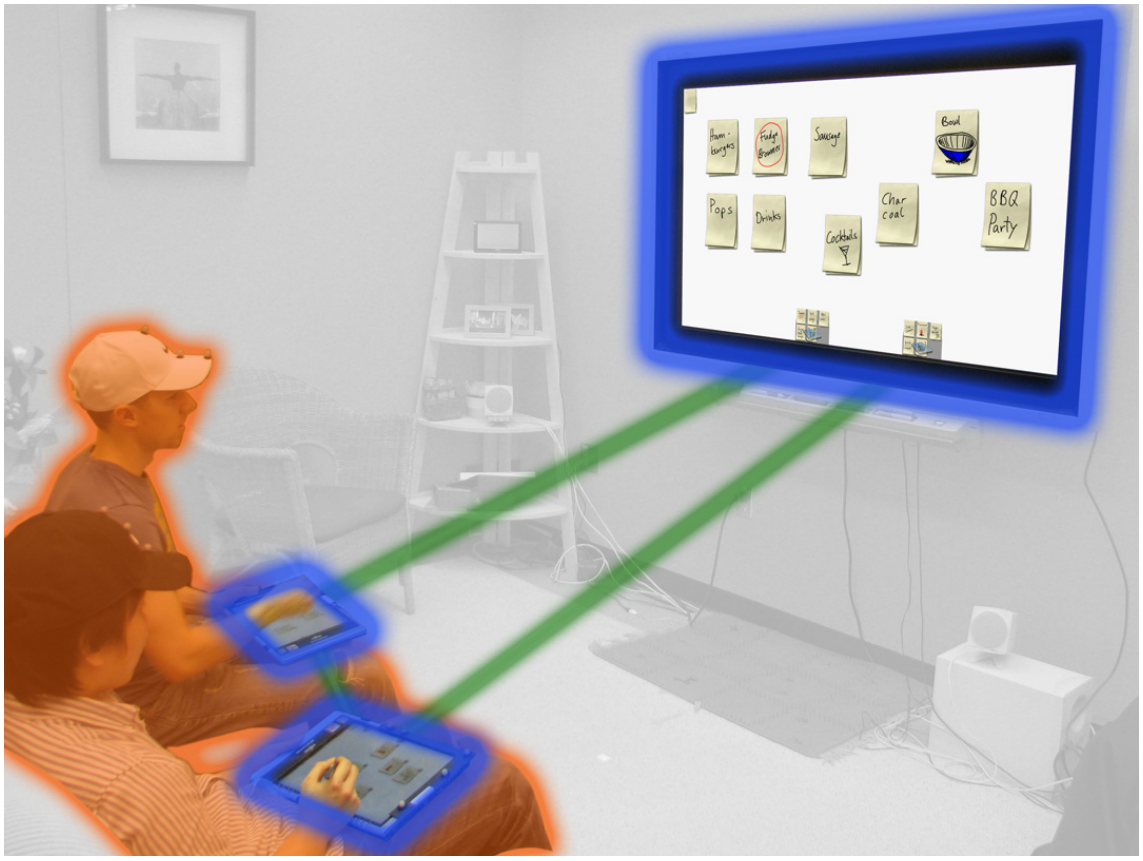

Figure 4.11: The default setup of the Proxemic Brainstorming application: a tablet computer for each person and a large shared display (marked in blue) - all interconnected over a wireless network (indicated by green lines).

4.2 Proxemic Brainstorming

The second application I developed is called *Proxemic Brainstorming*. It is a collaborative system for editing, sharing and discussing virtual sticky notes. Every person has a mobile tablet computer that can be used with a stylus as an input device. A large shared display (Smartboard) with touch input capabilities supports collaboration (compare figure 4.11). All these devices are interconnected over a wireless network. The application knows about the position of the furniture like chairs and a couch and the complete room layout and accurately tracks people and their tablet computers - receiving their exact orientation and location in space.
I will first explain all the features of the application in detail and later revisit the interaction in regard to spatial relationships.

4.2.1 Scenario

All the images and examples are taken from several sessions where participants were asked to use the application to plan a lab BBQ party.

Creating and editing sticky notes:
By default, all tablet displays and the large shared display show a canvas with the created sticky notes. These notes can be dragged around and arranged to support a specific

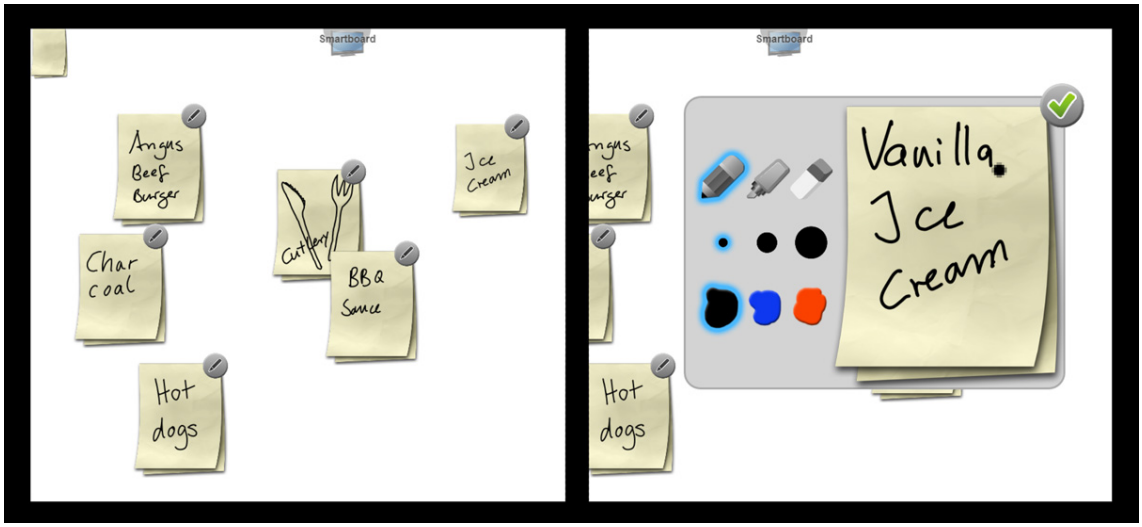

Figure 4.12: Screenshots of a tablet display - **left)** the default note canvas showing six sticky notes, the empty sticky note on the top-left and the icon of the large display in the top. **right)** shows the same canvas with one sticky note currently in editing-mode with a tool palette.

task. To create a new sticky note, a person can either use the large screen or his tablet computer. In both cases he drags a new note from the empty note-stack in the top left (see figure 4.12 left). Unless he explicitly wants others to see how he creates the note, he would probably take the tablet, as he can sit and hold it in a comfortable position while editing it. Once a new instance of the note has been created, a small pencil icon at the top right of the note can be pressed to switch into editing mode. On the tablet computer, these pencil icons are always visible, while on the large display they only appear when a person comes close to it. From a distance it does not make any sense to show these buttons - they can‘t be reached and only distract from the content itself.

Figure 4.12-left shows the canvas of a tablet computer with a couple of sticky notes on it. In figure 4.12-right the person clicked the edit symbol of the ’Ice cream’ note and wrote ’Vanilla’ on it by using the default pen tool. As the figure shows, the note sized up and a tool palette appeared left to it. The available tools include a pen for writing and a marker for highlighting. They can both draw in three different sizes and colors. Atop an eraser tool can be used to clear strokes. After a note has been edited with the stylus pen, pressing the green checkmark symbol brings the sticky note back into its default state.

Exchanging notes directly between tablet computers:
If there is more than one person with a tablet computer in the room, the tablet screen shows an icon of each person‘s device at its border. The icon consists of a half circle background, the symbol of a tablet computer and the name of the related device. It is positioned in a way, that it spatially relates to the position of the other device (see figures 4.14). This means it updates its position continuously whenever the orientation between the devices change. So the icon always points at the related device and helps the person to easily identify this correlation. Moreover the identification is facilitated by continuously adapting the size of the icon - it grows when the devices get closer to each other and becomes smaller when they are further apart (compare figures 4.13 and 4.14).

As described above, it is easy to identify a nearby tablet through its corresponding icon.

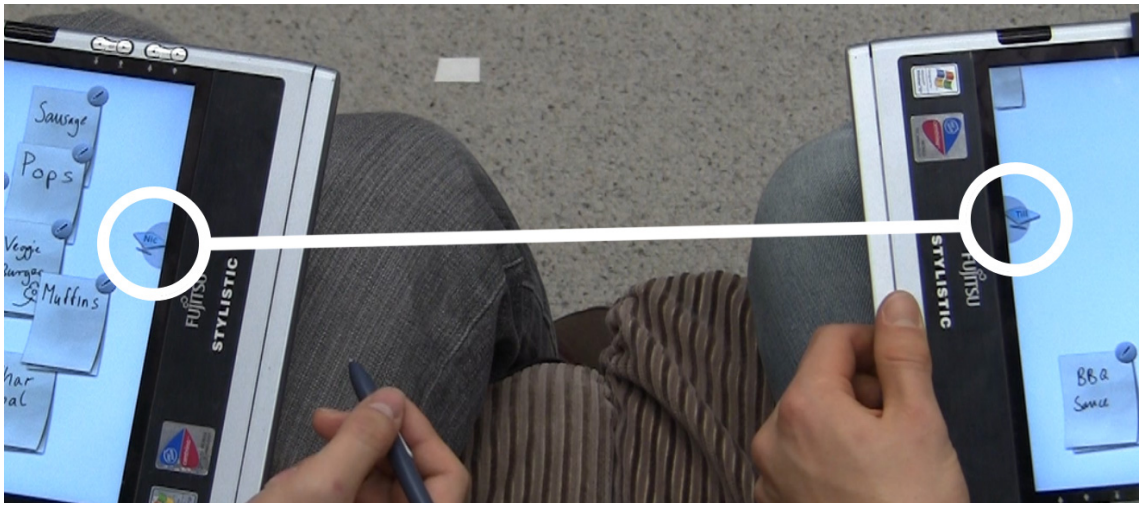

Figure 4.13: The spatially related tablet icons are smaller at a distance.

Figure 4.14: The tablet icons grow when bringing the tablets closer together.

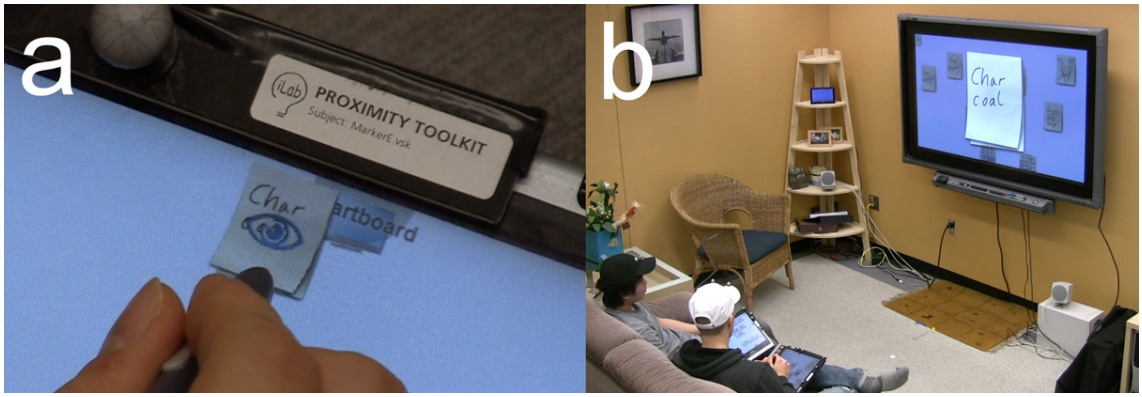

Figure 4.15: **a)** Dragging a sticky note onto the icon causes it to **b)** be presented on the large display for everyone to see.

People can now transfer a sticky note to another device by simply dragging the note on its icon. While dragged above the icon, a blue arrow indicates the transfer possibility (see figure 4.16). When released, the note is sent over the network to the other device, but it remains on both icons as a smaller representation moving with the icon (see figure 4.17). It is even possible to have multiple notes in this 'offer state' - they would appear as a stack of items. This behaviour ensures that one person can not take control over the other person's device and interfere with his current actions. There are two possible actions now. The first one is shown in figure 4.18, where the receiver drags the note from the icon into his canvas and thereby accepts the transfer. The note is scaled up again to the default note size for this canvas and simultaneously the note on the sender's side animates into the icon and disappears. The other possibility is that the sender retracts his note by dragging it back on his canvas. Hence the offered note disappears into the icon on the receiver's side.

Presenting notes on the large display:
Similar to the tablet icons that I just explained, there is an icon on every tablet screen with a symbol of a TV representing the large display. It has the same bahaviour of continuous movement along the screen border - always pointing to the large display depending on the spatial relations. As long as every person is in the *watching zone* or *distant zone*, people can drag notes onto the screen icon and they will be presented on the large display. Figure 4.15-a shows how a note is dragged over the large display icon and how it indicates the presentation possibility with a blue eye symbol. After releasing the note, it shrinks down and is assigned to the display icon - showing the eye symbol on top and following its movements. At the same time, the icon appears on the the large display in 'presentation mode' as a temporary overlay above the note canvas (see figure 4.15 b). Multiple people can drag multiple notes onto the display icon at a time. They appear stacked on top of the display icon on the tablet screen and the large display presents them side by side so that they are visible for everyone in the room and can be discussed. The presentation mode exits once all the notes are dragged back to the tablet canvases or when a situation occurs that requires the large display to show the note canvas. This can be a person standing up and approaching the display to interact with it or someone pointing with the tablet towards the display to select a particular sticky note.

Exchanging notes at the large display:
The third kind of icon that creates awareness of spatial relations is a representation of

Figure 4.16: A note is dragged over the device icon and the blue arrows indicate the transfer possibilities.

Figure 4.17: Once released the sticky note stays on both icons in an intermediary 'offer state'.

Figure 4.18: The receiver accepts the note by dragging it onto his canvas, while it disappears on the sender‘s side.

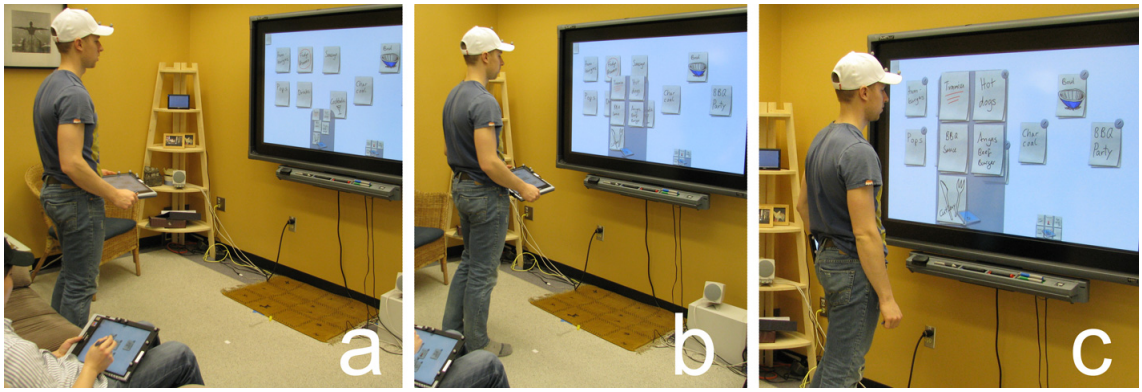

Figure 4.19: Approaching the large display with a tablet computer - **a) + b)** growing note area creates awareness, **c)** fully interactive fixed area for exchanging notes.

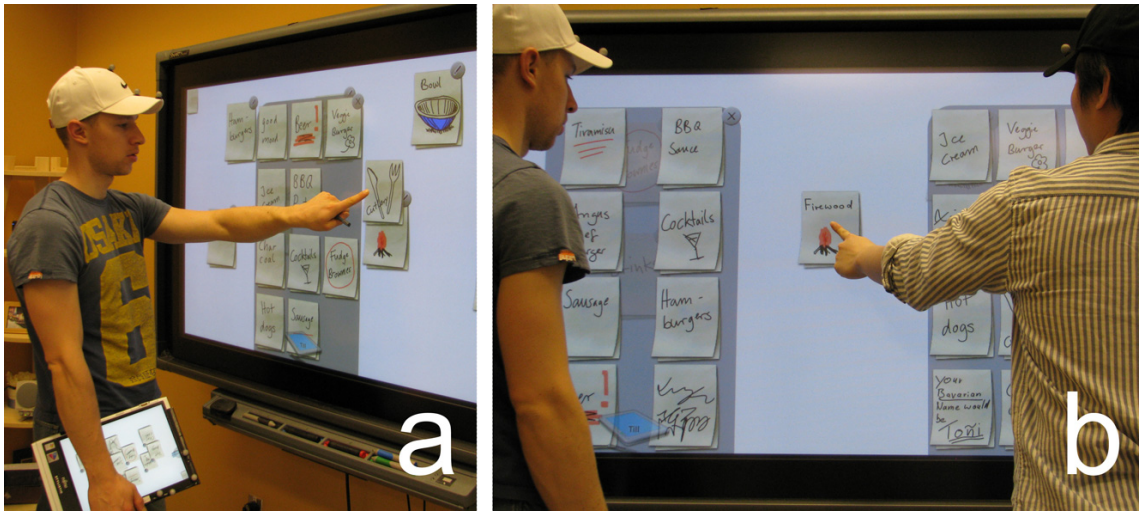

Figure 4.20: **a)** One person exchanging notes between his note area and the display, **b)** two people standing at the display and exchanging notes.

each tablet on the large display. This icon does not only show a tablet symbol, but also displays all the sticky notes that are currently on the tablet computer. When sitting on the couch or standing at a larger distance, the sticky notes are very small to fit into the box beneath the icon (see figure 4.11). The notes are synchronized at all time and provide ambient awareness of what other people are working on, without revealing too much information.

When a person now approaches the screen with his tablet, the box beneath the representing icon - with all the notes in it - grows continuously until it spans a complete interaction region around the person‘s body (see figure 4.19). In the intermediate zone the icon moves and grows continuously providing the best possible relation and instant awareness to the tablet computer. This behaviour changes into a more static note area, which only reacts to significant location changes. This is to not interfere with the interaction at the screen.

When standing close to the large display, a person can use the note area as an interactive

container representation of his tablet. He can drag sticky notes to the display canvas and vice versa to transfer them between the display and his tablet computer (see figure 4.20 a). So the display can be used as an intermediate storage before another person drags it into his note area. Also a person can directly drag items on other people's note areas to transfer them to their tablet. This can be done either when both people stand at the large display (see figure 4.20 b) or even when one person is sitting on the couch. If a person wants to use the display canvas to arrange or create notes, but his tablet note canvas is in the way, he can also minimize it and bring it back later.

Select and edit notes from the display by touching or pointing with the tablet computer:
A person can directly access sticky notes from the large display to edit them on his tablet computer, either by touching the display with his tablet when standing close or by pointing with the tablet from a distance.

When standing close a person lifts up his tablet directly in front of the large display. Doing so automatically causes the note area to minimize and clear the display canvas. Also a pointer is projected from the tablet onto the screen and follows its movements (see figure 4.21 a). As it intersects with a note on the display, the note is highlighted, thereby indicating that it can be selected. This is done by touching the note with one of the corners of the tablet computer. It causes the note to disappear from the display canvas and show up on the tablet's screen in editing mode as an overlay above the note canvas (see figure 4.21 b). The person can now edit the sticky note on his personal device and then bring it back to the screen, in two ways: either by pressing the green check mark of the overlaying editing mode which lets the note appear back at the pick-up position, or by again touching the display with a corner of the tablet which lets the note appear just at the point of the second touch.

Figure 4.21: **a)** Holding the tablet in front of the large display shows a pointer and highlights the selected note before touching it for selection. **b)** Pointing from a distance to select a note. The red annotated line emphasizes the pointing direction and the highlighted sticky note.

When sitting or standing at a distance, notes can be selected by pointing with the tablet. To to so, the person holds the tablet further away from his body to indicate a pointing action. This again minimizes the tablet's note areas and highlights the note on the display, which the pointing gesture directs to (see figure 4.21 b). The pointing direction is

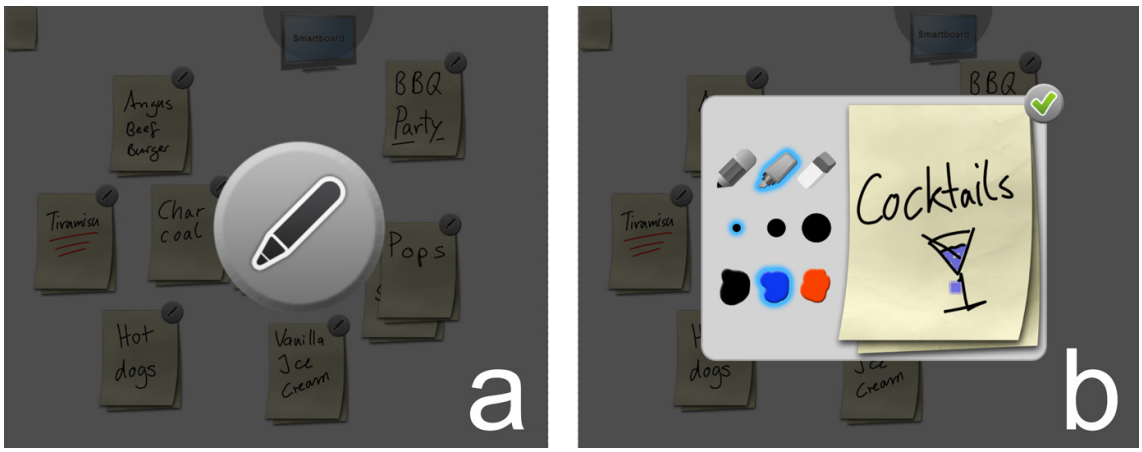

Figure 4.22: **a)** A button appears as an overlay while pointing and can be pressed to select the current note for editing. **b)** The temporary editing-overlay, that appears after selection of a note.

estimated by a line starting at the persons body through the position of the pointing tablet. Whatever note is closest to the intersection of this line with the display gets the focus. The tablet's screen shows a big button that triggers a selection (see figure 4.22 a). Once a sticky note has been selected remotely, it animates from the display canvas into the tablet icon and shows up in editing mode as an overlay on the tablet (see figure 4.22 b). Once the person is done editing the note animates back to its original position on the display canvas.

4.2.2 Interaction

In the previous section I described the application scenario and the single features. Now I want to revisit some interaction aspects in terms of proximity. In the application, the tablet computer can take different roles and have different functionalities depending on its proximity in relation to the large screen and the person holding it. In detail these are the following four roles (also illustrated in figure 4.23):

- **Editor and interaction canvas:** By default, the tablet device is used as a canvas where people can create, move and edit sticky notes. This role is active, when the person holds the device in front of him with the screen tilted so that he can look at it. This is normally the case when a person is sitting with the tablet lying on his lab. There are the two aspects the application considers when evaluating the tracking information: is the person currently in the *watching zone* of the large screen and is the tablet's position close to the user's body. As mentioned before, in this role the tablet offers further interaction possibilities with other devices: presenting notes on the large display and exchanging notes with others. This is accomplished by dragging a note on the corresponding icon.

- **Container:** Once a person grabs his device, stands up and approaches the large display, the tablet computer acts as a container with sticky notes. The content of this container is visualized on the large display inside a resizing note area, that continuously grows when approaching the display. It is placed and shaped in a way that it covers a projected shadow of the persons body onto the display. It lets

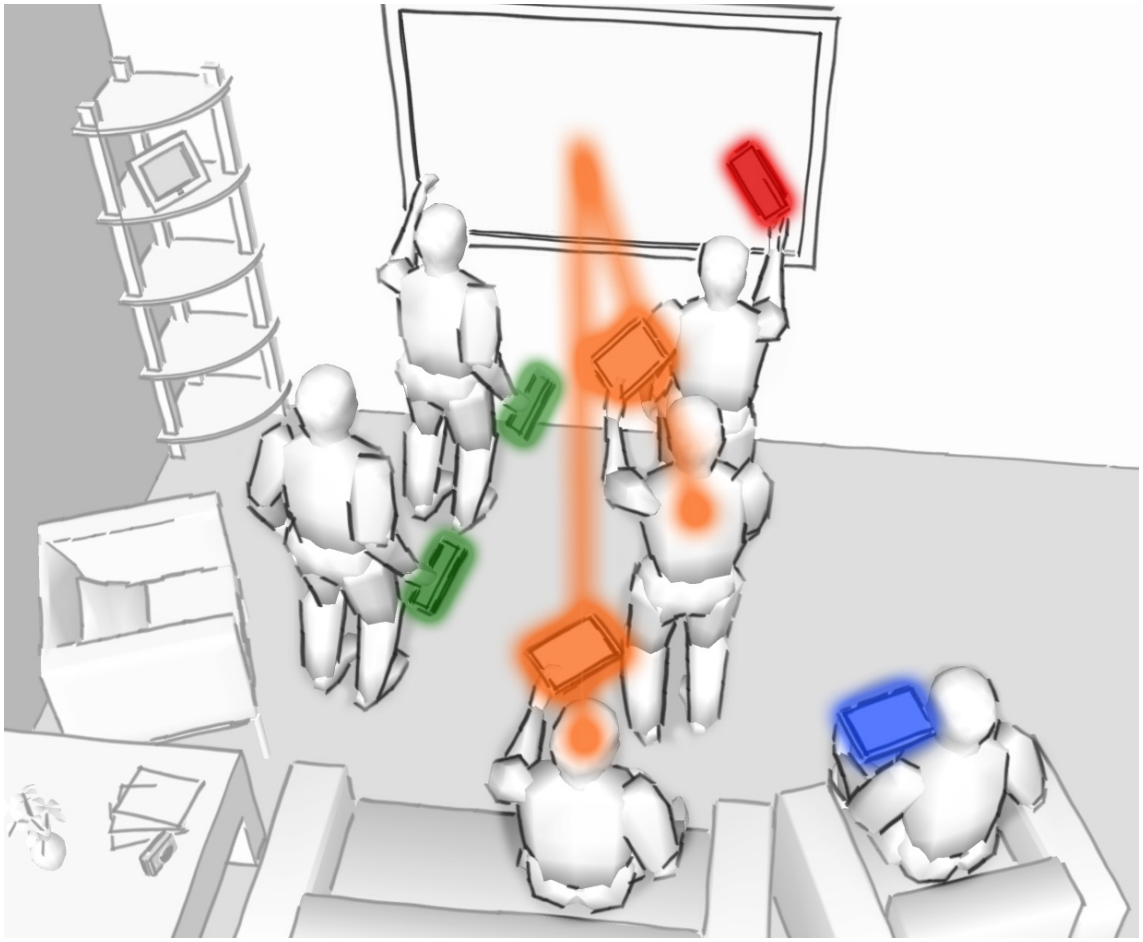

Figure 4.23: The role of a tablet computer defined by proximity - **blue)** Editor, **green)** Container, **red)** Selector, **orange)** Pointer.

the tablet‘s owner feel in control of his content at every time and provides an easily identifiable relation between the screen content and the tablet. When standing close to the display it seamlessly transforms into a personal interaction area, which can instantly be used to access the same content as on the tablet computer, because they are synchronized at all times. In terms of proxemics this notion of a container role is active when the person attends the display (his body orientation is roughly towards it) and stands either in the *intermediate or close zone* and when the position of the tablet is close to the persons body. When standing in reach of the large display, the tablet has to be held below a certain hight in order to be a container (more details in the next paragraph).

- **Selector:** The tablet itself can be used as a selector device, which picks up notes by physically touching them. It brings together the physical and the virtual world and builds upon the natural expectations of proxemics. The role is active when a person stands in the *close zone* and lifts the tablet above the lower border of the large screen - holding it directly in front of it. Open note areas are minimized and a projected pointer that highlights notes provides awareness of the selection while moving the device. When touching a sticky note with a corner of the tablet, it is temporarily transferred to the users device. He can instantly modify it with the provided editor on his device and put it back to a desired point on the display canvas

by again touching it with a corner of the tablet. The role of a selector can quickly switch back to the container role when holding the tablet at the side of the body again.

- **Pointer:** Similar to the selector role the tablet can take the functionality of a pointing device, when not standing close to the large display. This pointing role is activated whenever the person holds his device away from his body towards the display. Hence the system checks if the person stands in the *watching, intermediate or distant zone* and if the line starting from the person's body through the position of the tablet device intersects with the display region. Performing this pointing action highlights the closest sticky note to this intersection point and provides a button on the tablet that triggers the selection. The selected note is temporarily transfered to the tablet's screen, can there be modified, and is then sent back to the display. Note that a pointing action can interrupt an ongoing presentation of notes on the Smartboard.

These roles employ a variety or proxemic dimensions - building upon existing interaction techniques with the physical world in everyday situations. The application's interface provides seamless transitions between the roles and mediates actions of multiple people in a room.

5 Concepts of Proxemic Interaction between Digital Devices

In this chapter I describe higher level concepts as a merging of the most important techniques for proxemic interaction with digital devices, primarily extracted from designs of the previously presented proxemic applications.
There are three concepts that I want to highlight. For each I will first explain the general idea and then describe scenarios that illustrate them. These concepts however do not have to be separated - in fact they are more powerful when fluently interweaved to support each other - as I will revisit later.

5.1 Awareness through Continuous Visualization of Proxemic Dimensions

One thing Proxemics are in particular good for, is to create awareness about other devices nearby. A powerful way to do this is to visualize proxemic dimensions in a continuous manner using distance and orientation between devices combined with discrete proxemics as identity and location.
In this regard, awareness can be split into three steps: identity awareness, content awareness and interaction awareness. I explain why they are important and how they finally lead to interaction. Then I show how techniques from the presented proxemic applications cover each step.

1. **Identity Awareness** relates to the problem of identifying nearby devices in a computer network. Gellersen [18] and Rekimoto [47] presented techniques that employ the idea of displaying a representation of another device in a way that it spatially relates to its physical location. This idea can be very powerful as one can easily recognize the physical instance of a device and can thereby relate it to the representation on his screen. I extend on this existing work in several ways: I *continuously* visualize the *orientation* and *distance* between devices by showing a representing icon that moves in very fine gained steps whenever the orientation between devices changes - combined with continuous changes in size and detail depending on their closeness. The immediate and fluent response to spatial relations lets one instantly identify other devices. This is further facilitate by showing the representing icon on the displays of both devices, even combining vertical and horizontal screens. Atop I combine this notion with *discrete* proxemics. I react to the *location* of devices by completely hiding the icon, as a device leaves the shared space (e.g. exits the room). The device's *identity* is used to display an appropriate icon depending on the type of device (e.g. tablet computer, a camera or a TV). As *movements* are encoded in fast continuous changes of distance and orientation, they can also be identified with my approach. Therefore I cover the complete range of proxemic dimensions to best support the goal of identifying other devices.

2. **Content Awareness** is the next step, as knowing what content a device offers is important when deciding on further interaction. I continuously reveal content of other devices around the mentioned icon that represents the device. The amount of content shown depends on distance between devices. Chris Harrison's Lean and Zoom [23] is a related project that exploits the idea of reacting to continuous distant changes: a website is magnified depending on distance of a persons head to the computer. My approach applies this idea to distances between multiple devices, where one device sees a representation of another device on its screen. These representations continuously change the amount of information revealed. They move from low fidelity (e.g. indicating the amount of images or type of content on a device) to high fidelity content (e.g. a full size sticky note).

3. **Interaction Awareness** is the final step that can lead to interaction. I indicate the interaction possibilities that a device offers by adapting the representation of interactive elements. This can be done in a continuous manner: scaling an item gradually to finally fit the same size as other interactive elements on a screen; or continuously transforming and moving an area to where a person could interact (e.g. adapting the sticky note box, that represents tablet content on a large display, to span around a person‘s body when he comes closer); or indicating interaction results by e.g. projecting a pointer that follows a device and highlights items to indicate a selection possibility. Discrete measures of proxemics can be used to change the representation when a device enters a new interaction zone. For example showing buttons when standing close to an interactive display suggests interaction possibilities.

I want to show how my example applications support these steps and how they apply the concept with a fluent visualization of devices that finally guides to interaction.

The Proxemic Photo Canvas enables exchange of photos between a camera and a large interactive display (Smartboard) and therefore applies the proposed concept. When a person enters the room with his camera, a small icon that looks like a camera appears on the screen (see figure 5.1 a). This first step is triggered by discrete proxemics: *location* provides the information that the device is now present in the same room and *identity* helps to determine that the device-type is a digital camera.

The icon is placed at the border of the Smartboard‘s screen so that it *orients* towards the position of the camera in the person‘s hand. While moving the camera in the room, the icon continually updates its position, thereby creating an unambiguous association between the icon and the device. Even if multiple cameras would be present in the room, the corresponding item can be identified through matching *movements* and relating size to closeness and *distance*. Once the person approaches the display, a stack of photos slowly appears below the icon (see figure 5.1 b). The person would first recognize the fact, that the camera contains photos and then get an impression about the type of photos (e.g. distinguish between technical documentation, winter or summer vacation photos). While coming closer to the display, the stack transforms into a spiral fan of images arranging around the icon, continually growing in size (see figure 5.1 c). The person is able to identify single pictures and see their details, showing him what exactly the camera con-

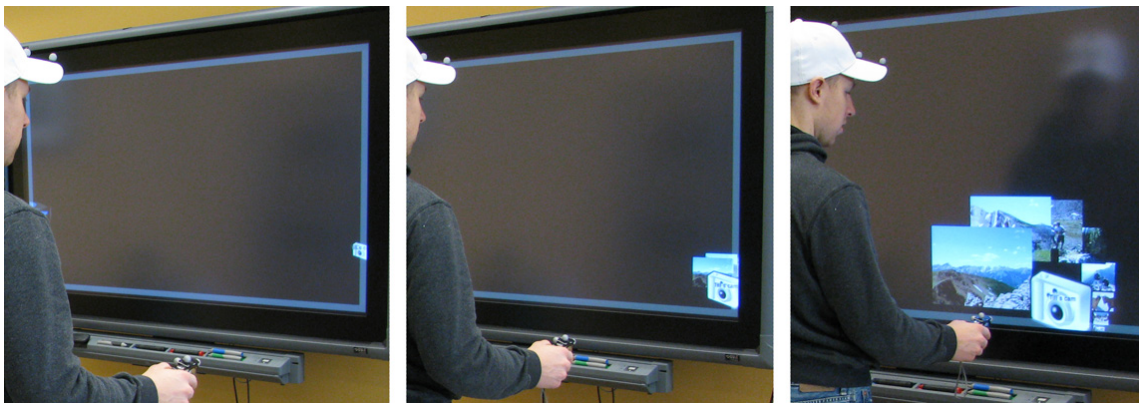

Figure 5.1: The camera symbol on the screen spatially relates to the physical device, gradually revealing photos and arranging them as a stack and then in a spiral, while coming closer.

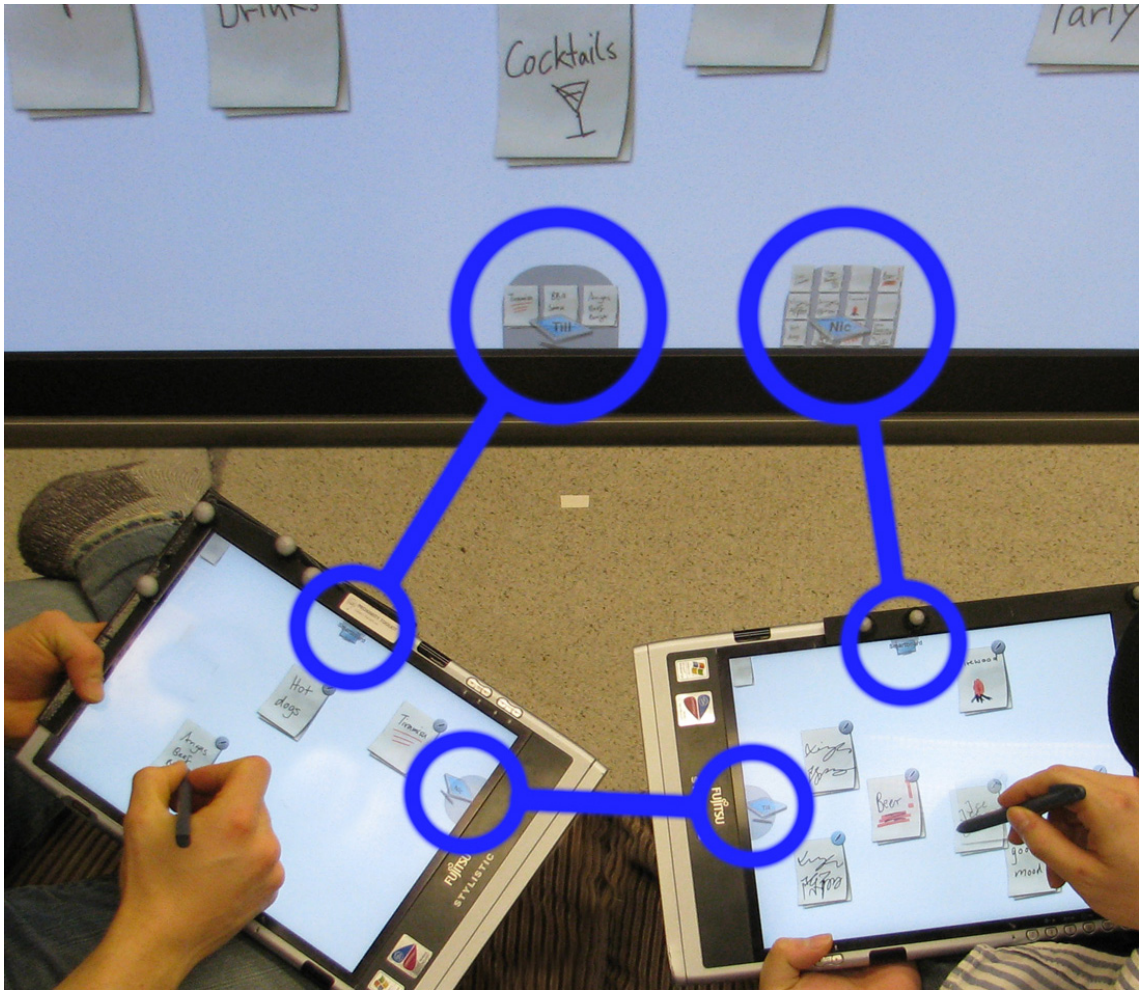

Figure 5.2: Two horizontal tablet screens and a vertical Smartboard show spatially oriented icons that represent each other.

tains. By then the pictures around the fan appear in the same way as the ones already present on the Smartboard canvas, suggesting the same drag-interaction by using direct touch.

Another feature of the Picture Canvas illustrates this concept. When holding the camera in front of the display - either at a distance or close by - a person can directly pin the latest image to the Smartboard canvas by either touching it with the camera or performing a throw action. I create awareness for this action by the following behaviour: when bringing the camera in front of the display, the icon is projected onto the display (using the normal vector to the vertical display plane). The icon again follows the movements of the camera while showing the latest image beneath the camera symbol (see figure 4.5). This image represents a preview of what it will look like, when the person executes the appropriate action to pin it to the display. So it creates awareness of the camera‘s identity and its interaction possibilities.

The Proxemic Bainstorming application enables the exchange of sticky notes between tablet computers and the Smartboard and also demonstrates how all three steps of awareness engage peoples interactions. As figure 5.2 shows, each device is represented

on the other device‘s screens - vertical and horizontal - by an icon that spatially relates to the physical position of its device. These icons continually move and change their size depending on distance and orientation, therewith facilitating the identification of the physical device that relates to the icon on the display. The two boxes around the icons on the Smartboard are filled with synchronized notes from the tablet computers. From afar people recognize that the other tablet contains sticky notes and can estimate the amount of notes. When they move closer with their tablets, the box and the containing sticky notes grow larger and larger until they are visible in full size creating awareness of the tablets content. As a person reaches the Smartboard, the box has transformed to span the shadow of his body projected to the display, indicating a personal interaction area. Furthermore the sticky notes inside have reached the same size as existing notes on the screen and controls for editing notes and manually closing the box are revealed (see figure 4.19), all suggesting interaction possibilities by using direct touch.

5.2 Discrete Proxemic Zones for Appropriate Interaction Modalities

The second concept suggests to base interaction possibilities on proxemic zones, while taking into account the assigned spatial meaning if the respective zone.

The notion of proxemic zones has first been explored by Edward T. Hall in a social context [21]. He defined specific distances between people and assigned a meaning to them: intimate (less than 50cm), personal (0.5 to 1m), social (1 to 4m) and public (greater than 4m). Researchers in computer science followed this notion and applied zones as a way to mediate interaction with public ambient displays [65] and digital whiteboards [30] or to negotiate information exchange in between mobile digital devices [32]. What all these projects have in common is, that closer zones suggest more detailed, personal and explicit interaction, while distant zones are used for ambient, public and implicit interaction. Atop, these zones can be influenced by peoples attention towards devices (related to AUIs [63]) by taking into account a person‘s body orientation and the way a person holds a device toward another device. Again providing more detailed and explicit interaction when attending a device and moving to ambient and implicit interaction when a device moves in the persons periphery.

The third influential factor of proxemic zones is the environment itself. Edward Hall distinguishes between fixed and semi-fixed features [21] - meaning fixed room layout and walls as well as movables like furniture. He identified that this space layout would influence our use and perception of personal space. As a consequence, distance zones can be interrupted by walls or doors and seating arrangements can create further meaning.
When designing a proxemic system for interaction between devices, my concept is to apply such proxemic zones around devices and the people using them, and then choose interaction methods that fit the derived meaning. I further illustrate the concept by explaining interaction scenarios from the proxemic applications for each zone:

Private and intimate zone:
The Proxemic Brainstorming application illustrates how detailed and explicit interaction between devices is possible, as their personal zones intersect. When a person comes close to the Smartboard with a tablet computer in his hands, a personal interaction area appears just where he stands. The area contains the sticky notes from the tablet computer and spans the personal interaction bubble of its owner (see figure 5.3). He can drag notes in between the display canvas and his personal box of notes to explicitly trigger an exchange of the data.

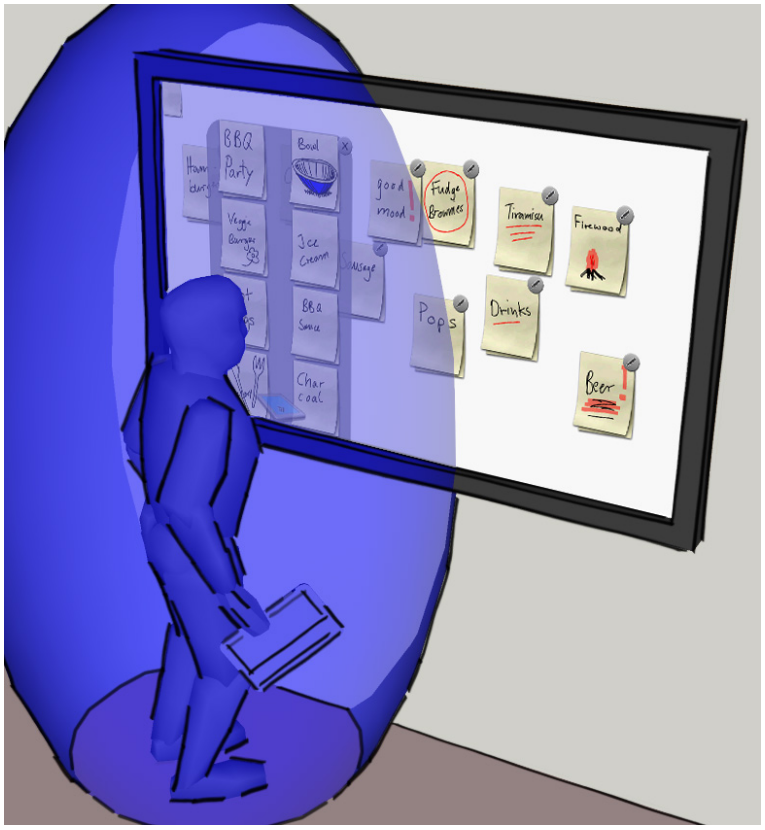

Figure 5.3: The personal interaction bubble of a tablet computer and its owner intersects with the Smartboard. This creates a personal interaction area on the screen to exchange information with the tablet.

Another interaction possibility is a direct touch of a digital note on the screen with the tablet computer itself. This brings the devices intimately close together and initiates an editing possibility of the touched note from the Smartboard on the tablet computer.
Similarly, the Proxemic Photo Canvas offers direct touch interaction when standing in close proximity to the Smartboard. Images from the digital camera appear as a spiral around its personal space-'bubble' on the Smartboard. Dragging a picture to the screen canvas, transfers a copy of it (see figure 5.1 right).
Again, a direct touch with the camera on the screen is a possible interaction modality. Doing so sends and pins a photo from the camera to the touched point at the screen.

Social zone:
When moving further away from the large display (into the social zone), interaction is possible using a throw gesture. As this zone intersects with the digital picture frame's zone (see figure 4.8 b), it is further limited by the the attention of the person and his camera. He can direct his action by holding the camera device towards the screen he wants to interact with (see figure 4.9) and then perform the throw action to transfer an image.
The Brainstorming application offers a selection possibility of sticky notes through pointing and pressing a button when located in the social zone of the Smartboard (see figure 4.21 b).

Public zone and sitting zone:
Both applications do not offer explicit interaction in the public zone, but only provide ambient awareness of device relations through small representing icons. However, as the arrangement of chairs and the couch around the Smartboard infer a social meaning of 'watching', the applications define a sitting zone around this semi-fixed environment. In the Proxemic Brainstorming application interaction with the Smartboard is possible while in the sitting region. People can drag sticky notes onto the representing icon of the display to show them in a presentation mode to others (see figure 4.15).
The Proxemic Photo Canvas provides a slide show on the Smartboard, presenting images from its canvas, while people are seated.

5.3 Explicit Interaction through Physical Devices

For explicit interactions between devices this concept suggests do use interaction techniques that incorporate proxemic dimensions of these devices and appropriate their contextual meaning. The general guideline is to let proxemic actions with a device affect its associated information. They can become a physical tool for interaction depending on the proxemic context. This can require to use the first concept, to create awareness of an association of displayed data with a device.

For example, I described how photos from a digital camera are revealed on a large display around the physical position of the camera and how the person holding the camera is aware of the association of the camera's pictures with the screen. To navigate through the spiral of pictures the person can tilt the camera to the left or to the right. This causes the spiral of images to rotate and reveal further images (see figure 5.4). The tilt interaction itself is inspired by existing systems that use tilt of mobile phones or PDAs [6] [43]. What they don't consider - but what is essential to achieve a seamless experience - is the complete context of proxemics and its preceding events. The context for example defines that a tilt action is only possible while attending the display with the camera, and not while the camera is in a person's pocket or while one is watching the slideshow. As mentioned, preceding events create awareness of the association between the camera and the representation on the display. This refers back to the concept of 'Moving from Awareness to Direct Interaction' as described in chapter 3.4.

While holding and moving the camera in front of the large screen a camera symbol and the first picture is projected to the display, following the device's movements (see figure 5.5). The camera itself can now be used as a drop-tool to pin the associated image to the canvas of the screen by touching its surface with the camera. This technique is inspired by Rekimoto's pick and drop [44], but is not limited to a dedicated device like a pen. It uses the actual device which is the source of the dropped information. Additionally, continuous proxemics create further awareness of the action, as the visualization previews the dropable image while hovering with the camera in front of the screen.

When standing at a distance, the same drop-action it possible by performing a throw gesture with the camera. This means the person accelerates the device towards the display to drop the projected image onto the canvas. Throw gestures have been used to transfer information between a mobile phone and a distant display [9]. However these techniques just consider a coarse binary action, whereas the Proxemic Photo Canvas facilitates fine grained proxemics to create awareness and enable exact placement of information: bringing up the projected icon with the preview image when holding the camera in front of the display creates awareness of the exact position on the canvas, where the image can be

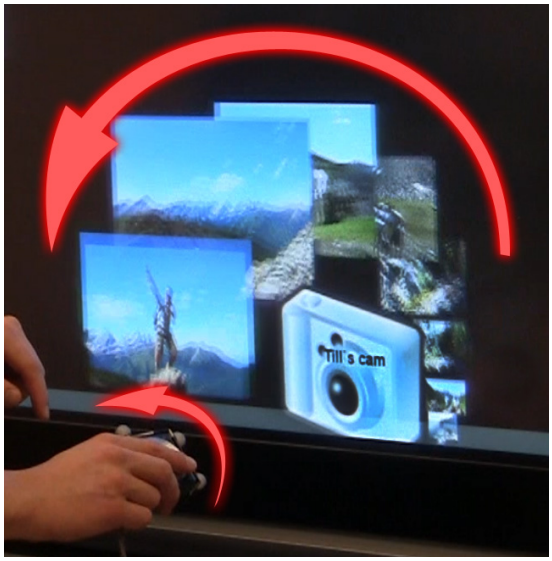

Figure 5.4: Browsing through images is possible by tilting the digital camera to the left or right.

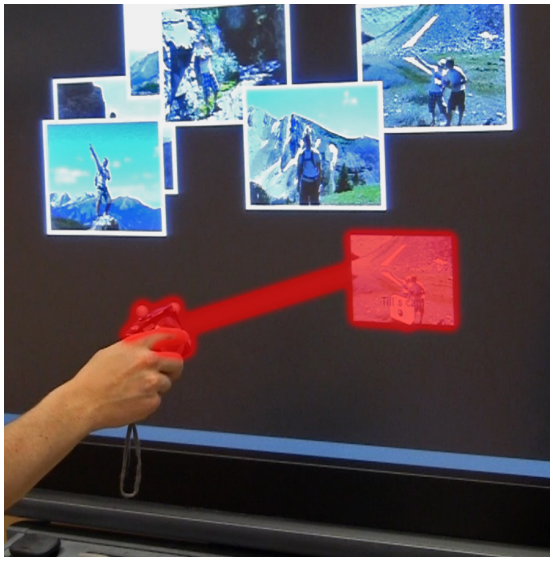

Figure 5.5: An icon of the camera and the image is projected, before dropping it to the screen by touching or throwing.

dropped with the throw action (see figure 5.5). Another point, where it extends existing projects, is that the same throw action can affect different screens (the digital picture frame or the Smartboard) estimated through proxemics.

Similarly, the Proxemic Brainstorming application allows a tablet computer to turn into a picking-tool for sticky notes. When holding it in front of the screen a pointer is projected onto the canvas (see figure 5.6 left). It creates awareness of which note the tablet would select by highlighting a note while hovering. When the tablet touches the screen at that point, the note is transfered to be edited on the tablet. It can then be dropped back to the canvas by physically touching it again.

At a distance, the tablet computer can act as a selector. The selector-tool is activated when the tablet is held towards the screen (see figure 5.6 right). The note where the tablet points at is highlighted and can be selected by a click on the tablet‘s screen (it shows one big selection button). This technique does not consider height changes but only side to side movements, so that the tablet computer can be held in a comfortable position during the selection.

As stated throughout the description of the techniques: these three concepts are most powerful when applied in combination with each other. This way techniques can create awareness of other device‘s identity, their content and appropriate interaction possibilities; and then fluently move to interaction facilitated through proxemic dimensions between devices and their owners.

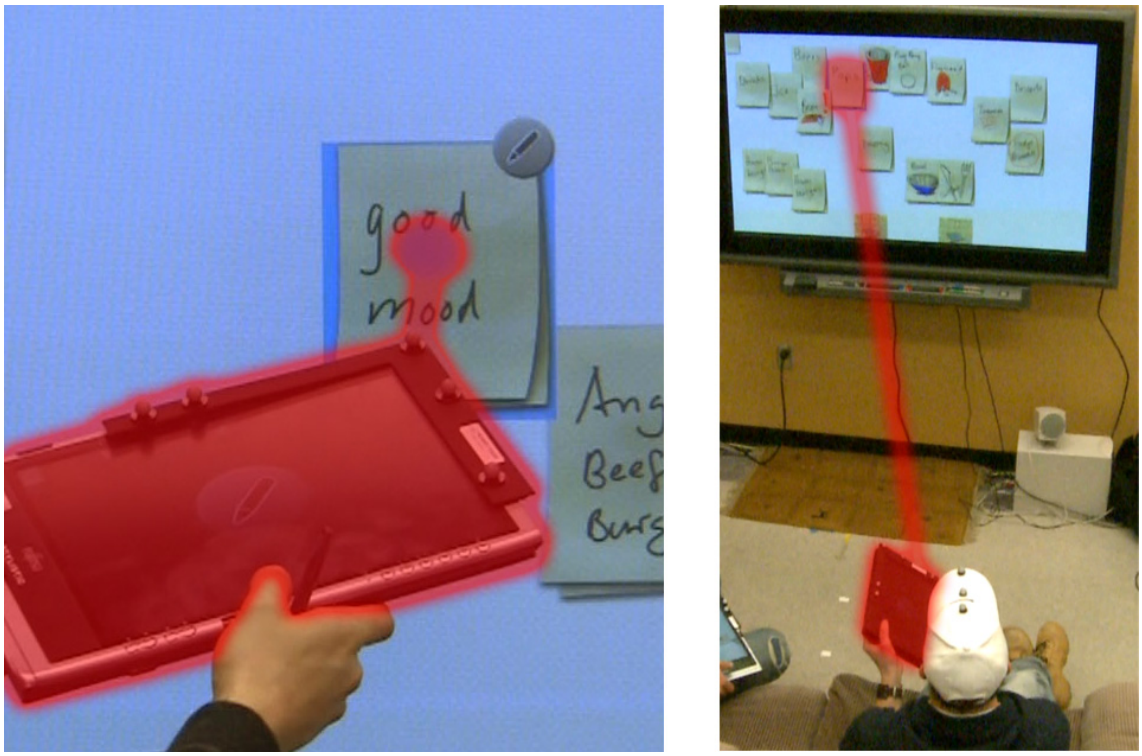

Figure 5.6: Selection of sticky notes in close proximity to the display and from a distance. **left)** Holding the tablet in front of the screen before selecting a sticky note by directly touching it with the device. **right)** Pointing at the screen with the tablet to select a sticky note.

6 Implementation

6.1 Tracking Technologies

In the previous chapters I pointed out that applications illustrating proxemic concepts may require fine grained knowledge about the *five Proxemic Dimensions* in a small space environment. While there are a variety of technologies available which provide different degree of detail for different dimensions, I focus on a prototyping platform that provides very accurate information about all these dimensions: a Vicon Motion Capture System [64]. It is a prototyping platform because it has several downsides that make it very hard to be employed in peoples homes or offices, as I will explain later. Even if some applications may not require this high tracking accuracy, I believe using this system is very valuable because one can explore the possibilities of Proxemic Interaction without being limited by technology. Focusing on the design of proxemic interaction is even further facilitated by the use of the Proximity Toolkit [11] for the Vicon system. It provides accessible and high level access to spatial information.

6.1.1 Vicon Motion Capture System

The Vicon Motion Tracking System is a high-end installation that uses a number of infrared (IR) cameras and IR emitting lights distributed around a trackable area. Sets of passive IR-reflective markers, which can be attached to objects, are captured by the cameras. A hardware unit computes their three dimensional position in space from multiple camera images and provides them to a computer for further processing. I will now explain these hardware components, their capabilities and setup and the software in detail.

Vicon infrared cameras:
Vicon offers several types of cameras for different purposes. This system uses MX-F40 Vicon infrared cameras (see figure 6.1) which each contain a four megapixel 370Hz high speed image sensor that is tuned for detecting infrared light. The lens is surrounded by a ring of IR light-emitting diodes (LED) which is necessary to see the passive markers. As figure 6.2 shows, the latest setup includes eight of those cameras mounted to the ceiling at a hight of around three meters. They are distributed in a way that they each capture a different angle and that at least three cameras can see every point in the room that is important to the applications. However it is hard to not have any blind spots, especially if there are objects that can cause occlusion. As the main interest is in the area in front of the large screen, there are cameras focusing there from different sides. This is possible by using a variety of lenses for cameras in different distances to this area, ranging from 12.5 millimeter wide angle for a smaller distance to 20 millimeters from a greater distance. The result is a reliable tracking in an area of roughly four by five meters around the large display.

IR reflective markers:
IR reflective markers are tracked from the cameras in the above described area. They are small plastic balls as shown in figure 6.3, covered with reflective tape from 3M. Their sizes can range from a diameter of some millimeters up to about two centimeters. In general the tracking is more reliable the bigger the markers are. For the setup that I used, a marker diameter of eight to ten millimeters worked very well. To identify a certain object and distinguish it from others, a set of these markers has to be attached to the object. For creating such a set one as to take info account the following:

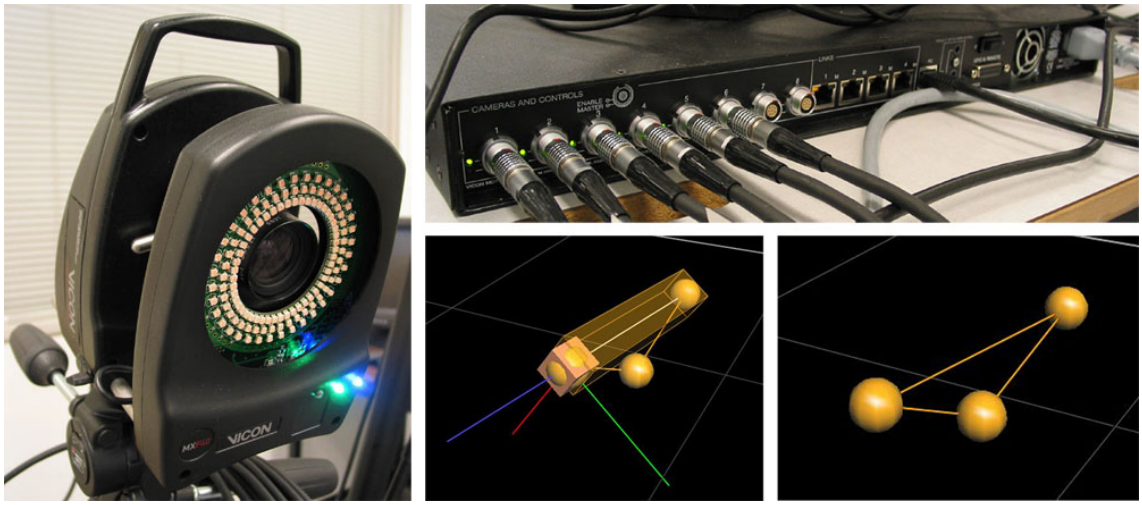

Figure 6.1: **left)** MX-F40 camera with a IR LED ring around its lens, **top-right)** MX ultranet server, **bottom-right)** Body models of tracked markers in the Nexus Vicon software.

Figure 6.2: **top)** The prototyping environment as a living room setup with eight Vicon cameras mounted to the ceiling, **bottom)** three additional cameras are placed on the other side of the room.

Figure 6.3: IR reflective markers: **top-left)** sets of markers directly attached to objects, **bottom-left)** single markers in different sizes, **right)** reusable marker bars.

- At least three markers have to be arranged in a unique pattern. More markers can help to improve the tracking reliability of the pattern orientation.
- Avoid symmetrical placement of markers, so that the pattern does not look alike from different viewing angles.
- Make sure that the set of three base markers is visible to at least three cameras at any time. Markers should be placed on an object where they will not be occluded by other markers, objects or people.
- Leave a clearly visible gap between markers of at least two centimeters (depending on the marker size). Otherwise the image processing might detect them as one blob.
- Apply markers only to firm material so that they can‘t move and change their spatial relation, otherwise recognition might fail. An exception to this are joints, but they are not supported by the Proximity Toolkit and non of the presented scenarios required them.

As shown in figure 6.3, for my applications several objects are equipped with those marker sets; either by glowing them directly onto an object (e.g. a baseball hat or a pen) or by creating reusable cardboard marker bars that can be easily attached to various objects and devices (e.g. tablet computers or a digital camera). Also they can be taken off and put back on at any time so that a device can be used without markers for other tasks.

Vicon Server:
The MX ultranet acts as a pre processing unit of the camera‘s raw data. Up to eight cameras can be plugged into a unit. If more cameras are necessary for tracking, it is even possible to add a second ultranet server. It does real time processing of the raw image

data to first extract markers in each camera image and then triangulate their positions from a minimum of three cameras into 3D points in the room. Also it maps these points into predefined rigid body models that correspond to a set of markers attached to a physical object. These bodies have to be calibrated and labeled in order to be recognized by the system. This is done on a computer to which the MX ultranet server is connected, using gigabit Ethernet and TCP/IP to exchange information.

Vicon Software:
The Vicon Nexus software can be used to calibrate and maintain the described hardware components. First it is necessary to calibrate the tracked space and then define the marker sets so that they can be recognized as rigid bodys with the ultranet. As the software shows a live 3D model of markers in the observed space, markers can be detected visually and combined to a body model using a small wizard. Once a body is created and labeled, the information is stored in a file which can then be loaded whenever the corresponding object with the set of markers should be tracked by the cameras. Also the software shows the single camera images and lets one identify reflection of infrared light that could interfere with the tracking of markers. The reflections can be handled by either moving the reflection-causing object out of the area or by marking the reflections in the software so that they appear as blind spots to the camera.

The result of this setup is a very high speed and millimeter precision tracking of the position and orientation in six dimensions of freedom of several marker equipped objects in a room. However there are a few drawbacks of the system: first the hardware components are very expensive and as described the setup requires a certain effort and know how. The markers of course can be disturbing or hard to install and as it is a vision based approach, occlusion can become a real problem. In particular if there are multiple people with multiple devices in the room, this technology quickly reaches its limits and becomes unreliable.

6.1.2 The Proximity Toolkit

Accessing raw data of the Vicon system requires a lot of effort and basic steps have to be repeated over and over for every single information in every single application. Hence it is hard to quickly prototype and evaluate ideas. This is why Rob Diaz-Marino, Nicolai Marquardt and Saul Greenberg designed the Proximity Toolkit [11]. It takes the Vicon raw data as an input, extracts the relevant proximity information and offers it to the developer through an accessible event based application programming interface (API) for .NET C# (My latest applications use version 1.2.1 of the toolkit).

The proxemic information offered covers all of the five proxemic dimensions: distance, orientation, motion, identity and location. In addition to accessing the information by code, it can be explored in a visual editor as shown in figure 6.4. The editor provides an interactive 3D model of the observed space including live representations of three kinds of objects: every marker set defined in the Nexus Vicon Software can be accessed in the toolkit as a *Presence*. *Volumes* represent static objects like furniture or walls and can be defined with a small tool which generates volumes from captured locations of a specified presence. *Displays* represent movable or static screens and can be assigned to software screens from the operating system. Proxemic dimensions and specific proximity values between these three object types can be explored in the visual editor and can be accessed in code. These values include: distance between two objects as well as their absolute position as a 3D point in the room, orientation of an object as angles of pitch yaw

and roll, motion as values for acceleration and velocity, pointing as an intersection point of a casted ray with another object and collision as a binary value or an intersection point of two object's collision volumes. It is even possible to access higher level information such as if two objects are directed towards or away from each other. The editor program integrates a server for proxemic information that can be accessed through the API.

Once the objects are set up, accessing them in code is done in a simple object oriented way. Figure 6.5 shows an example of how to access the distance between a hat and a display. The first block connects to the proximity server via TCP/IP. In this case the server is running on the same machine on port 888. Then a `TrackedPresence` object is created for the movable hat and a `DisplayPlane` object represents the display. A relation between these objects provides an `OnLocationUpdatedAsynch` event which is fired every time the spatiality between the two objects changes. The `LocationRelationEventArgs` provide access to distance and other related values. The API provides access to all other proxemic values in the same straight forward object oriented and event based way. Missing dimensions can be retrieved by the advanced user of the toolkit as raw 3D vectors of all the markers. I will talk about this when I describe the additional modules in the following chapter.
As tracking technology is a subject of rapid changes, the architecture of the toolkit is designed in a way that it allows any potential sensing hardware as an input without requiring changes in the developer's code. At this point it has not been done, but is planned in near future work.

6.1.3 Kinect Sensor

The Vicon Motion Capture system is one of the most accurate tracking systems available and it meets the demands for most proxemic applications. Hence it is perfectly suited to explore new ideas without being limited by technology. Still, for some application other tracking technologies might be a better fit. Maybe an application does not require certain proxemic dimensions and hence a cheaper system can be used that does not require

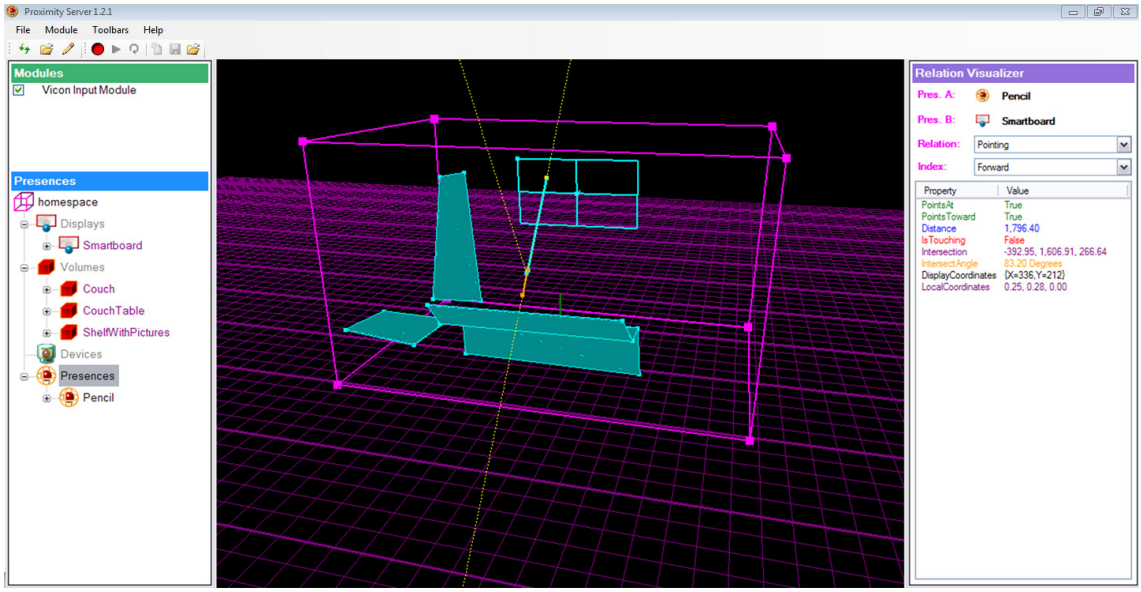

Figure 6.4: Visual editor of the Proximity Toolkit.

```
public DistanceExample(){
    //initialize a connection to the Proximity Server
    ProximityClientConnection client = new ProximityClientConnection();
    client.Start("127.0.0.1", 888, true);

    //get two objects from the observed space (a movable hat and a static dispaly)
    ProximitySpace space = client.GetSpace();
    TrackedPresence hat = space.GetPresence("WhiteHat");
    DisplayPlane display = space.GetDisplay("SmartBorad");

    //listen to an event that tells you whenever the Location Relation between the two objects changes
    RelationPair hat2Display = new RelationPair(hat, display, RelationMonitor.Location);
    hat2Display.OnLocationUpdatedAsynch += new LocationRelationHandler(hat2Display_OnLocationUpdatedAsynch);
}
void hat2Display_OnLocationUpdatedAsynch(LocationRelationEventArgs args){
    Console.Write("Distance between person and display: " + args.Distance);
}
```

Figure 6.5: This is all the code required to retrieve the distance between a tracked entity and a static display with the Proximity Toolkit API.

markers or is easier to transport and setup. I always keep this in mind and also explore other technologies.

For example I ported the Proxemic Pong game (as described in chapter 3.3) to work with a Microsoft Kinect sensor [36]. the Kinect sensor is a piece of hardware designed as a motion controller for Microsoft's Xbox console that lets people control games by their body movements (see figure 6.6 right). It contains a multi-array microphone, a RGB camera and a depth sensing camera. Both cameras provide a 30Hz video image with a resolution of 640 by 480 pixels covering an angular field of view of 57° horizontally and 43° vertically. While the RGB camera provides color information for each pixel, the depth sensor provides distance information to the closest object for each pixel point as an 11-bit depth value - which are 2048 levels of sensitivity. It consists of an infrared laser, a monochrome CMOS sensor and a processing unit that computes the depth information through stereo triangulation of the IR structured light image from the CMOS sensor and hard coded positions based on the distance between the laser and the sensor. The technique is called Light Coding and is patented by PrimeSense [41]. This information is then used to generate a 3D skeleton model of up to six people with 20 joints each.

I built an adapter cable (see figure 6.6 left) that connects the Kinect to a computer and use the NITE Middleware drivers provided by PrimeSense [41]. The OpenNI Framework [40]

Figure 6.6: **left)** Custom USB-adapter to connect a Kinect sensor with a computer, **right)** Microsoft's Kinect sensor [36]. Copyright 2010 Microsoft

provides access to the skeleton model through an API with a .Net C# wrapper. This way I can get accurate location information of people in a room. Public PrimeSense documents claim a depth resolution of 1cm at a 2m distance to the camera. Every skeleton of a person has an assigned ID that can even be recovered after leaving the camera‘s visible area. With this information I was able to implement all the same features of the Proxemic Pong game as with the Vicon system.

In comparison to the Vicon system, the Kinect sensor has the big advantage, that it does not require markers, is a lot cheaper and can be moved around easily. However it can not replace the vicon system completely at this point. First it has a lower accuracy and speed, which might be required in some applications, but more importantly it is not possible to track particular devices or objects, nor does it provide their orientation or identity. That‘s why the Kinect could not be used to implement any of my other proxemic applications.

6.2 Proximity Modules and Widgets

When developing applications that consider spatial relations between devices, there are some common tasks which go along with the proposed concepts (see chapter 5), but are not covered by the Proximity Toolkit. This is why I built three reusable modules providing a small programming library that can be used on top of the Proximity Toolkit: The *Region Detector* module notifies the programmer, when an object enters or leaves a defined area. The *Follow Me* module calculates a screen position that is oriented towards a physical object in space. The *Networking Module* provides easy distribution of proxemic values and exchange of data between devices over a TCP/IP network.
I will now explain the features these modules provide and how they can be used and talk about implementation details.

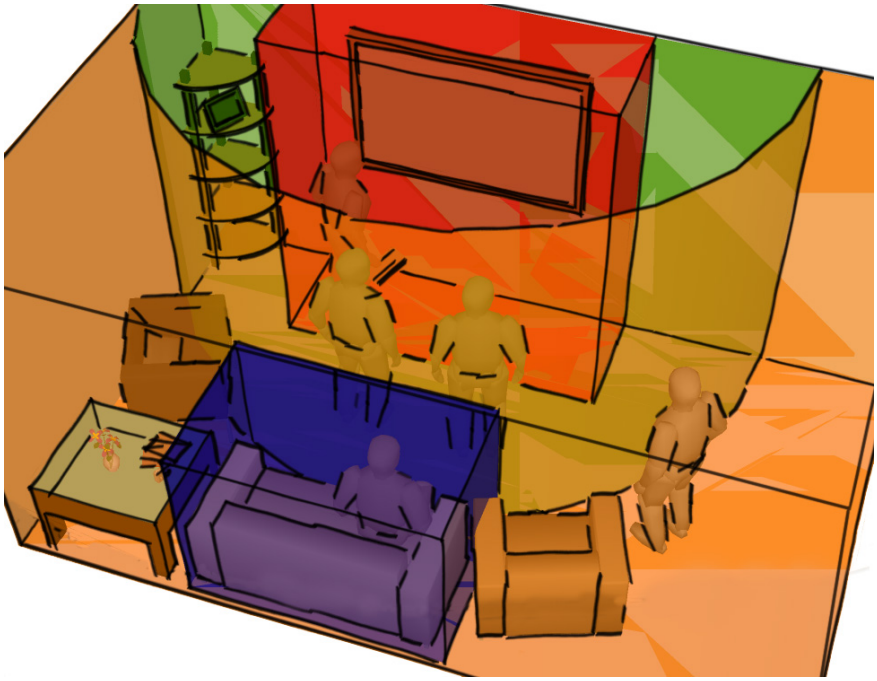

Figure 6.7: The four regions as defined in the code example 11.1: **red)** touch region, **green)** intermediate region, **blue)** sitting region, **orange)** far away region.

6.2.1 Region Detector

The Region Detector module allows a programmer to define and combine a variety of shapes in the tracked area, as regions. These shapes can be a boxes defined through width, depth and height values or cylinders defined by radius and height. The values for radius, width and depth can also be intervals with a minimum and maximum value, so that a region has a ring- or a 'box with a whole'-shape. As the code example in figure 11.1 shows, region objects are created through the described values and a label, and are then added to the `RegionDetector` object. Thereby one can define a priority for an added region. This is useful when region shapes overlap - only the one with the highest priority will be considered. Using priorities allows the creation of hardly any shape by intersecting shapes through priority values. Regions can be placed in the space by setting a `Center` point in coordinates of the Proximity Toolkit's tracked area. Figure 6.7 illustrates the region shapes as they are defined in the code example 11.1 (see appendix).
To actually use the defined regions, one has to add at least one tracked entity to the `RegionDetector` object. This is done by passing the name of a presence as it appears in the Proximity Toolkit. Once all the regions are defined and presences are added, the programmer registers to the `OnRegionChange` event. This event is fired, whenever an observed presence changes in between regions. Passed arguments contain the name of the region and the `PresenceBase` object from the Proximity Toolkit.
One important property of the Region Detector module is a hysteresis threshold between regions. This means, whenever an entity enters a region, the region shape is extended by the `DynamicRegionBelt` value (which can be assigned to every region). This is useful, because it prevents quick region changes when a tracked presence (e.g. a person) is located at the border of two regions. Especially when tracking accuracy is low, this technique assures very reliable detection of region changes, which for example helps in the design of applications that provide different interfaces or visualizations depending on regions.
Another feature provided by the module, are the `OnBecomesInvisible` and `OnBecomesVisible` events for each tracked presence. These events fire whenever the tracking system can or cannot see an entity, meaning that the entity left or entered the room or has been covered or uncovered (e.g. a cellphone has been put into a pocket). While the Proximity Toolkit gives access to a visibility value, it does not provide notification about their changes. This is a common task, because proxemic applications often react to binary presence information.

6.2.2 Follow Me

The idea behind the Follow Me module is related to Gellersen's Relate Gateways [18] and Rekimoto's M-Pad in Proximal Interactions [47]. They both show a graphical representation of a physical entity at the border of a digital display in a way that the representation is oriented towards the entity in space. These projects had either no or only very coarse proxemic information in their systems. To my knowledge the Follow Me module is the first implementation that actually provides this information in a continuous manner for a variety of horizontal and vertical, mobile or fixed displays.

The module offers two classes to access this information:
The `FollowCanvas` extends the WPF `Canvas` container and autonomously moves around in the application window. Figure 6.8 shows a code example of how to use the module. The constructor requires the window container and the names of the physical display where the `FollowCanvas` is shown and the presence which it should follow. It

```
followCanvas = new FollowCanvas(displayCanvas, "SmartBoard", "WhiteHat");
followCanvas.Width = 100;
followCanvas.Height = 100;
followCanvas.Background = Brushes.Black;

//additional settings
followCanvas.PositioningBehaviour = FollowCanvas.ScreenAlignmentStrategy.Inside;
followCanvas.AnimatedScaling = true;
followCanvas.ScalingAnimationSpeed = .3;
followCanvas.AnimatedMovement = true;
followCanvas.MovementAnimationSpeed = .3;
followCanvas.AllowProjection = true;
followCanvas.PositionCalculationStrategie = DeviceFollowPoint.CalculationStrategie.AngleFromCenter;
followCanvas.MovementThreshold = 0;

followCanvas.OnAngleChange += new AngleChangeHandler(followCanvas_OnAngleChange);
followCanvas.OnPositionChange += new PositionChangeHandler(followCanvas_OnPositionChange);
```

Figure 6.8: This code example illustrates the basic use of the Follow Me module.

can be configured to either position the whole canvas inside the window bounds, or just place its center at the calculated point, by setting the `PositioningBehaviour`. The movement of the canvas can be animated to provide an attractive visualization. The only thing a programmer has to do, is to place his own graphical representation into the canvas.
The second option is the `FollowPoint` class which does not offer a graphical representation, but otherwise has the very same functionality. Following the code example 6.8, it is possible to set a `MovementThreshold` which lets the following point only react when the entity‘s position change exceeds the defined value. This ’stickyness’ of movement is useful e.g. when a person should be able to interact with a following visualization and small movements would interfere. Another feature is, to allow movements not only at the border of the display, but project the position into the display when an entity is directly in front of it. This is don by setting the `AllowProjection` property. It is possible to choose a `PositionCalculationStrategy`, which sets the algorithm that calculates the point. Both of the two algorithms project the center point of the the followed entity into the plane of the display, using the normal vector to it. The `SimpleProjection` approach calculates the point by the shortest distance to the display border, while the `AngleFromCenter` approach lets one define an origin point on the display and then calculates the intersection point between the line from the origin point to the the projected point of the entity with the display border. The `OnAngleChange` event handler can be used with the second approach to receive the angle from the origin point to the entity. Both strategies offer the calculated position of the entity representation as pixel coordinates of the application window on the display through the `OnPositionChange` event handler.

6.2.3 Networking Module

The main goal of this module is to distribute proxemic values to devices connected over the network, so that they can react accordingly. Atop it lets devices transfer data in between them.
The implementation is based on the .Networking GT Toolkit [10] developed by Brian de Alwis and Mike Boyle. I used version 1.2.8 in a TCP/IP network. The main component of the toolkit is a distributed shared memory system called the shared dictionary. Every

device in the network can access this dictionary as if it was a hash table data structure with tree-paths as keys. The first instance to connect to a shared dictionary, acts as a server that stores all the data. It is not only possible to add basic types like strings or doubles, but also binary data can be added into the dictionary by either a client or a server. The toolkit offers programmers the possibility to subscribe to a path in the dictionary and notifies about any changes of data in the specified path.
The Networking module consists of two parts: The `ServerConnection` runs in the background on the computer, that receives the tracking information. It uses the Proximity Toolkit and the prior presented modules to collect and then share the required proxemic values with local or remote applications. It first creates a shared dictionary and then waits for other applications to connect. Every application, that wants to receive proximity information, uses the `DeviceConnection` and has to provide the name of the device it is running on (as it appears in the Proximity Toolkit) and what kind of device this is. Once connected, the server checks if there is tracking information available and writes this information into the shared dictionary using the structure shown in figure 6.9. Depending on the type of device, it computes and provides information relative to others. For example, whenever Nic's tablet (a 'mobile surface') connects, it creates `FollowPoint` instances for every other connected device in the room and writes the angle values into the dictionary. That way Nic's tablet knows which other devices are available in the room and at what angle they are located relative to his tablet. Other information includes distance, visibility and the name of the current proximity region. Every `DeviceConnection` has easy access to those changes through event handlers. Example 6.10 shows the use of the `OnDistanceChange` event handler. A List of available event handlers can be found in the documentation.
Another feature of the Networking module is the possibility to easily communicate with other devices in the room. Example 6.10 shows how an application checks if other devices are in uncomfortably close proximity to it and sends a warning to them if they cross a threshold of half a meter. The module provides the possibility to send data like images, videos, audio, double and strings or application specific commands to other devices in the room.
Note that all the modules are provided in one library and implement a Singleton design pattern approach for the Proximity Server connection, meaning no matter how many modules are used simultaneously, there will only be one connection.

6.3 Application Specifics

After I explained the general setup of the tracking system and talked about underlying work on libraries that make certain proxemic dimensions easier accessible for programmers, I will now show how these are applied to the development of the proxemic applications. Also, I will describe challenging parts of their implementation and talk about the technologies used.

One more thing, that I want to introduce upfront is the vertical interactive display, that is use in all of the following applications. It is a 52 inch TV equipped with a touch enabling overlay from SMART Technologies [54] - a glass layer with a bezel around it that contains four IR cameras - one in each corner. LED's emit IR light from along the edges to illuminate arbitrary objects that come close to the surface. The cameras can recognize them and an integrated chip triangulates the touch positions. This vision based approach is called DViT (Digital Vision Touch) and can distinguish up to two simultaneous touches. From now on I will refer to this display as the Smartboard.

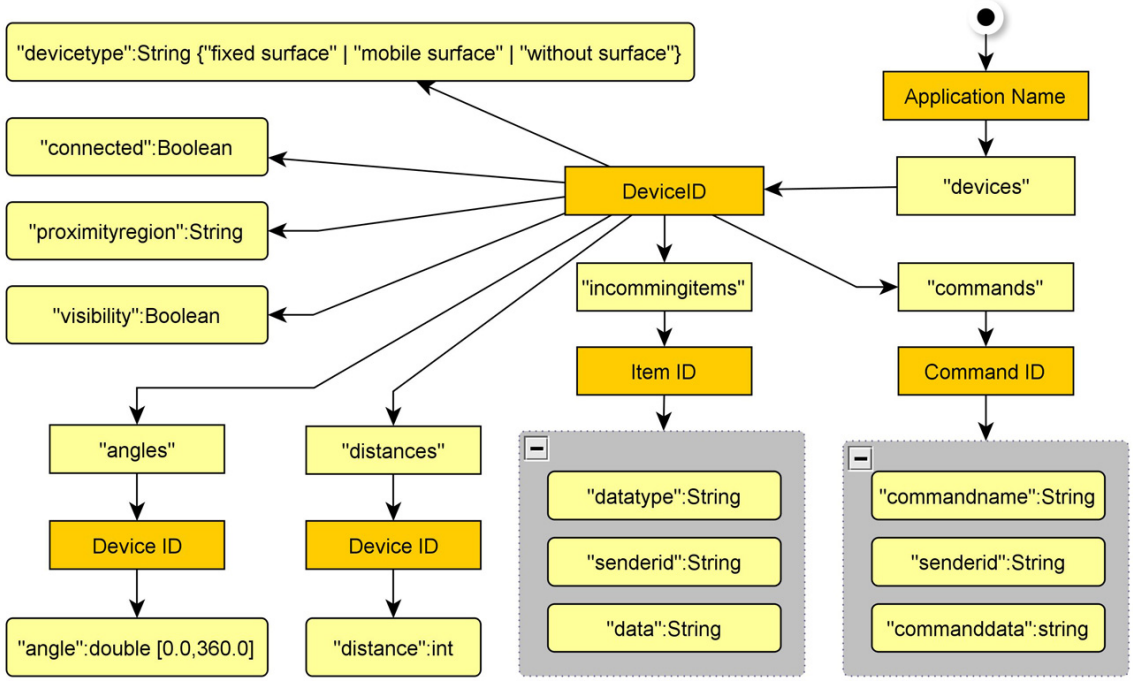

Figure 6.9: This graph shows the Network module's structure of the shared dictionary. Leafs that contain actual data are represented in boxes with round edges. Sharp edged boxes represent nodes and dark yellow nodes represent a arbitrary number of concrete nodes. The gray box represents a map data type which is provided by the .Networking GT toolkit.

```
private void sendreceive(string serverIP)
{
    myConnection = new DeviceConnection(serverIP, "tabletnic", DeviceVO.DeviceType.mobilesurface);
    myConnection.OnIncommingData += new NetworkDataUpdateHandlerClient(myConnection_OnIncommingData);
    myConnection.OnDistanceChange += new NetworkUpdateHandler(myConnection_OnDistanceChange);
}
void myConnection_OnIncommingData(DataItemVO incommingItem)
{
    Console.Write("I received data from " + incommingItem.SenderID + ": " + incommingItem.Data);
}
void myConnection_OnDistanceChange(DeviceVO deviceInfo)
{
    if (deviceInfo.Distance < 500) {
        myConnection.TransferDataTo(
            new DataItemVO(87621, DataBase.DataType.String, deviceInfo.ID, "Too close!"));
    }
}
```

Figure 6.10: This code illustrates how a device can use the DeviceConnection class to receive from and send data to other devices and get proximity information like the distance to others.

6.3.1 Proxemic Pong

I already described the idea behind this game in chapter 3.3 and now want to describe the main parts of its implementation. This includes the physics engine, a particle filter, the flexible MVC architecture and the stone jagged paddles.

The complete behaviour of the game play is managed by a physics engine. The engine provides a mathematical model of the game world and computes changes of this model over time while simulating the rules of physical behaviour as they exist in our real world. Using such an engine in the pong game, has several advantages: ball movements appear very natural, as parabolic motion and acceleration are based on gravity. This way, common flaws in pong implementations, like slow linear movements from left to right can be avoided. Also it calculates correct exit angles for bounces from any shape. Atop it is easy to change the gameplay in terms of difficulty or experience, just by adapting parameters like gravity, masses, friction or impulses. However there are some disadvantages come along with physics engines. First it becomes computational expensive, in particular if there are many objects to calculate. Also it can be hard to implement behaviour that does not coincide with the laws of physics (e.g. paddles moving in space not being influenced by bounces or friction). And ofcourse physics API's can be very complex and may require a lot of learning. Hence, before deciding to use physics engine one should carefully evaluate the amount of time it saves versus the amount of time he has to put into it.
I use the open source Farseer 2D Physics engine for C# in version 2.1.3.0 [69], which has no visual debugging component for WPF. So aligning game visuals to the models in the engine can sometimes be tricky, as the coordinate systems do not match. However, this changed with version 3, which was released after the Proxemic Pong was finished and seems to be a promising engine to use with C# and WPF.
In this game, all the visual objects have representations in the physics engine. The walls are static boxes and the paddles are represented by polygons. This is one of the major advantages of using the engine: A player can arbitrarily modify the paddle polygon during the game and the ball (the only dynamic object in the engine) still bounces accordingly. Collisions with the paddle add an additional impulse to the ball's movements. To increase the difficulty level, this impulse is increased over time.

The pong ball simulates a burning metal orb. The fire effects are created with a particle system [28] which is available for the Silverlight platform, but could easily be ported to the desktop. Such a system continuously generates visual objects at a defined location - in this case at the position of the ball. One can define their appearance by setting size, color values, blurriness and other parameters and affect behaviours through parameters like their livespan, movement speed or size and color variance. As figure 6.11 shows, the particles create a blazing trail of fire while the orb moves in the game area.

The architecture of the game is designed using a Model-View-Controller(MVC) pattern. This strictly separates the visual game objects (View), from their abstract data representation (Model) and the game play logic together with handling input from the user (Controller) (see figure 6.13. This has the general advantage, that every of these components can be easily substituted without any changes necessary in one of the other parts. That way I could develop the game logic first while using only simple shapes as visuals and later create a more appealing version of them. Another advantage is the abstraction from the input provider. While developing the game I used keyboard commands to control the paddle and simulate game events like an entering person or a per-

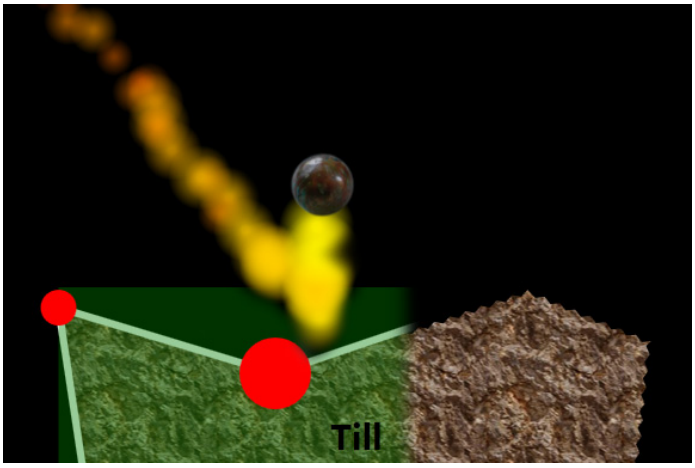

Figure 6.11: The pong paddle - **left half)** editing mode with control points, **right half)** jagged stone style paddle.

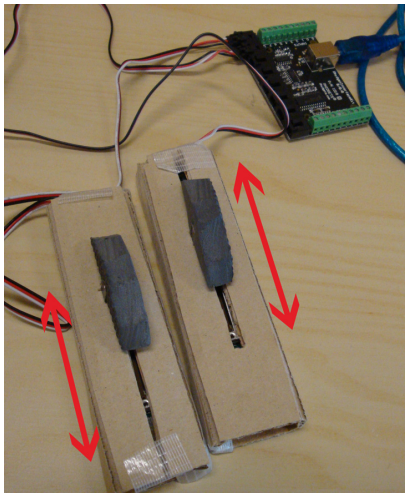

Figure 6.12: A Phidget slider that can be used to control a pong paddle [33].

son coming close to the screen. Later I substituted this `KeyboardController` with the `ViconProximityController` as shown in figure 6.13-top.

The `ViconProximityController` observes the room and lets people control the complete game play through presence, positions and motions of their body. Its implementation is straightforward: it uses the Proximity Toolkit to detect side to side movements and the Region Detector module to define region shapes around the couch, the play area and the display.

As figure 6.13-top shows, I created two more input controllers that simply plug into the architecture by implementing the `InteractionController` interface. The `SimpleKinectController` triggers game events and enables control through the Kinect sensor, as described in section 6.1.3. Another controller that illustrates the flexible architecture is the `PhidgetSliderController`. It uses two linear potentiometers with 60 mm of travel, connected to a Phidget micro controller [33] (see figure 6.12). Each player can control his little slider unit with a handle, that looks like a paddle and the position of the slider determines the paddle position.

One aspect of the game that is not determined by proximity is the modification of the paddle shape. Players can use direct touch on the Smartboard‘s display [54] to move control points around and thereby change the visual appearance and the bounce bahaviour of their paddle. Figure 6.11-left shows the editing mode while the control point in the center is modified. Visually, the paddle appears as a stone with a jagged border. In order to generate a jagged border on an editable polygon, a custom algorithm was required: as the position of a point in the polygon changes, the new length of the adjacent edges is calculated and split into a number of sub points. These points are then alternately moved up and down in a randomized angle and distance. This new complex jagged shape is filled with a stone texture and represents the paddle (see figure 6.11 right).

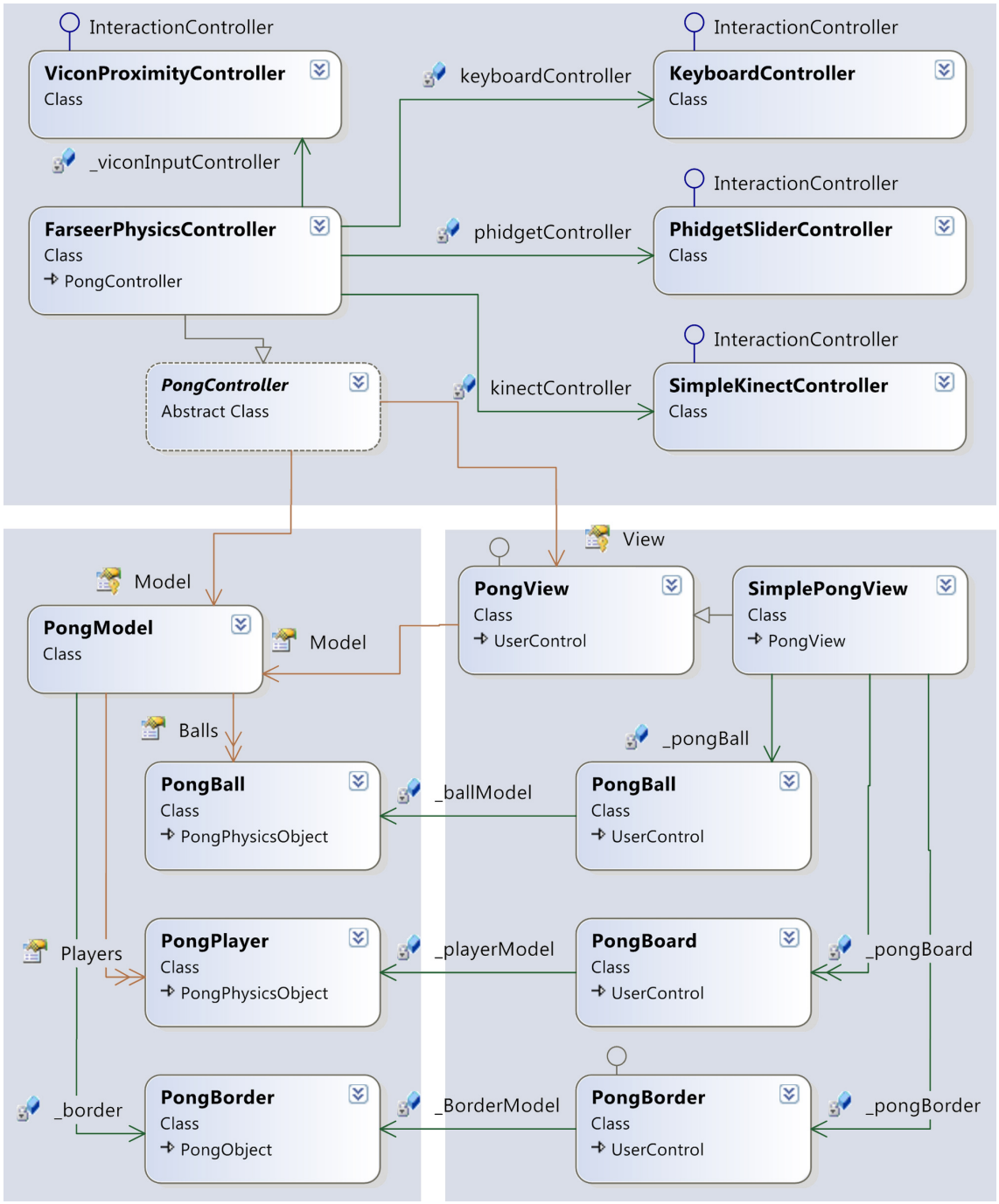

Figure 6.13: The main classes of Proxemic Pong illustrate the MVC architecture - **top)** Controller, **bottom-left)** Model, **bottom-right)** View.

6.3.2 Proxemic Photo Canvas

The goal of the photo canvas application is, to enable seamless exchange of photos facilitated through the proximity between a digital camera, a digital picture frame and a large interactive display. Hence the first step is to set up these hardware components and interconnect them.

The digital picture frame is a seven inch touch enabled monitor from Mimo [37]. It is connected via USB to the main computer, that also runs the Smartboard and the Vicon tracking system. This way, the application could just create an additional window and show it on this specific monitor. Input from the monitor‘s resistive touch layer could be handled as mouse inputs. To receive reliable events from a touch of the camera with the picture frame and distinguish from finger touches or accidental touches, the position of the digital camera in relation to the frame‘s position in the room is matched with the mouse events on the monitor. Hence, the picture frame is modeled as a static object in the Proximity Toolkit. Figure 6.14 shows the picture frame on the shelf and the digital camera after it touched the frame and initialized an image transfer.

The digital camera shown in figure 6.14 is an ordinary point and shoot camera. A connection to the other devices was established through a Wi-Fi enabled memory card from Eye-Fi [16] (Eye-Fi Connect X2 card as shown in figure 6.15). The camera connects to the same router as the main machine and then automatically synchronizes new photos into a specified folder. However at this point, there was no explicit access to the camera‘s memory and pictures could only be transfered in one direction - from the camera to the computer. But the main picture canvas application had access to all the data on the camera through watching the synchronized folder and thus could instantly react when a new picture was taken.The camera itself is tracked with the Vicon system through an attached marker bar as shown in figure 6.14. This provides access to the exact position and orientation of the camera in the room using the Proximity Toolkit.

Figure 6.14: The digital camera directly touches the picture frame surface to transfer an image.

Figure 6.15: The Eye-Fi card is used to wirelessly connect the digital camera with other devices.

One main idea of the application is to reveal the camera's images on the border of the large display while approaching it - always moving the visualization so that it is oriented towards the physical position of the camera. The Follow Me module together with the distance between those two devices comfortably provides all the proxemic information to accomplish this goal. To emphasize the chronology of images on the camera, it first displays a stack and then a spiral fan of images, where the last taken image is the largest and prior ones appear below, becoming smaller and smaller (see figure 6.16 shows the fist sketches). The tricky part about this implementation was, to provide a fluently animating spiral visualization when the camera moves along the corners of the display. After several approaches with a circle as a path for image alignment, I came to the following solution: The Follow Me module provides a point on the display that is always oriented towards the physical camera device. I create a square box with this point as a center, which continuously changes its size depending on the distance. The intersection of this box and the display area defines the content square, inside which the camera icon and the image fan should be placed (see figure 6.17 a,b - outer gray box with black a dot in the center). I start by estimating the position of the largest and the smallest image, then draw a half circular arch between their centers and distribute the remaining images along this path. The camera icon is then placed in the center of the resulting spiral fan of images. This is repeatedly done for every position change or change of distance of the camera and all visuals are smoothly animated in between. Figure 6.17 illustrates the algorithm. After crossing a defined distance threshold, the visualization slowly changes into a stack by scaling down and tilting the arch path, which is used for aligning the images (see figure 6.17 c). The result with real pictures in the final application can be seen in figure 4.4.

This design even enables browsing through images when a person tilts the camera. E.g. when the camera is tilted to the right, a new image is loaded, scaled to a small size and placed so that it is hidden below the display border. Now all images animate their position and size to fit into their successors spot, while the last image disappears below the display boarder. This allows fluent navigation through all the pictures on the camera.

As described in chapter 4.1, the application offers different interaction modalities to transfer an image from the camera to a surface. This ranges from direct touch to ambient visualization - where interaction possibilities are defined by a person's presence in a proximity region around the display. These regions are again implemented using the Region Detector module. When a person can actually reach the display, he can use direct

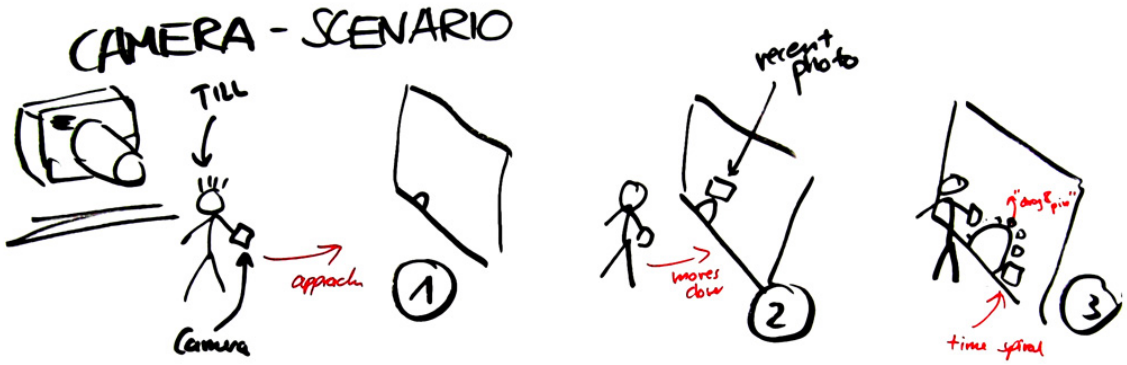

Figure 6.16: The first sketches of the interface bahaviour, when a person approaches the screen with his camera (drawn by Nicolai Marquardt).

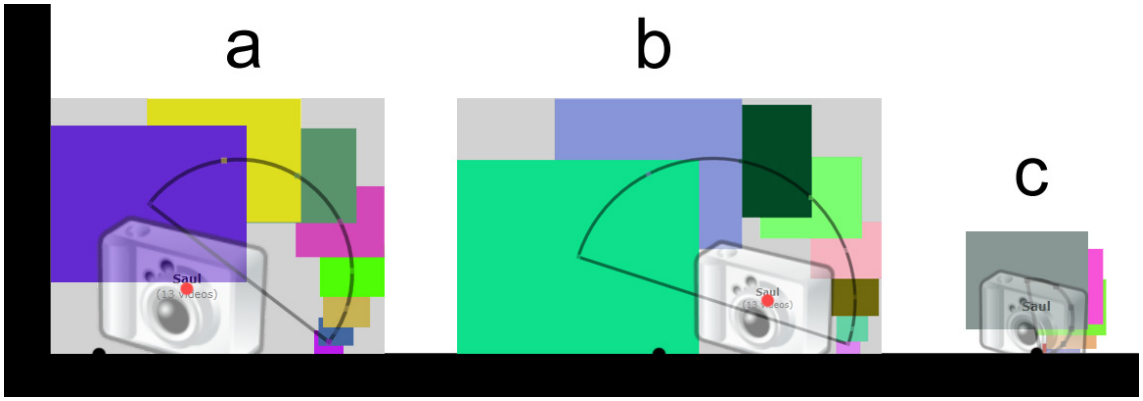

Figure 6.17: Illustrates the alignment algorithms of images in the camera visualization - **a)** picture fan when the camera is held at the corner of the display, **b)** picture fan when the camera is held at the lower edge of the display, **c)** picture stack when the camera is further away from the display.

touch to drag images out of the fan or use his camera to touch the screen. To distinguish between finger and camera touches, the digital camera has a collision volume around it. Whenever a touch is detected by the Smartboard while the collision volume intersects with the display, the application checks if the touch is inside this intersection area. If so, it assumes that the touch comes from the camera and transfers the image, otherwise it is a finger touch. Throw gestures are possible, whenever the camera is held outside the touch region, but inside a box in front of the Smartboard display. This is, because only then it made sense to display the camera icon as a projection inside the screen that creates awareness of which picture will be transferred. This pointing and throwing action is also possible when the person is inside the sitting region around the couch and chairs. It has higher priority than the ambient slideshow of images and thus interrupts the presentation and shows the projected image from the camera.

6.3.3 Proxemic Brainstorming

This application is distributed on different devices, so there are several sub-applications:

One part is a *server application* running in the background of the main machine. It uses the `ServerConnection` class from the Networking module as described in chapter 6.2.3. Hence it applies all the other modules and the Proximity Toolkit to provide proximity information to clients and enable easy communication between them. A simple GUI allows monitoring the connection status of other applications and manually adding observed entities. While the modules provide easy to use event handlers and methods, internally all information is exchanged using a shared dictionary from the .Networking GT Toolkit [10]. Its structure is illustrated in figure 6.9.

The *Smartboard Brainstorming application* runs on the same machine but could potentially run on any computer in the network as it uses the `DeviceConnection` class. It receives information about new devices in the room by registering to the `OnDeviceAdded` event. If a new tablet computer has been added, the application initializes monitoring the device's sticky note collection by calling the `startMonitoringDeviceData` method. It then receives all sticky notes from the tablet computer and changes made to them through `OnMonitoredDeviceData -Changed -Removed -Added` events. This way the ap-

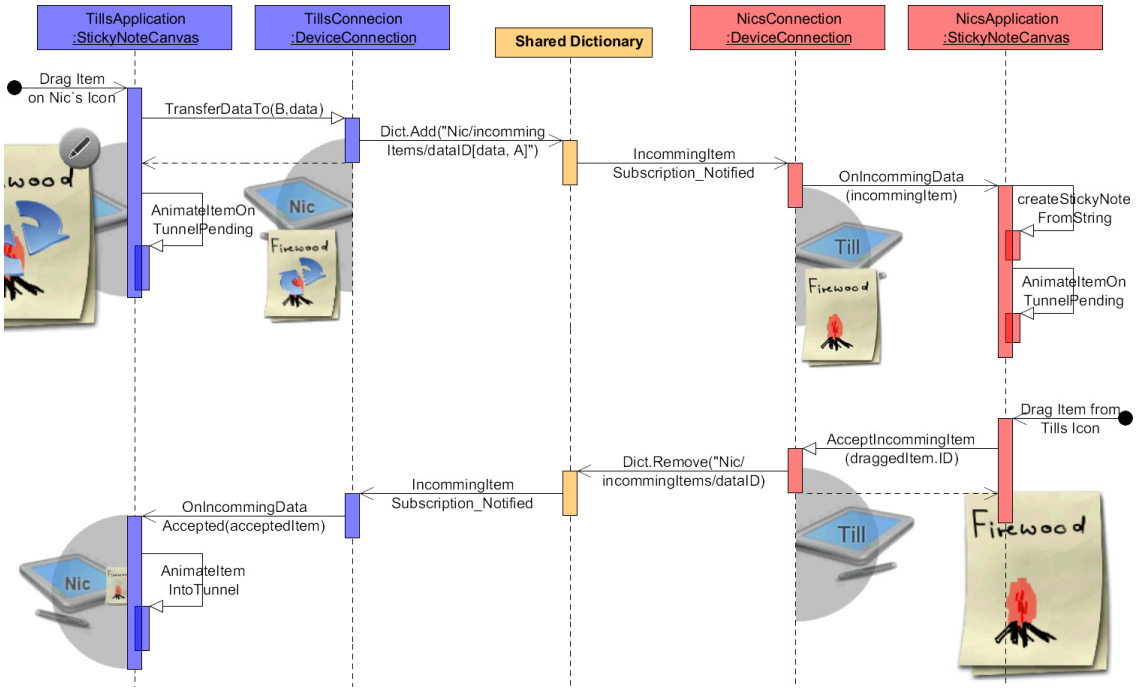

Figure 6.18: The sequence diagram shows the procedure of how a sticky note is transfered between Till‘s and Nic‘s tablet computer using the Networking module and a shared dictionary

plication can synchronize the sticky note containers, that represent the tablet computers. The proxemic values necessary to adapt the size, position and interactivity of the containers can again be accessed by events like `OnAngleChange`, `OnDistanceChange` and `OnAnyProximetyRegionChange`.
These containers continuously grow and change their shape to span the covered area of a person‘s body while approaching the display. The sticky notes inside grow accordingly and adapt their layout inside the container to best fit into the available space. This is possible with a custom designed, self managing component that extends the WPF canvas class. It also allows dynamic adding or removing notes and smoothly resizes and rearranges them.

The third application is the *Tablet Brainstorming application* that runs on each tablet computer. Again it uses the `DeviceConnection` class to receive information about proxemics, update data to the dictionary and exchange commands and data with other devices. This way the application connects to the server that is running in the background and then calls the `GetOtherDevicesInSpace` method. Depending on the type of device, it displays different icons on its border that represent these other devices. It continuously changes the icon‘s size and position depending on proxemic values that the application receives. When a sticky note is dragged over an icon it indicates the resulting action by showing a blue transfer- or presentation-symbol on top.

For example when Till enters the room with the application running on his tablet and Nic is already present. An icon will appear on Tills‘s screen that shows a tablet symbol with Nic‘s name on it and moves along the screen border so that it always orients towards Nic‘s tablet. As he comes closer to Nic, the icon grows. Till wants to transfer one of his

ideas to Nic and hence drags a sticky note over the icon. The sequence diagram in figure 6.18 shows the simplified progress of how the applications transfer the item using the Networking module and the shared dictionary. As Till releases the note, the application uses the `TransferDataTo` method offered by its instance of the `DeviceConnection` class. Also the note shrinks to half of its size and animates onto the icon and stays there until the end of the process. Hidden to the application, the `DeviceConnection` writes the data into the shared dictionary to the "imcommingItem" path for Nic's device. Because Nic's instance of the `DeviceConnection` reacts to changes in the path, his application receives an `OnIncomminData` event. he receives the sticky note as a serialized XAML string and recreates the instance on his device. This is possible as the `StickyNote` object is completely XAML compatible and can be represented as XML. As the diagram shows, the note appears on Till's icon and is in an intermediary state - showing a small representation on both screens - where either Till could retract the offered note or Nic could accept it. This is what he does in the example, by dragging it onto his canvas. His application tells the `DeviceConnection` instance by calling the `AcceptIncommingItem` method. Behind the scenes the module removes the entry from the dictionary, which causes Till's `DeviceConnection` instance to react and passes an `OnIncommingDataAccepted` event to the application. It lets the note disappear inside the icon - thereby completing the transfer.
The example shows how the Networking module hides the complexity of the shared dictionary and its sometimes cryptic handling. This becomes even more important when handling multiple and simultaneous transfers. The documentation for the Networking module shows all its functionality (see appendix), which is extensively used by the Proxemic Brainstorming application.

Another feature of the application is the possibility to save and load sets of sticky notes. This is again done by serializing `StickyNote` objects into XAML strings and writing them into an XML file. This format has the big advantage, that one can open the file in any text editor to modify it or even use scripts to do so. A loader function uses an XML parser to iterate through sticky notes in a file and recreates them on the Canvas.

As a connection is already established between the devices, the Networking module offers simple methods to exchange messages containing a command name, a text and the identity of sender and receiver. This allows application specific notifications between devices. For example the Brainstorming application sends a command from a tablet to the surface when a person presses the select button while performing a pointing action (compare figure 4.22 a).

7 Discussion

In this work I presented a number of user interfaces that react to proxemics and in many cases exploit people‘s expectations of how devices could behave within a Ubicomp ecology. This however requires a set of well defined ’rules of behaviour’. How to configure these rules is one of the largest unsolved issues in proxemic interaction. While it is easy to define a believable set for a particular scenario, there will arise problems applying it to others - e.g. an entity reacting in the wrong and unexpected way. While a growing number of researches are investigating in this area, we still don‘t understand the HCI of proxemics and there is a long way to go until a theory can fully describe people‘s spatial expectations of Ubicomp.

Another aspect that makes the design of proxemic systems challenging, are the differences in perceiving and interpreting proxemics. People‘s perception of proxemic relationships are influenced by gender, cultures, age, work hierarchies, and other factors [21]. These differences also affect the design of proxemic interactions. 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.

However there are guidelines and frameworks that can support the design of proxemic systems and are worth mentioning:
Proxemic systems have to regulate both implicit and explicit interaction. This is a major concern of Ju, Lee and Clemmer [30]. They highlight the ability of a system to repair mistakes and present three techniques that prevent, mitigate, or correct errors when implementing proactive behaviour:

1. **User Reflection:** The system should indicate what it infers from user actions, what actions it recognises and what effect these actions will have.

2. **System Demonstration:** The system should indicate what it is doing.

3. **Override:** A user should be able to override system decisions (e.g. interrupt or stop an action).

I believe that repairing mistakes is a central requirement for proxemic systems, especially as to the differences of people‘s perception of proxemics.

Belotti et al. proposed a framework for designing ubicomp systems [4]. Even if not exclusively focusing on proxemics, it might be a good start to consider their guidelines while designing for proxemics. Derived from Normans seven stages of execution [39], they identified five design challenges that focus on communication rather than cognition:

1. **Address:** This Raises the question of how to address or not address a device, while considering sensing failure and unintentional communication?

2. **Attention:** How does one know if a system can attend or is ready to attend to user actions? How can the system give appropriate feedback, so that the user is aware of that?

3. **Action:** How can a person perform a meaningful action, and how can he specify its target?

4. **Alignment:** How can the system show what it is doing and that it is doing the right thing?
5. **Accident:** How can a system avoid mistakes or inform the user about his mistakes?

Analyzing these points and trying to offer solutions for each of them - maybe by looking at our or other‘s projects - can help to overcome design mistakes in proxemic systems, but can of course not create or validate a set of proxemic behaviours.

While designing the presented systems I asked those questions and found them very helpful. I studied related systems and their solutions and often build upon them. As a merging of the most important techniques I presented a set of concepts that are particularly suited for fine grained proxemic interaction between devices in small space environments. With ’Awareness through continuous visualization of proxemic dimensions’ (see chapter 5.1)I address challenges of ’Addressing’, ’Attention’ and ’Alignment’. The concepts of ’Discrete proxemic zones for appropriate interaction modalities’ and ’Explicit interaction through physical devices’ (see chapters 5.2 and 5.3) mainly focuses on the ’Action’ challenge.

In this regard, all implementations - while fully functional - serve just as examples that illustrate, explore and evaluate design possibilities. I do not suggest that they are the ideal, nor that they achieve the perfect balance between adjudicating proxemic information and implicit or explicit interaction. Also my goal was not to implement feature complete solutions for an application domain, but rather focus on very particular tasks, try to design possible solutions and conceptualize them.

Proximity related fields of computing, like Attentive user interfaces [62] or Context-aware computing [50] provide further conceptual toolkits and guidelines. These are good sources for certain design aspects of proxemic systems and inspiration of further use cases.

Network connections:
All of my current implementations require prior setup of a network connection and installation of software. While this can be convenient in a home scenario, where people regularly use the same devices in the same network, it might be a burden for spontaneous interaction in public or unfamiliar environments. Still there are a couple of approaches that try to overcome this: Speakeasy [13] for example provides a framework for creating spontaneous peer-to-peer connections. It provides flexible discovery protocols, network usage and data transfers, so that it is easily possible to securely connect and communicate between different devices, even bridging the boundary between network technologies.

Tracking technology:
One obvious downside of my implementation is the tracking technology they rely on. As explained in chapter 6.1.1, they currently use a very expensive Vicon infrared tracking system [64] that requires markers to be attached to tracked objects. This makes it impossible to be deployed in peoples homes or offices. However I believe that tracking technology with similar fidelity will soon become cheaper and available for a broad audience. The first step has already been taken with Microsoft releasing the Kinect sensor [36] and making it available for consumers. As described in chapter 6.1.3, I was able to acquire all necessary information from this sensor to use it as a tracking source for the Proxemic Pong game. There is other technology like infrared sensors, RFID, depth sensors or vision and scene analysis and depending on the required proxemic information, the current technology may already be sufficient. While I am always looking for new and

more advanced tracking technologies, the main goal of this work is to explore the possible proxemic interaction concepts. I see the Vicon system as a rapid prototyping platform for new ideas. It provides detailed information about all proxemic dimensions, but I believe it can soon be replaced by consumer affordable products.

Privacy:
When a system uses tracking technology, it always raises the question of privacy. For example a person might not feel comfortable if a system knows, that he is in a particular location and fears that the information could get into wrong hands. This is of course a valid concern, which applies to many 'smart' systems - in particular when they are connected over a network. Applying proxemic interaction will always be a trade-off between the amount of information a person is willing to share with a system and the advantages it can provide. In this regard people will have different opinions and it is important to follow and try to protect their concerns.
However, proximity can also be a chance for privacy and security. Examples are Marquardt's *Distant-dependent disclosure* RFID tags [34] that reveal certain information only in closer physical distance, or projects that use proximity sensing for secure authentication [47, 2, 35]. If a system is well designed it can exploit physical means to peoples privacy and help them better understand and control how much information they want to reveal (e.g. only a physical touch lets others access information from your tablet computer, or only line of sight reveals the presence of your phone on other peoples mobile devices). These potentials of proximity can become even more important as we connect and digitalise our lives (e.g. through social networks and internet enabled smartphones) and one has to manage a vast amount of privacy settings.

8 Conclusion and Future Work

I conclude this thesis by summarizing the research contributions. First I shortly reiterate the problems identified in chapter one, then outline my contributions and explain how they address these problems. Finally I suggest areas of future work.

8.1 Research Problems

In chapter one I stated four research questions that relate to interaction between devices in small space proxemic environments:

1. **How does a person know which devices can intercommunicate in his immediate surrounding?**
   With the multitude of network technologies, communication protocols and types of data that devices can support or not support, often it is not obvious which devices can be interconnected and exchange information.

2. **What kind of information do these devices contain and which interactions do they support?**
   In order to find out which devices a person wants to interact with, it is essential to be aware of the information devices currently contain and in which ways this content can be accessed or shared.

3. **How can a person address a particular digital device in his nearby environment?**
   In many cases a piece of digital technology can be easily identified and addressed by physical means. This however in most cases does not solve the problem of identifying it in a digital network interface. So are there means of proximity that better support this task?

4. **How can a person share information between devices?**
   Once a device is addressed, what are the interaction possibilities? Do they build upon natural expectations and are they easy to use?

8.2 Contributions

I presented two systems that try to address the stated problems in concrete scenarios. They both employ a variety of interconnected digital devices of different form factors and react to fine grained proxemic relationships between them. The *Proxemic Photo Canvas* supports exchange of pictures between a digital camera, a large interactive display and a digital picture frame. The *Proxemic Brainstorming Application* lets people share their ideas on digital sticky notes between their tablet computers and a large interactive display.

These applications illustrate interaction concepts for devices in small space proxemic environments, that I presented in chapter 5. They are based on prior work on *Proxemic Interactions* [1]. I revisit these concepts and explain how they address each of the four stated problems:

- **Awareness through Continuous Visualization of Proxemic Dimensions** suggests to visualize proxemic dimensions between nearby devices in a continuous manner using distance, movements and orientation, combined with discrete proxemics as identity and location. *Location* tells if a device is in the same space and

should be used to decide whether to visualize an instance of it. The visualized instance is further defined by *identity*, as it represents the type of device and possibly its name. This addresses problem one, as only addressable devices are shown. Problem three is approached by taking *orientation* into account. The visualized instance orients on a screen so that one can identify the relation to the physical instance in space. This it further facilitated by matching physical device-*movements* to movements of the visualization. *Distance* is the main indicator for the amount of content shown. This answers problem two, as more detailed information and interaction possibilities are revealed while approaching devices.

- **Discrete Proxemic Zones for Appropriate Interaction Modalities:** Proxemic zones are defined by the relative distance or the location in the environment. I suggest different modalities of interaction depending on the zone in which people and devices are located. Interaction modalities should consider the notion of providing more explicit and private interaction with detail control in a closer zone ranging to more implicit and public interaction from afar. It relates to problem two and four: as proximity suggests offered interactions, information exchange is possible when approaching or attending towards a particular device.

- **Explicit Interaction through Physical Devices:** For explicit interactions between devices this concept suggests do use interaction techniques that incorporate proxemic dimensions of these devices and appropriate their contextual meaning. The general idea is to let proxemic actions with a device affect its associated information. I illustrate how information exchange is possible by using a device as a physical tool (this addresses problem four).

A takeaway of this is that these three concepts are most powerful when applied in combination with each other. This way techniques can create awareness of device identity, its content and appropriate interaction possibilities; then fluently move to interaction facilitated through proxemic dimensions between devices and their owners. I want to encourage others to apply these concepts in their own interactive environments with digital devices.

8.3 Future Work

In this work I applied concepts of proxemic interaction to specific application scenarios. While they include a number of different devices, they only scratched the surface of the possible design space for proxemic applications. Mainly two aspect could be further explored. First, how do proxemics differ in a professional work environment rather than a home environment? And second, how do the proposed concepts scale for digital appliances (e.g. coffee machines, thermostats or lights) and devices of other form factors (e.g. interactive tabletops)?

The presented applications support only specific tasks, that are facilitated by proxemics. But they are not feature complete systems that could be employed in peoples homes or offices. It could be valuable to integrate a number of proxemic techniques into existing applications and observe how people use them in real life. Due to the limitations of current tracking technology this is still hard for many of the proposed concepts. However, there are dimensions that can be sensed and used in applications. Location, discrete distances and identity might be easiest to employ.

Figure 8.1: Sony's Intelligent Presence Sensor [56]. **left)** Presence Sensor, **center)** Distance Alert, **right)** Position Control. Copyright 2010 Sony Corporation.

Figure 8.2: HP's Touch to Share technology [25]. Copyright 2011 Hewlett-Packard Development Company.

In this regard I want to mention two examples (other than gaming) where consumer electronics already made the first steps:

The latest televisions from Sony integrate an *Intelligent Presence Sensor* [56] which supports three features (see figure 8.1). A presence sensor turns the picture off if no viewer is detected. A distance alert recognizes viewers that are too close to the screen and shows a warning while turning off the picture. A position control detects people's seating positions and balances the volume between left and right speakers accordingly.

HP announced a feature for their upcoming tablet computers and smartphones which is called *Touch to Share* [15] and uses their *Touchstone technology* [25]. It enables one device to pick up the current webpage shown on another device. This is done by physically tapping the devices together. That way the address of the webpage is instantly transfered and displayed on the tapping device (see figure 8.2).

References

[1] T. Ballendat, N. Marquardt, and S. Greenberg. Proxemic interaction: Designing for a proximity and orientation-aware environment. In *Proceedings of the ACM Conference on Interactive Tabletops and Surfaces - ACM ITS'2010*, pages 121–130, Saarbruecken, Germany, November 7-10 2010. ACM Press. Best Paper Award.

[2] J. Bardram. The trouble with login: on usability and computer security in ubiquitous computing. *Personal and Ubiquitous Computing*, 9(6):357–367, 2005.

[3] M. Beigl. Point & click-interaction in smart environments. In *Handheld and Ubiquitous Computing*, pages 311–313. Springer, 1999.

[4] V. Bellotti, M. Back, W. K. Edwards, R. E. Grinter, A. Henderson, and C. Lopes. Making sense of sensing systems: five questions for designers and researchers. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, pages 415–422, Minneapolis, Minnesota, USA, 2002. ACM.

[5] H. Brignull and Y. Rogers. Enticing people to interact with large public displays in public spaces. In *Human-computer interaction: INTERACT'03; IFIP TC13 International Conference on Human-Computer Interaction, 1st-5th September 2003, Zurich, Switzerland*, page 17, 2003.

[6] S. Cho, C. Choi, Y. Sung, K. Lee, Y. Kim, and R. Murray-Smith. Dynamics of tilt-based browsing on mobile devices. In *CHI'07 extended abstracts on Human factors in computing systems*, pages 1947–1952. ACM, 2007.

[7] H. Clark. Pointing and placing. *Pointing: Where language, culture, and cognition meet*, pages 243–268, 2003.

[8] J. Cooperstock, S. Fels, W. Buxton, and K. Smith. Reactive environments. *Communications of the ACM*, 40(9):65–73, 1997.

[9] R. Dachselt and R. Buchholz. 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*, pages 3253–3258. ACM, 2009.

[10] B. de Alwis, C. Gutwin, and S. Greenberg. GT/SD: performance and simplicity in a groupware toolkit. In *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*, page 265–274, 2009.

[11] R. Diaz-Marino and S. Greenberg. The proximity toolkit and ViconFace. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems - CHI EA '10*, page 4793, Atlanta, Georgia, USA, 2010.

[12] P. Dourish. *Where the action is: the foundations of embodied interaction*. MIT Press, Sept. 2004.

[13] W. K. Edwards, M. W. Newman, J. Z. Sedivy, T. F. Smith, D. Balfanz, D. K. Smetters, H. C. Wong, and S. Izadi. Using speakeasy for ad hoc peer-to-peer collaboration. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 256–265, New Orleans, Louisiana, USA, 2002. ACM.

[14] S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, E. Pedersen, K. Pier, J. Tang, and B. Welch. Liveboard: a large interactive display supporting group meetings, presentations, and remote collaboration. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 599–607, Monterey, California, United States, 1992. ACM.

[15] Engadget. Hp's touch to share. http://www.engadget.com/2011/02/09/hps-touch-to-share-eyes-on-starring-the-touchpad-and-hp-pre-3/. last accessed Feb. 14, 2011.

[16] I. Eye-Fi. Eye-Fi wireless sd cards. http://www.eye.fi/. last accessed Feb. 07, 2011.

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

[18] H. Gellersen, C. Fischer, D. Guinard, R. Gostner, G. Kortuem, C. Kray, E. Rukzio, and S. Streng. Supporting device discovery and spontaneous interaction with spatial references. *Personal and Ubiquitous Computing*, 13(4):255–264, 2009.

[19] S. Greenberg, N. Marquardt, T. Ballendat, R. Diaz-Marino, and M. Wang. Proxemic interactions: The new ubicomp? *ACM Interactions*, 18(1):42–50, January-February 2011. Invited cover story.

[20] S. Greenberg and M. Rounding. The notification collage: posting information to public and personal displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 514–521, Seattle, Washington, United States, 2001. ACM.

[21] E. Hall. Distances in Man: The Hidden Dimension. *Garden City, New York: Double Day*, 1966.

[22] E. Hall. Proxemics. *Current anthropology*, 9(2/3):83, 1968.

[23] C. Harrison and A. K. Dey. Lean and zoom: proximity-aware user interface and content magnification. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 507–510, Florence, Italy, 2008. ACM.

[24] T. Hesselmann, S. Flöring, and M. Schmitt. Stacked Half-Pie menus: navigating nested menus on interactive tabletops. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 173–180, New York, NY, USA, 2009. ACM.

[25] Hewlett-Packard. Hp touchpad - better together. http://www.palm.com/us/products/pads/touchpad/index.html. last accessed Feb. 14, 2011.

[26] K. Hinckley. Synchronous gestures for multiple persons and computers. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 149–158, Vancouver, Canada, 2003. ACM.

[27] L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H. Gellersen. Smart-Its friends: A technique for users to easily establish connections between smart artefacts. In *Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 116–122, Atlanta, Georgia, USA, 2001. Springer-Verlag.

[28] R. Ingebretsen. Silverlight particle generator. http://blog.nerdplusart.com/archives/silverlight-particle-generator/. last accessed Feb. 06, 2011.

[29] S. Izadi, H. Brignull, T. Rodden, Y. Rogers, and M. Underwood. Dynamo: a public interactive surface supporting the cooperative sharing and exchange of media. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 159–168, Vancouver, Canada, 2003. ACM.

[30] W. Ju, B. A. Lee, and S. R. Klemmer. Range: exploring implicit interaction through electronic whiteboard design. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 17–26, San Diego, CA, USA, 2008. ACM.

[31] T. Kindberg, J. Barton, J. Morgan, G. Becker, D. Caswell, P. Debaty, G. Gopal, M. Frid, V. Krishnan, H. Morris, et al. People, places, things: Web presence for the real world. *Mobile Networks and Applications*, 7(5):365–376, 2002.

[32] C. Kray, M. Rohs, J. Hook, and S. Kratz. Group coordination and negotiation through spatial proximity regions around mobile devices on augmented tabletops. *Proc. of IEEE Tabletop 2008*, page 3–10, 2008.

[33] N. Marquardt and S. Greenberg. Distributed physical interfaces with shared phidgets. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 13–20. ACM, 2007.

[34] N. Marquardt, A. Taylor, N. Villar, and S. Greenberg. Rethinking RFID: awareness and control for interaction with RFID systems. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 2307–2316. Citeseer, 2010.

[35] R. Mayrhofer, H. Gellersen, and M. Hazas. Security by spatial reference: using relative positioning to authenticate devices for spontaneous interaction. In *Proceedings of the 9th international conference on Ubiquitous computing*, pages 199–216. Springer-Verlag, 2007.

[36] Microsoft. Kinect motion sensor. http://www.xbox.com/kinect. last accessed Feb. 03, 2011.

[37] Mimo. MIMO monitors. http://www.mimomonitors.com/. last accessed Feb. 07, 2011.

[38] M. Nacenta, C. Gutwin, D. Aliakseyeu, and S. Subramanian. There and back again: cross-display object movement in multi-display environments. *Human–Computer Interaction*, 24(1):170–229, 2009.

[39] D. Norman. *The design of everyday things*, volume 16. Basic Books New York, 2002.

[40] OpenNI. Openni framework. http://www.openni.org/. last accessed Feb. 03, 2011.

[41] PrimeSense. Natural interaction. http://www.primesense.com. last accessed Feb. 03, 2011.

[42] G. Ramos, K. Hinckley, A. Wilson, and R. Sarin. Synchronous Gestures in Multi-Display Environments. *Human–Computer Interaction*, 24(1):117–169, 2009.

[43] J. Rekimoto. Tilting operations for small screen interfaces. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 167–168. ACM, 1996.

[44] J. Rekimoto. 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*, pages 31–39, Banff, Alberta, Canada, 1997. ACM.

[45] J. Rekimoto. A multiple device approach for supporting whiteboard-based interactions. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 344–351, Los Angeles, California, United States, 1998. ACM Press/Addison-Wesley Publishing Co.

[46] J. Rekimoto, Y. Ayatsuka, and M. Kohno. SyncTap: An interaction technique for mobile networking. *Human-Computer Interaction with Mobile Devices and Services*, pages 104–115, 2003.

[47] J. Rekimoto, Y. Ayatsuka, M. Kohno, and H. Oba. Proximal interactions: A direct manipulation technique for wireless networking. In *Proceedings of INTERACT 2003*, 2003.

[48] M. Ringwald. Spontaneous interaction with everyday devices using a PDA. In *Proceedings Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings, Ubicomp*, 2002.

[49] D. Russell, J. Trimble, and A. Dieberger. The use patterns of large, interactive display surfaces: Case studies of media design and use for blueboard and MERboard. In *System Sciences, 2004. Proceedings of the 37th Annual Hawaii International Conference on*, page 10 pp., 2004.

[50] D. Salber, A. Dey, and G. Abowd. The context toolkit: Aiding the development of context-enabled applications. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 434–441. ACM, 1999.

[51] A. Schick, F. van de Camp, J. Ijsselmuiden, and R. Stiefelhagen. Extending touch: towards interaction with large-scale surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 117–124, Banff, Alberta, Canada, 2009. ACM.

[52] B. Schilit, N. Adams, and R. Want. Context-Aware computing applications. In *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society, 1994.

[53] G. Shoemaker, A. Tang, and K. S. Booth. Shadow reaching: a new perspective on interaction for large displays. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 53–56, Newport, Rhode Island, USA, 2007. ACM.

[54] SMART. Technologies, industry leader in interactive whiteboard technology, the SMART board. http://smarttech.com/. last accessed Feb. 06, 2011.

[55] R. Sommer. Studies in personal space. *Sociometry*, 22(3):247–260, 1959.

[56] Sony. Bravia tv - intelligent presence sensor. http://esupport.sony.com/docs/imanual/NA/EN/1a/ipsensor_uc_1a.html. last accessed Feb. 14, 2011.

[57] N. Streitz, T. Prante, C. Röcker, D. V. Alphen, C. Magerkurth, R. Stenzel, and D. Plewe. Ambient displays and mobile devices for the creation of social architectural spaces. *Public and Situated Displays: Social and interactional aspects of shared display technologies*, page 387–409, 2003.

[58] N. A. Streitz, J. Geißler, T. Holmer, S. Konomi, C. Müller-Tomfelde, W. Reischl, P. Rexroth, P. Seitz, and R. Steinmetz. i-LAND: an interactive landscape for creativity and innovation. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 120–127, Pittsburgh, Pennsylvania, United States, 1999. ACM.

[59] C. Swindells, K. M. Inkpen, J. C. Dill, and M. Tory. That one there! pointing to establish device identity. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, UIST '02, pages 151–160, New York, NY, USA, 2002. ACM.

[60] P. Tandler, T. Prante, C. M
"uller-Tomfelde, N. Streitz, and R. Steinmetz. 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*, pages 11–20. ACM, 2001.

[61] P. Välkkynen, M. Niemelä, and T. Tuomisto. Evaluating touching and pointing with a mobile terminal for physical browsing. In *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles*, NordiCHI '06, pages 28–37, New York, NY, USA, 2006. ACM.

[62] R. Vertegaal, J. Shell, D. Chen, and A. Mamuji. Designing for augmented attention: Towards a framework for attentive user interfaces. *Computers in Human Behavior*, 22(4):771–789, 2006.

[63] R. Vertegaal and J. S. Shell. Attentive user interfaces: the surveillance and sousveillance of gaze-aware objects. *Social Science Information*, 47(3):275 –298, 2008.

[64] Vicon. Motion capture systems from vicon. http://vicon.com/. last accessed Feb. 02, 2011.

[65] D. Vogel and R. Balakrishnan. Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. pages 137–146. ACM, 2004.

[66] F. Vogt, J. Wong, S. Fels, and D. Cavens. Tracking multiple laser pointers for large screen interaction. In *Extended Abstracts of ACM UIST*, page 95–96, 2003.

[67] R. Want, K. P. Fishkin, A. Gujar, and B. L. Harrison. Bridging physical and virtual worlds with electronic tags. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 370–377, Pittsburgh, Pennsylvania, United States, 1999. ACM.

[68] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1):91–102, 1992.

[69] J. Weber and I. Qvist. Farseer physics engine. http://farseerphysics.codeplex.com/. accessed Feb. 06, 2011.

[70] M. Weiser. The computer for the 21st century. *Scientific American*, 272(3):78–89, 1995.

[71] A. Wilson and R. Sarin. BlueTable: connecting wireless mobile devices on interactive surfaces using vision-based handshaking. In *Proceedings of Graphics interface 2007*, pages 119–125. ACM, 2007.

List of Figures

9 Publications and Press

Materials, ideas, and figures from this thesis have appeared previously in the following publications:

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, pages 121-130, November 7-10. Best Paper Award.

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

Ballendat, T., Marquardt, N. and Greenberg, S. (2010)
Proxemic Interaction: Designing for a Proximity and Orientation-Aware Environment. SurfNet - Technology of the Future - Today!, Vol 1, Issue 3 page one, September/October 2010. Newsletter

Ballendat, T., Marquardt, N. and Greenberg, S. (2010) **Proxemic Interaction: The Video**. Research report 2010-963-12, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, June. Duration 3:11

10 Acknowledgements

This work would not have been possible without the support and guidance of many people:

To my supervisor Nic, you have been involved in every part of this work and I learned a lot from you during my thesis time. Thank you for the valuable discussions and excellent guidance. They motivated many of the presented ideas and this thesis would not have been possible without your help. Thanks for being a good friend and mentor!

To Saul, You gave me the opportunity to work in a wonderful place, the Interactions Lab. You believed in me at all times and gave me valuable advice for every important decision. Thanks for sending me to conferences and having me in your ubicomp class. Those where amazing sources of inspiration! I really enjoyed working with you and admire how you approach things.

To the members of the Interactions Lab, thanks for all your support in the lab, the lunch table discussions, fun events, amazing potlucks, movie nights, sport buddies and much more. I had an amazing time with all of you!
Special thanks to those who helped me with this work by giving valuable feedback, having great discussions and helping with my videos and photos!

11 Appendix

```
//Example lets`s you define custom regions and observe people moving in and out of them
void createRegions() {
    peopleObserver = new RegionsDetector();

    //add two hats that will be observed as people
    peopleObserver.AddObservedPresence("WhiteHat");
    peopleObserver.AddObservedPresence("Blackhat");

    Vector3 displayCenter = new Vector3(-380, 0, -89);
    //define and add custom regions
    //Touchregion is a rectangular Box around the display
    RectangularRegion touchRegion = new RectangularRegion("touch", 500, 750, 3000);
    touchRegion.Center = displayCenter;
    touchRegion.DynamicRegionBelt = 150;
    peopleObserver.AddRegion(touchRegion, 2);

    //Sitting region is a Rectangular Box around the couch
    RectangularRegion sittingRegion = new RectangularRegion("sitting", 600, 1500, 1350);
    sittingRegion.Center = new Vector3(1500, 0, 0);//center of the couch
    sittingRegion.DynamicRegionBelt = 150;
    peopleObserver.AddRegion(sittingRegion, 2);

    //Intermediate Interaction Region is a Cylindrical Region around the display`s center
    CircularRegion intermediateRegion = new CircularRegion("intermediate", 1800);
    intermediateRegion.Center = displayCenter;
    intermediateRegion.DynamicRegionBelt = 150;
    peopleObserver.AddRegion(intermediateRegion, 1);

    //Far away Region is a circular belt around the intermediate interaction region
    CircularRegion farRegion = new CircularRegion("far away", new RegionValue(2000, 4000));
    farRegion.Center = displayCenter;
    farRegion.DynamicRegionBelt = 150;
    peopleObserver.AddRegion(farRegion, 1);

    //Be notified whenever a person changes in between Regions
    peopleObserver.OnRegionChange += new RegionChangeHandler(peopleObserver_OnRegionChange);
}
void peopleObserver_OnRegionChange(ExtendedPresence presence){
    Console.Write(presence.Presence.Name + "entered region" + presence.Region.Name);
}
```

Figure 11.1: Shows how the Region Detector module can be used. This example creates four regions of different shapes and fires an event whenever a person enters or exits a region

12 DVD Content

This is a list of folders and files on the DVD that is attached to every copy of my thesis:

- `diploma_thesis_tillballendat_final.pdf`
  The complete thesis document
- `Literature`
  All referenced literature documents
- `Publications`
  This folder contains documents that we published during my thesis
  - `2010-ProxemicInteractions-Paper.pdf`
    ITS 2010 paper on "Proxemic Interaction"
  - `2011-ProxemicInteraction.Interactions.pdf`
    Interaction magazine Jan/Feb 2011 cover article on "Proxemic Interaction"
- `Software`
  This folder contains all the software I wrote for this thesis
  - `ProximityModulesDocumentation.doc`
    A code library documentation of the presented proxemic modules
  - `ProxemicInteraction`
    Code of the proxemic applications and the proxemic modules
  - `ProxemicPong`
    Code of the Proxemic Pong game
- `Talks`
  Presentations I gave related to this thesis
  - `DA_exit_talk_till_ilab.pdf/.pptx/.wmv`
    Diplom thesis exit talk given at the University of Calgary on 20/12/2010
  - `DA_exit_talk_till_muc_final.pdf/.pptx`
    Diplom thesis exit talk given at the LMU Munich on 01/02/2011
- `Thesis source`
  This folder contains the latex sources and images of the thesis document
- `Videos`
  This folder contains videos that show every application in action
  - `PhotoCanvas01.wmv` `PhotoCanvas01.wmv`
    Videos documenting the Proxemic Photo Canvas application
  - `ProxemicBrainstorming.wmv`
    A video documenting the Proxemic Brainstorming application
  - `ProxemicMediaPlayer.wmv`
    A video documenting the Proxemic Media Player application
  - `ProxemicPongVicon.wmv`
    A videos documenting the Proxemic Pong game implemented with the Vicon tracking system
  - `ProxemicPongKinect.wmv`
    A video documenting the Proxemic Pong game implemented with the Kinect sensor