

Enriching Touch

Raf Ramakers

Advisor: Prof. dr. Kris Luyten

Supervisor: Davy Vanacken

Thesis proposed to achieve the degree of master in computer sience:
Human-Computer Interaction

Academic year 2011–2012

Acknowledgments

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance during the creation of this thesis.

First of all, I would like to express my gratitude to my supervisor Prof dr. Kris Luyten. He gave me lots of inspiration and suggestions to make this topic even more interesting.

Special thanks go to Davy Vanacken. He introduced me to the interesting problems discussed in this work. During the past year, He guided me through this thesis with lots of ideas, advice, and feedback. He also gave me a lot of suggestions to improve the writing style of this work. Thanks also to Jonny Daenen for the feedback that he gave me on the first part of this thesis.

I also want to thank dr. Cedric Vanaken for the advice on the camera setups. I am also grateful to all my fellow students and EDM researchers who participated in the tests that were conducted for this thesis.

On a personal note, I would like to thank my parents for giving me the opportunity to study, and for supporting me throughout my whole life and education. Also thanks to all my friends for the moments of humor and relaxation during the creation of this thesis. Last but certainly not least, I want to thank my girlfriend, Evelien, for her support during my studies at the times that were the most difficult for me.

Raf Ramakers, January 2012

Enriching Touch

Raf Ramakers

Thesis proposed to achieve the degree of master in computer sience:
Human-Computer Interaction

Advisor: Prof. dr. Kris Luyten
Supervisor: Davy Vanacken

Hasselt University

Academic year 2011–2012

Abstract

In recent years, multi-touch capable input systems have increasingly been a focus of research in the UI community. There is no doubt that the success of surface computing stems in parts from its directness of interaction and of course its multi-touch and hence multi-user capabilities. This new accumulation of parallel input information from multiple users poses new challenges. For example, one user can open or alter files of a second user, or impersonate the second user, while she is absent or not looking. Furthermore, it is not possible to present content in the right direction to face specific users.

In this thesis, we will specifically focus on techniques to reveal the identity of touches and positional information of the users that are working on an interactive display. This information can be very useful to reach consensus in conflicting situations when users are working together. For example, when two users are trying to interact with the same object, the system can determine the result of the outcome. In addition, we will discuss novel applications and interaction techniques that exploit this new set of information. We believe that the identity of a touch point and the position of the user in front of a large display will become an essential feature of surface computing as the enabling technology is readily available.

Contents

Acknowledgments **i**

Abstract **ii**

Contents **iii**

1 Introduction **1**
1.1 Purpose of this Thesis 1
1.2 Overview of this Thesis 2

2 Touch Technologies **4**
2.1 Introduction 4
2.2 Sensor-Based Touch Technologies 4
2.2.1 Resistance-Based Techniques 5
2.2.2 Capacitance-Based Techniques 5
2.3 Computer Vision-Based Technologies 7
2.3.1 Purely Vision-Based Techniques 8
2.3.2 Optical-Based Techniques 9
2.4 Conclusion 15

3 Co-located Groupware **16**
3.1 Introduction 16
3.2 Co-located collaboration 16
3.2.1 Simultaneous Interaction 17
3.2.2 User Arrangement 18
3.2.3 Public and Private Spaces 20
3.2.4 Representative Content 24
3.3 Coordination Policies 25
3.3.1 Motivation 25
3.3.2 Conflict Dimensions 26
3.4 Conclusion 27

4 Touch Identification **29**
4.1 Introduction . . . 29
4.2 Technology Classification for Touch Identification . . . 29
4.3 Simple Software Approaches . . . 31
4.4 External Tokens . . . 32
4.4.1 Radio Frequency Identification . . . 33
4.4.2 Cricket Sensors . . . 35
4.4.3 Tokens Using Infrared Light . . . 38
4.4.4 Tracking Gloves . . . 40
4.5 External Devices . . . 42
4.5.1 Mobile Phones . . . 43
4.5.2 Pen Computing . . . 45
4.6 Bare Touch Identification . . . 48
4.6.1 Tracking Users . . . 48
4.6.2 Biometric Touch Identification . . . 53
4.7 Conclusion . . . 62

5 Multi-touch Interaction Patterns **64**
5.1 Introduction . . . 64
5.2 Structure . . . 64
5.3 Format . . . 65
5.4 The Catalog . . . 66
5.5 Conclusion . . . 80

6 Non-Intrusive High-Accuracy User Identification **81**
6.1 Introduction . . . 81
6.1.1 The Goal . . . 81
6.2 Finding New Identity Features . . . 82
6.2.1 The Approach . . . 82
6.2.2 Candidate Features . . . 82
6.3 Feature Extraction Techniques . . . 84
6.3.1 Measuring Hand Metrics . . . 84
6.3.2 Color Matching . . . 85
6.3.3 Eigenimages . . . 87
6.3.4 Local Binary Pattern Matching . . . 89
6.4 Hand Recognition . . . 92
6.4.1 Skin Segmentation . . . 92
6.4.2 Finger Detection . . . 94
6.4.3 Wrist Detection . . . 96
6.5 Conclusion . . . 102

7 Study Setup and Execution **105**
7.1 Introduction . . . 105
7.2 Quick-and-Dirty Test . . . 105
7.2.1 Tasks & Procedure . . . 106
7.2.2 Apparatus . . . 106
7.2.3 Participants . . . 106
7.2.4 Results . . . 106
7.3 Extensive Study . . . 109
7.3.1 Tasks & Procedure . . . 109
7.3.2 Apparatus . . . 111
7.3.3 Participants . . . 111
7.3.4 Results . . . 111
7.3.5 Discussion . . . 124
7.4 Conclusion . . . 125

8 The Identity Server **127**
8.1 Introduction . . . 127
8.2 System Architecture . . . 127
8.2.1 Hardware . . . 127
8.2.2 Software . . . 129
8.3 Demo Application . . . 130
8.4 Conclusion . . . 131

9 Discussion **133**
9.1 Introduction . . . 133
9.2 Enriching Touch . . . 133
9.2.1 Additional Touch Information . . . 133
9.2.2 Non-Intrusive High-Accuracy User Identification . . . 134
9.3 Future Work . . . 137
9.4 Contributions . . . 138
9.5 Conclusion . . . 139

10 Conclusion **140**

Bibliography **141**

List of Figures **156**

List of Tables **161**

Chapter 1

Introduction

Interactive graphical devices that use tactile technologies for direct on-screen manipulation, are known as touch displays. These displays are particularly suitable for co-located collaboration due to their natural interaction using fingers, as well as their similarities to the collaboration style of users in real world (e.g. working around a tabletop display). Although these systems are generally quoted as being multi-user capable due to being able to track multiple points of contact at the same time, these systems treat every input alike. It is therefore impossible to tell apart and respond individually to interactions from different users working together. Furthermore, the system does not know the position of a user around the table and as a result, content cannot be presented directly in the right direction to the user.

1.1 Purpose of this Thesis

Current interactions on direct-touch surfaces are almost all modeled as x-y coordinate information. However, we think that touch interaction is much higher dimensional than 2D. By leveraging additional information available from interactions on and above the surface can possibly result in richer and novel interactions possibilities. In this thesis, we specifically explore the use of identity and positional information of users around a touch display. When adding this information to each touch event, the system can individually respond to interactions from different users working together and present all content in the right direction. We will use the concept of touch identification in a very broad sense. for example, techniques for revealing a user's true identity will be presented as well as techniques for differentiating between hands and fingers of a user to provide new interaction possibilities.

In this thesis, we want to enrich the touch capabilities and collaborative aspects of large interactive displays by extracting additional information from touch contacts and interactions above the surface. The strengths and weaknesses of current techniques to realize this will be

discussed. Furthermore, applications that exploit this new set of information will be presented. Finally, we will try to develop our own enriched interactive system. We think that only by enriching the touch capabilities of current interactive displays can make these technologies more natural and easy to use. As a result, the true widespread use of these displays can be realized.

Envisioned Application

We envision a collaborative multi-touch system that can be used by multiple users at the same time. The system needs to be able to identify every touch point with high reliability while preserving the natural interaction that is inherent to multi-touch displays. As a result, custom functionalities can be provided, such as, multiuser undo buttons and parallel-moded interactions. Furthermore, the system can even provide support for customized content (e.g. translating text, display extra help, etc.). Another option is to give users privileged access to their private documents, contact lists, bookmarks, etc.

1.2 Overview of this Thesis

In this thesis, we will present a state-of-the-art review of possibilities to enrich touch interaction and also present our own contributions. There are 10 chapters:

- Chapter 2: A brief overview of popular touch technologies, as well as the latest techniques that researchers are currently investigating.
- Chapter 3: A discussion of important aspects of the social behavior of people that are working together around a large display.
- Chapter 4: An extensive overview of techniques to extract more information from touch contacts.
- Chapter 5: We will present 13 multi-touch interaction patterns in this chapter. All of them use additional information sources to enrich touch interaction.
- Chapter 6: In this chapter, we will present a new technique to identify users that are working simultaneously on an interactive display.
- Chapter 7: In this chapter, an extensive evaluation of our presented identification technique will be provided.
- Chapter 8: We will present a real-time solution in this chapter that integrates all our research findings.

- Chapter 9: In this chapter, we will reflect back on our study and discuss the future work that can be done.
- Chapter 10: The conclusion of this thesis.

Chapter 2

Touch Technologies

2.1 Introduction

Interactive graphical devices that use tactile technologies for direct on-screen manipulation, are known as touch displays. These technologies exist since 1970 and have been available in different forms [13]. Patents from that time [63, 74] demonstrate how camera based touch sensitive surfaces can be constructed. Touch technologies have a long history, which is comparable to the mouse that was invented in 1965 and became ubiquitous in windows 95. Because of the disseminated work from Wilson (e.g. [150]) and Han [45], we can say that in the past years touch input has hit the mainstream. The demand for these systems is pushing touch interaction to modalities ranging from camera-based wall-sized installations seen on CNN to the iPhone's [4] capacitive screen and the Microsoft Surface [24]. All these new products have helped to encourage a fundamental rethink of how we interact with computers because touch interaction provides a more intuitive and direct way of interacting with rendered content.

Various technologies have been introduced to develop touch capable devices. These technologies are diverse in terms of their approach to the problem of recognizing and interpreting touches. We classify these techniques in two categories: *sensor-based touch technologies* and *optical touch technologies*. This chapter will give a brief overview of recent approaches to develop touch capable devices and especially multi-touch devices.

2.2 Sensor-Based Touch Technologies

Sensor-based touch technologies use electronic sensors and circuitry to monitor changes in a particular state. Because these circuitries can be manufactured in very small components, these techniques already found their way into consumer products, albeit in small interactive surfaces such as mouse pads and phones. There are two types of techniques that are commonly

used in the industrial fabrication of small electronic devices with touch capabilities: *resistance-based techniques* and *capacitance-based techniques*.

2.2.1 Resistance-Based Techniques

Resistive touch screens have a flexible transparent top layer and a rigid bottom layer (typically a glass substrate) separated by insulating dots. The inside of these two layers are coated with a transparent conductive coating. Voltage applied to the these conductive materials produces a gradient across each layer. When pressing on the touch screen, the circuit is closed because the flexible top sheet makes contact with the bottom layer as shown in Figure 2.1 [114].

There are many techniques to determine the position of a touch point on a resistive touch-screen. The most common technique uses a unidirectional voltage gradient which is applied on the conductive material on the top layer. When the two sheets are pressed together, the bottom layer measures the voltage as distance along the top sheet, providing the x-coordinate. When this coordinate has been acquired, the uniform voltage gradient is applied to the conductive coating of the bottom layer to ascertain the y-coordinate. These operations occur within a few milliseconds, registering the exact touch location as contact is made. Because this technique relies on changes of electrical current along the axises instead of each point on the screen, most resistance technologies cannot sense multiple touch points at the same time. This type of resistance sensing is used in many mobile devices that support touch interaction, such as the Nintendo DS, the LG viewty phone and the Nokia Internet Tablets.

2.2.2 Capacitance-Based Techniques

Capacitive touch screens use a layer of capacitive material to hold an electrical charge. As human body conducts electricity, touching the surface results in a small transport of charge from the electric field of the screen to the finger. When there is a change in electrical charge, the system can register the location of the input. Capacitive technologies can even register the locations of multiple finger contacts and thereby supporting multi-touch interaction. However, this technique cannot recognize touches with gloves or objects because these materials are typically not conductive. There are two types of capacitive techniques: *Mutual Capacitance* and *Self Capacitance*.

Mutual Capacitance

This type of touch surface consists of grid-shaped transmitter and receiver electrodes as shown in Figure 2.2. The vertical wires are transmitter electrodes, and the horizontal wires are receiver electrodes. When a finger approaches a crossing point, it capacitively couples to the

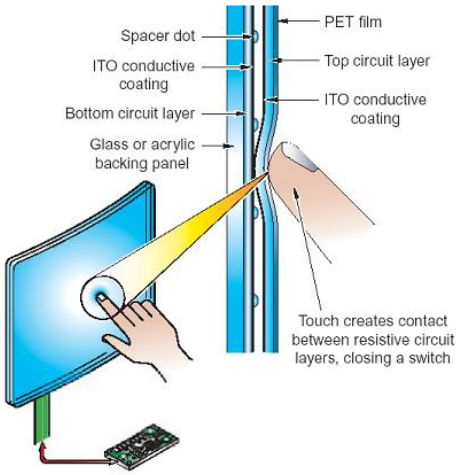


Figure 2.1: Resistive touch technology [46].

electrodes and drains the wave signal. As a result, the received signal amplitude becomes weak. By measuring this effect, the system can detect touch contacts on the screen.

Rekimoto et al. experimented with this technique and created SmartSkin [102] (see Figure 2.2). The system uses time-devision multiplexing to independently measure the values from each of the receiver electrodes and thereby support multi-touch interaction.

Self Capacitance

In self capacitive systems a grid of electrodes is located underneath the surface. The circuitry can sense changes in the charge of each capacitor along this grid because every cell generates its own signal when touched. Each capacitor extends over a single area of the screen to be unambiguously identified: the system cannot tell where on the electrode a user touches, just

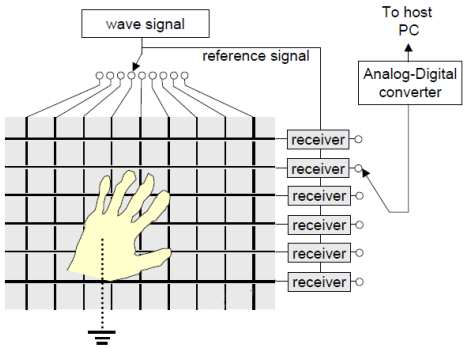


Figure 2.2: The mutual capacitance sensor configuration in SmartSkin[102].

that the user touches that cell. The touch screen of the iPhone [4] for example, is based on this technique. The capacitors are arranged according to a coordinate system and every capacitor in the grid relays its own signal to the iPhone's processor when touched.

The DiamondTouch [28] works with a similar technique. Here, the system transmits a different electrical signal to each part of the table surface that needs to be uniquely identified. However, instead of detecting changes in the charge of a capacitor from inside the surface, signals are capacitively coupled from directly beneath the touch point, through the user, and into a receiver unit located in the chair of this user (see Figure 2.3). This receiver can then determine which part of the surface the user is touching. Each user has a separate receiver inside her chair, therefore, the touch signals that are relayed to the computer for further processing can be uniquely identified (see Chapter 4). This setup can even be created using conductive floor plates instead of sitting in conductive chairs. However, the identification procedure is ambiguous when users are moving around the table or touching each other or the conductive material connected to the receiver.

The two largest groups of sensor-based touch technologies that are described in the previous sections are alternatives rather than opponents. Each technology has its strengths and weaknesses. For example, resistive touch screens can be operated on with fingers but also with other kinds of materials such as gloves, styluses, etc. On the other hand, capacitive touch screens can only be used with fingers. However, these displays have typically a better quality also in direct sunlight because there are no layers which reflect ambient light such as in resistive touch screens. However, sensor-based systems require industrial quality fabrication facilities and are therefore are almost impossible to build from off-the-shelf components. Furthermore, the costs of these components are very high. For example, In the DiamondTouch system, a special antenna pattern in combination with multiplexing was needed to reduce the number of antennas.

2.3 Computer Vision-Based Technologies

Due to the decreasing cost and improved performance of computers, computer vision technologies have been greatly improved. This enables systems to process and analyze video signals at high-speed to meet the requirements [142] for real-time interaction. Furthermore, computer vision techniques are very scalable and easy to setup. Due to this flexibility, every vision-based technology can track multiple touch contacts simultaneously and thereby support multi-touch interaction. For these reasons, researchers are putting forward a number of multi-touch systems based on computer vision which will be described in the next sections.

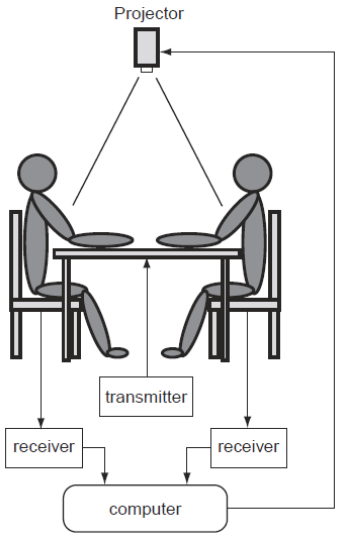


Figure 2.3: The DiamondTouch setup [28].

2.3.1 Purely Vision-Based Techniques

Purely-vision-based multi-touch systems rely solely on image processing techniques to track the position of touches. Multi-touch systems that employ this technique are of very high portability and can be used on any flat surface without the need for a dedicated display device. However, the flexibility of pure vision systems comes at the cost of precision.

Wilson et al. [151] developed a system which uses a depth camera to track touches. Dept-sensing cameras report the distance to the nearest surface at each pixel. The depth camera included with the Microsoft Kinect [23], projects a fixed pattern of infrared light in the environment. An infrared camera is used to calculate the precise manner in which this pattern is distorted as a function of depth of the nearest physical surface. When such a camera is mounted above a surface, the system can detect the presence of a hand or object by subtracting the depth image of the current frame from a referential depth map which was captured at the system startup. The depth map gives an indication of the distance in each pixel to the nearest surface. Therefore, this approach can even be used to sense touches on non-fat surfaces. To retrieve the touch points from a depth map, a thresholding technique can be used to classify regions with high depth values as touches.

Microsoft's PlayAnywhere [150] is a relatively compact and mobile desktop system with a front camera. Most notably is the shadow-based touch detection algorithm, which can accurately and reliably detect touch events and their contact position. However, Aragwal et al. [2] pointed out that the algorithm could achieve the best result only when the point of the

finger is vertical. For this reason, Aragwal et al. [2] developed a computer vision algorithm which uses a pair of cameras mounted above the surface to detect touches with a precision of a few millimeters. The algorithm relies on machine learning methods and a geometric finger model to achieve the required precision.

2.3.2 Optical-Based Techniques

Purely vision-based touch techniques are not very accurate and a clear line of sight is needed for the camera to capture a touch. For this reason, optical-based multi-touch technologies have become very popular. These techniques track touches from behind a semi-transparent surface and are using light sources (e.g. infrared, lasers, etc.) to track the position of a touch contact very accurate. In most cases, all the components of an optical touch display are integrated in a single table to make this system more mobile and robust. Many optical-based touch technologies exist, we will describe the most popular techniques in short.

Frustrated Total Internal Reflection

Frustrated Total Internal Reflection (FTIR) [44] is one of the most popular optical multi-touch technologies. The system introduces infrared light into a medium (typically acrylic) with an index of refraction significantly different than the air around it. When the light reaches the interface between the two mediums at an appropriate angle of incidence, it is reflected (see Figure 2.4). However when the surface is touched, the difference in the index of refraction between the two interfacing mediums (acrylic and skin in this case) is reduced and the reflection is frustrated, causing the light to escape out the opposite side. This light can then be detected with an infrared camera (see Figure 2.5-1). Feedback to the user is presented by projecting a view onto the diffuser material on top of the surface. This material serves not only as a projection screen but this layer also ensures that only bright touches are visible to the camera. In the FTIR setup, friction may reduce the FTIR effect, therefore a compliant layer (soft transparent material such as silicon) on top of the acrylic is needed to improve the sensitivity of the surface.

Diffused Illumination

Diffused Illumination (DI) [114] is the second very popular optical-based technology. In contrast to FTIR, DI-based systems rely on “seeing” what is on the screen, rather than detect touch, and so, DI is able to detect objects and hands hovering the surface. DI comes in two main forms: Rear Diffused Illumination and Front Diffused Illumination.

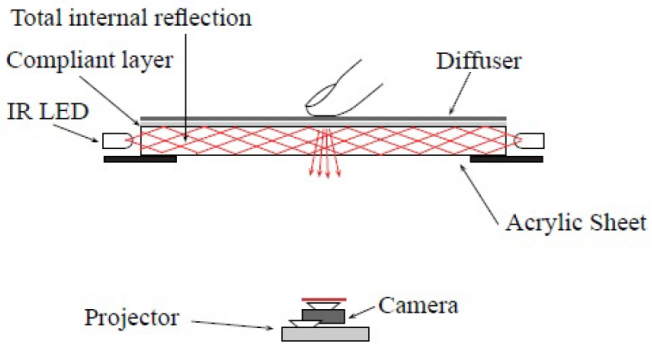


Figure 2.4: Frustrated Total Internal Reflection (FTIR) setup [114].

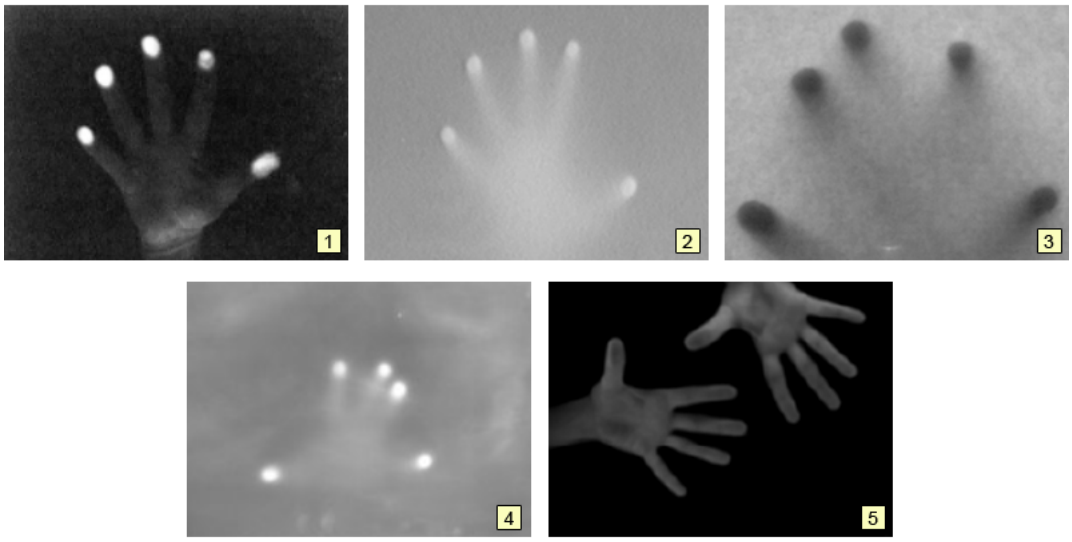


Figure 2.5: Captured images with an infrared camera in a (1) FTIR setup, (2) rear DI setup, (3) front DI setup, (4) DSI setup, (5) DI setup without diffuser.

Rear Diffused Illumination Rear Diffused Illumination is used in many multi-touch setups today [24, 78] and refers to infrared radiation which reaches the touch screen from the bottom, as shown in Figure 2.6. When an object or hand touch/hover the screen, infrared light will be reflected to the infrared camera (see Figure 2.5-2). Because this this type of diffused illumination can capture a bright image of the entire surface, visual markers [62, 21, 72, 16] placed on the touch screen can be identified.

Front Diffused Illumination In this setup, an infrared light source is mounted above the table to create shadows of hands and objects that are on or above the surface (see Figure 2.7). These shadows can be tracked by an infrared camera inside the table (see Figure 2.5-3). This type of diffused illumination cannot track visual markers because of the dark shadows underneath each object.

In both types of diffused illumination the quality of the diffuser is a significant factor for the performance of the touch screen because an even illumination of infrared light is required to track touch contacts on every part of the surface. For this reason, some systems use multiple infrared light sources to ensure that the infrared light is spread evenly over the surface. Compared with the simple use of a blob tracking algorithm in FTIR, DI uses a more complex image processing technology. In addition, DI systems are more vulnerable to external light sources.

Diffused Surface Illumination

Diffused Surface Illumination (DSI) is a variation of the standard FTIR setup. The setup is basically the same as a traditional FTIR setup (no compliant layer needed), with a special acrylic layer (called "EndLighten"). This special acrylic plate consists of small particles acting like thousands of small mirrors. When infrared light is shined into the edges of this material, the light gets redirected and spread to the surface of the acrylic as shown in Figure 2.8. The

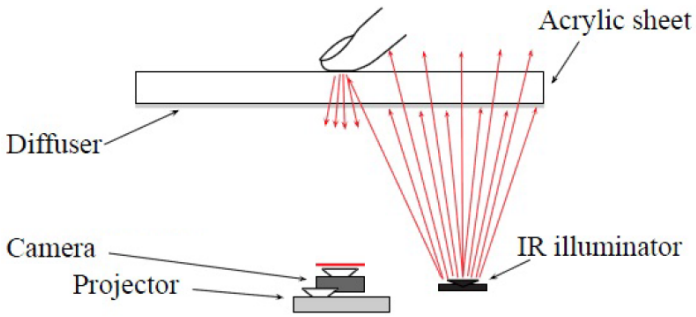


Figure 2.6: A rear diffused illumination setup [114].

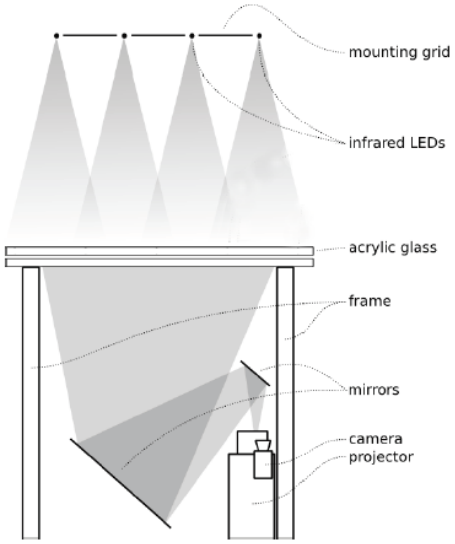


Figure 2.7: A front diffused illumination setup [33].

effect is similar to DI (see Figure 2.5-4), but with even illumination, no hotspots, and the same setup process as FTIR.

Transparent Projection Material

This thesis is all about enriching the touch capabilities of touch screens. It can therefore be interesting to capture images from the interaction above the surface or capture more clear images of users' hands for identification purposes 4. Traditional optical-based touch technologies however, prevent the capturing of clear images (see Figure 2.5-1-4) of the interaction above the surface because of the opaque diffuser which primarily serves as projection material. Recent research [149, 60, 77, 56] has therefore explored the use of semi-transparent diffuse materials.

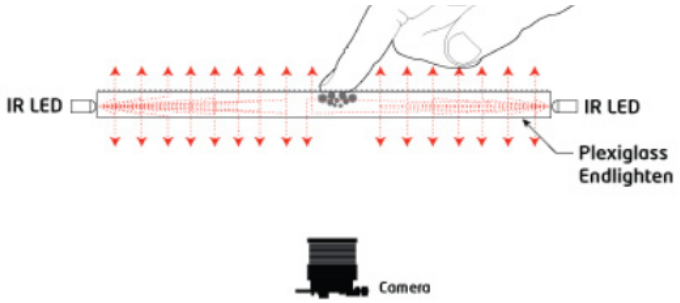


Figure 2.8: A Diffused Surface Illumination setup [114].

Kakehi et al. and Lida et al. [77, 60] investigated in the use of a lumistry film as projection material in an optical multi-touch setup. A Lumisty film is a layer which is diffuse from one privileged direction and has high transparency to incoming light from other directions. As a result, by positioning the projector and camera in the right angle, the system can project on the surface and capture images through the table at the same time. However, the quality of the projected image depends on the viewing angle of the user. For this reason, Wilson used a DNP Holoscreen [29] as the projection material for the TouchLight [149] display. A holographic screen consists of transparent material on which images can be projected. As a result, cameras behind the display can capture bright images of the interaction in front of the touch screen. This type of projection material is however very expensive and the projected images are not as bright as on traditional diffuser material.

Microsoft introduced a new technique to see beyond the display surface in the setup of SecondLight [56]. SecondLight uses a switchable diffuser which is a material that is commonly used as privacy glass in offices and store fronts. The optical properties of this material can be changed from frosted to clear at the flick of a switch. The diffuser consists of liquid crystal molecules which are randomly oriented and therefore scatter light in all directions. However, these molecules become untwisted and aligned in response to a suitable electric field which may be generated by applying a voltage across two parallel, transparent substrates on either side of the screen. When the projection material is transparent, an infrared camera integrated in the setup can capture a bright image of the interaction above the table as shown in Figure 2.5-5. When the projection material is diffused, the interface can be projected on the surface. In SecondLight, the system switches between these two states 60 times per second to avoid flicker perception. To turn the projection of the interface off when the diffuser is transparent, an optical shutter can be attached above the lens of the projector which switches at the same frequency as the projection material.

Flat Optical-Based Technologies

All optical-based multi-touch techniques described in the previous sections require bulky components under the table, such as a projector and a camera. As a result, users need to sit awkwardly or stand around the table. Therefore, researchers investigate new techniques to display content and track fingers at the same time on a multi-touch screen to get rid of the projector and camera.

First of all, the projector can be replaced by an LCD screen. When these displays are stripped, the liquid crystal panel is transparent to infrared light and can therefore be used in traditional optical multi-touch technologies [114]. Secondly, the camera can be replaced by a grid of infrared sensors placed behind the LCD screen. FLATIR [49] and ThinSight [55] for

example, have successfully integrated multi-touch capabilities in LCD panels (see Figure 2.9). Microsoft even used a special LCD display produced by Samsung with optical multi-touch capabilities for the production of the Microsoft Surface 2 (see Figure 2.10) [24]. This display uses a special LCD technology which has an infrared photo detector at every pixel. The backlight of the display is composed of visible and infrared light. When touching the surface, infrared light is reflected back to the infrared sensors at the pixels directly underneath the touch contact.

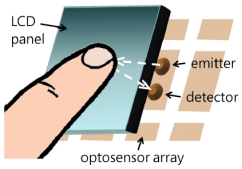


Figure 2.9: The basic construction of ThinSight [55].



Figure 2.10: The Microsoft Surface 2 in which infrared sensors are integrated in every pixel of the LCD display [24].

The multi-touch screens mentioned in the previous paragraph, integrated optical multi-touch techniques in LCD displays with significant modification of the display panel. ZeroTouch [80] however, provides an empty frame with multi-touch capabilities (see Figure 2.11). This frame can detect the position of all fingers inside, and can be placed on top of every surface (e.g. an LCD display). The frame consists of 32 infrared LEDs and 256 infrared sensors which can be read in parallel, allowing for very fast readout. Because visual information from a large number of perspectives is combined, the frame is able to track more than 20 fingers at the same time. The system does this by turning one infrared source on at a time and taking readings from all sensors on the three other sides of the frame. As a result, the system can provide an image of all objects and fingers inside this frame.

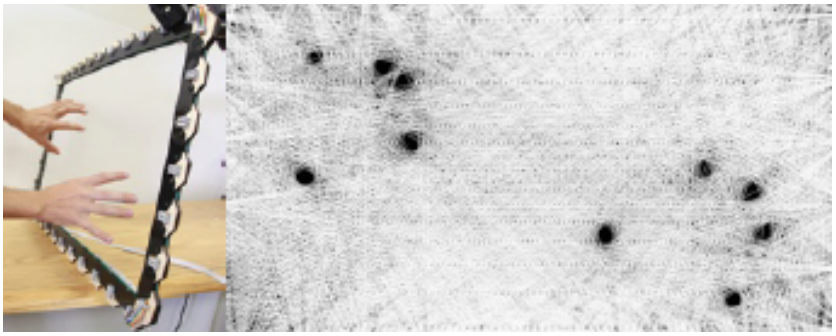


Figure 2.11: ZeroTouch: a frame which can detect the position of all fingers and objects inside [80].

2.4 Conclusion

In this chapter, many touch technologies are introduced. Furthermore, the advantages and disadvantages of these technologies are analyzed briefly. We mainly focused on multi-touch technologies because these displays provide many new interaction techniques such as interacting with multiple fingers and hands, supporting simultaneous interaction by multiple people, etc. These new interaction possibilities can be supported on both sensor-based and computer vision-based technologies. However, computer vision techniques are mostly used in research projects because they are very flexible, scalable and do not require industrial fabrication. Although multi-touch technologies provide a more natural and efficient way of interaction, there are still many problems to solve, such as identifying touches of multiple users, automatic orientation of content in collaborative work, more precise target selection, creating displays that are more convenient and portable, etc. Only by solving these problems can the true widespread use of these interactive displays be realized.

Chapter 3

Co-located Groupware

3.1 Introduction

The world's first personal computer, the Alto, had only a single keyboard and mouse. This fundamental design legacy has carried through to nearly all computer systems. Traditional applications have no special support for multiple users. In contrast, groupware applications are group-aware, they have a fundamental knowledge of multiple users. Networks have allowed computers to become very well suited to support remote collaboration. These applications are intended to be run on multiple workstations and can communicate with one another across a computer network. Another subset of groupware systems is co-located collaboration. In these systems multiple users are working together at a single system at the same time and place. These systems provide a single output channel and a shared input channel that can be used by multiple users at the same time [128]. Many research projects [54, 85, 34] exist in which co-located collaboration is supported. In addition, recent research [140, 102, 121] noticed the use of tabletop displays to support this type of collaboration.

3.2 Co-located collaboration

Through centuries, humans have created special skills for working, talking and collaborating around tables. For example, preserving a small private area, shielding objects in the personal area from prying eyes until the object is ready, making various piles at locations on the table, etc [119]. It is essential to take these skills into account when integrating computer technology into a table. Wu et al.'s landmark paper on Multi-finger and Whole gestural interaction techniques [153] mentioned that despite of all the high-end computing technology at their labs, they inevitably resorted to work out the design of their new offices on a conference room table with little paper cutouts of furniture pieces. This indicates that despite of all the digital

collaborative systems [129, 140, 102, 121] that are available, even users familiar these systems do not prefer to use them in real world situations.

In this section we will provide four important concepts to support collaboration in co-located groupware. We think that these concept are very valuable when creating new applications and interaction techniques for supporting co-located groupware. We believe that in the near future, interacting around these digital tabletops can become as easy as working together with physical objects around a traditional table.

3.2.1 Simultaneous Interaction

Traditional desktop computers do not effectively support co-located, multi-user interaction because of their underlying one-user/one-computer concept. Although these systems may be collaborative in the sense that multiple people can gather around and discuss the digital information, only one person can manipulate digital artifacts at any given time. In order to work collaboratively, users are forced to share the available input devices and monitor collaborators to know when the system is available. As a result, users need to adapt to this technology limitation and learn to take turns [53].

The adaption of users to technology can cause breakdowns in collaboration because users cannot work intuitively and need to concentrate on the interaction with the digital device instead of focusing on the collaborative activity. For this reason, Stewart et al. extended the design of the personal computer and introduced the concept Single Display Groupware (SDG) [128]. These systems enabled co-located users to collaborate via a shared computer with a single shared display and simultaneously use multiple input devices. Many research projects are based on the original configuration of Stewart e.g. Dynamo [54], Pebbles [85], LiveBoard [34], etc.

Dynamo [54] is a communal multi-user interactive display for supporting cooperative sharing and exchange of a wide range of media that can be brought to the surface. Once each user has attached a USB mouse and keyboard to the surface they can concurrently interact with the system by using a colored cursor (see Figure 3.1). In a usability test, subjects created an interactive poster on the Dynamo surface. They observed that although users had problems with overlapping objects because of the single pointer each user had, social protocols between participants tended to be sufficient to coordinate the collaboration. For example:

> L: Can anybody move this one? (has trouble grabbing title bar) I'm trying to get the old lady to the front.
> P grabs an occluding window and moves it out of the way



Figure 3.1: The Dynamo system provides a single mouse and keyboard for each user [54].

> R then grabs another occluding window and moves it out of the way leaving L with space on the surface. [54]

More recent work in Single Display Groupware [102, 28, 24] has discovered the abilities to use multi-touch displays for co-located collaboration. These interactive displays have great potentials for supporting collaborative work because their inherent ability to be used by multiple users at the same time. Multi-touch surfaces provide the ability to easily follow actions of group members since whole arms are moving in space. In contrast, when using a mouse, only a small cursor moves. This increase in awareness may be worth a decrease in precision of selections and speed because larger gestures need to be performed [10] [118]. When working with multi-touch displays, users can use both hands and multiple fingers to directly manipulate the content on the display. In addition, users tend to create their own private spaces on the table (see Section 3.2.3) [119]. Because of this behavior, users typically have less overview of the entire system when working in their own private space compared to the Dynamo system. Consequently, social protocols are less likely to be effective in conflicting situations, especially when results of unintentional actions have consequences for all collaborators. In these cases *coordination policies* (see Section 3.3) are needed.

3.2.2 User Arrangement

During co-located collaboration, people sit or stand around the display at a variety of locations. These arrangements can be affected by four factors: the social organization of the environment, the group task, the culture of the group members and the properties of the display.

First of all, the social environment can influence the arrangement of users around an interactive surface. In public outdoor spaces people are less likely to approach the surface and

collaborate with people they have never seen before. Brignull et al. [19] noticed that people have to overcome two thresholds before they can start interacting with a display. First, they have to withdraw from other activities. Then, when they are standing at the display, they have to wait for a free place and feel willing to use the display in the presence of others. In addition, Peltonen et al. [94] analyzed the recorded interactions of a public display installed in a central location in Helsinki. They noticed that people mostly noticed the wall when someone is using it. Furthermore, they observed that people often approached the display in a stepwise manner. Even though the wall was large enough for multiple groups to work simultaneously, people formed a queue to make their presence visible to the other group using the screen and waited for the other group to leave before approaching the display.

When designing collaborative applications developers need to consider that not all people take an equal part in the collaboration activity. When the display size is small or the technology only supports single touch, users need to take turns. Furthermore, during the observation of the *Citywall* [94], researchers noticed that assumed roles can exist, such as the teacher-apprentice setting, the comedian or the commentator. Some people can also hesitate to use new technologies and therefore need some encouragement of other participants [104].

Secondly, the type of the task that needs to be performed collaboratively can influence the preferred positions of a user at a table. When personal information is displayed on the table, the user probably wants to sit right in front of these objects so she can protect and cover this information with a hand. When the group activity is focused on conversation, people generally prefer to sit in a face-to-face or corner seating arrangement. In contrast, when working on a shared document without additional orientation sensing techniques (see Chapter 4) users prefer to sit on the same side of the table to read the document at the same time.

Thirdly, the culture and age can affect a person's preferred interaction distance. Children tend to favor side-by-side or corner seating arrangements while adults tend to prefer face-to-face arrangements during tabletop activities [118].

Finally, the psychical properties of a tabletop display, such as size or shape can influence the positions of people. Each person has a specific *distance zone* at which they interact comfortably with others [139]. Group members may feel socially awkward when they need to interact within a person's "intimate" space. In the CollabDraw system[82] some cooperative gestures were implemented on the DiamondTouch to achieve consensus among group members. For example, exiting the application could only be achieved when all group members are holding each others hands, and one member of the chain touched the surface with a single finger as shown in Figure 3.2. In a user study, particularly male groups made nervous jokes before making the gesture. Furthermore, they commented that the gesture was unpleasant and felt awkward. People feel more comfortable when they can work at arm's length since

this preserves their personal space.

Using a vertical interactive display to collaborate requires a much larger surface to support as many people as with a tabletop variant. This is because users can only take a position in front of the display. As a result, a turn-taking approach is usually used to collaborate on smaller vertical displays [118]. For example, the BlueBoard [107] is a vertical display introduced by IBM. Multiple users can login and view, compare and exchange their calendars and other digital media. On the one hand, due to the turn-taking protocol, users could easily see how someone else could do things they wanted and pick up this behavior. On the other hand, many moments of hesitation were noticed during a usability test when controlling the board required reaching across other persons standing in a controlling position.

Horizontal tabletops are more suitable for larger groups to collaborate because all sides of the display are available for one or more users to sit down and collaborate [118]. However, at some horizontal tables users need to stand or sit awkwardly because of the bulky components under the table, such as projectors and mirrors for bottom-projection. Furthermore, when users are sitting at various locations around a table, the displayed information may not be oriented appropriately for all users. We will discuss this problem extensively in Chapter 4.

3.2.3 Public and Private Spaces

When working together at a traditional table with physical objects there is an inevitable division of tabletop space into shared common areas and smaller private areas. This is because people want to work on a preliminary version in a small private space before it is evaluated by the other workers. Or sometimes there is a competitive behavior in which each user wants to discretely work on their version of an object. These social behaviors are natural for humans and for this reason a digital collaborative system must provide features for users to work privately and in public.



Figure 3.2: Group members need to form a chain of hands to exit the CollabDraw application [82] and feel socially awkward.

Digital Private Spaces

A private space can take many forms depending on the type of the application and the security needed for the objects presented in this personal space. When the application is small and only objects need to be viewed and shared, a single personal icon for each user can be sufficient. The IBM BlueBoard [107] integrated this type of personal spaces in their collaborative surface. When a user logs in, a personal icon or p-con is displayed on the right side of the display as shown in Figure 3.3. This icon is the link to the users homepage. Objects can be shared by dropping them on a user's p-con.

The Dynamo system [54] provides a similar small personal area called a *personal palette* in which personal media is collected. In addition, users can "carve off" parts of the large public surface by dragging their cursor on the background. These private regions called *carved regions* can be used to work privately on documents and other media (see Figure 3.4). Furthermore, a carved region can be shared with another group member by dragging a key in a user's personal palette. In this system, access to personal objects can be easily and reliably controlled because each user has its own input device as shown in Figure 3.1. To realize this level of user identification on an interactive surface, additional technologies are required as described in Chapter 4.

On interactive tabletop surfaces, a private space is often located immediately in front of the user (e.g. the Table for N [122] shown in Figure 3.5). In addition a table can use resizable personal spaces that can either be customized by the user or automatically positioned by the system based on the intent of the action. In these arrangements, the users can work discretely on private objects and can even cover their work with a hand [153]. If the system supports shape sized gestures, the user could be assisted with additional personal information on the object as shown in Figure 3.6. In contrast, hiding a personal object when working with

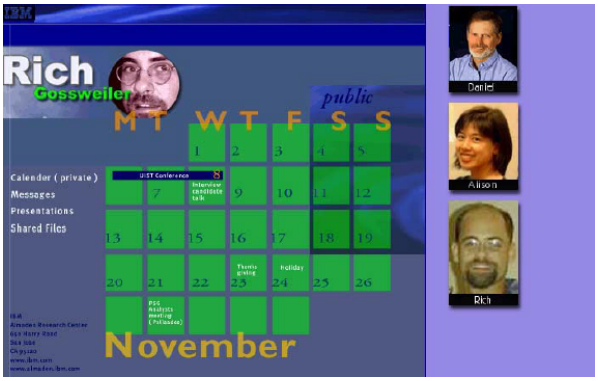


Figure 3.3: The public and private areas in the IBM BlueBoard system [108].



Figure 3.4: The Dynamo workspace arrangement: a personal palette for each user, at the top of this figure a carved region is shown shared by the two users [54].

traditional input devices (e.g. a mouse) instead of direct touch, such as the Dynamo system is not possible. The shape of a tabletop system may also influence its ability to provide private spaces. An experiment has shown that users find it more difficult to partition round tabletop systems in individual workspaces compared to square or rectangular tables [137].

Another technique to provide personal information is using an external device on which a private workspace is visible. Shirazi et al. [123] experimented with the combination of mobile phones and multi-touch tables, they implemented a collaborative poker game on an interactive display in which each user has a small space with her cards displayed backwards. Each user

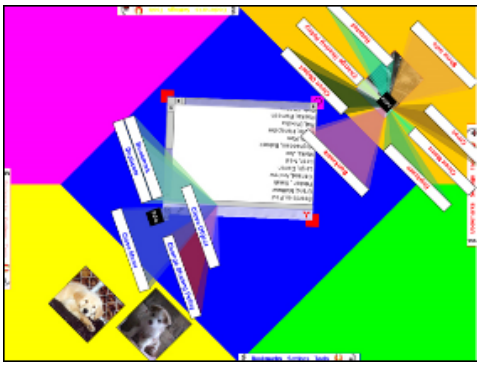


Figure 3.5: The Table for N created with the DiamondSpin SDK [122].



Figure 3.6: Covering a personal object with a hand [153].

has a mobile phone which serves as a private space to show the player's hand. Furthermore, this mobile phone provides a tangible feeling and allows the user to hold their "cards" in the hand. Other research project such as I-LAND [130] explored how a secondary device (i.e. a tablet) can be placed on or adjacent to the tabletop display and serve as a private workspace. Separate displays create obvious boundaries between the personal and shared workspaces, although this separation may hamper interpersonal interaction.

Digital Public Spaces

A public space is usually a larger area in the center of the screen that is easily accessible for all people (see Figure 5.8-1). In this area, users can share objects, work collaboratively or view the preliminary result of the project on which the group is working. This single representation of an object ensures that each participant sees the same updated version of the work. In addition, each participant can see other participants place new elements. As a result, these movements can be easily interpreted during collaboration which increases the awareness of other participants.

The DiamondSpin [122] provides a *black hole* or a *central focus* technique for arranging documents in the public space. When using the black hole layout, the size of objects become smaller when moving them towards the center of the display, keeping with the principle that many objects are displayed in this shared space. In contrast, when using the central focus layout, objects become larger when moving them towards the center of the surface. As a result, all users can see and manipulate the shared object. To share objects easily in the DiamondSpin tabletop arrangement, the black hole layout can be extended to a *wormhole* layout. In such an arrangement, sharing of objects is supported by the connected endpoints of the "wormholes". Each endpoint would be located in different private spaces and objects can be shared by dragging them to the mouth of a wormhole.

3.2.4 Representative Content

When working collaboratively on an interactive display, the ability of the application to provide relevant content, and the extent to which the application addresses privacy concerns are two important issues. Huang et al. indicated in their research on semi-public displays for small, co-located groups [53] that the success of a collaborative application primarily depends on these two factors because it influences the awareness of the system. On the one hand, a system that is highly aware of each user around a collaborative display can greatly enhance the cooperation of these users in collaborative tasks. A participant can for example use and share personal documents that she has been working on. On the other hand, applications that provide a high level of awareness in public settings, can cause problems which will be described in the next paragraph.

Applications intended for large groups that provide high levels of awareness by integrating more detailed information about individuals often face difficulties because of privacy concerns. It is difficult to make personal details available to large groups without impinging upon privacy. For this reason, applications deployed in public spaces are most successful when this system provides only low-level awareness, for example, an indication of the presence or absence of group members. In other words, a tradeoff exists between privacy and depth of awareness. This tradeoff is based on three factors: the type of environment (public, semi-public or private space), the size of the group (pair, small or large) and the type of the display. For example, when large groups are using traditional desktop computers or systems like Dynamo [54] in public settings, makes the consideration of privacy concerns even more challenging because it is difficult to assess who is looking at what information and how often. In contrast, large interactive tabletop systems can use new authentication and identification mechanisms to protect private information and enhance the awareness of the system. These possibilities will be explored in Chapter 4.

The other challenge for presenting detailed awareness in collaborative work is the relevance of the information to the audience. Some systems rely on the users to supply content explicitly, for example in the Dynamo [54] system, the user can integrate personal media by connecting a storage device. Accordingly, the I-LAND environment [130] uses the *Passage* mechanism to move digital information easily from a computer to a shared display. In contrast, systems intended for large groups that rely entirely on user-submitted content tend to be uninformative because the content is not of interest to much of the audience.

Another type of content that can enhance collaboration is the integration of physical objects on interactive tables. This allows users to apply the years of experience they have accumulated collaborating around traditional tables. In addition, the use of a robust surface such as the DiamondTouch [28] display encourages users to treat the system surface as a

table, not as a delicate display. To support the practice of using physical objects on a table, researchers have investigated tangible user interfaces [38] as an alternative and an extension for interaction techniques that are currently available. Many technologies exist for identifying objects on tabletops such as bar-codes [78], RFID tags [106] or computer vision technologies [62, 21]. The disadvantage of these technologies is that objects must be pre-tagged or the contours of the objects must be defined in the system. For this reason, systems can never recognize all objects. On the other hand, the system can recognize the placement of an object and avoid displaying relevant information in the physical space occupied by that item.

3.3 Coordination Policies

Current systems for supporting co-located collaboration rely almost entirely on social protocols for coordinating the actions of a group of users. However, this strong dependency on standards of polite behavior can have shortcomings when the size of the group increases and it becomes difficult to monitor everyone else's actions. For this reason, *coordination policies* - direct mechanisms for avoiding and resolving conflicts can potentially improve groupware interfaces.

3.3.1 Motivation

In some cases social protocols provide sufficient mediation in groupware for example the Dynamo system [54]. In this system, a small group of users, each with it's own input device has an overview of the entire collaborative workspace. As a result, group members can follow their coworker's mouse cursor and provide help when performing tasks or react when a coworker is trying to execute an action that affects other users. In contrast, when using an interactive surface in a collaborative setting, users do not always have an entire overview of the system because they tend to focus on their own private space. As a consequence, conflicting situations are more likely to occur. Furthermore, the ease of "reach out and touch" on direct-manipulation devices, such as shared multi-user tabletops makes social protocols even more inappropriate for coordinating all actions because accidental activation of actions are very likely to occur.

Invocation of potentially destructive application events such as quitting the application, deleting a large amount of content, etc. can be problematic in groupware systems. This is because social negotiation becomes challenging when the size of the group grows. In addition, these actions can be invoked accidentally by a group member. Previous work in this area has explored the potential of cooperative gestures [82] for coordinating actions in a painting application called *CollabDraw*. CollabDraw primarily uses *additive gestures* to involve all group members in executing important actions. An additive gesture is a gesture which is

meaningful when performed by a single user, but whose meaning is amplified when simultaneously performed by all members of the group. For example, the CollabDraw application supports an erase gesture which can be performed by rubbing the palm of a hand back and forth on the surface of the table. When all users simultaneously perform this motion, the effect is magnified and the entire table is instantly cleared. On the one hand, users can reach consensus by performing gestures simultaneously. On the other hand, gestures can be difficult to remember especially when the resulting action depends on the number of group members that perform the gesture simultaneously. For this reason, we will discuss a general method to reach consensus in collaborative work called *Coordination Policies*.

3.3.2 Conflict Dimensions

The number of coordination policies are probably endless and can be combined in various ways in a single application. Morris et al. proposed 18 general coordination policies in their study of co-located groupware [83]. They classified these coordination policies along two dimensions: *conflict type* and *initiative strategy*. The type of the conflict refers to the level at which the conflict occurs, whether it affects the state of the entire application or only a single object. In contrast, initiative strategy refers to whose actions determine the outcome of the conflict.

Conflict Types

In general, there are two levels of conflicts: *global conflicts* and *whole-element conflicts*. Global conflicts can change the application state and are therefore potentially disruptive to other group members. For example, when the system provides a "magnet" feature to reorient all open documents on the table to face a single user. Global conflicts can be resolved by applying *global coordination policies*. These policies are used to mediate in the execution of an action that affects the application as a whole. The system can coordinate this activity and execute this functionality only when there are for example *no selected objects*, or when all users have *voted* or there can be a *ranking* technique, etc.

In contrast to global conflicts, whole-element conflicts involve access to a single object. For example, when multiple users are trying to select an item from the same menu. These conflicts can be resolved by applying *whole-element policies*. Consider, for example, the case where two users are trying to take the same object. The system can coordinate this activity by *duplicating* the object or by creating a *personalized view* of the object by displaying the content customized for each user who takes it. On the other hand, the application can even make the object temporarily inactive to both users (*stalemate* policy) which can encourage conversations.

Initiative Strategies

Coordination policies can also be classified by taking in perspective whose action determines the outcome of the conflict. These strategies called *initiative strategies* can be categorized in three policies: *proactive*, *reactive* and *mixed-initiative*. Proactive policies are based on the assumption that the initiator of the action, or the owner of the object on which an action is performed determines the outcome of the conflict. When using reactive policies, the outcome of a conflict is based on the actions of the other users (e.g. the users who are affected by the global change or the user who tries to use someone else's object). In mixed-initiative policies the outcome of a conflict is determined by the action and state of all parties involved in the conflict.

Table 3.1 gives a classification of all coordination policies introduced by Morris et al. [83] grouped along the two described key conflict dimensions.

3.4 Conclusion

In the first part of this chapter we explored four elementary concepts for collaborative systems. We have seen that it is not only important to provide simultaneous interaction of multiple people but also to consider the arrangement of group members because there are many aspects (e.g. the environment, type of the task, culture, etc.) that can influence users' ability to work successfully in a group setting. In most collaborative applications there is also a need to provide some kind of private area for each user. These kind of private spaces can be integrated in the application in various ways, such as static predefined rectangles, resizable regions, a single folder for each user with private documents, etc. In addition, we described the importance of representative content and privacy concerns in a collaborative system, for example, easy integration of personal documents.

In the second part of this chapter we identified possible issues when co-located groupware systems are used to support collaborative work. For example, trying to reach consensus before executing a potentially destructive event. Coordination policies were described to handle these kind of problems. However, many of these coordination policies require the support of a user identification mechanisms. For example, when a user moves two fingers on an object to the opposite side, the object is increased in size. In contrast, when the same action is performed by two users, each trying to move the document towards themselves as shown in Figure 3.7, the system needs to execute a coordination policy. In this case, the system can only unambiguously determine which action needs to be executed when a reliable user identification mechanism is integrated. Furthermore, other degrees of freedom such as orientation can significantly improve the usability of a couple of coordination policies. For example, displaying a dialog

	Proactive	**Mixed-Initiative**	**Reactive**
Global	privileged objects anytime	rank	no selections no touches no holding objects voting
Whole-Element	sharing explicit dialog	rank speed force	public private duplicate personalized views stalemate tear

Table 3.1: Coordination policies grouped along the dimensions of conflict type (rows) and initiative strategy (columns) [83].

box to face the user. For this reason, techniques to realize all these new interaction possibilities will be discussed in Chapter 4.



Figure 3.7: Two different users executing a pulling away gesture [26].

Chapter 4

Touch Identification

4.1 Introduction

Large interactive displays are particularly suitable for co-located collaboration due to their natural interaction using fingers, as well as their similarities to the collaboration style of users in real world, for example, working around a tabletop displays. In Section 3.2.1 we mentioned *simultaneous interaction* as an important requirements for devices intended for collaborative work. Although many interactive systems designed for co-located collaboration [44, 24, 102] can sense multiple input points at the same time known as *multi-touch sensing*, they have no real mechanism to distinguish between the touches of multiple users.

In this chapter we will discuss many possibilities to extend current multi-touch technologies with touch identification methods. We believe that the association of touches to users or hands is certainly a huge benefit and will become essential information as applications mature and new application areas emerge. Touch identification could possibly enable a whole set of new interaction possibilities, including multi-user-aware interfaces [109, 110] and access control [117, 113]. For example, different users can chose their own line widths and colors in a painting application, users can instantaneously access personal data or customize content (starting a browser application will bring up users' customized start page). These new interaction possibilities for user interfaces will be further discussed in Chapter 5. We will first start exploring how some touch technologies are already able to identify multiple users in some ways and how other technologies can be extended to support and improve this ability.

4.2 Technology Classification for Touch Identification

Multi-touch systems are generally quoted as also being multi-user capable due to being able to distinguish between multiple points of contact, regardless of whether these points stem from a

single user or multiple users. While some capacitive systems such as the DiamondTouch (discussed in Section 2.2.2) can identify touch points for up to four users, optical-based tabletops generally cannot. Without this user identification, all input is alike: it is impossible to tell apart and respond individually to interactions from different users working together.

As argued in the previous sections, there is clearly a distinction between supporting multiple concurrent touch points and supporting multiple concurrent users. Therefore, Figure 4.1 shows how some important touch technologies which were described in Chapter 2 can be classified based on the number of distinguishable users (single or multiple), as well as the number of concurrent touches they support (single or multiple) [59].

Currently, there is only one system that can provide some kind of identity when a user is touching the surface: the DiamondTouch. We described the technology behind this system in Section 2.2.2, but what this system really does is identifying the chairs of the users. As a result, users cannot move around the table during a collaborative session. Popular capacitive-based multi-touch handheld devices like for example the Apple iPhone [4] or Apple iPad [5] can sense multiple touches at the same time but cannot truly support multiple users. Because these devices are relatively small they are usually not qualified for concurrent use. On the other hand, identifying the user who currently holds the device can help to personalize the device based on a predefined user profile. For this reason, Apple is working on a technique that uses capacitive sensors to recognize a user's identity by the way she holds the device [50]. The largest group of touch technologies in the category multi-touch, single-user are the optical-based touch technologies (see Section 2.3.2). The intention of this chapter is to describe possible mechanisms that can be integrated into optical multi-touch technologies and as a result move this large group of interactive displays to the multi-user multi-touch category.

In Section 2.3.2 we addressed the most important optical-based multi-touch technologies. We noticed that the captured finger blobs in a typical FTIR setup are brighter and thus more easier detectable than those in captured images on a DI touch-table. On the other hand, each touch on a FTIR table ultimately remains a standalone piece of data. When using a DI-setup in contrast, the users' hands reflect more infrared light. However, there is not enough information to relate a hand to a user. Furthermore, the brightness of the contours of a hand on a DI multi-touch table highly depends on the posture of this hand, the lighting situation in the environment and the type of diffuser.

In this work, we will use the concept of user and touch identification in a broad sense. On the one hand, techniques for revealing a user's true identity such as fingerprints (see Section 4.6.2) will be discussed. On the other hand, we are especially interested in assigning touches to users and relating touches to hands which can for example provide more advanced gestural interaction techniques. In this case, gestures involving multiple fingers or hands can

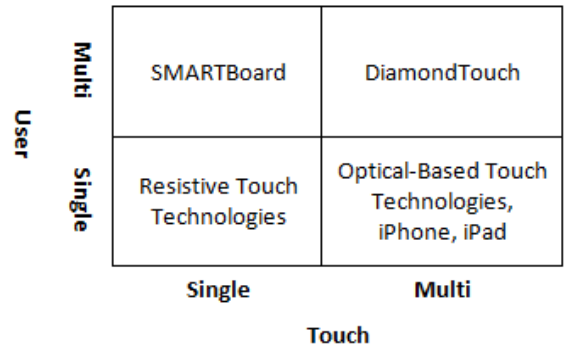


Figure 4.1: Classification of some important touch technologies based on their multi-touch and multi-user capabilities [59].

be interpreted unambiguously [84]. For example, the system could be able to distinguish a dual finger stretch gesture performed by fingers of the same hand from a dual hand duplicate gesture where a copy of a visual object is produced (see Figure 4.2).

Providing support for touch identification is a challenge and a salient issue for tabletop system research. Simple software heuristics that will be described in Section 4.3 are probably not reliable enough as applications mature. Therefore, an extensive overview of touch identification techniques will be discussed in this chapter. We classified all these techniques in three categories: techniques using *external tokens*, *external device* and *bare user identification*.

4.3 Simple Software Approaches

One of the simplest techniques to identify touches is to divide the surface of the table into discrete territories as described in Section 3.2.3. In this case, any action occurring in such a

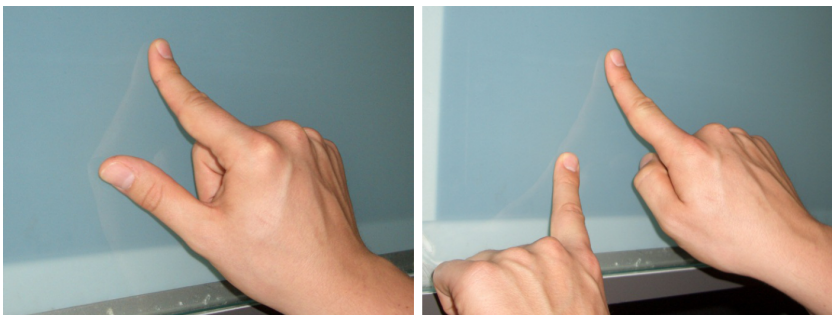


Figure 4.2: Without assigning fingers to hands, the gesture on the left cannot be distinguished from the gesture on the right [26].

territory belongs to the owner of that private space. Touches in the public space however, cannot be assigned to any user. To solve this problem, each user can use a movable and resizable region called an *interaction workspace* [59]. In this approach, users first need to move their interaction region over the space of the display which they want to manipulate as shown in Figure 4.3. These interaction workspaces can provide some clever features such as maintaining an interaction history to support undo and redo operations for each individual user. The downside of these kind of private spaces is that users need to move their region every time they want to interact with another part of the display. In addition, this identification method relies entirely on social protocols. When users want to work together on the same object they need to interact carefully to ensure that they don't touch a group member's interaction workspace because these regions are mutual exclusion locks on parts of the user interface.

4.4 External Tokens

A token is a tag which provides no direct functionality for a person. In this Section we will provide an overview of the possibility of tokens to identify persons which are interacting with a multi-touch table. We will see that the use of tokens is generally the easiest method for user identification because they are designed for this purpose. In contrast, these tags can be unwieldy and each user needs to be instrumented with a token before interacting with the table.

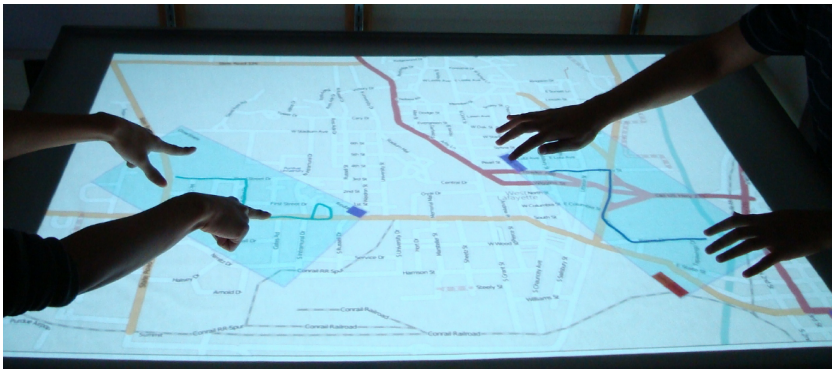


Figure 4.3: Two users interacting on their interaction workspaces to give input in a bus route map application [59].

4.4.1 Radio Frequency Identification

Radio frequency identification (RFID) [106] is an effective automatic identification technology for a variety of purposes, identification of manufactured products as well as humans and animals. This technology has many benefits over traditional visual code markers such as barcodes [16], QR-codes [72] or other fiducial markers [62, 21]. For example, visual code markers are read-only and can therefore never send out any information. Also, a clear line of sight is needed to identify a visual code. In addition, such visual representations cannot encode much data compared to traditional RFID tags which can store at least 2 kilobytes depending on the type of the tag. Because RFID tags are cheap and have many technical benefits they can be employed in a wide variety of application domains such as grocery stores and human identification. In this section we will investigate if RFID can be used to identify touches on an interactive table.

An RFID system consists of 2 important components: a reader and RFID tags. The overall idea is that the reader uses radio waves to broadcast a signal to all RFID tags in the vicinity. When a tag receives such a radio signal, it reflects back a modulated signal that carries some information, typically a global unique identifier. The reader then provides the list of sensed IDs back to the host application.

To use this popular technology for identifying touches on an interactive table, each user needs to hold an RFID component in his hand (or integrated in a glove). A touch table with integrated RFID capabilities can now detect the users that are currently interacting with the system. On the one hand, the table can reliably identify the presence of multiple users. On the other hand, the system cannot assign an identity to a single user when multiple group members are interacting with the table at the same time. This is because the traditional RFID technology is not designed for localization purposes and therefore reports only the presence of a tag. Several research projects in ubiquitous computing [101, 36, 98, 99] investigated the use of custom or modified RFID readers and tags for positioning purposes in addition to identification.

Sugimoto et al. [131] contributed to the problem of tracking the location of an RFID tags. They constructed the ePro board to locate tangibles that are placed on the surface. The ePro board is a checkerboard (20*24 matrix) and consists of 480 integrated readers to identify the position of a tangible based on the reader that identified the tag on top of the board. Parallel processing with 30 units each controlling 16 readers where needed to reduce the delay. In these kind of setups, the range of the readers need to be very small (the size of a single cell in the matrix) so that the detection of an RFID tag can serve as a high-confidence indicator that the tagged object is exactly on top of that reader. To use this technique in a tabletop setting, the users need to hold their tags attached to their palms close to the surface while

interacting. Another option is to make the ranges of the readers (and the size of the cells in the matrix) bigger which would result in even less accurate information. Furthermore, when using this setup, the back of the surface will be entirely covered with readers which has many disadvantages, for example, top projection and other methods to track touches using cameras above the table are needed (see Section 2.3.1). Because the tags are much more cheaper then the readers, this construction will probably have a lower production cost if each user holds an RFID reader and the back of the table is covered with tags. However this technique has still the drawbacks of low accuracy and the need for top projection and alternative touch tracking.

To make touch identification using RFID more integrable in the design of optical based multi-touch displays, less components are needed. Olwal et al. [91] presented a technique to determine the position and identify tangibles on a camera-based touch table using a single reader inside the table and an RFID tag on each object. Figure 4.4 shows this setup. Optical-based touch technologies and RFID are two complementary sensing modalities. The first one can track locations while the second technique tracks the identity. Synchronizing these two sources can result in a touch identification technique. However, when multiple users are interacting with the display at the same time, this synchronization procedure can be ambiguous when multiple tags are placed on the display at the exact same time.

Fishkin et al. [36] discusses another technique to estimate the location of an RFID tag by approximating the signal strength based on the response rate (the number of successful responses divided by the number of attempted polls by the reader). However, traditional interactive tables are small and free of obstacles. For these reasons, we think that there will be no significant difference in response rates between tags, even if multiple readers with small ranges are used.

The position of a hand above an interactive surface can be determined more reliably using

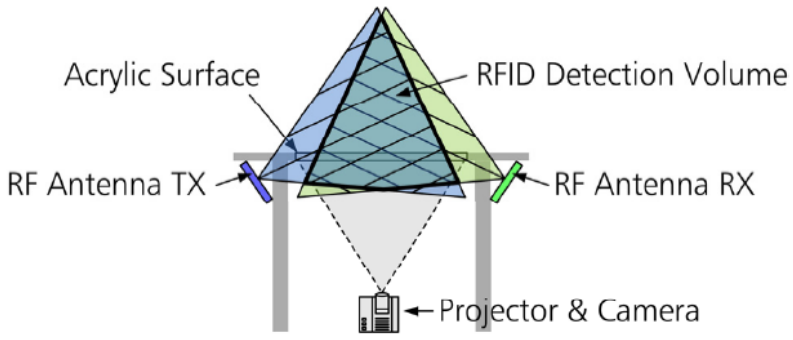


Figure 4.4: Extending a traditional optical-based touch table with RFID identification capabilities [91].

Radio Frequency Identity and Geometry (RFIG) tags in combination with structured light [99]. RFIG tags are RFID tags with an integrated photo-sensor. When a user moves a hand above the display, the RFIG tag attached to her hand will be awoken by the broadcasted radio waves of the reader inside the table. The photo-sensor on the tag now takes a reading of ambient light in the environment to be used as a zero for subsequent illumination measurements. A second projector above the table (or inside when using a switchable diffuser as discussed in Section 2.3.2) now illuminates this tag with a sequence of binary patters (binary structured light). Each projector pixel emits a unique temporal gray-code, and thus encodes its position. The tag records the gray-code that is incident on its photo-sensor, and then makes a transmission of its identity plus the recorded gray-code back to the reader (see Figure 4.5). When the reader has received all this information, it is straightforward for the system to relate a touch to an RFIG tag and the user's identity.

Extending RFID tags with location tracking techniques could provide some possibilities for user identification when these tags are attached to the hands of each user. When the location information is reliable enough, this technique can even be used to provide a hover state on multi-touch tables. For example, displaying extra information when a user moves a hand over an object. Furthermore, the fusion of hand and touch point locations increases the robustness of touch sensing in the presence of unwanted IR light because of the redundancy of the positional information. In contrast, the RFID technology is only designed for identification purposes, it is therefore relatively complex to extend this technology to track the locations of tags. For this reason, other tracking technologies using tokens such as *cricket sensors* or *infrared wristbands* are probably more suitable to identify touches and will therefore be described in the next sections.

4.4.2 Cricket Sensors

The cricket technique [96] is a location-support system that was originally intended for in-building, mobile, location dependent applications. It allows an application running on a mobile or static node to learn its physical location by using an attached *listener* that analyzes

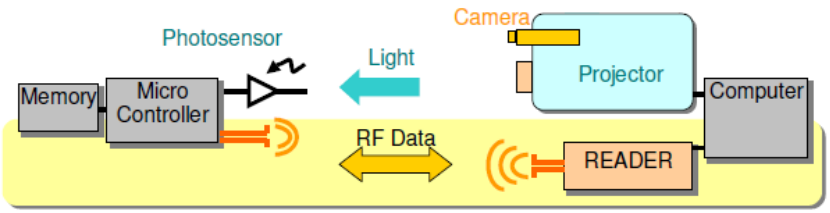


Figure 4.5: Determining the position of an RFID tag using an integrated photo-sensor called RFIG tag [99].

information received from *beacons* spread throughout a building. Beacons are electronic units that can be attached to walls or ceilings. These components have fixed position coordinates which they broadcast regularly to nearby listeners. These listeners can then estimate their position in the environment. The cricket system is entirely decentralized because there is no centrally maintained database that stores location information for all mobile units. The system is therefore called a *location-support* system instead of a *location-tracking* system.

An undergraduate student of the Iowa State University explored in his master thesis how these sensors can be used to identify touches on an interactive tabletop [125]. In this work, four beacons were mounted on a frame above the interactive table as shown in Figure 4.6. In addition, a cricket listener was attached to the hands of each user (see Figure 4.7). By estimating the position of a hand above the table using the cricket system, each touch point could be related to a single user. In the next paragraphs we will describe his study in short.

The cricket location-support system consists of 3 major components: listeners, beacons and base stations. A beacon communicates with a listener using a combination of radio and ultrasonic waves. On each transmission, a beacon concurrently sends its unique identifier and static location information over the radio frequency spectrum, together with an ultrasonic pulse. The radio frequency signals are transmitted at the speed of light. In contrast, ultrasonic signals have a much lower propagation time because they are transmitted at the speed of sound. This combination of signals enables a listener to estimate the distance to a beacon with an accuracy of 4-5 centimeters by measuring the delay between the receipt of the two signals. Each listener transmits the distance to each beacon to the base station using radio wave signals. The base station is a cricket node, which listens to transmissions from all the listeners and acts as a bridge between these listeners and the computer. When the computer has received at least 3 distance measurements of the same listener to different beacons, it calculates the position of the user's hand using triangulation techniques. The system can now relate all touch points to the nearest hand and as a result identify these touch points.

While cricket has the attractive property that its decentralized network is easy to configure and manage, it comes at the absence of explicit coordination. There is no explicit scheduling or coordination between the transmission of different beacons that may be in the close proximity. This lack of coordination can cause radio frequency transmissions from different beacons to collide. The cricket protocol solves the problem of collisions using *randomization*. The choice of the random interval is governed by the number of beacons and the time it takes for the transmitted information to reach the listener. The experiment of Siddalinga on a tabletop surface [125] used an average frequency of 800 ms which resulted in an average update interval of 2 seconds for the position of a hand. If the number of active users around a collaborative table is less then the number of beacons, an active mobile architecture can be used instead of the traditional cricked setup to increase the update rate of the positions of users' hands.

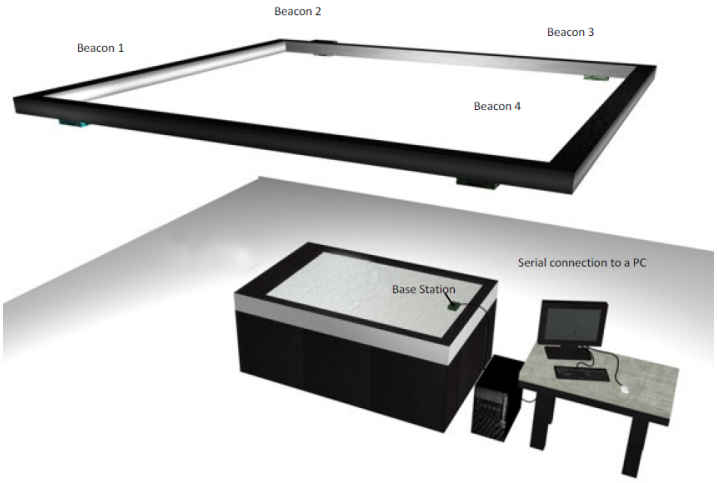


Figure 4.6: Cricket based infrastructure for touch identification on a multi-touch table [125].

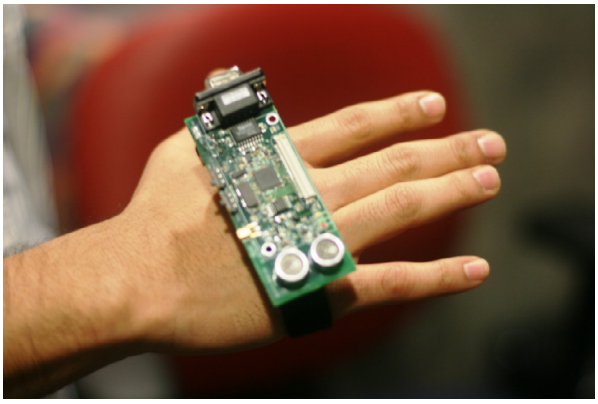


Figure 4.7: A cricket listener attached to a user's hand [125].

In an active mobile configuration, four listeners are mounted on the ceiling and the beacons are the mobile units attached to the user's hand. In such a setup, the beacons can broadcast signals at a higher frequency when there are less than four group members interacting with the table at the same time.

To identify touches, each touch point needs to be associated with a user identity. The easiest method to do this is to relate each touch point to the nearest cricket listener. Because the resolution of the cricket system is around 4 cm, this naïve approach for user ID association would only work well when the distance between the users' hands is large enough (at least 4 cm). For this reason, Siddalinga implemented an algorithm based on *linear extrapolation* [125]. The idea in this method is to find the listener location that best fits a touch point given the path the hand has traversed in the recent past. For higher accuracies, a curve fitting approach may be employed. However, when users' hands are moving fast, the wrong user can still be associated to the detected touches. This usually occurs because the update frequency of the randomization protocol is low as described in the previous paragraph. A higher frequency can be obtained when the randomization protocol of the cricket system is replaced by a carrier-sense-style channel-access protocol. Another disadvantage of this system is that all users as well as the ceiling above the interactive table need to be augmented with cricket components.

4.4.3 Tokens Using Infrared Light

In previous sections we described technologies using radio waves such as RFID and the cricket system to identify users on a multi-touch display. These technologies are relatively complex and many new electronic components need to be integrated in the interactive table because traditional optical-based touch tables are using other waves in the electromagnetic spectrum (i.e. infrared signals) to track touches (see Section 2.3.2). Reusing the infrared tracking components in tabletop systems to identify users will certainly simplify the design of a user identification mechanism.

Users interacting with a touch-table can be identified by wearing a special glove, wristband [79], ring [105], etc. with integrated infrared LEDs. A user's identity can be encoded using these LEDs with a spatial or temporal approach. The former uses a unique pattern of multiple infrared LEDs for each user whereas the latter technique uses a specific blinking pattern to determine the identity of a user. when a hand is above a multi-touch display, the system can locate these unique patterns in the same way as it locates touches. The system can now associate all touches in the direct vicinity of the authenticated location with the owner of this region.

Roth et al. [105] created a ring-like device that transmits a continuous pseudorandom

bit sequence in the form of infrared light pulses. The prototype consists of a single IR LED integrated in a circuit board which transmits a Manchester-encoded [6] bit sequence by means of on/off states of its LED. Because a temporal approach is used to encode the unique identifier, the table needs to track this pattern for a certain time before the system is 100% confident of the identity of the user. This delay depends on the length of the unique bit sequence and the data rate. The Nyquist-Shannon sampling theorem [120] bounds the maximum data rate at half the receiver's sampling frequency (i.e. the frame rate of the tracking camera in the tabletop setup). This is because the theorem states that the sampling frequency should be at least twice the highest frequency contained in the signal to avoid aliasing. Furthermore, the Manchester encoding requires the transmission of two bits to encode one data bit because a logical "1" is defined as a transition from low to high and a "0" is a transition from high to low. This further reduces the theoretical maximum rate to a fourth of the receiver's sampling frequency. For example, with a 60 Hz camera setup the maximum data rate will be 15 bits per second.

Because the technique for tracking touches and identifying users are both transmitting infrared light to the same receiver (i.e. the camera), the system cannot differentiate between a touch and a part of an authentication sequence. Therefore, another important property of bit sequence used in this system is the occurrence of a zero bit after at most two consecutive 1 bits. This allows the system to distinguish a touch from the IR LED based on time: a blob that persists for at least one and a half cycles must be a touch. Another problem is that the low data rate for user identification can make the system unusable because a user first needs to hold her hand above the surface for a specific amount of time to transmit the unique bit sequence before touching a private object. This will certainly be a problem when a long pseudorandom signal is used to increase security. For this reason, the test application in the initial experiment with the irRing [105] uses a security policy that attaches variable authentication probabilities to user interface elements. With the receipt of each bit of the authentication sequence, the probability that the hand of a specific user is above a tracked position increases. As a result, this user gets access to interface elements with higher authentication requirements as time progresses.

When a bit sequence is transmitted partially or entirely, the region underneath the user's hand is associated with her identity. The simplest method to associate touches with such an authenticated region is nearest neighbour allocation, shown in Figure 4.8-1. When using this technique, every touch point is allocated to the region next to it, while the distance is restricted to a maximum. This allows every touch in a circle around the user's hand to be identified and has a good rate of true positive allocations. On the other hand, it also allows touches in a large area where the hand could not reach, and thereby increases the potential of false positive allocations. To reduce these errors, algorithms such as linear extrapolation or

quadratic curve fitting (see Section 4.4.2) can be used to predict the movement of a hand and thereby exclude the regions of the circle where fingers of this hand can never reach. Because of the low-cost production of infrared LEDs and the potential to track their positions very precisely, a hardware approach with more LEDs per user can be used to reduce the circle around the user's hand to a circle segment. Meyer et al. [79] created a wristband augmented with two infrared LEDs. Each LED has a specific blinking pattern and the system knows which bit of the transmitted code should be received from which LED and uses this information to determine the orientation of the wristband. Hence the approximate position of the hand is known as shown in Figure 4.8-2. Corresponding touches are searched for in a smaller region, decreasing the possibility of false positive allocations.

In conclusion, tracking a hand above an interactive surface using infrared is very accurate, easy to construct and has a low production cost because the basic infrared tracking components are already integrated in traditional optical-based touch tables. On the other hand, wavelengths of infrared light are much shorter than radio waves which are used in the technologies discussed in sections 4.4.1 and 4.4.2. This may lead to occlusion of the signal, e.g. if the user curls his fingers for convenience and to avoid hand fatigue. Furthermore, an adversary may attempt to capture reflections of infrared light pulses of another user with the goal of requiring access to his private data.

4.4.4 Tracking Gloves

The technologies discussed in previous sections tried to identify touches by tracking the user's hand with a single token attached to the hand, wrist or a finger. These techniques allow every touch in a specific region around the token to be identified which can result in false positive allocations when users are interacting together in a small area. Previous work tried to minimize the regions where users can possibly touch by predicting the movement of the hand with linear extrapolation algorithms [125] or by estimating the orientation of the user's

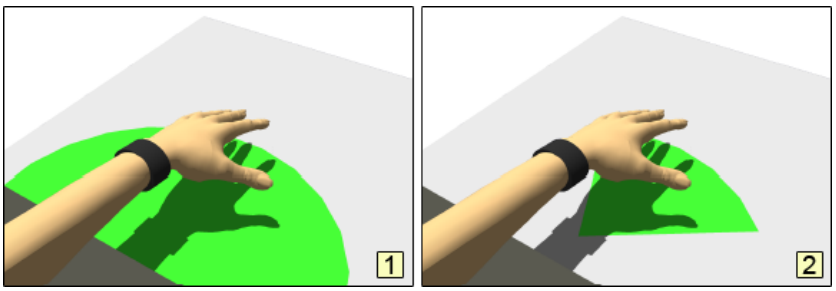


Figure 4.8: 1. Basic allocation through nearest neighbour, 2. advanced allocation using the wristband's orientation [79].

hand [79]. While these methods decrease the possibility of false positive allocations they do not provide a complete solution. For this reason, the association of touches to a single hand is only reliable when each finger can be tracked. This can be achieved by tracking the position of specialized gloves which each user wears.

Data gloves or wired gloves are widely used in many applications such as virtual reality, telerobotics, etc. Essentially, a dataglove (see figure 4.9) is a glove fitted with sensors to measure physical data such as the motions or the bending of fingers. There are several types of data gloves commercially available [135, 1]. The movements of the hand and each finger are interpreted by software and this data could be used to identify a touch. On the other hand, these gloves are delicate, expensive, not very accurate, unwieldy and an extensive calibration process is needed because the measured data is relative. Therefore, a more accurate alternative to wired gloves are computer vision techniques to track gloves.

Identifying touches can be done accurately and relatively easy when the gloves are tracked with a camera. For example, each user can wear a glove with a specific color. When a user touches the interactive table, the identification mechanism just needs to look up the color at this touch position in the captured images from the camera above the table to know the identity of the touch. In addition, research in computer vision has investigated in many techniques to track hand postures very accurately by using color patterns printed on gloves (see Figure 4.10) [146, 136, 7]. By interpreting a hand posture, the system can determine with which finger of a hand the user is touching the surface. However, to track these gloves, a camera mounted on top of the table is needed. These overhead cameras require a clear line of sight which can result in occlusion problems when users are bending over a tabletop display.

To track gloves with a camera inside the tabletop setup, a transparent surface can be used (see Section 2.3.2). Another option is to attach fiducial markers [62, 21] at the fingertips of the gloves as shown in Figure 4.11 [75]. These markers consist of unique patterns that can be



Figure 4.9: The P5 dataglove.

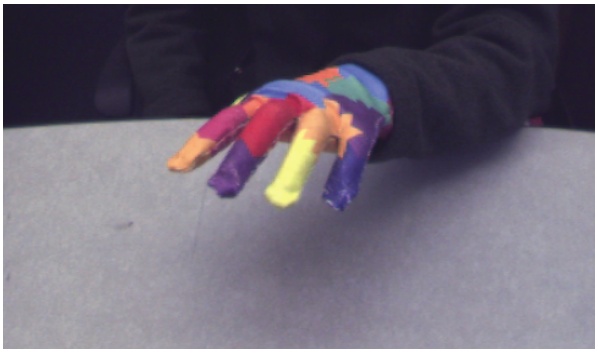


Figure 4.10: A glove with a color patterns to track postures of a hand very accurately [146].

captured by an IR-camera inside a DI or DSI setup. These tags do not only provide a unique identification method for each finger but they also reveal the orientation of a touch contact which the system can use. The described vision-based tracking algorithms that are currently available provide a very accurate touch identification mechanism. However, these techniques require more processing power than the previously described hardware solutions.

4.5 External Devices

In contrast to external tokens, digital devices provide specific functionalities for users. In this section we will consider if external devices such as mobile phones and digital pens can be used to identify users which are interacting with an interactive display. These devices are designed for interaction and some even provide the option (e.g. GPS, internet) to help users during the day, for this reason, people carry it with them throughout the entire day. When using such a device to identify users around an interactive touch display, no extra instrumentation is needed and users can start interacting with the display immediately. In addition, because the devices are personal they can provide a more reliable identification technique.



Figure 4.11: A fiduciary-tagged glove[75].

4.5.1 Mobile Phones

The usage of mobile devices for interaction with interactive display has been the focus of many research projects in the last decade [9]. Fitzmaurice was one of the first who designed and discussed applications in which a mobile device are used to interact with information displays [37]. He created a map on which the user can point their mobile phone to get additional information about that specific area.

A mobile phone is a general-purpose computer and can therefore be used to perform a variety of tasks in combination with a computer. For example, in the Pebbles project [85], mobile devices were used as an input device for interaction with a PC or whiteboard. Hardy et al. [47] used a mobile phone to select objects on a dynamic display and transfer them to the cellphone, etc. In this section we will primarily focus on the use of mobile phones to identify users which are interacting with a multi-touch display.

In many mobile phones additional sensors and technologies are integrated such as Bluetooth, Wi-Fi, camera's, accelerometers, flashlights, etc. Some phones are even available with buit-in Near Field Communication (NFC) chips. These NFC-chips are RFID tags with possibilities to operate in an active mode in which peer-to-peer communication is possible between two devices that support NFC. However, to support identification when such a devices touches the surface at a specific area, a mesh of tags needs to be integrated in the table as described in Section 4.4.1. For this reason, identifying touches using mobile phones with integrated NFC-chips has the same problems as using RFID tokens. In contrast, recent research [111, 116] has investigated in the use of mobile phones to identify touches on interactive displays using hardware that is currently available in optical-based touch displays.

Schöning et al. [116] created a technique to authenticate a user with a sub-region of a multi-touch wall. When a user wants to execute an action on the interactive display that requires authentication, this user needs to pass her identity to the display by touching the widget (e.g. a button) with her personal phone. The identification mechanism uses the accelerometer of the mobile phone to detect a touch event. When such an event occurs, the phone sends the user ID via Bluetooth to the interactive display. The multi-touch display detects this touch event and receives the user ID. The system can now relate the location information of the interactive display with the received identifier by synchronizing these events. If both events do not occur within a short time window or if one is missing or detected to late, the protocol is aborted. The detection of more than one phone touch or the receipt of more than one identifier in a single time window is considered as a collision. In this case, the server asks one of the devices that has sent an identifier to repeat the procedure. If a unique association of position and user ID is found, the server looks up the authorization data for the object at the respective position and checks whether the user is allowed to perform the action. If so, a

positive response is sent via Bluetooth and the action is executed.

Because performing all touches on a display with a mobile phone is not intuitive and a full multi-touch experience (for example multiple input points and drag-and-drop operations) is not achievable, only actions that require the identity of the user need to be executed using a mobile phone. As a result, actions can still be performed by touching the display with a finger. This requires methods for the system to discriminate between touches with fingers and mobile phones. Schöning et al. [116] implemented a technique using the flashlight of the mobile phone to distinguish finger touches from phone touches. In this technique, a light flash is generated simultaneously with the transmission of the user ID when a user touches the surface with her mobile phone. The multi-touch display detects this light spot at a certain position using a second camera installed behind the display. By relating the captured images of the the additional camera with the images of the camera to track touches, the system can discriminate between touches with fingers and touches using mobile phones. The entire identification process is summarized in Figure 4.12. When using this method, a problem can occur when a device generates an input request, but the corresponding light flash is not detected by the surface. This could happen if a touch event is triggered while the mobile device does not face the surface. In this case, an attacker could produce a light flash on some random display region. To solve this problem, a second light flash could be produced after a random delay whose duration is sent in the identification message. The attacker would then have to guess the right delay and produce the second flash at exactly the right moment. Another technique based on the work of Cuypers et al. [25] is to send a challenge by projecting a pattern on the surface next to the detected light spot. The camera of a mobile device can then detect the challenge and send it back to the server.

PhoneTouch [111] uses a similar technique to identify users interacting with a table. In addition to transferring the user's identity when touching the surface, this system also supports the transmission of personal photos to the surface for sharing purposes. In contrast to the technique with the flash light, they observed variations in blob sizes to distinguish a finger touch from a PhoneTouch. In an experiment they observed that a PhoneTouch results in a substantially smaller blob than finger touches even with different phone models and varying angles. Furthermore, they observed a high variance of contact size in the first frames when a finger touches the surface. This can be explained by an approaching phase when soft objects like fingers are used: the touch area is small and grows until full contact is made. When these observations are used to discriminate between a finger touch and a PhoneTouch, no additional camera is needed. On the other hand, the reliability of this technique is affected when phones are used with soft protective skins, or if users with smaller fingers (e.g. children) are using the system.

When users can use their own mobile phone to interact with the table, they don't need

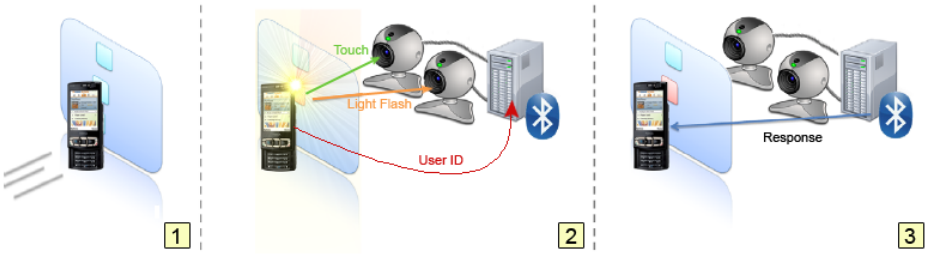


Figure 4.12: Authentication scheme of users interacting with a multi-touch display: 1. The user touches the wall with the phone. 2. The mobile phone sends a light flash to indicate the region the user wants to interact with and at the same time sends his identifier to the system. .3 The system combines the touch data, light flash and identifier to check the access rights of this user and responds to the device [116].

to be instrumented with additional tokens before interacting with the surface. On the other hand, many mobile phones with different technologies exist (e.g. camera, flash light, bluetooth, accelerometer, NFC-chip, etc). A touch table can never support all these technologies, as a result, specific requirements for devices are needed to enable interaction with the display.

4.5.2 Pen Computing

Touching with a mobile phone on a multi-touch display is not always as accurate and natural because these devices are not designed for this purpose. A pointing device such as a digital pen can therefore be more suitable. A digital pen looks like a normal pen but has electronic components integrated such as buttons, (IR)-leds, a laser, wireless communication modules, cameras, etc. These pens are generally used for tracking handwritings. On the other hand, recent research in this area [11, 42, 17] describes some possibilities to identify users when interacting with digital pens on an interactive surface.

The uPen [11] is a pointing device that can be used to interact with a digital table or with the system from a distance to give for example a presentation. The pen has an integrated laser emitter, a radio frequency communication module and tree push buttons to emulate traditional mouse operations and to turn the pen *on* or *off* (see figture 4.13). When pushing the left button, a laser beam is emitted. A camera behind or in front of a projection screen tracks this laser-pointer and as a result the system can perform the necessary operations, for example, clicking on a hyperlink in a presentation. When working with the uPen on a digital surface, the laser is turned on when the pen makes contact with the screen. The pen can sense this contact because a contact switch is integrated in the tip of the pen. The system can track this laser-pointer with a camera installed behind the surface. At the same time, the system receives the uPen's ID which the radio frequency module of the pen has

emitted. Because the matching of the laser-pointer and the received identifier is exclusively based on time synchronization, two users cannot start working exactly at the same moment. Furthermore, when the pen makes a movement over the surface, each laser spot needs to be related to the stroke to associate the entire stroke to the user's identity. In the uPen prototype this is done using a Kalman filter [147]. This algorithm calculates valid regions for the next point of a stroke based on its previous position and velocity. When a laser spot is located in such a region, the point is added to the specific stroke. Bi et al. [11] noticed that the algorithm only correctly associate each laser spot with the corresponding uPen when the user moves the pen less then 1cm per second. As a result, this technique is not suitable for highly interactive systems.

Instead of using a laser transmitter, the InfrActables [42] system uses a digital pen with integrated infrared LEDs. A traditional optical-based touch table can use the location and time multiplexing techniques described in Section 4.4.3 to localize and identify each pen.

Brandl et al. [17] and Leitner et al. [69] described a completely different technique to identify users interacting with a digital pen on a multi-touch screen. In these research projects an Anoto pen [71] tracks the position of the pen on the surface and sends these coordinates and the pen ID to the system. An Anoto pen is a ballpoint-pen with an embedded infrared camera that tracks the pen movements. The pen has to be used on a specially printed 600 dpi paper with a pattern of small dots as shown in Figure 4.14. The Anoto pattern needs to be printed on a special rear-projection foil to fulfill two functions: projection screen and pattern for the digital pen (see Figure 4.15). Once the user touches the screen with the pen, the camera tracks the underlying pattern. It can then derive its absolute coordinates on the pattern and send them to the touch table over Bluetooth.

In contrast to these approaches, Mohamed et al. [81] analyzed gestures from pen input to identify users. Their recognition algorithm is able to tell on which side of the table the user

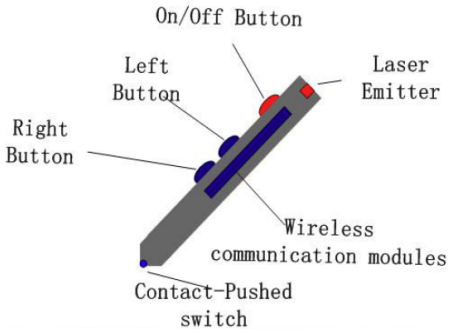


Figure 4.13: The design of the uPen [11].

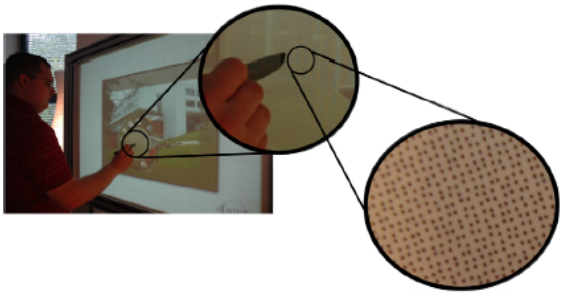


Figure 4.14: The Anoto pattern printed on a special rear-projection foil[17].

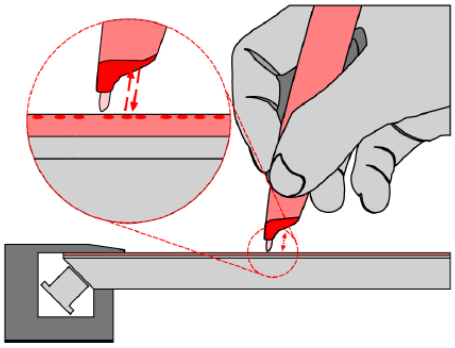


Figure 4.15: The Anoto pen used on a optical-based touchscreen to identify the users [69].

is located, only by pressure and angle information of the pen and without the need of extra hardware.

In conclusion, digital pens can be used to identify users working together around an interactive table. These pens provide more natural and precise interaction possibilities compared to mobile phones. On the other hand, using pen input instead of direct touch result in a slower and less expressive interaction technique because direct touch allows the user to manipulate data very fast with multiple hands and fingers.

4.6 Bare Touch Identification

Identifying touches on an interactive table with external devices or tokens requires instrumentation of the users. These extra devices can be very unwieldy and some techniques (see Section 4.5) even require the user to touch the system with these devices which can result in an unnatural interaction. In this section we will therefore examine if it is possible to identify touches without any instrumentation of the users.

4.6.1 Tracking Users

Tracking hands and arms above an interactive surface does not only allow interaction with the system without touching the surface, this information can also be useful to estimate the position of the user around the table. When this position is known, the system can relate each finger touch to the right identity. We will describe four techniques to track interactions above the surface: *Shadow Tracking*, using an *RGB camera*, using a *depth camera* and using a *thermal camera*.

Shadow Tracking

Many research projects [150, 138] exist in which shadow tracking is used to provide touch capabilities on an ordinary surface. On the other hand, Echtler et al. [33] developed a multi-touch table in which FTIR and front DI (see Section 2.3.2) is combined to track users' hands above the surface. This information can then be used to relate touches to users' identities.

In front diffused illumination setups, infrared light is illuminated from above the surface. When a hand touches or hovers the surface, this infrared light cannot reach the camera inside the table in this region. As a result, shadows are created underneath the hand and arm of the user which the infrared camera can track. Because the same camera tracks the infrared light from above the table and infrared light from the FTIR technique, the illumination of the table is alternated in odd and even frames. Two consecutive frames can thus be used to track finger

contacts as well as hand shapes. We will refer to these two images as contact and shadow image respectively (see Figure 4.16). For each blob in the contact image, a circular area with increasing radius around the centroid is searched in the corresponding shadow image. When a shadow blob is found, its identifier is assigned to the respective contact spot, thereby creating a relationship between shadows and surface contacts.

When the system has related touch contacts to hands, it can estimate on which side of the table the user is standing. This can be done by iterating over the points in the detected contour to find the points with minimum and maximum Y-axis values. When a user stands in front of the table and touches the surface, hand contours with very low and middle-ranged Y-axis values will be detected. On the other hand, when a user is standing on the opposite side of the table, hand contours with very high and middle-ranged Y-axis values will be detected. This technique can be easily extended to recognize users on the remaining two sides by doing comparisons with X-axis values. However, when multiple users are standing on the same side of the table, it cannot be unambiguously determined from which user a captured hand/arm is. Furthermore, when a user is working near a corner of the table, one of her arms can intersect a side of the table which is assigned to another user. For this reason, research need to investigate in complexer models to estimate the position of a user based on arm postures [30].

In contrast to the previous described shadow tracking technique, a single infrared light source integrated in a DI or DSI setup can also be used to illuminate both touch contacts and hands above the surface [112]. In this approach, a secondary infrared camera is mounted above the surface to track hand shadows, these are the regions where no infrared light is visible. When using this approach, no alternation between infrared sources is needed and as a result, the frame rate is not reduced because only one illumination source is needed. On the other hand, this type of shadow tracking cannot be used in combination with the FTIR technique because there is no illumination of infrared light above the table. Furthermore,

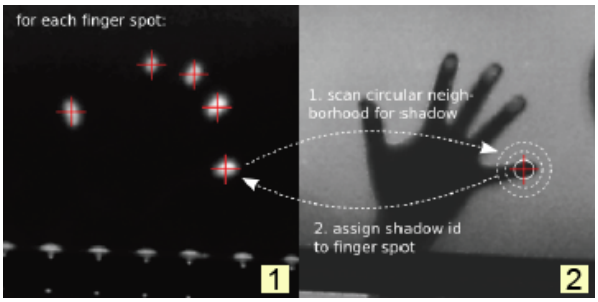


Figure 4.16: 1. An odd frame in which the finger contacts are visible. 2. An even frame in which hand contours are visible [33].

in both described techniques it is possible to unintentionally create additional shadows with the head or upper body when bending over the table which causes undesirable merging of shadows.

In this approach, the assignment of captured hands to a user's identity is based on estimations, therefore, the system can be very error-prone especially when users are moving around the table. This touch identification method can possibly be made more reliable when proximity sensors are integrated along the sides of a multi-touch table. Walther-Franks et al. [40] proposed the integration of infrared reflection sensors in multi-touch tables (see Figure 4.17). An infrared reflection sensor is an electronic circuit in which an infrared LED and phototransistor is integrated. This sensor scans a very limited area and reports a rough distance value that is influenced by the reflective properties of the surface of the object in front of it. To cover the whole space around a display, many sensors need to be integrated around the table at the same hight. A simple model can map each sensor to a specific location. The result is a proximity map that shows which area is occupied around the table. Research needs to investigate if estimates about a user's position around an interactive table are more reliable when places occupied around the table are known.

Using an RGB Camera

A second technique to track hand movements above a multi-touch screen is using RGB camera in front/above of the display. Dohse et al. [30] experimented with this technique and mounted a visible spectrum camera above a traditional FTIR multi-touch table to track hands via skin color segmentation techniques [88, 134]. Hands and arms are detected by creating a binary image of the pixels which fall within the appropriate RGB ranges. Data from both the overhead skin tracking and infrared touch tracking is handled simultaneously by the system. The system first relates the touch contacts to their corresponding hands and then estimates

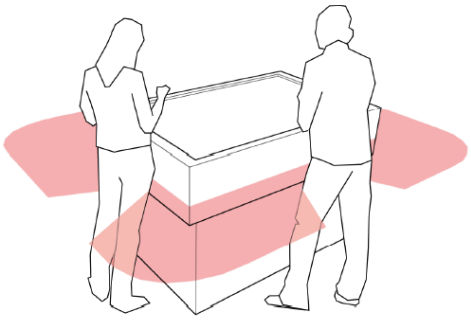


Figure 4.17: Integrating proximity sensors in multi-touch screens to know which places are occupied around the table [40].

the position of this user using the same algorithm as described in the previous section.

When mounting a camera above a multi-touch display, a clear line of sight is needed to track hands. This is not always assured because users can bend over the table and thereby obscuring hands. However, a camera can be mounted above the table at any hight which has the advantage of capturing not only the touch table but also the space near the display. This extra information can significantly improve the algorithm to identify touches because the real position of a user can be tracked. In addition, the system can then determine with which hand the user is interacting with the table.

The captured images of an RGB camera and shadow tracking (see Section 4.6.1) provide a very bright view of hands above the surface. For this reason, a possible promising next step is to apply a robust hand model to these captured images in order to determine which finger corresponds to which touch point. Several options are presented in [20] and [70]. This offers the possibility to transfer well-known interface metaphors like left and right mouse click to multi-touch screens by assigning different actions to, e.g, thumb and index finger [76]. Of course a thorough user study is needed to evaluate the usability of such an interface.

Using a Depth Camera

In contrast to the use of a traditional camera to track hands above a surface, a depth camera can be used to track hand movements and even touches as described in Section 2.3.1. However, the touch performance when using a depth camera for tracking touches can never reach that of a traditional optical multi-touch table (see Section 2.3.2) because the depth precision decreases as the distance to the camera increases. On the other hand, depth cameras can provide more information about the user doing the touching as shown in Figure 4.18. By combining optical-based multi-touch techniques (e.g. FTIR, DI or DSI) with a depth camera, the system can track the position of each user interacting with the display and relate each touch to a user's identity. Furthermore the system can track with which hand the user is interacting.

Using a Thermal Camera

A fourth technique to track hands in above a multi-touch display is using a digital thermal imaging camera. Thermal imaging is a technique for passively conducting a high resolution heat map of objects appearing in a scene without using an external illumination source. This is accomplished by measuring the quantity of far-infrared (F-IR) radiation emitted by any object. Objects that humans interact with on a daily basis are near room temperature and radiate mostly in the F-IR spectrum.



Figure 4.18: A depth camera can provide positional information of users around an interactive surface [151].

Larson et al. [67] investigated in the use of a thermal camera to track touches on arbitrary surfaces. They noticed that hovering a surface can be differentiated from a touch because of the short-lived heat transfer from the hand to the surface when contact is made. Furthermore, they indicated the possibilities of detecting the level of pressure at a touch contact which is usually simulated by observing variety in the shape size of a touch contact on traditional optical-based touch technologies [27]. However, we are more interested in the use of thermal imaging to identify users around an interactive surface.

To identify touches with a thermal camera, positional information in the captured heat map can be used to relate touches to user identities by using the techniques described in previous sections. However, thermal imaging has largely been used for military and surveillance applications [155], where the heat signature produced by the human body is used to track individuals in arbitrary environments and conditions. Thermal imaging has also been used in hospitals and border crossing to identify individuals with a fever within a crowd of people using face temperatures [143]. These people can be tracked because they have different thermal gradients on their body. Although these gradients can change over long time intervals (e.g, coming inside from a winter's day), Larson et al. [67] think that the overall heat signature remains unchanged for the duration of an interactive session at a display. For this reason, we believe that several users can be uniquely identified within an interactive session based on their heat signature (see Figure 4.19).

A captured heat map does not only provide possibilities to differentiate between users, the underlying technology has also many good implications for the applicability of the interactive system. Tracking a user with a thermal camera is independent of the illumination and are far less susceptible to changes in light intensity than traditional RGB or depth cameras. As a result, thermal sensing can work in complete darkness and direct sunlight. Furthermore, this technology provides options for integrating pressure sensitivity and a hover state in a multi-

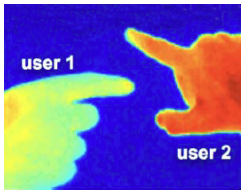


Figure 4.19: Thermal heatwave of two users interacting on a surface [67].

touch display. On the other hand, a thermal camera is very costly, the cheapest cameras cost just under 3000 euro [93].

4.6.2 Biometric Touch Identification

In the previous section we discussed how the tracking of hands/arms and users can reveal the identity of touch points. However, we have seen that this technique becomes complex when users are working near the corners of the table and intersect multiple edges of the screen with their arms. Furthermore, the presented methods are not reliable when multiple users are standing on the same side of the display or when arms are intersecting each other. For these reason, we will examine if biometric feature extraction techniques such as *hand geometry feature extraction*, *face recognition*, *touch contact feature extraction* and *fingerprint identification* can be used to provide a more reliable and usable touch identification technique.

Hand Geometry Feature Extraction

Without additional instrumentation of the interactive table or users, DI and DSI tabletop setups are able to capture a clear view of the contour of a flat hand which is entirely in contact with the surface. In this section we will discuss if a multi-touch system can extract features from such a hand contour and relate this information to a single user. Researchers started extracting features out of hand contours by designing a prototype hand geometry based identity system [58]. In this prototype, a user needs to place her hand in a specific arrangement as shown in Figure 4.20. Pins are used as control points for appropriate placement of the hand. The user's hand is observed by two cameras from different angles as shown in Figure 4.20. Features are measured along 16 dimensions such as length and width of fingers, aspect ratio of the hand palm and fingers, thickness of the hand, etc. Although an experiment turned out that these extracted features provided enough unique information to discriminate hands from 50 individuals, this system is not invariant to rotation and translation of hands.

In contrast to the previous described feature extraction technique, Boreki et al. [15] presented a technique in which hand geometry features where extracted independent of the

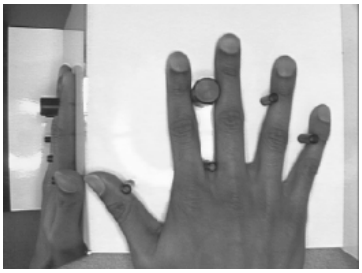


Figure 4.20: Hand geometry sensing device [58].

position of the hand. A traditional flatbed scanner was used to capture a flat hand with an arbitrary position and orientation. This image is processed in two steps: extraction of the contours and extraction of features. In the first step, a thresholding method is used which is responsible for counting the frequency of pixels of each color and choosing which one belongs to the background and to the hand. The second step starts with a segmentation of fingers from the hand image by observing the slope differences in the contour. This analyses of the curvature profile is done using the DOS (difference of slopes) technique [87]. This is a method which consists basically in roaming the hand contour with two vectors of the same size and calculating the angle formed by these two vectors. High changes in direction indicate finger tips and valley points which mark the finger locations. In the next step, the length and the mean width of each finger is extracted from the contour. The length of each finger can be found by drawing an imaginary triangle among tree points. The main line of the triangle is then divided into twenty new points. At these points widths of the finger will be sampled (see Figure 4.21). The average of all width samples will result in the finger width. The average length and width of all fingers in one hand is then used as an individual biocode for the user. Boreki et al. [15] observed that for a candidate biocode of a hand contour to be accepted, it should have an average finger length and width that comprises two times the standard deviation of this user's training samples.

HandsDown [112] is a system in which the described hand geometry feature extraction

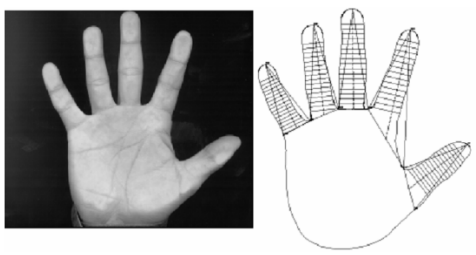


Figure 4.21: Hand contour feature extraction [15].

techniques are used to identify users working on a DSI multi-touch setup based on the captured hand contours when users put down their flat hands on the surface. Out of the hand silhouettes, the length of the fingers, the widths of each finger at five equidistant points, the mean width, the palm width as well as three distances between different finger valley points are extracted as characteristic features for a user. The algorithm ensures that neither the hand's location on the surface nor its rotation is restricted. It is important to note that HandsDown is not a general user identification technique in a sense where every touch can immediately be associated to a user hand because the user can only be identified when a HandsDown event occurs. However, when equipping the system with an overhead mounted camera (see Section 4.6.1), all subsequent finger interactions originating from an already identified hand can be associated to the same user.

Although the properties of hand contours vary greatly among different people, it is yet insufficient to claim that any hand contour is exclusively unique from the rest of the entire world. Instead, HandsDown enables robust identification for small user groups. Furthermore, the described technique cannot be used on an FTIR setup because the space above the surface is not illuminated with infrared light, as a result, the system cannot capture a clear view of hand contours.

Face Recognition

When mounting a camera on a multi-touch table or above an interactive wall facing the user, it is possible to track the heads of users that are interacting with the display [66]. Research [154] investigated in many algorithms to extract features out of these captured images in order to identify a user. In contrast, research has not yet explored the usefulness of this technique to improve the identification of users around an interactive display. Unlike the hand geometry feature extraction approach described in the previous section, face recognition is a more passive, non-intrusive technique to verify a persons identity in a user-friendly way without having to interrupt a user's activity. However, face recognition cannot be used as a general user identification technique because a face cannot unambiguously be associated to a single touch when multiple users are interacting with the system at the same time. To relate a touch to a recognized face, interactions above the surface need to be observed (see Section 4.6.1). By relating for example touch events, hand/arm movements and recognized faces, a touch identification mechanism could be created.

The reliability of face recognition has been getting very good at full frontal faces and 20 degrees off. Other conditions where face recognition does not work well include poor lighting, sunglasses, long hair, low resolution images, etc. Furthermore, the algorithms are less effective if facial expressions (e.g. mouth expressions) vary [41].

In a horizontal multi-touch setup, the camera which captures the users' faces can also be mounted behind the surface. In this kind of setups, transparent projection material (see Section 2.3.2) is needed to see beyond the display surface. To integrate the face tracking camera in a horizontal setup, additional mirrors inside the table are needed to capture the faces of users that are standing on one side of the table [60]. However, tracking the faces of users standing on all four sides of a multi-touch tabletop setup is not possible with a single camera. Because face tracking algorithms can also work in the infrared spectrum [65], the infrared camera behind an optical-based multi-touch wall can possibly be reused to track users' faces. When identifying faces in the infrared spectrum, face detection, location and segmentation is easier. Furthermore, the system will be nearly invariant to illumination changes, facial expressions and can even work in total darkness [65].

Touch Contact Feature Extraction

In the previous section we described a technique to identify touches by tracking the faces of the users in front of the display. However, this technique requires the combination of many captured sources, such as touches, above the table interaction and face recognition which can all introduce errors. For this reason, touch identification would be more easier and reliable if information can be extracted directly from the touch contact.

First of all we need to investigate which information can be extracted from a finger contact captured by a camera inside a traditional optical-based touch screen (see Section 2.3.2). Wang et al. [144, 145] created an algorithm to extract the orientation of a finger based on the shape of an *oblique touch* contact. An oblique touch occurs when the finger lands on the surface at an oblique angle as shown in Figure 4.22-1. The contact shape can then be fitted into a perfect ellipse and the angle θ between the major axis distance of the ellipse and the x-axis can be calculated (see Figure 4.23). In contrast, a vertical touch occurs when the finger is directly pointing downward, toward the surface as shown in Figure 4.22-2. In this case, the contact shape is circular and does not provide usable orientation information.

Wang et al. hypothesized that oblique touches are more likely to happen when people

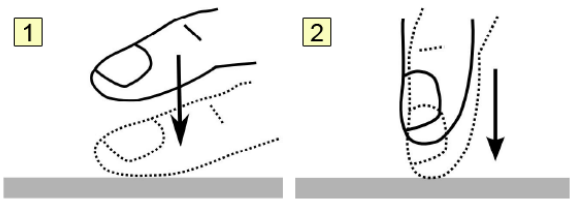


Figure 4.22: Two ways of finger touch. (1) oblique touch. (2) vertical touch. [144]

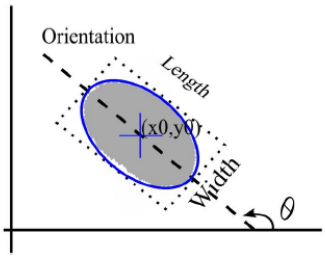


Figure 4.23: Finger contact region fitted to an ellipse [144]

are interacting with multi-touch tables because they considered common practices of humans when handling physical objects, as well as the necessity to accommodate long fingernails by some people. As a result, they used the orientation of the elliptical contact to determine the orientation of a user's finger. However, the true direction of the finger can be either θ or $180°+\theta$. This ambiguity can be solved by examining the deformation of the finger's contact region in the process of it landing on the surface. Figure 4.24 shows the contact region across time when a finger is landing. It is apparent that the center point of the finger contact moves inward, towards the user's palm. By considering this deformation, the system can roughly infer on which side the user's palm lies, and in turn which direction the finger is pointing to.

Finger orientation can only provide a very rough estimate of the users position around an interactive table. Therefore, it is not very reliable to identify touches based on this information only. However, the orientation of fingers can be exploited to provide new widgets and interaction possibilities as we will discuss in Chapter 5. Furthermore, researchers explored techniques to use the orientation of a finger contact to determine if multiple touches are originated from the same hand.

Many of the previously described technologies to identify users around an interactive display are based on tracking the users' hands and assigning finger touches to these hands based on distance measurements, linear exploration techniques (see Section 4.4.3), curve fitting al-

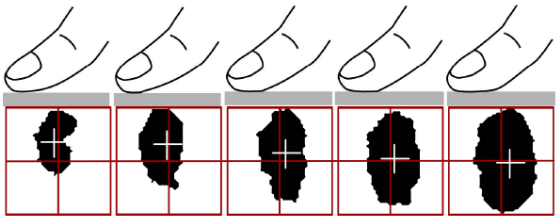


Figure 4.24: Finger contact deformation over time. The crosshair shows the center of the contact region. [144]

gorithms, estimates about the orientation of the hand (see Section 4.4.2) or computer vision techniques (see Section 4.6.1). Here, we will describe a more advanced technique that does not require the instrumentation of traditional multi-touch tables or users with extra hardware. Although the association of finger touches to hands are required in many user identification methods, this technique can also be used as a standalone mechanism for enriching interaction possibilities.

Dang et al. [26] presented an approach to exploit information on the anatomy of human hands and finger orientation to map touches onto hands. They analyzed all possible finger arrangements of human hands an created a model in which all these postures are covered. Basically the algorithm uses two properties of finger touches to achieve correct assignments: the position and the orientation. The positional information can be retrieved from the touch tracking techniques integrated in a traditional interactive display. The orientation of a blob on the other hand, can be determined using the algorithm described in the previous paragraphs.

For each captured frame from the touch tracking camera integrated in a multi-touch display, the system evaluates the touch points pairwise, to decide if they originate from the same or different hands. This evaluation is based on 4 conditions:

1. Distance condition: The distance d (see Figure 4.25) between two touch contacts is treated as a first indicator for being from the same hand or not. When this distance exceeds an empirically determined maximum distance D_{max} (in the original implementation D_{max} has a value of 26,8cm), the considered two touch contacts are from different hands. The touch points are considered as adjacent fingers if the distance between these contacts is less then an empirically determined distance D_{adj} (in the original implementation D_{adj} has a value of 8,9 cm) and the interior angle θ (see Figure 4.25) is smaller than an empirically defined angle θ_{adj} (in the original implementation θ_{adj} has a value of 45°). Figure 4.25 shows a situation in which two touch points are captured that are not originated from adjacent fingers and the distance between them is smaller then D_{max} ($D_{adj} < d < D_{max}$). As a result, the system will categorize this case as two touch points originated from the same hand.

2. Intersection condition: This condition ensures that the intersection point of touch points originated from the same hand is located behind both touch contacts by considering the gradient vectors v_1 and v_2.

3. Distance relationship condition: Condition 1 is sufficient to recognize adjacent finger touches. However, condition 1 and 2 are not sufficient because the situation shown in Figure 4.25 can still be formed by fingers of two different hands. Therefore, limitations are imposed based on the properties of the limbs and joins of human hands by considering the distance between the two touch contacts and their intersection point defined as

the normal distance n. The farther two touch contacts are, the higher the interior angle between their orientated lines is, and the shorter the corresponding normal distance becomes. When the normal distance is smaller than an empirically determined value N_{max} (in the original implementation N_{max} has a value of 35,7cm), the system categorizes the two touch points as touches from the same hand. When the normal distance is higher than N_{max} the touch contacts are from different hands or are originated from adjacent fingers.

The two situations shown in Figure 4.26 are categorized as touch points from different hands by applying condition 2. However, the thumb in combination with another finger of the same hand is able to create such an arrangement when for example performing a grab gesture (Figure 4.26-1) or a pump gesture (see Figure 4.26-2) [127]. To handle these cases, the algorithm is extended to support temporal information because when performing these gestures, the system usually assigns the initial arrangement of fingers ready to perform these gestures correctly to the same hand.

In an experiment, Dang et al. noticed that even without perfect accuracy of this hand mapping algorithm, the described technique is sufficient to distinguish hands for two-handed input on interactive tabletops. False recognition happened mostly because of the thumb being more flexible as for the constellation with fingers. Enhancing the algorithm with respect to the thumb and with the support of vertical touches can significantly improve this hand distinction approach. Furthermore, research needs to investigate in the recognition rates when multiple users interact in parallel on a multi-touch screen.

Fingerprint Identification

In the previous section we described techniques to exploit touch contact features. We concluded that the orientation of a touch contact was not enough to provide a reliable touch identification mechanism. However, when observing a finger in more detail, we can see unique markers for every person: *fingerprints*. Modern fingerprint matching techniques are used for personal identification for a very long time [57]. In the early 20th century, fingerprint identification was formally accepted as a valid personal identification method by law enforcements agencies and became a standard procedure in forensics [68]. Fingerprints are a unique marker for a person, even for an identical twin. For this reason, fingerprint analysis is mostly used for crime investigation and security. There are two major types of fingerprint scanners: capacitance scanners and optical scanners.

Capacitance scanners are made up of one or more chips containing an array of tiny cells. Each cell includes two conductor plates and are smaller then the widths of one ridge on a

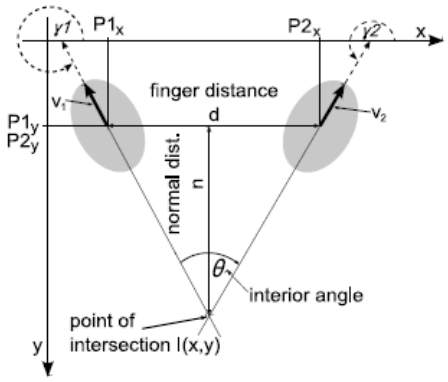


Figure 4.25: Two touch points originated from the same hand [26].

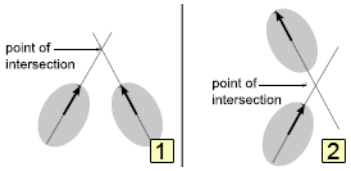


Figure 4.26: Two situations in which both touch contacts are originated from the same hand: 1. Performing a grap gesture. 2. Performing a pump gesture. [26].

fingerprint. The two conductor plates form a basic capacitor, an electrical component that can store up a charge. A finger touching the surface acts as a third capacitor plate, separated from the other plates by insulting layers in the cell structure and in the case of fingerprint valleys, a pocket of air. When a finger touches the screen, the capacitance of the capacitors in the underlying cells will change. As a result, the capacitor in a cell under a ridge of a fingerprint will have a greater capacitance than the capacitor in a cell under a valley of a fingerprint. The scanner processor can now read the output voltages out of each cell in the sensor array and determines whether the value is characteristic for a ridge or a valley. Finally, the processor can put together an overall picture of the fingerprint [48]. This type of fingerprint sensors are only produced as very small devices on which just a single finger can be placed. On the other hand, One can imagine the integration of this technique in a large surface on which is projected from above to create an interactive display similar to the DiamondTouch (see Section 2.2.2).

The second type of fingerprint scanners also generates an image of ridges and valleys that make up a fingerprint. But instead of sensing electrical current, optical scanners are based on the optical frustrated total internal reflection (FTIR) concept. This technique is similar to the FTIR touch tracking technique in a multi-touch screen. An array of light-emitting diodes in

the device illuminate the surface. When a user places her finger on the surface of the scanner, light under the ridges in a finger is scattered while light incidenting on the valleys suffers total internal reflection. The reflected light is captured by a charge coupled device (CCD). A CCD is an array of light-sensitive diodes called photosites and is also used in digital cameras. Each photosite records a pixel, a tiny dot representing the light that hit that spot. The CCD then generates an electrical signal in response to the light photons. Afterwards, the captured image is inverted, dark areas represent the ridges of the finger whereas lighter areas represent the valleys between the ridges [57]. This technique is used extensively in guardian fingerprint scanners which are even used in research projects [51]. Furthermore, researchers have tried to integrate this technique in other devices. For example, Sharp, one of the manufacturers of the iPhone sceen, was working on a new display in which a photosite is integrated in each pixel of the display. As a result the multi-touch display would be able to recognize fingerprints in every region of the screen [141].

To integrate an optical fingerprint reader in an optical-based multi-touch screen Schöning et al. [115] proposed a technique to mount a steerable high-resolution camera under the interactive surface. When using transparent projection material (see Section 2.3.2) in a DI or DSI multi-touch setup, the high resolution camera can focus on the potential touch region after the infrared camera captures a hand that is performing a touch action. Schöning et al. experimented with this technique on a large wall size display which resulted in lots of "ghost" fingerprints because the steerable camera was to slow to cover his large display. On the other hand, implementing this technique on a smaller table can perhaps work but has not been investigated yet

When a fingerprint is captured by the scanner, the system compares specific features of the fingerprint with a set of previously captured fingerprints. These features are generally known as *minutiae* and are classified in three levels as shown in Figure 4.27. Typical algorithms concentrate on points where ridge lines end or where one ridge splits in two (called *bifurcations*). When 100% reliability is needed (e.g. in law enforcements), level 3 feature extraction is required in which ridge contours are analyzed. On the other hand, level 3 feature extraction requires a camera of 1000 ppi. Furthermore, executing the feature extraction algorithm will be time consuming and would require lots of processing power [100]. However, for touch identification purposes, a 300 ppi camera would be sufficient to extract level 1 and 2 feature [100]. Results of experiments indicate that if two prints have three ridge endings and two bifurcations, forming the same shape, there is a high likelihood that they are from the same print [57].

Touch identification based on the analysis of fingerprints introduces many new possibilities to enrich multi-touch interaction in a collaborative setting. For example, a single touch contact can be related to the exact finger, hand and user. However when multiple fingers need to be

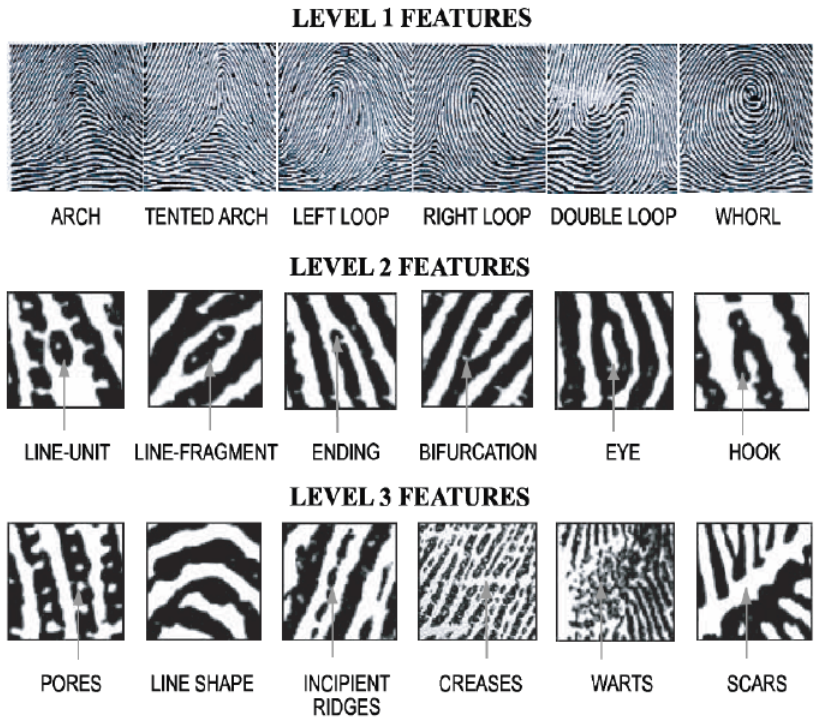


Figure 4.27: Levels of feature extraction in the analysis of fingerprints [100].

supported, users first need to touch the surface with all fingers in different pitch angles in a registration fase before the system can relate all possible touches to the right users [51]. Furthermore, the identification process can be optimized when the system tracks the hands of users (see Section 4.6.1). In this situation, the fingerprint identification algorithm needs to be executed only when a new hand approaches the display.

4.7 Conclusion

In this chapter, we discussed many techniques to support touch identification. We have seen that some techniques are suitable for revealing the true identity of users while others could barely discriminate between users but provided other possibilities, such as estimating the position of users around the table, discriminating between hands and fingers of a user, etc. We classified all these techniques based on the amount of extra instrumentation that is required. Identifying users using external tokens or devices (see sections 4.4 and 4.5) required extra instrumentation of the users. On the other hand, techniques such as camera tracking (see Section 4.6.1) required the instrumentation of the environment around or above the touch table. Only biometric techniques (see Section 4.6.2) or tracking techniques with cameras inside the table provided identity information of touches while preserving a self-contained system in

which all hardware is integrated in the display. This will severely inhibit the mobility of these displays. However, this increase in mobility can come at the cost of reliability of the system. Therefore we think that each technology has its own strength and weaknesses and research in this area is needed to create more superior identification methods.

Chapter 5

Multi-touch Interaction Patterns

5.1 Introduction

In chapters 2 and 4, we provided a large overview of many technologies that can be used in touch interfaces. Furthermore, we provided important concepts for supporting collaboration on tabletop displays in Chapter 3. However, we have not yet investigated how new applications and interaction techniques can be created using these new technologies with respect to the collaboration concepts of tabletop groupware. In this chapter we will therefore provide a catalog of patterns with new interaction techniques that can be created.

Of course, many other applications than the ones described in this chapter can be developed, especially when new and probably more reliable techniques emerge. However, with these multi-touch interaction patterns we try to give an overview of what research in tabletop groupware and touch identification has already made possible. First of all, we will explain the structure of the pattern language. Then, information about the format of each pattern will be provided.

5.2 Structure

The structure of our catalog is listed in table 5.1. We classified all new applications and interaction techniques in thirteen patterns and categorized these interaction patterns in three main categories:

- Providing identity-awareness
- Support for rotating user interfaces
- Solving related problems in tabletop research

Providing identity-awareness
(01) Privileged Access **
(02) Customizing Functionalities **
(03) Cooperative operations **
(04) One and Two Handed Gestural Interaction
(05) Customizing Content *
(06) Holding Commands On Fingers
Support for rotating user interfaces
(07) Automatic Content Orientation *
(08) Orientation-Sensitive Widgets
(09) Orientation Invariant Input
(10) Occlusion Problems
Solving related problems in tabletop research
(11) Providing Hover states
(12) Optimizing Touch Accuracy
(13) Extending Reachability

Table 5.1: The structure of the pattern language.

Many techniques described in this work tried to reveal the identity users working around an interactive table. Therefore a large number of applications are related to the support of *identity-awareness* in tabletop systems. A couple of techniques described in Chapter 4, exploited the orientation of touch contacts and tried to estimate the position of the users around a multi-touch table. Applications using this information will be covered in the category *support for rotating user interfaces*. Finally, some of the described techniques also provide solutions for other problems in tabletop research (e.g. providing hover states, enhancing target acquisition, etc.), these will be discussed in the last category.

5.3 Format

The format of each pattern is highly oriented towards the format of Borchers' HCI design patterns for interactive exhibits [14] and Remy's pattern language for interactive tabletops in collaborative workspaces [103]. Each interaction pattern has a name and is preceded by a unique ID and followed by a ranking from zero to two asterisks. A higher ranking (i.e. more asterisks), indicate a pattern that is more suitable in collaborative situations. While the name of a pattern gives the reader an idea of the kind of applications that will be discussed, the *intent* is a short statement that describes the rational and the problems that will be addressed. Section *application scenarios* describes a wide variety of possible applications in

the context of real world scenarios. These applications are all based on common problems in tabletop research and can therefore be very useful. Finally, in the *recommended techniques* section, we suggest techniques to develop all these applications. All mentioned techniques are already described in previous chapters, however, more practical explanations are provided where needed.

5.4 The Catalog

(01) Privileged Access **

Intent

Providing highly secure access to private documents or functionalities for a special subset of users.

Application Scenarios

- Providing document ownership for personal files that are send to the collaborative display (e.g. with a PhoneTouch [111]).
- When a user moves a document from her private space to the shared space, she is still the owner and can only access options to edit this document.
- Each user can grant and revoke access permissions to her private documents.
- Provide access for specific users to some special functionalities (e.g. control panels, functionalities to change the appearance of the whole system).

Recommended Techniques

Because security needs to be guaranteed at any time when accessing personal documents, we recommend the use of very reliable techniques:

- The use of fingerprints (see Section 4.6.2) is formally accepted as a valid person identification method by law enforcements and therefore provides much reliability.
- Infrared tokens (see Section 4.4.3) are less reliable because users need to wear these tags and they can therefore be interchanged. However, by combining this technique with another identification method such as hand geometry feature extraction (see Section 4.6.2) more reliability is guaranteed. For example, users can first register the attached infrared token with the system by making a HandsDown event (placing their hand flat on the surface). The system can then assign the real identity of the user to the tag and enable interaction possibilities for this user.
- Colored gloves (see Section 4.4.4) also provide much reliability because the color of the glove at a touch location can be unambiguously observed with computer-vision

techniques. However, a glove can also be interchanged and the system cannot assign the users real identity to a colored glove. This technique can therefore be made more secure by entering a password with this glove. Kim et al. explored the use of passwords on multi-touch tables [64]. They noticed that in a collaborative setting users will typically authenticate in full view of a number of group members. The use of a password is therefore not resistant to observational attacks called *shoulder surfing*. For this reason, they designed many authentication schemes such as support for shielding PIN entries, to more elaborate visual PINs and pressure-based systems (e.g. the PressurePin widget) that do not require accompanying shielding actions [64, 148]. When using PressurePIN, parts of a password such as numbers or in this example images are aligned in a grid and can be selected to form the required authentication string as shown in Figure 5.1. Such a selection can be performed by applying additional pressure on the predefined regions, displayed in front of each row and column. This can be repeated until the entire sequence of objects is selected. If fingers are removed from the table during the input, the login is canceled as the user may be at risk of exposing selections.

- Identification with a mobile phone (see Section 4.5.1) can provide a high level of reliability. However, this technique is time consuming because it requires the use of an extra device. This can be acceptable when permission needs to be granted for only a small amount of actions (e.g. changing an option in a control panel). The system can also support authentication with a small area of the screen to enable the execution of more authenticated actions at a time. In this case, time constraints and clear visualizations are needed to protect regions from other users.

(02) Customizing Functionalities **

Intent

Providing a system which looks the same for all users, but behaves differently depending on who is interacting. This technique can significantly reduce clutter because only one copy of each option appears in the application instead of providing separate widgets for each user.

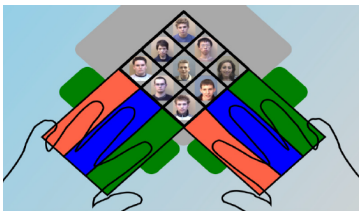


Figure 5.1: PressurePIN: a pressure-based authentication mechanism for inputting PIN-codes [64].

Application Scenarios

- Supporting multi-user undo: when an undo button is pushed by a particular person, it causes the last action of that person to be undone, rather than the last global action.
- When a user opens the address book, web browser, bookmarks, etc. respectively her own contact-persons, start-page and favorite websites are displayed.
- Varying the paintbrush properties (e.g. thickness, color) on a per-user basis in a painting application.
- Supporting parallel moded interactions: for example one person is in delete mode while another person is simultaneously in annotate mode.

Recommended Techniques

Most of the scenarios listed are not as critical as providing *Privileged Access* ***(01)*** to documents. For this reason, less reliable techniques can also be used in these applications:

- Fingerprint identification (see Section 4.6.2).
- Estimating the identity of the user based on positional information obtained by tracking users around the table with a camera (see Section 4.6.1).
- Attaching infrared tokens on users' hands (see Section 4.4.3).
- Tracking gloves (see Section 4.4.4).

(03) Cooperative operations **

Intent

Differentiating between an action performed by one user and the same action performed by multiple users.

Application Scenarios

- All users reach consensus by performing the a gesture simultaneously (see Figure 5.2). As a result, actions which involve global changes (e.g. organizing all documents in a single pile) can be executed.
- Providing a vote button for each user to reach consensus [83].
- Two users that are touching a document at the same time to transfer ownership (see Figure 5.3).
- When two users are touching the same document, the contested item duplicates itself. Possibly both views can be linked to the same source to reflect changes in both versions [83].

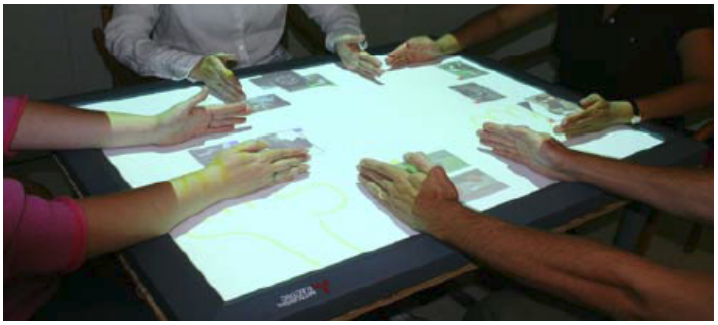


Figure 5.2: Reaching consensus by performing the same gesture simultaneously [82].

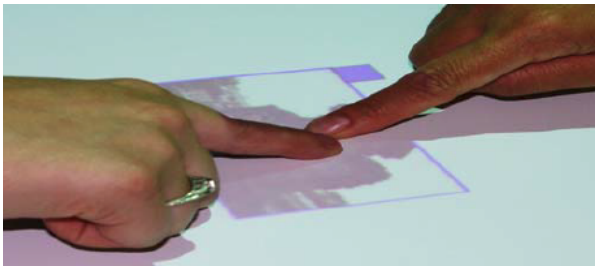


Figure 5.3: Two users simultaneously touch a photo to transfer ownership. [82].

Recommended Techniques

- Estimating the identity of the user based on positional information obtained by tracking users around the table with a camera (see Section 4.6.1).
- Attaching infrared tokens on users hands (see Section 4.4.3).
- Tracking gloves (see Section 4.4.4).
- To differentiate between users when an action is simultaneously performed, the heatwaves of a thermal camera can be used (see Section 4.6.1).
- When only touches are used in cooperative operations, fingerprint identification (see Section 4.6.2) can even be used.

(04) One and Two Handed Gestural Interaction

Intent

Differentiating between the same gesture performed by one and two hands. One-handed input and two-handed input is not interreplaceable, since handedness has an impact on the ease and accuracy with which gestures may be conducted [84].

Application Scenarios

- Distinguishing a dual stretch gesture performed by fingers of the same hand from a dual hand duplicate gesture where a copy of a visual object is produced (see Figure 4.2).
- When touching a display with two fingers of the same hand, an object can be scaled or rotated. Instead, when touching the display with two different hands, objects in the region between the two touch contacts can be selected. (Moscovich et al. [84] showed that two handed multi-touch manipulation is only compatible with a task when there is a clear correspondence between the fingers and the manipulated control points. Unimanual multi-touch manipulation on the other hand, is compatible with many tasks even when lacking a clear point correspondence between fingers and object.

Recommended Techniques

To implement these scenarios, techniques which provide a clear distinction between touches of different hands can be used:

- Differentiating between hands by extracting features from touch contacts. However, this technique does not support vertical touches (see Section 4.6.2).
- Relating fingerprints to the corresponding hands (see Section 4.6.2).
- Tracking techniques such as shadow tracking and tracking with RGB and depth cameras (see Section 4.6.1) provide a clear view of the interaction above the table in which hands can be distinguished.

(05) Customizing Content *

Intent

Providing a personalized view of the user interface for each user.

Application Scenarios

- Displaying extra help information near every widgets for a person who needs it.
- Translating text to match each user's language preference (see Figure 5.4).
- Users with vision problems can read text in a bigger font than other users.
- Each user has some private documents which other users are not able to see.
- Credentials are automatically filled in when a user wants to login on for example a website to avoid shoulder surfing (see Figure 5.5).



Figure 5.4: Personalizing the interface appearance by translating text to the user's preferred language [112].



Figure 5.5: An example of customizing content: moving a lens over a login area automatically fills in credentials [113].

Recommended Techniques

To provide each user a customized view of the user interface, a lumistry film (see Section 2.3.2) can be used. However, this hardware solution is not very flexible and does not provide techniques to identify users or obscure private content for other users. For this reason, we recommend the use of *idLenses* [113] which are basically identity-aware *magic lenses* [12]. When a user reveals her identity to the system, a special transparent window called *magic lens* is displayed on top of the user interface (see Figures 5.4 and 5.5). Moving this window over content in the interface creates a personalized view for this user.

The activation of idLenses can be supported using many events. However, we do not suggest the use of a touch event to activate an idLens because this action is traditionally used for all interactions on a multi-touch display. Techniques that can be used:

- Activating an idLens when a HandsDown event is performed (hand geometry feature extraction (see Section 4.6.2)) (see Figures 5.4 and 5.5).
- Fingerprint identification (see Section 4.6.2): activating an idLens when for example two fingers are touched down near each other. Another option is to support a hover state on touch displays with pressure simulation (see *providing hover states **(11)***).
- Activating an idLens when an infrared token (see Section 4.4.3) is hovering the display (see *providing hover states **(11)***).
- Using a depth camera (see Section 4.6.1) to detect when a hand is hovering the display (see *providing hover states **(11)***).

(06) Holding Commands On Fingers

Intent

Differentiating between touches with different fingers of a hand.

Application Scenarios

- Supporting a button which behaves differently depending on which finger is clicking on this button. For example, when the index finger is used, the web browser is started whereas the ring finger opens the email client [132].
- Simulating full mouse functionality by mapping different buttons to different fingers (see Figure 5.6) [76].

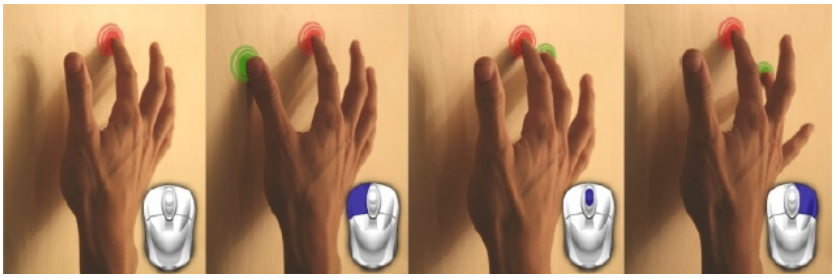


Figure 5.6: The SDMouse to emulate the functionality of a 3-button mouse with multi-finger input [76].

Recommended Techniques

- Differentiating between fingers when the entire hand is placed on the screen when performing a HandsDown event(see Section 4.6.2)
- Fingerprint Identification (see Section 4.6.2)
- Using gloves (see Section 4.4.4) with unique colors or tags for each finger.
- Tracking hand movements with shadow tracking or an RGB camera (see Section 4.6.1) can provide very bright images of the hands above a multi-touch display. A robust hand model can therefore be applied in order to determine which finger corresponds to a touch point.

(07) Automatic Content Orientation *

Intent

Providing functionalities for automatic orientation of documents to display content in the right direction for a user who wants to read it.

Application Scenarios

- Automatic reorientation of content to face the target user when passing documents between users.

Recommended Techniques

- Cameras above the table (see Section 4.6.1) can be used to track the position of the users around an interactive table.
- Proximity sensors (see Figure 4.17) can be used to determine which places around the table are occupied.
- When the positions of users around a table are known, vector fields [32] can be generated to provide smooth transitions when passing a document (see Figure 5.7).

- The DiamondSpin [122] provide a software solution to rotate documents to face the user (see Figure 5.8). However, this solution does only work on a circular multi-touch table.

(08) Orientation-Sensitive Widgets

Intent

Treating the orientation of a finger as an additional direct input dimension for interface widgets.

Application Scenarios

- Providing an orientation-sensitive button (see Figure 5.9-1) to allow users to use the finger orientation to specify the parameters of the button functionality while hitting the button. As a result, function invocation and parameter specification are combined in a single step.
- Providing an orientation dial (see Figure 5.9-2) to allow continuously adjustments to parameters with high precision while minimal finger movement and screen space is needed.

Recommended Techniques

- The orientation of a finger can be extracted from the touch contact (see Section 4.6.2). However, this technique does not support vertical touches.
- Pitch, yaw and roll angles can be extracted from a fingerprint (see Section 4.6.2) [51].
- Hands above the table can be tracked with a camera (see Section 4.6.1) to determine the orientation of a finger.

(09) Orientation Invariant Input

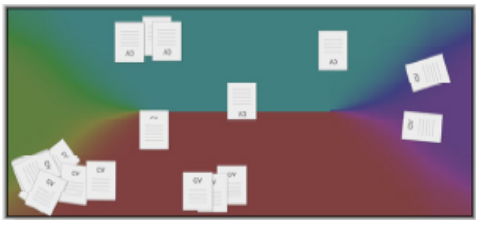


Figure 5.7: A colored visualization of the vector fields on the table to provide automatic orientation of documents [32].

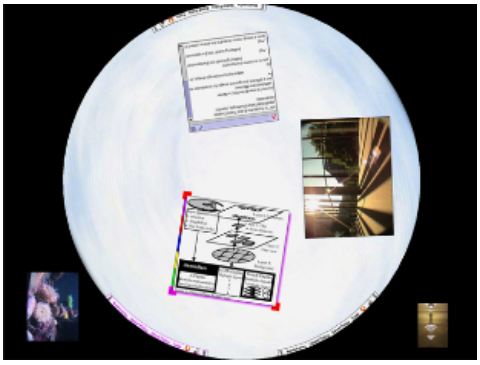


Figure 5.8: The DiamondSpin software [122].

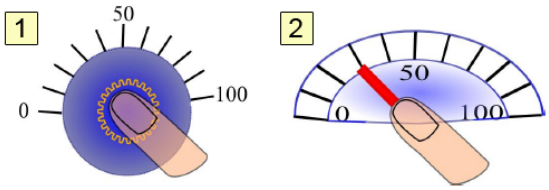


Figure 5.9: Orientation-sensitive controls: (1) Orientation-sensitive button. (2) Orientation dial. [144]

Intent

Enabling the system to interpret input in an orientation invariant way.

Application Scenarios

- The orientation of a gesture performed on a multi-touch table can be arbitrary depending on the user's position. This can cause problems when a gesture has a different meaning in another direction (e.g. an arrow as shown in Figure 5.10).
- Since many numerical characters are inherently orientation-specific, the system needs to support handwriting recognition in an orientation invariant way (see Figure 5.11).

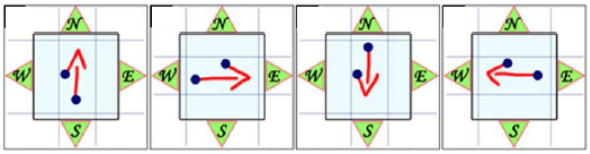


Figure 5.10: Orientation of the input is needed to discriminate between these four gestures [81].

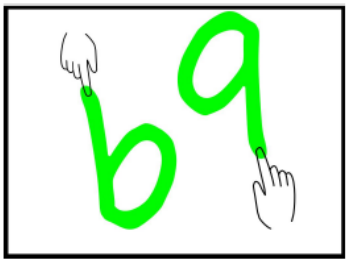


Figure 5.11: Ambiguous gestures caused by different hand orientations [144].

Recommended Techniques

To determine the orientation of a gesture, not as much precision of the angle of the finger contact is needed as in *orientation-sensitive widgets* ***(08)***. Therefore we recommend the following techniques:

- While dragging the finger the orientation can be extracted from the touch contact (see Section 4.6.2). However, this technique does not support vertical touches.
- Interaction above the table can be tracked with a camera (see Section 4.6.1) to determine the orientation of a finger.
- Infrared tokens that are attached to the wrist (e.g. a wristband) can be used because when the position of the touch contact and wristband is known, the system can calculate the orientation of the input.
- Because pens are usually used for writing text, the orientation of a digital pen (see Section 4.5.2) can be used to determine the orientation of the input.
- (Fingerprints can also be used but this precision is not really needed for these kind of applications.)

(10) Occlusion Problems

Intent

The user can interact with the display without occluding content with her hands. As a result, users can interact very fast with minimal additional finger movements.

Application Scenarios

- Dynamically placing content and interface elements outside the occlusion region of the hand.
- Dynamically reorient special widgets such as occlusion-aware pie-menus (see Figure 5.12) with a gap on one side so that the gap is always aligned with the body of the finger and no menu-items are occluded.



Figure 5.12: Design of a menu that avoids occlusion caused by the user's hand [18].

Recommended Techniques

- When tracking hands above the surface (see Section 4.6.1) or tracking gloves (see Section 4.4.4), the system can capture the occluded area very precisely.
- The orientation of a finger can be extracted from the touch contact (see Section 4.6.2). However, this technique does not support vertical touches.
- Infrared tokens attached to the wrist (e.g. a wristband) can be used because when the position of a touch contact and wristband is known, the system can estimate the position of the hand.
- The position of a hand can be extracted from the orientation of a fingerprint (see Section 4.6.2).
- When using DI or DSI multi-touch tables (see Section 2.3.2) it is possible to capture a bright image of the hand in which the position of the hand can be detected.

(11) Providing Hover states

Intent

Providing an extra state on multi-touch systems so the user can get information about an action before executing (touching) that action.

Application Scenarios

- Providing information about the properties of a cursor (e.g. brush thickness) in a painting application running on a multi-touch table before the user starts painting.
- Providing extra information about an action by using for example a traditional tooltip.

Recommended Techniques

- Tracking the users' hands with a depth camera (see Section 4.6.1).

- Infrared tokens attached to a user's finger (i.e. the irRing (see Section 4.4.3)) can be used to enable a hover state because they can be tracked even when the user is not touching the display.
- Depending on the type of diffuser and the light conditions in the environment, hands hovering above the surface can be tracked on a DI or DSI multi-touch setup (see Section 2.3.2).
- By observing the contact area when a touch is performed, it is be possible to simulate two (or even more) levels of pressure. For example, a vertical touch can be used to activate the hover state whereas the oblique touch can be used to execute an action. This technique has the advantage that orientation information can always be extracted from the finger contact shape when an action is executed (see Section 4.6.2).

(12) Optimizing Touch Accuracy

Intent

Exploiting finger orientation to make the selection of small targets on touch displays easier.

Application Scenarios

- Improving the accuracy of fingers touches to make touch interfaces more reliable.
- Providing a faster selection technique for small targets.
- When increasing the touch accuracy, smaller and more controls can be packed up in the same user interface to reduce clutter (assuming that people have perfect vision).

Recommended Techniques

- By considering the orientation of a finger touch (using contact area information (see Section 4.6.2) or tracking hand postures (see Section 4.6.1)) the *bubble cursor* [43] can be improved to provide a *directed bubble cursor* (see Figure 5.13-1) [144]. The bubble cursor dynamically resizes the activation region of an area cursor [8] so that the target at the shortest distance to the center of the cursor is always selected. A directed bubble cursor further improves this target acquisition technique by being biased to towards targets in the front of the finger. The shape of the activation region is also slightly skewed to reflect this (see Figure 5.13-2). Figure 5.13-3 shows the distance weights of this directed bubble cursor. Another technique to provide enhanced target acquisition by exploiting the orientation of the finger is the cursor-offset technique [95]. In this technique, a cursor is displayed at a certain distance from the touch contact.

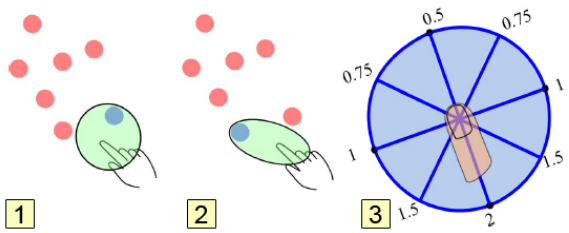


Figure 5.13: (1) Regular bubble cursor. (2) Directed bubble cursor. (3) Distance weights according to finger orientation. [144]

- Holz et al. [51] exploited the pitch, roll and yaw angles of a touch contact extracted from fingerprints (see Section 4.6.2) to increase the accuracy of touch interfaces. In an experiment they noticed that when a user needs to touch a small target a several times under the same conditions (equal pitch, roll and yaw angles), they will hit exactly the same spot. However this spot can be different for each user. For this reason they extracted the identity and orientation from a finger contact to automatically recalibrate the screen when a touch is performed. As a result, the error rate was reduced to a third when touching small objects.

(13) Extending Reachability

Intent

Exploiting finger orientation to select objects that are far away on interactive displays.

Application Scenarios

- When the touch display is very large and users cannot easily reach to every corner, techniques can be provided to select objects from a distance.

Recommended Techniques

By considering the orientation of a finger touch (using contact area information (see Section 4.6.2), tracking hand postures (4.6.1) or extracting the finger orientation from fingerprints (see Section 4.6.2)), the system can support for example the *aim and grab* or *finger cross selection* techniques:

- When using the aim and grab technique [144], the user touches the finger on the surface to cast a selection ray aligned to the finger orientation as shown in Figure 5.14-1. To switch between multiple objects intersected by the selection ray, the user can move the finger forward or backward along this ray (see Figure 5.14-2).
- In order to select a distant target with the finger cross selection technique [145], the orientation of two fingers can present two radial lines. The user can select the

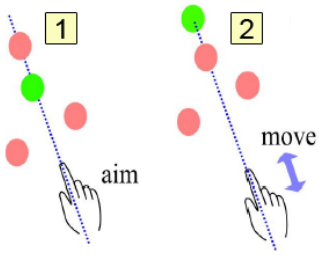


Figure 5.14: (1) Aim finger to select an object. (2) Move finger along the orientation vector to switch between multiple objects. [144]

target by controlling the position of the intersection of these two lines (see Figure 5.15).

5.5 Conclusion

In this chapter we discussed many new applications that can be supported using the techniques described in Chapter 4. We provided an overview of techniques that can be used to develop each application. However, the final decision to use a specific technique will highly depend on the context in which an application will be used (e.g. what reliability is needed, is it acceptable to instrument users or the environment with additional hardware).

With this catalog of interaction patterns, we want to encourage developers to participate in the process of developing additional applications that exploit identity information of touches. We see this set of applications as "Hello World" applications in the domain touch identification. For this reason, there is much potential to extend this set of patterns especially when better and more reliable touch identification techniques emerge.

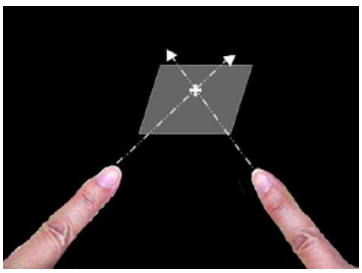


Figure 5.15: The finger cross selection technique to select objects from a distance [145].

Chapter 6

Non-Intrusive High-Accuracy User Identification

6.1 Introduction

In Chapter 5, we discussed a large number of applications and new interaction techniques that can be created when touch technology is able to discriminate between the input of multiple users. As we saw in Chapter 4, revealing the identity of the user while working on an interactive display has been the focus of many projects and is still a challenging and salient issue in tabletop research. Many technologies that are already discussed, required instrumentation of the user with extra hardware (see Sections 4.4 and 4.5). Other techniques supported bare hand interaction but with a reduced reliability of the touch identification mechanism (see section 4.6.1), or required the user to make specific postures (e.g. the HandsDown technique described in Section 4.6.2), which can disturb the natural interaction. In this chapter, we propose a couple of new techniques to reveal a user's identity while she is working on an interactive display. In Chapter 7, we will evaluate and optimize these techniques using a large database of data that we collected during user tests.

6.1.1 The Goal

In the introduction of this thesis we proposed our envisioned system. Based on our findings from previous chapters, we are now able to provide a more detailed description.

Our goal is to develop a new technique to identify at least four users that are working simultaneously on an interactive surface. The system needs to be able to relate an identity to each touch point without disturbing the interaction. In our envisioned collaborative setup, users can move around the display, so the technique needs to be independent of the user's

position. Furthermore, the system needs to have the potential to work in real-time. We also aim to develop an identification technique that can be used in combination with many touch technologies that are already available (sensor-based touch techniques as well as computer vision-based techniques discussed in sections 2.2 and 2.3).

6.2 Finding New Identity Features

Based on the requirements that we imposed on our identification technique, biometric features (see Section 4.6.2) are probably the only metrics that can be used to develop the envisioned system. Finding features on the human body that are variable enough to be used as identity features is, however, very hard, because variability is the raw material of natural selection, it is what nature selects from. Locating these features and measuring them with an algorithm can even be more difficult.

Papillary ridges like fingerprints (see Section 4.6.2) are features that are entirely unique. Pappilary ridges are at their finest on the fingertips and at their coarsest on the rest of the fingers and the palm. However, it is not possible yet to get a clear view of the palmar region of the hand when touching a display. As a result, we are forced to find new features on the back of the hand that reveal a user's identity. The back of the hand can easily be captured by an overhead camera. These kind of setups are also very flexible, which ensures that the technique can be used in combination with different touch technologies.

6.2.1 The Approach

In many situations in HCI research a feature is measured during a test and the consequences can be computed by fitting a function to it. Unfortunately, we do not know yet what feature to measure. This forces us to take a more general approach to the problem [35, 52]:

1. Guess the law (or the feature in our case).
2. Compute the consequences.
3. Compare the computation to experiment/observation. If it disagrees with experiment it is wrong.

6.2.2 Candidate Features

Hands are the most flexible limbs of the human body [86]. It can therefore be very hard to detect features on a hand reliably over different postures. To get an indication of how much

the human skin can deform in a specific region we can look at the number of flexure lines. Flexure lines are skin hinges that open and close during grasping and gripping, following the movement of the underlying joints. In the palmar region of the hand, there are many flexure lines (e.g. the heart line, head line, line of density, etc.). In the dorsal region (back of the hand), however, only a couple of flexure lines are visible near the thumb. This indicates that the back of the hand is not very flexible. The back of the hand is also the region that is visible to a overhead camera most of the time when touching a display (the arm can be invisible when touching near the border of the screen and the fingers can be invisible when performing a vertical touch). These characteristics make that the dorsal region of hand is the best candidate for extracting features while interacting on a multi-touch display.

Now that we have selected a region of the hand that can be captured reliably by an overhead camera, we need to guess which features to extract. Afterwards, we will check if these features are variable enough among different people. To make an educated guess, studies and observations of the human hand by Prof. dr. John R. Napier can help. He was preeminent among the founders of modern primatology and focused on topics including the anatomy and functions of the human hand and other species.

> "Variation in the shape of the hand is particularly obvious. The human hand comes in all sorts of shapes and sizes".
>
> "The skin surface of the human hand, far from being smooth and perfect like alabaster, as so often in romantic novels, is neither smooth nor white but highly textured and multicolored. It is deeply etched with lines, grooves, folds, and furrows and studded with pits for hairs and open ducts for the excretion of sweat. In living hands veins show blue, arteries show ping, and for good measure freckles and liver spots show brown; only dead hands are lily-white."
>
> John R. Napier [86]

Based on this discussion, we propose that the following features have the potential to be unique over many people:

1. Geometry metrics
2. Skin color
3. Skin texture

We present four computer vision techniques to extract these features from an image of the human hand:

1. Measuring hand metrics
2. Color matching
3. Eigenimages
4. Local binary pattern matchings

6.3 Feature Extraction Techniques

In Section 6.2.2, we proposed 4 potential techniques for extracting features out of the images of the back of the human hand taken by an overhead camera. To make it possible to compare features from different hands they need to have the same scale. Therefore, we suppose that the camera is mounted above the interactive display at a fixed position. In this section, we will discuss the general principles of each of these techniques. In chapter 7, we will evaluate these techniques based on the results of studies that we conducted.

The general principal that we use in all the techniques is that the system has training data for all of the registered users. When a users touches the display, features will be extracted from the back of the hand captured by the overhead camera. These extracted features will be compared to the training data. The image/person that matches the most is the user that is most likely touching the screen.

All algorithms are implemented in C++. We used OpenCV (version 2.3.1) [92] to implement the image processing and computer vision algorithms.

6.3.1 Measuring Hand Metrics

In Section 4.6.2 we discussed how geometrical features can be extracted from a flat hand placed on a surface with the fingers spread. However, research has not yet investigated if there are unique geometry features that can be extracted from the hand in other postures, such as a hand that points on a surface (see Figure 6.1). Because the back of the hand is the least flexible region of the hand and also the region that is visible to an overhead camera most of the time over different postures, we prefer to extract most of the geometry features from this area. Potential features in this area can be: the width of the hand at different places, the height and the area of the hand (see Figure 6.1). Other geometry features, such as the length and width of fingers or the width of the arm at different places can also be measured when they are visible to improve the reliability of the technique. All these measurements can vary over different hand postures. However, it is possible that these variations can be taken into

account by training the system with enough samples of each user in different postures over the entire area of the interactive display.

6.3.2 Color Matching

The skin on the human hand is highly textured and multicolored [86]. The color of the skin is therefore a very good candidate to be different over many people. On the one hand, we can compare two images by doing a pixel-based comparison of the intensity value of each pixel for each channel. This approach requires that the system can recognize the hand 100% accurate which is not possible. On the other hand, spatial information can be eliminated when we use the histogram of the image as a unique feature. By doing this however, all spatial information is lost and the histogram of an image with for example a dark spot in the upper left corner can match with an image with a dark spot located in the bottom right corner. It is clear that an intermediate solution between these two approaches is needed.

Using Patches to Preserve Spatial Information

By dividing the skin region of the back of the hand into patches of equal sizes, we can create a histogram for each color channel of this patch and still preserve some spatial information by using the location of the patch inside the hand region (see Figure 6.2). When analyzing the histograms of such a patch (see Figure 6.2), we see that the number of bins is very small and highly concentrated in one region. This is because there are only very subtle variations in the skin color of a single patch. These histograms do not provide much data, therefore, it is also possible to use the average skin color of the patches as a feature instead.

The ideal size of these patches can only be determined by analyzing experimental results. When the size of a patch is only 1 pixel, the patch can match with to many patches of other hands. When a patch is very large, on the other hand, the information in the corresponding histogram will be to general and the patch will also match with to many patches.

Matching a patch of an image with other patches of images in the trainings set can be done by finding patches at corresponding locations in the training set and comparing their features (e.g. average color, histograms). However, it is not likely that a hand recognition algorithm is able to localize a patch very precisely. In practice, there will be many situations in which small differences in the posture of the hand or the location of the hand on the interactive screen will make another skin region visible in the image, this causes all patches to shift (see Figure 6.3). To handle this problem, we can specify a region in which to search for the best match of a patch(as shown in the highlighted region in Figure 6.3-2). It is important to note that this region needs to be small enough to ensure that the algorithm will not match a patch

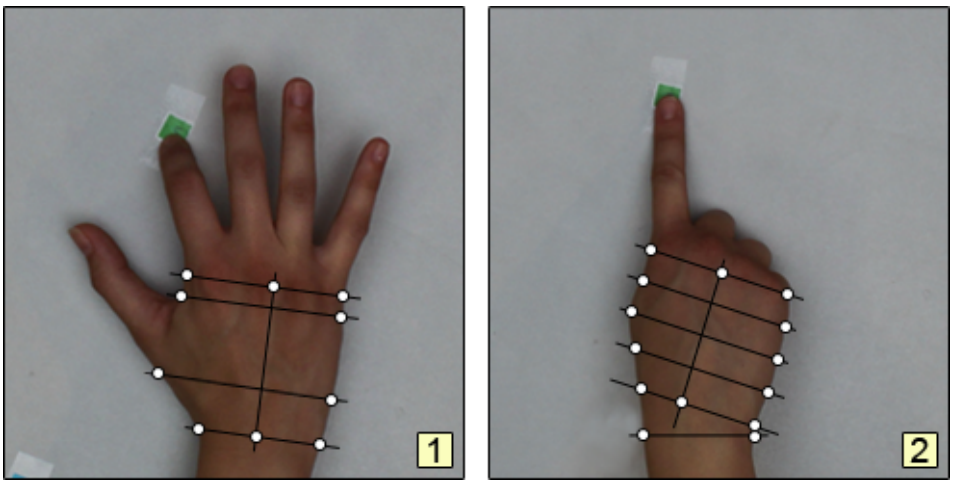


Figure 6.1: Extracting geometry features from the human hand.

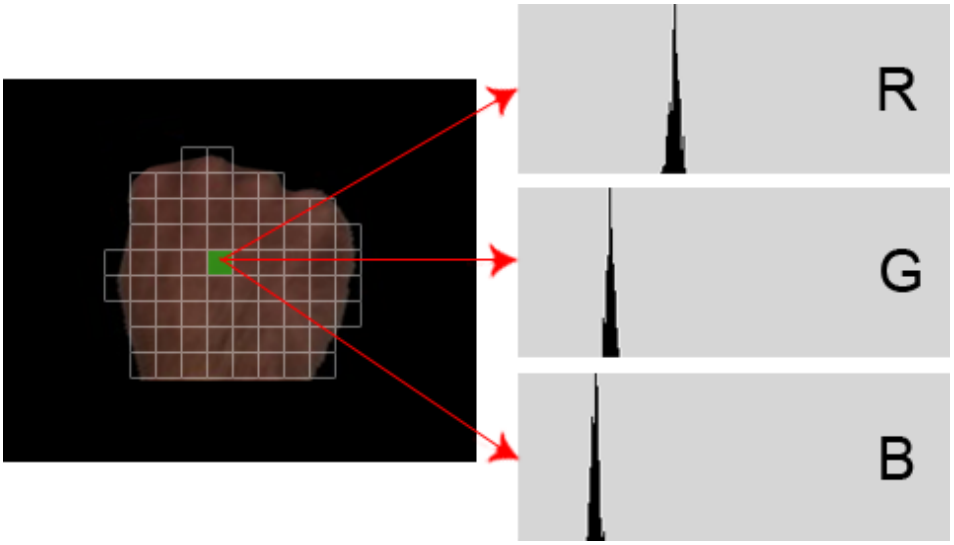


Figure 6.2: The histograms for the three color spaces of a single patch of an image of the back of the hand.

with another patch in the training set that is located on the other side of the hand. Instead of extending the region in which to search for patches, it is also possible to build a training set of images in which all possible postures on every region of the interactive screen are included. When using this approach however, the training set needs to be so large that there is a risk of overtraining. In this situation, a patch will match with so many patches in the training set that the performance of the system will drop significantly. We can conclude that the amount of train data, the size of the area in which to search for the best match of a patch and the quality of the hand recognition algorithm are interdependent. Experiments are needed to decide what the best trade-off is between the two approaches.

6.3.3 Eigenimages

Eigenimages [126] is a very popular technique in face recognition, therefore, it is sometimes called *eigenfaces*. The basic principle of this technique is that the system builds a set of standard eigenimages from the training set. Input images can now be considered as a combination of these standard eigenimages. For example, a test image might be composed of the average image plus 55% from eigenimage 1, 20% from eigenimage 2 and even -5% from eigenimage 3. The identity of the eigenimage with the highest contribution can be considered as the identity of the input image. We will explain the basic principles of the algorithm in short:

1. All training images are stored in a single matrix T. Each row represents a training image, obtained by concatenating the rows of the pixels in the original image.

2. Calculate the average image and subtract it from each original image T.

3. Compute the covariance matrix S of T.

4. From this covariance matrix, the eigenvectors and eigenvalues can be computed. Each eigenvector has the same dimensionality as the original images, and thus can itself be seen as an image. The eigenvalues associated with each eigenvector represent how much the images in the training set vary from the mean image in that direction. This result can be visualized by placing each eigenvalue in the corresponding pixel location. When doing this, we get the familiar ghostly look of an eigenimage (see Figure 6.4). Computing the eigenvectors and eigenvalues can be very computational expensive. Therefore, we first reduce the dimensionality of the image using the principal components analysis technique [124].

5. The N x N covariance matrix will result in N eigenvectors. By selecting the principal components (eigenvectors with the largest eigenvalues), we can find a transformation matrix P.

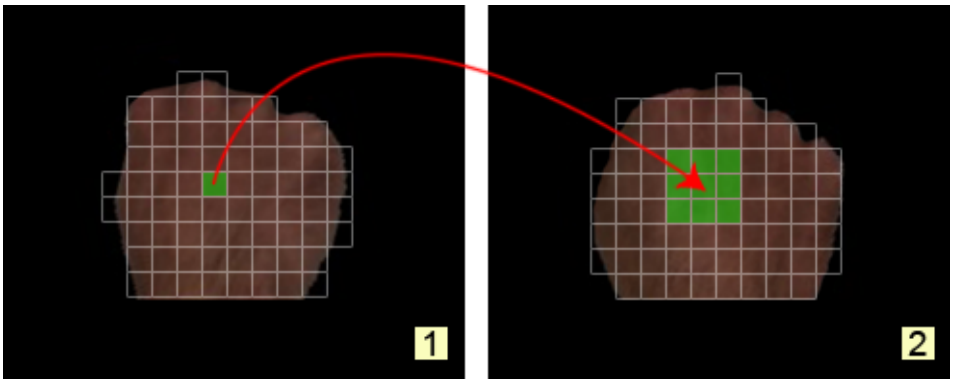


Figure 6.3: Matching a patch of an image to valid patches of an image in the training set based on the location of the patch.

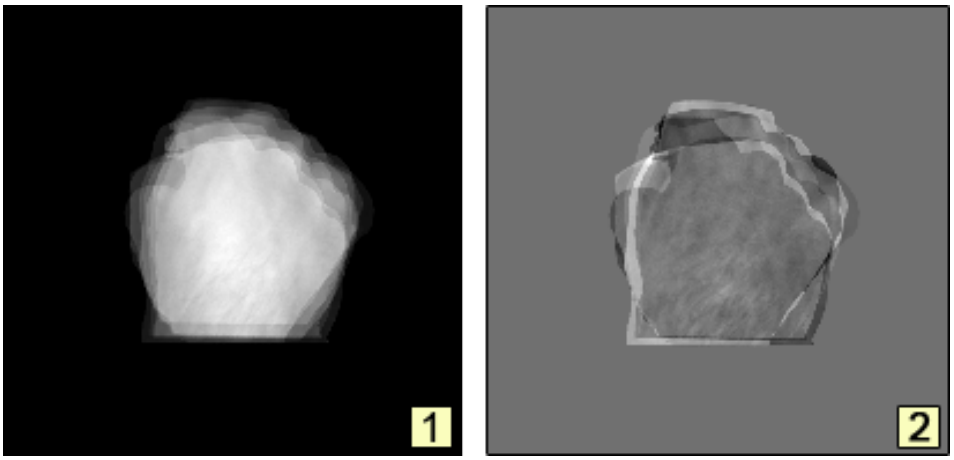


Figure 6.4: 1. The average image 2. An example of an eigenimage.

The identity of an image can now be extracted by first subtracting the mean from it and then project the image on the eigenimages using the transformation matrix P. By doing this, we can record how much the tested image differs from the mean image. The identity of the image can now be extracted by finding the nearest neighbor eigenimage from the set of training images.

6.3.4 Local Binary Pattern Matching

A Local Binary Pattern (LBP) [89] is a special representation of an image in which fine details of the texture are encoded. Not only is this technique very popular for matching textures but also for face recognition [3]. The idea is not to look at the whole image as a high-dimensional vector, like the technique discussed in Section 6.3.3, but to describe only local features of an object. The features that are extracted in this way will have a low-dimensionality.

The Encoding Technique

The local binary pattern operator labels the pixels of an image by thresholding the 3x3-neighborhood of each pixel with the center value and considers the result as a binary number. In other words, if the intensity of the center pixel is greater or equal than its neighbor, denote it with 1 and 0 if not. We will end up with a binary number for each pixel. The decimal representation of this number will be the intensity for that pixel in the local binary pattern (see figure 6.5). If we do this operation for all pixels in each channel of an image we are able to capture very fine grained details (see Figure 6.6-1). Furthermore, this representation of an image is illumination invariant. Figure 6.6 shows two images of the back of a human hand in the same posture under different illumination. We can see that the local binary patterns are identical.

An extension of this operator is the Circular Local Binary Pattern (CLBP) operator [90]. This operator allows us to encode the variations in intensity levels of an arbitrary number of pixels on circles with a predefined radius any radius. If a point on the circle does not correspond to image coordinates, the point gets interpolated (for example with a simple bilinear interpolation). The CLBP-operator is very powerful for capturing details (see Figure 6.7) at

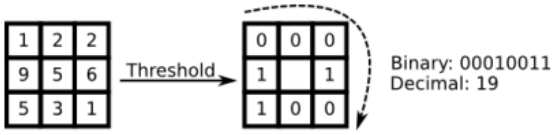


Figure 6.5: The basic local binary pattern operator.

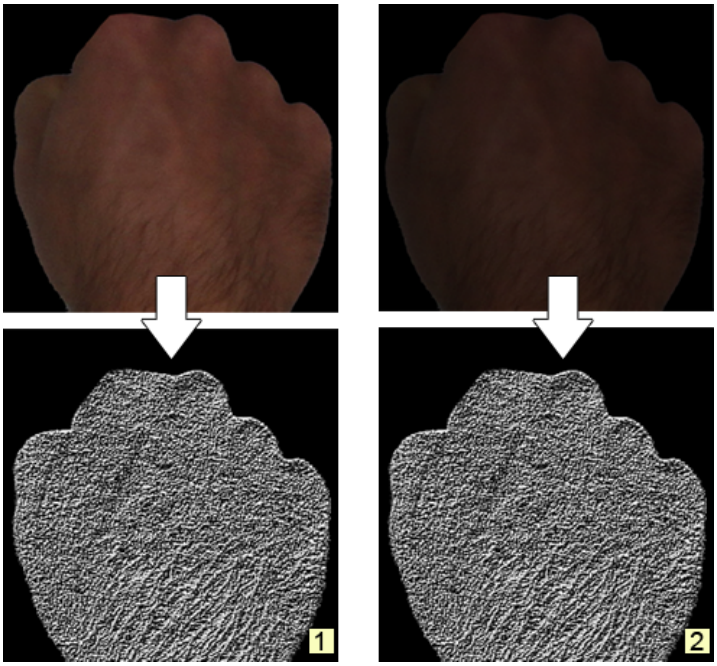


Figure 6.6: The local binary pattern representation of a single channel (in this case the red channel) of two images under different illumination. The local binary patterns are identical.

different levels: the smaller the radius, the more local details will be captured. By varying the number of samples on the circle we can control the number of patterns that can be encoded. In general, the more patterns can be encoded, the more discriminative power the operator has. However, when the number of neighbors on the circle exceeds 8 we need to take into account that the number of patterns increases significantly. This can introduce performance issues when we want to generate and compare the histograms of the local binary patterns: 16 sample points on the radius already produces a histogram of 65536 bins.

Ojala et al. [90] developed an extension to the CLBP-operator called the uniform patterns. When using this operator, the number of patterns does not increase exponentially in function of

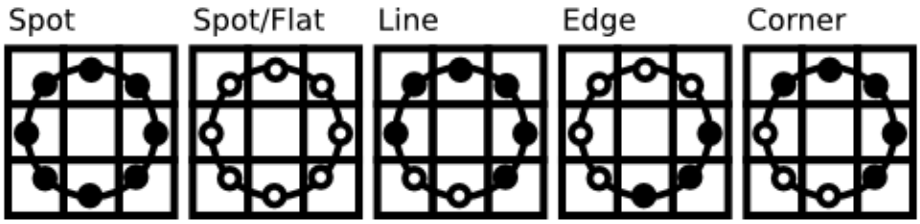


Figure 6.7: Examples of neighborhoods that can be captured with the circular local binary pattern operator.

the number of sample points on the circle. A local binary pattern is called uniform if it contains at most two bitwise transitions from 0 to 1 or vice versa when the binary string is considered circular. For example, 00000000, 00011110, 10000011 are uniform patterns. Because of the circular interpretation of the binary patters, all patterns with the same number of ones/zeros represent the same uniform pattern. For example, 11100000, 00111000, 10000011 are the same uniform patterns and can therefore have the same label. All non-uniform patterns have the "miscellaneous" label. In short, all patterns can be encoded in P + 2 groups (with P the number of sample points on the circle used in the CLBP-operator): P + 1 groups for the uniform patterns and 1 group for the nonuniform patterns (see figure 6.8). Ojala et al. noticed in their experiments with textured images that uniform patterns account for 90% of all patterns when using 8 sample points and a radius of 1. When there are 16 sample points and the radius is set to 2, they observed that uniform patterns account for 70% of all patterns. We can conclude that uniform patterns can encode local binary patterns very compact, only 18 bins are needed instead of 65536 when 16 sample points are used. This is especially useful when using a relatively large amount of sample points.

Matching the Local Binary Patterns

In Section 6.3.2, we discussed that patches can be used to preserve spatial information while matching images. This technique can also be used for the comparison of the binary pattern representation of images. The discussion in Section 6.3.2 regarding the patch size and the maximal search region for patches also needs to be taken into account in this case.

Figure 6.9 shows the histograms of the binary patterns for the three channels of a single patch. These histograms are more dense than the histograms of the patch in Figure 6.2 because the local binary pattern representation provides much more detailed information of the texture inside the patch. To compare two patches, we can simply compare the histograms of the corresponding color channels. A very popular, easy and fast technique to compare histograms is the *intersection technique*. This method simply compares, for each bin, the two values in each histogram, and keeps the minimum one. The similarity measure is then the

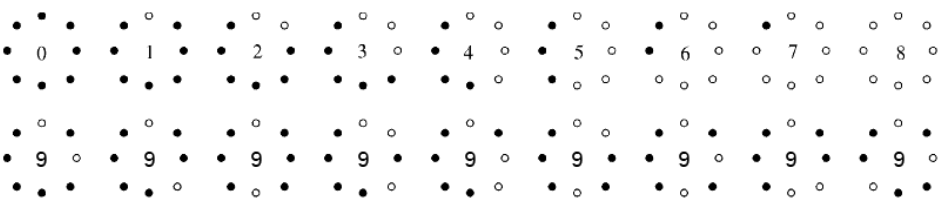


Figure 6.8: If there are 8 sample points on the circle with the CLBP-operator, all patterns can be categorized under 10 labels using uniform patterns.

sum of these minimum values. Consequently, two images having histograms with no colors in common would get an intersection value of 0, while two identical histograms would get a value equal to the total number of pixels.

6.4 Hand Recognition

Now that we have an idea of potential feature extraction techniques that can be used to identify users, we first need to find an algorithm to recognize the hand of a user in a captured image before we can test the feature extraction techniques on a larger scale (see Chapter 7). In Section 6.2.2 we concluded that the back region of a hand is very suitable to extract features from. This region can be extracted from an image captured by a camera above an interactive display in 3 steps:

1. Skin segmentation
2. Finger detection
3. Wrist detection

6.4.1 Skin Segmentation

A very popular technique to extract human body parts out of an image is *skin segmentation*. This technique is used in a wide range of applications, such as face detection and tracking, naked people detection, hand detection, etc [73, 61]. The main goal of skin segmentation is to build a decision rule that will discriminate between skin and non-skin pixels. Identifying skin-colored pixels involves finding the range of values for which most skin pixels would fall in a given color space. A good skin color model must detect most skin pixels while minimizing the amount of non-skin pixels classified as skin. Unfortunately, RGB is not the best color space to label a range of colors as skin color because the color space is not perceptually uniform. This means that two colors at a given distance might look very similar, while two other colors separated by the same distance will look very different.

The YCrCb color space is a very attractive color space for doing skin color segmentation [73]. The Y component describes the brightness, the other two values describe the color difference rather than a color, making the color space unintuitive. However, the separation of luminance and chrominance components makes that this color space is luminance independent, resulting in a better performance. Figure 6.10-1 shows an image of a hand in the YCrCb color space. The skin cluster that we use in our skin segmentation algorithm is chosen by experiment: $Y>20$, $85<Cb<135$, $135<Cr<180$. Figure 6.10-2 shows an example of the result

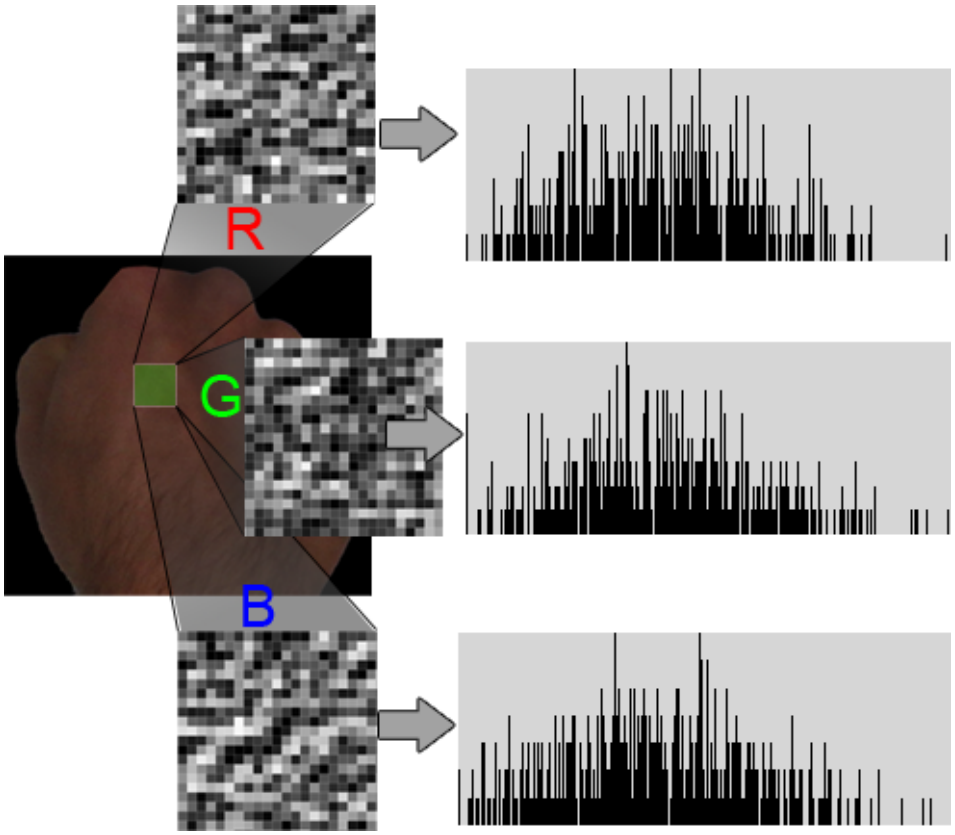


Figure 6.9: A patch on a hand of which each of the three color channels are encoded as a binary pattern.

of the skin segmentation algorithm. To eliminate the noise in the binary mask produced in the skin segmentation step, we apply an opening and closing filter (see Figure 6.10-3). The opening filter removes small blobs introduces by image noise, while the closing filter connects together objects erroneously fragmented into smaller pieces. By applying the opening filter before the closing we prioritize the noise filtering.

The skin segmentation step provides us a binary image of the skin region from which we need to extract the contours. This can be done using the Suzuki 85 algorithm [133] which is implemented in the OpenCV library. This algorithm generates a collection of lists of coordinates that represent all the external contours in the binary image (see figure 6.10-4). We can use these coordinates in the following steps of our hand extraction technique.

6.4.2 Finger Detection

In section 4.6.2 we already mentioned in short how fingertips and valleys can be extracted from a hand contour using the Differences Of Slopes (DOS) technique [87]. This approach developed by Boreki et al. [15] performs good when a hand is placed flat on a surface. We want to detect the fingers of a hand reliably under different postures. Therefore we present another contour analysis technique.

We start by calculating the curvature of the contour with the traditional DOS technique. This can be done by roaming the hand contour in counterclockwise direction with two vectors of the same size and calculate the angle formed by these two vectors (see Figure 6.11). We also take the direction of the vector into account: when the absolute value of the angle is greater than 180 degrees we take the supplement and change the sign of this angle. In Figures 6.12-2 and 6.13-2 the plots of the differences of slopes of the contours in figures 6.12-1 and 6.13-1 are shown. We can see from these plots that there is a high fluctuation in the DOS values and that this does not provide us much information. However, when for each point in the contour the DOS values of the 3 previous and the 3 next points are added to the DOS value of the central point (see Formula 6.1), each point provides information of the curvature of the curve in the neighborhood of that point. In Figures 6.12-3 and 6.13-3 the results of this operation are shown (the cumulative DOS functions). Very high and very low peaks in these plots can be interpreted as regions in the curve with a very high curvature in the same direction. In regions of the curve where a finger top is present, the sum of the DOS values will be very high. On the other hand, in a region in which a finger valley is present, the sum of the DOS values will be very low. When the DOS values in a region vary highly in both directions (e.g. when much noise is present in the contour), the values will cancel each other out when taking the sum. The cumulative DOS function will therefore be very flat in this region.

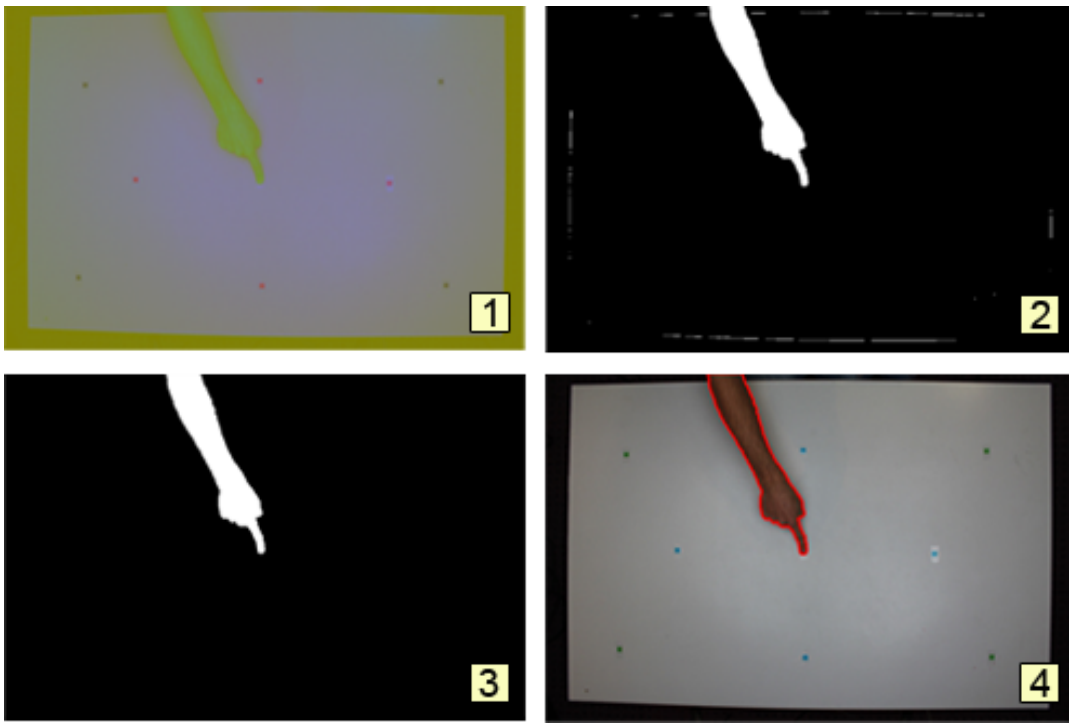


Figure 6.10: 1. An image of a human hand in the YCrCb color space. 2. Result of the skin segmentation. 3. Result of the opening and closing filter. 4. The extracted contour visualized in the original image.

$$CumulativeDOS_i = \sum_{j=i-3}^{3} DOS_j \tag{6.1}$$

The regions in which a potential finger top is located can be extracted by thresholding the cumulative function at -100 degrees. By thresholding this function at 50 degrees, regions in which a potential finger valley is located will be extracted. Afterwards, we label the center points of these regions as a potential finger top/valley (see Figures 6.12-4 and 6.13-4). In Figure 6.13-4, we can detect some points that are erroneously labeled as finger tops or valleys. This is primarily because the bones at the base of the fingers have the same curvature as the finger tops in this posture. These points can be eliminated by thresholding the size of the fingers or the angle between the two phalanges of the fingers (see Figure 6.13-5). We can see in Figure 6.12-4 that the beginning of the thumb and end of the little finger are not detected. These two points require special treatment because they are located in low curvature regions. We will estimate these points with extra conditions (see Figure 6.12-5).

6.4.3 Wrist Detection

Before we can extract the back of the hand from images, the detection of the wrist is very essential. However, this region is sometimes difficult to detect reliably because in an image captured from above, the hand sometimes blends perfectly into the arm in some postures. It is also not possible to base or entire algorithm on the position of the fingers extracted in Section 6.4.2, because in some postures no fingers are visible (e.g. when touching a screen vertically). There are two steps in our wrist detection algorithm: the extraction of the direction of the wrist and the position of the wrist.

The Direction of the Wrist

Instead of trying to detect the direction of the wrist directly, it is easier to extract the direction of the arm first. Choi et al. [22] presented an algorithm to approximate the direction of the

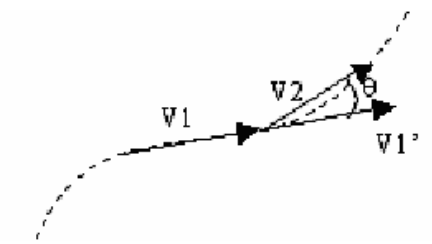


Figure 6.11: Calculating the DOS of a curve.

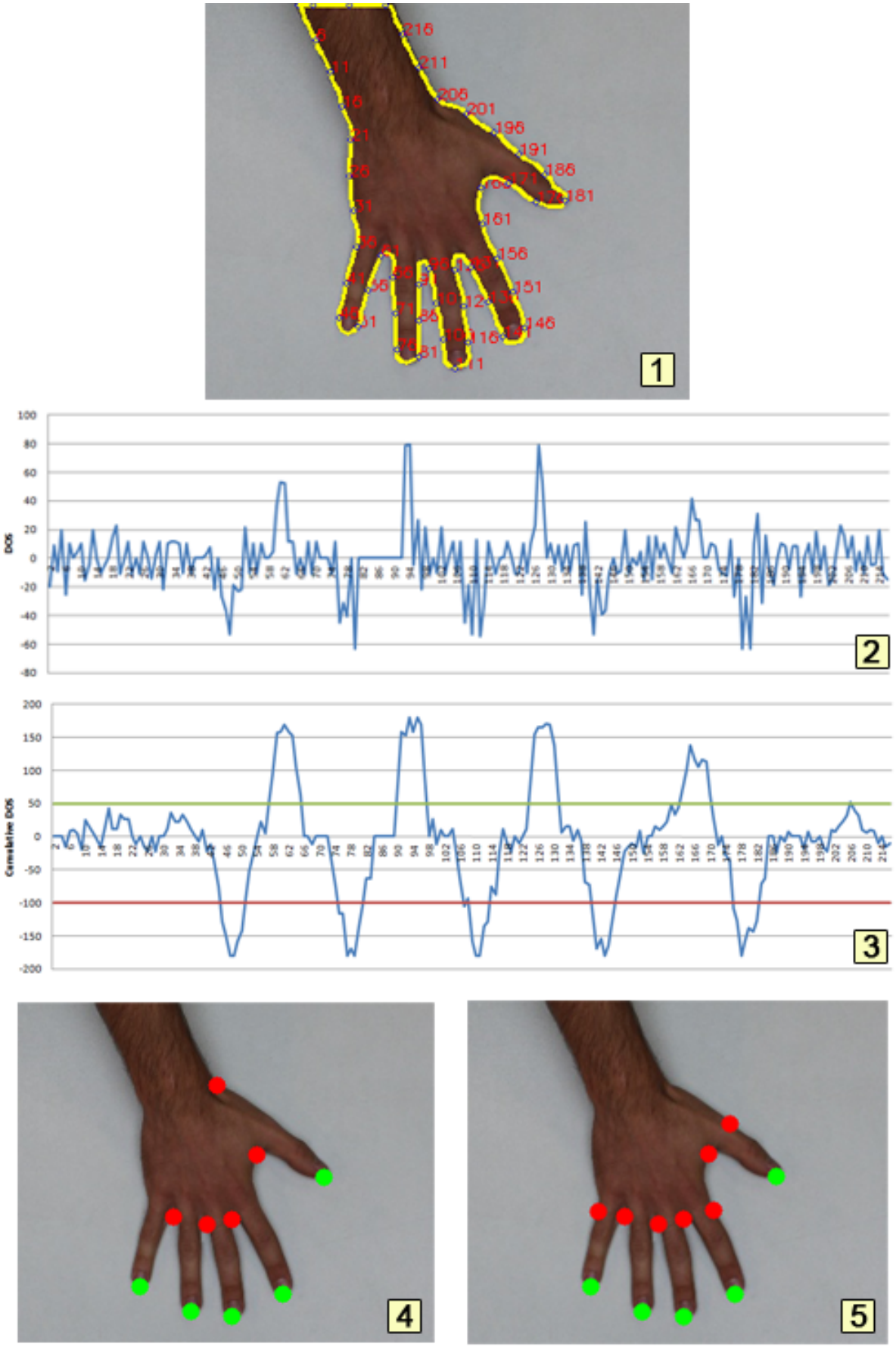


Figure 6.12: 1. A flat hand extracted from an image (skin segmentation). 2. The DOS values of the contour in each point. 3. The cumulative DOS values. 4. The extracted finger tops and valleys obtained by thresholding the cumulative DOS values. 5. The final result.

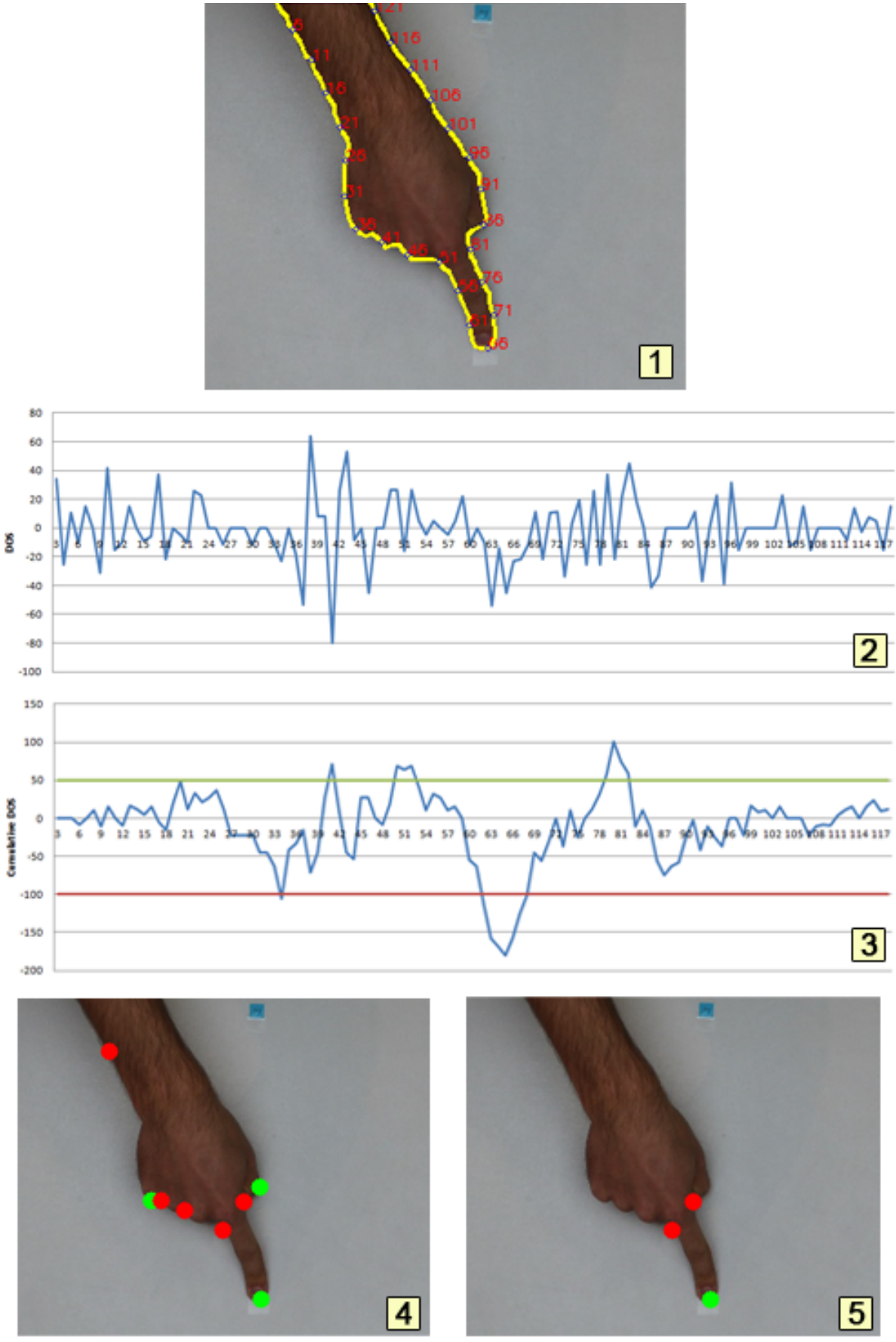


Figure 6.13: 1. A hand in a pointing posture extracted from an image (skin segmentation). 2. The DOS values of the contour in each point. 3. The cumulative DOS values. 4. The extracted finger tops and valleys obtained by thresholding the cumulative DOS values. 5. The final result.

arm by fitting a line to the contour points of the hand. We noticed however that their approach was not very accurate and the technique even failed when the hand bends to the left or right when only a small portion of the arm is visible. Therefore we take another approach.

Our algorithm is based on the assumption that the longest straight line in the arm region of the hand contour approximates the direction of the arm most perfectly. Before this line can be extracted however, we need to detect the arm region. This region can only be roughly approximated; if we could discriminate this region very precisely the problem of detecting the wrist was already solved. To be able to approximate the arm region we will first detect the start and end point of the hand contour. The start point of the hand contour is defined as the rightmost intersection of the arm with the border of the image. The end point is defined as the leftmost intersection of the arm with the image border. These points can be easily detected by searching for the points in the contour that are located very close to the border of the image. When using this approach however, we assume that an intersection of the skin region with the border of the image exists. This is not the case when for example the sleeve of a shirt is visible in the image (see Figure 6.14-1). In this case, we will do a background subtraction with an image of the scene that was previously captured when no skin regions were visible (see Figure 6.14-2). When thresholding each channel of the resulting image, we get a binary image on which the entire arm is visible (see Figure 6.14-3). The contour of this binary image can be analyzed and the intersections with the image border can be detected easily, as we just described (see Figure 6.14-3). Now that we have the intersection of the entire hand with the image border, we can map these points to the contour of the skin region (see Figure 6.14-5). The start and end point of the contour define a line which we will call the *baseline* of the arm (see Figure 6.14-5).

In order to estimate the arm region we first approximate the length of the entire contour (hand + arm). This can be done by calculating the perpendicular distance between the baseline of the arm and the farthest point of the contour which is probably a fingertip. Now that we have the approximate length of the contour, we can calculate the length of the arm region by subtracting the maximal length of a hand which was chosen by experiment. The resulting length is the minimum length of the arm that is visible in the image. We can even take into account the length of the fingers that we extracted in Section 6.4.2 to make this approximation more precise. With the baseline of the arm and the minimum arm length at hand, the contour points that are definitely inside the arm region can be selected (see Figure 6.15-1).

Lastly, the longest straight line in this region needs to be selected, as this line will represent the final direction of the arm. Finding straight lines in a contour can be done by simplifying the contour of the hand (reducing the number of points) with the Douglas-Peucker algorithm [31], implemented in the OpenCV library. This algorithm reduces the number of vertices in

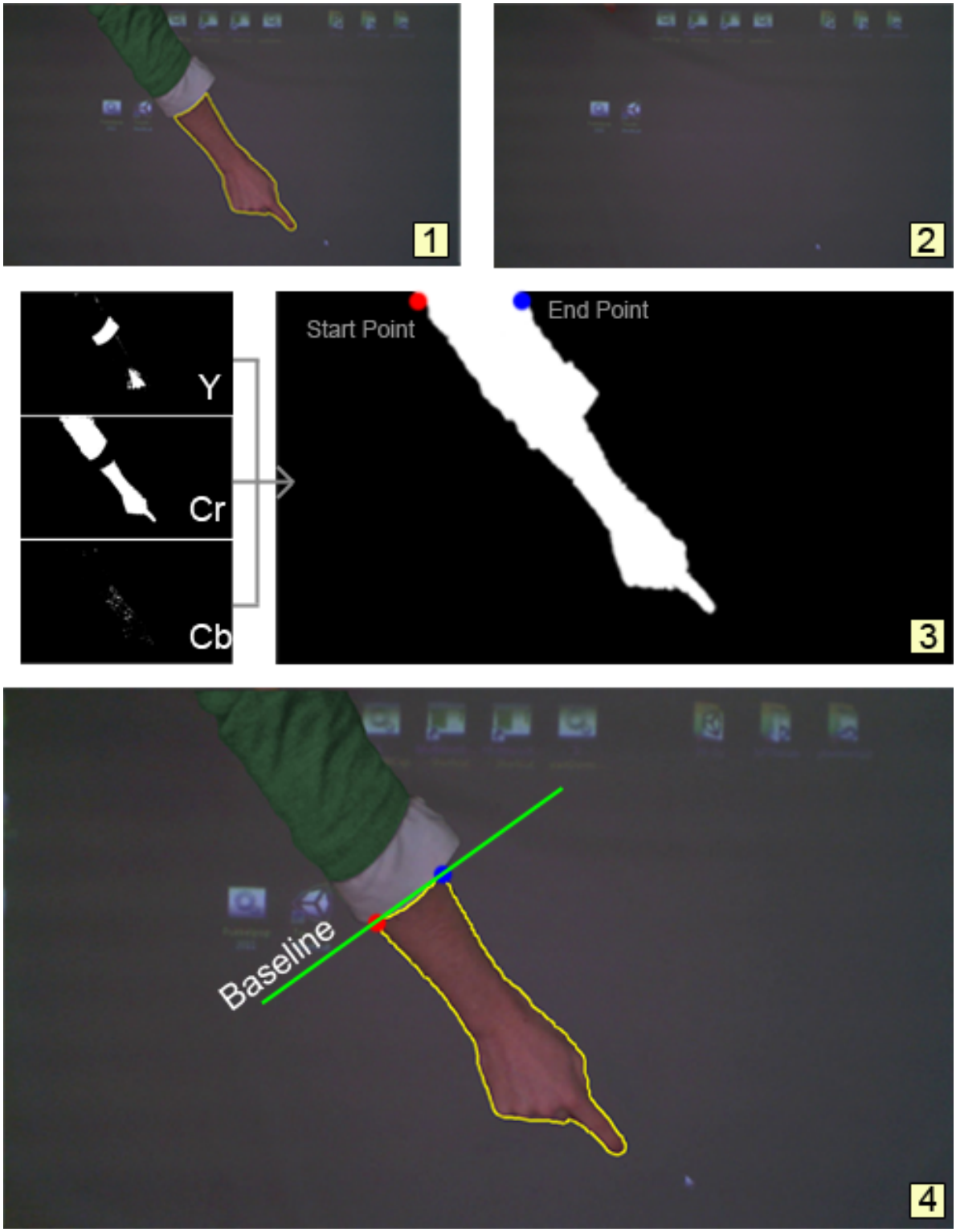


Figure 6.14: 1. An image in which the sleeve of a shirt is visible. The skin region is already highlighted (see Section 6.4.1). 2. The reference image. 3. Background subtraction of image (1) with image (2). 4. The resulting image in which we can now detect the base line of the arm.

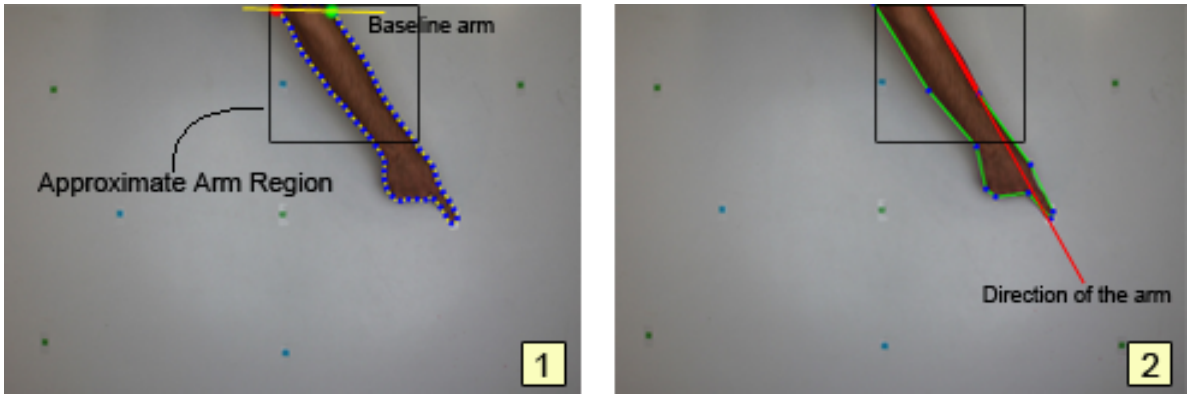


Figure 6.15: 1. An image in which the skin region and the baseline of the arm is detected. 2. Extracting the final direction of the arm.

the contour, so that the distance between the original and reduced contour is less or equal than a specified precision (see Figure 6.15-2). Now that the longest line of the arm can be found, we have a very good approximation of the direction of the arm and also the wrist (perpendicular to the direction of the arm).

The Position of the Wrist

Before the wrist of a hand can be extracted precisely in an image, we need to detect the position of this wrist in addition to the direction that was calculated in the previous step. To determine this position, we first calculate the distance of the hand contour along the direction of the wrist (see Figure 6.16). This distance can be computed at a specific position by measuring the distance between the two outermost intersections of the hand contour with a line that has the same angle as the wrist. Figure 6.17 (blue function) shows a plot of the lengths of all lines in Figure 6.16. The variations in these distances have very specific characteristics at the position of the wrist: in the direction of the hand they will vary significantly but in the direction of the arm these distances will be relatively constant. It is easy to measure the variation of the lengths of the lines before a specific point by computing the backward difference in a neighborhood (see Formula 6.2). To measure the variation of the lengths of the lines after a specific point, the forward difference can be computed (see Formula 6.3). Figures 6.18 and 6.19 (blue functions) show plots of respectively the backward and forward difference (with a neighborhood size of 5) for each point of the (blue) function in Figure 6.16. The red functions in these graphs provide a rough estimate of the region in which the final wrist will be located. This region was specified using the length of the arm in the longitudinal direction, two predefined constants (chosen by experiment) which represent the minimum and maximum length of the back of a human hand and the length of the fingers that were extracted in Section

6.4.2.

$$BackwardDifference_i = \sum_{j=i-5}^{i-1} |Distance_i - Distance_j| \tag{6.2}$$

$$ForwardDifference_i = \sum_{j=i+1}^{i+5} |Distance_i - Distance_j| \tag{6.3}$$

The final position of the wrist will be located in the valid region of the wrist at the position where the backward difference has a high value (much variation in distances in the direction of the hand) and the forward difference has a low value (the computed distances are relatively constant in the direction of the arm). By maximizing this ratio a single point can be selected. This point represents the final position of the wrist. For the current example, this ratio is maximal at position 13. Line 13 in Figure 6.16 will therefore be selected as the final wrist.

Now that we have extracted the skin region (see Section 6.4.1), the finger tops and valleys (see Section 6.4.2) and the position and direction of the wrist (see Section 6.4.3), it is straightforward to extract the back of the hand (see Figure 6.20). This image can now be used as input for the feature extraction techniques discussed in Section 6.3.

6.5 Conclusion

In the first part of this chapter we discussed which parts of the human body are the best candidates to extract features from. We concluded that the back of the hand is a very good candidate because it is the largest region of the hand that is the least flexible. This makes it easier to match regions over different postures. The back of the hand contains many features that are likely to be unique over different people, such as the shape, color, texture, etc. These potential features can all be captured by an overhead camera because this region of the hand is visible to the camera most of the time. In Section 6.3 we provided an extensive overview of techniques that can be used to extract these features. To test these techniques on a large scale we need algorithms to extract different parts of the hand. Therefore, a hand recognition technique was discussed in Section 6.4.

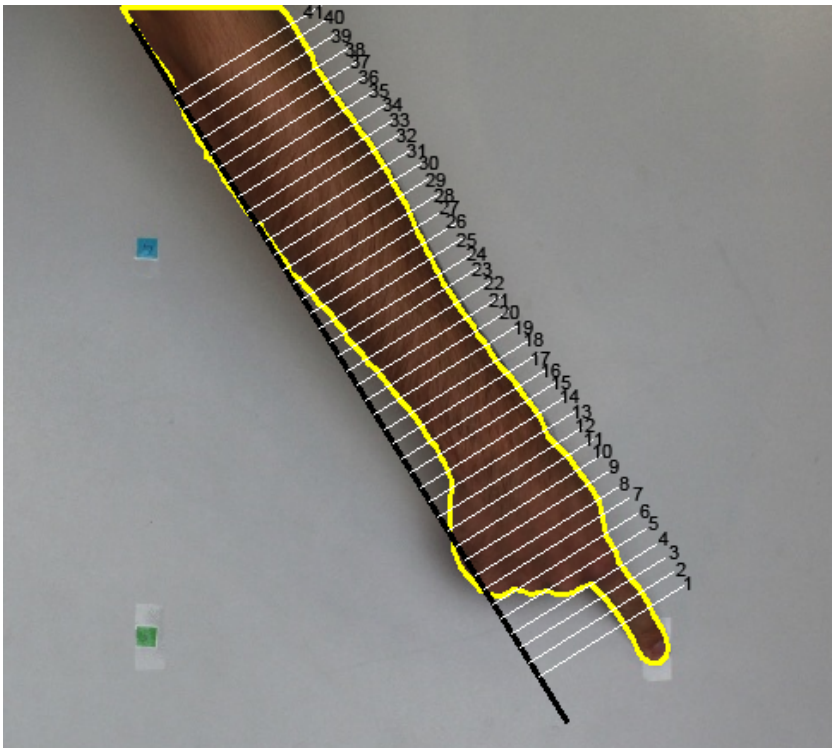


Figure 6.16: Determining the position of the wrist.

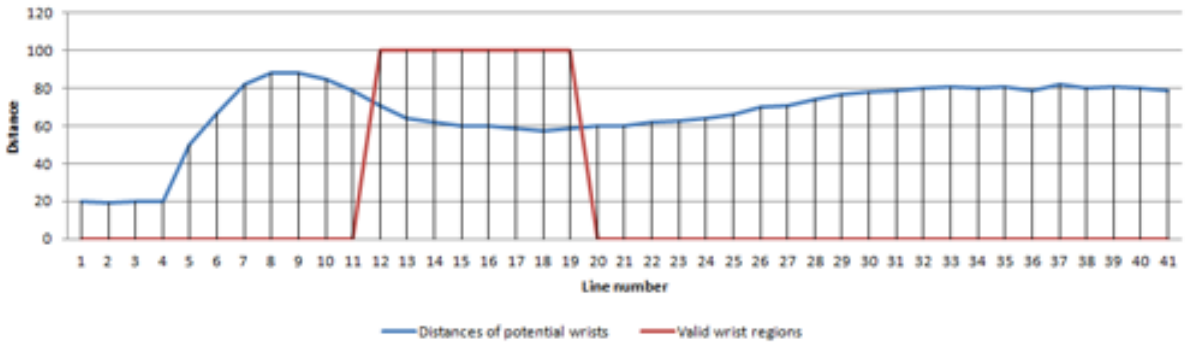


Figure 6.17: Blue function: the lengths of all potential wrists in Figure 6.16. Red function: the valid region for the wrist.

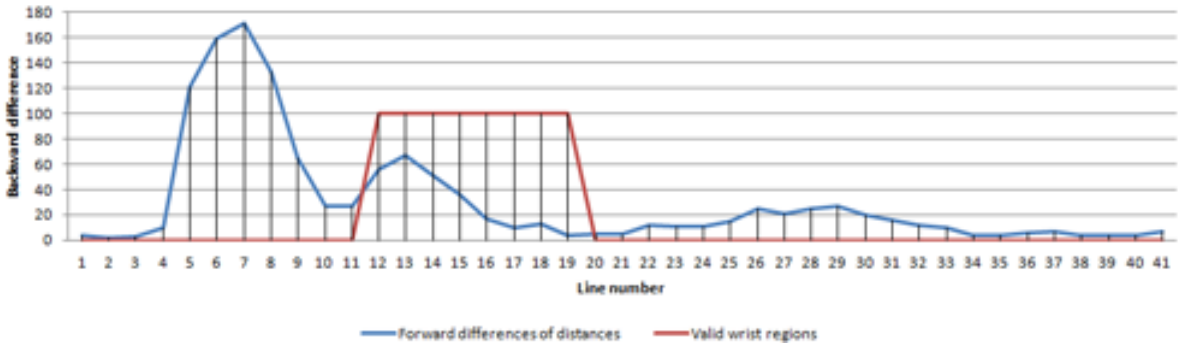


Figure 6.18: Blue function: the backward difference (neighborhood size of 5) for all the values of the blue function in Figure 6.16. Red function: the valid region for the wrist.

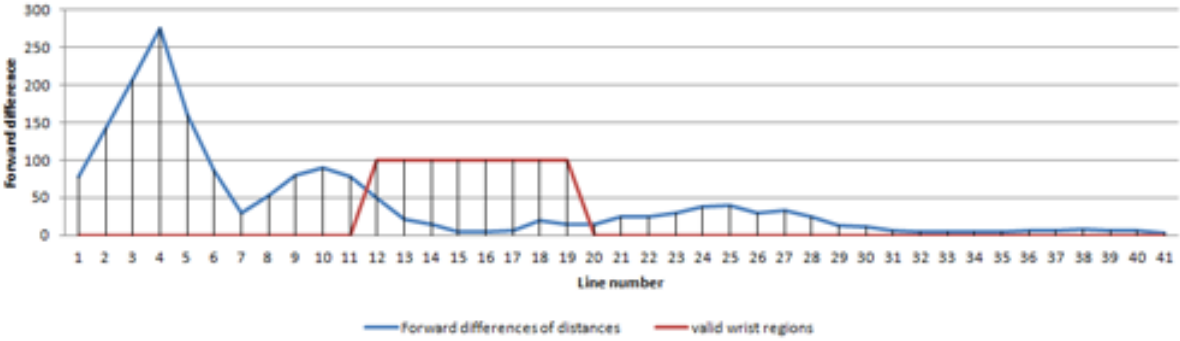


Figure 6.19: Blue function: the forward difference (neighborhood size of 5) for all the values of the blue function in Figure 6.16. Red function; the valid region for the wrist.

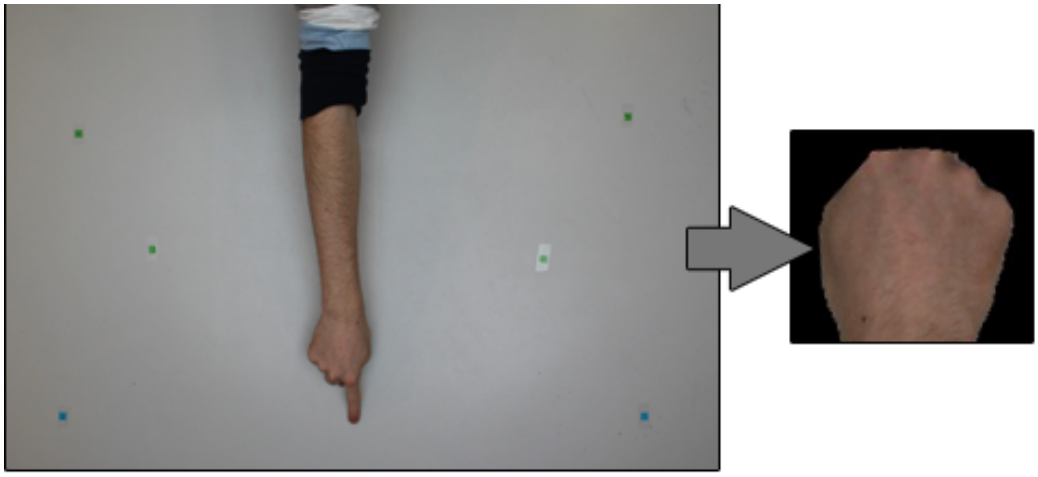


Figure 6.20: The back of the hand extracted from an image of a hand on a table captured by an overhead camera.

Chapter 7

Study Setup and Execution

7.1 Introduction

In Chapter 6 we proposed that users can be uniquely identified on an interactive display by analyzing the human hand captured by an overhead camera. We presented 4 potential techniques to extract biometric features from the back of the human hand. In addition, we discussed an algorithm to extract this part of the hand in a captured image. In this chapter, two studies will be described which were conducted to test the reliability of the feature extraction techniques. The results of these tests will not only be used to check if the technique works in practice and how unique the features are, but also to improve the feature extraction algorithms presented in Section 6.3. First of all, we conducted a quick-and-dirty test to see if there are techniques that can quickly be excluded because they are definitely not reliable enough. This first test is also used as a pilot for the more extensive study. The results of the second, larger study will be examined in more detail so that we are able to give an indication of how good our final technique works and how it can be improved.

7.2 Quick-and-Dirty Test

The purpose of this test is to get a first indication of the suitability of the feature extraction techniques for biometric identification of users. Furthermore, the results of this test can give more insight in the specific requirements for each technique. For example, the required lightening condition, minimal dpi of the camera, number of images needed to train and test the system, necessary hand postures and covered regions of the screen to simulate a real interaction on a multi-touch screen. If it already turns out during this preliminary test that some techniques are not useful to realize our envisioned non-intrusive identification technique (see Section 6.1.1), than these techniques can be excluded.

7.2.1 Tasks & Procedure

Each participant was instructed to place his hand flat on a traditional table at four positions with the fingers spread. Afterwards, the participants were instructed to point at four different positions on the table as if they were touching that region of the table. When the participant performed a posture, a picture was taken by an overhead camera.

7.2.2 Apparatus

For this study a traditional off-white table was used (see Figure 7.1). We used a digital still camera (Canon EOS 500D) to capture the users' hands at 150dpi. Compared to video cameras and webcams, still cameras have better optics, larger sensors, and longer exposure times are permitted. This makes it possible to capture very fine grained details of the hands and reduce factors such as noise. We also installed a couple of artificial light sources to illuminate the surface and the hand evenly because there are some algorithms that depend on the luminance in the image (e.g. the color matching technique discussed in Section 6.3.2).

7.2.3 Participants

We recruited 8 participants (1 female). All participants were between 22 and 30 years old.

7.2.4 Results

After the collection of all the data, we extracted all the hands from the captured images using the hand recognition algorithm described in Section 6.4. From these results we could already give a decisive answer for the hand metrics and the color matching techniques.

Hand Metrics

As we already discussed in Section 4.6.2, Schmidt et al. [112] extracted geometry features from a hand that was placed flat on a multi-touch surface. In Section 6.3.1 we proposed to use this geometric feature extraction technique to extract metrics from the hand in other postures (e.g. while pointing to a position on an interactive surface). Figure 7.2-1 shows 4 images of the same participant while pointing at different positions on the table in the same posture. The results of the hand recognition step are shown in Figure 7.2-2. We can see that although the hand postures are the same, changing the position of the hand on the table causes different regions of the hand to be visible. As a result, the extracted geometry features will vary too much over different images to relate them all to the same user identity. These variations in



Figure 7.1: Setup of the quick-and-dirty test.

geometry metrics cannot be taken into account by training the system with images of the users' hands in all possible regions of the surface because we would definitely overtrain the system. Another problem that we noticed during the analysis of the captured images was that although the camera is mounted at a fixed distance from the table, the distance to the hand can still vary. This is because the back of the hand is located a couple of centimeters higher when performing a vertical touch compared to a horizontal touch. This variation can also ruin our extracted geometry metrics.

Color Matching

In section 6.3.2, we proposed the use of skin color as a feature to discriminate between different users. During the analysis of the extracted images we noted that even though extra artificial light was present while capturing the images, there are still quite some differences in skin color of the same participant over multiple images (see Figure 7.3). These changes in illumination of the hand are caused by the differences in shadows that the participant's body casts on his own hands while touching different regions of the surface. For this reason, the rough skin color is not very useful for biometric identification. Techniques such as local binary patterns (see Section 6.3.4) that are less sensitive to changes in illumination will probably provide more reliable features.

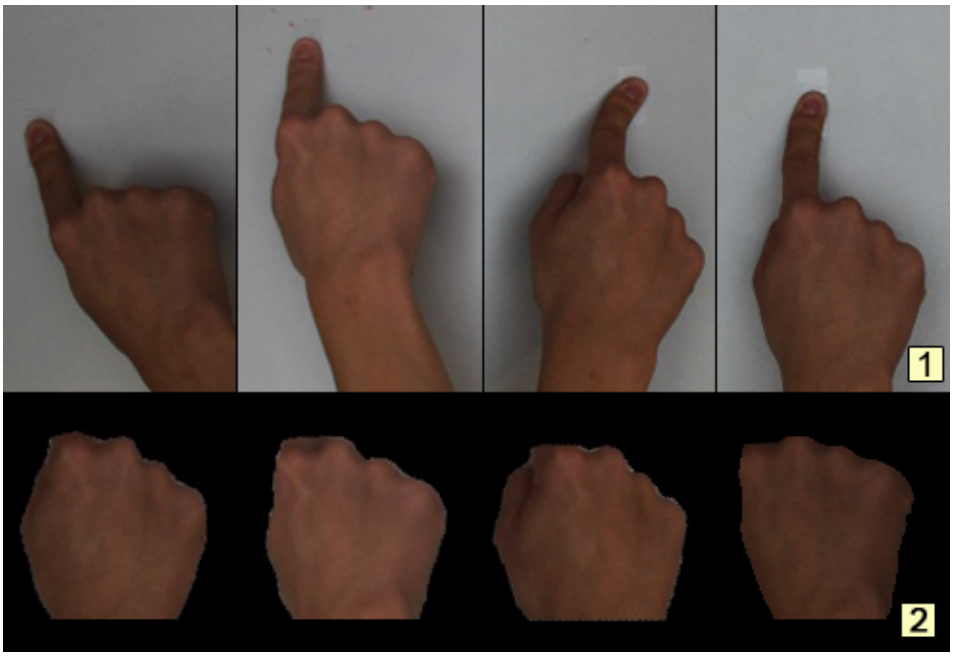


Figure 7.2: 1. Images of the hand of a single participant pointing at different regions of the surface in the same posture. 2. The hands extracted using the hand recognition algorithm discussed in Section 6.4.



Figure 7.3: Differences in the illumination of the hands because of the shadows casted by the body of the participant.

The other two techniques that we presented earlier, eigenimages and local binary patterns, provided more promising results and gave the impression to be able to match the extracted features reliably. However, with this very small dataset (small amount of users and training images) we are only authorized to exclude techniques that will definitely not work. When considering only the data of this small study we cannot claim that any of these techniques really work. For this reason, we conducted a second more extensive study.

7.3 Extensive Study

The results of the quick-and-dirty test suggest that the eigenimages (see Section 6.3.3) and the local binary pattern technique (see Section 6.3.4 are the most promising techniques to extract features from the back of the hand. Therefore, we will concentrate on these two techniques in this study. A large dataset is needed to test if these techniques are really suitable in highly interactive systems and to test how unique their features are. For these reasons, we will construct a large dataset which consists of images of hands at different positions on the surface in different postures.

7.3.1 Tasks & Procedure

We marked 9 positions on a traditional table that participants needed to touch (see Figure 7.4). These positions were chosen to cover the major regions of the surface. In addition, 3 important postures were identified that participants needed to adopt while pointing at these points. The first posture is *handsdown* (see Figure 7.5-1). In this posture, a hand with fingers spread is placed flat on the surface. The rotation of the hand can only change according to the x-y plane. This is also the rotation that we extracted in the hand recognition algorithm (see Section 6.4). As a result, the extracted region can be oriented almost perfectly and corresponding features can be found more reliably in images from the training set. Images of hands in this posture are therefore best suited to verify the uniqueness of the extracted features over multiple participants. The second posture is *pointing* (see Figure 7.5-2). While pointing at a point on the table, the rotation of the hand can change in all three dimensions which can result in a reduction of the accuracy of the feature matching algorithm. This is because features cannot be matched exactly and parts of the skin region can be skewed. However, we suppose that this posture is used mostly during touch interactions. The third posture is *free hand pointing*(see Figure 7.5-3). This posture is used most often when actions need to be performed fast and instantly (i.e. when the subject has no time to shape their hands perfectly in a pointing posture). It has the same properties regarding the rotation of the hand as the traditional pointing posture.

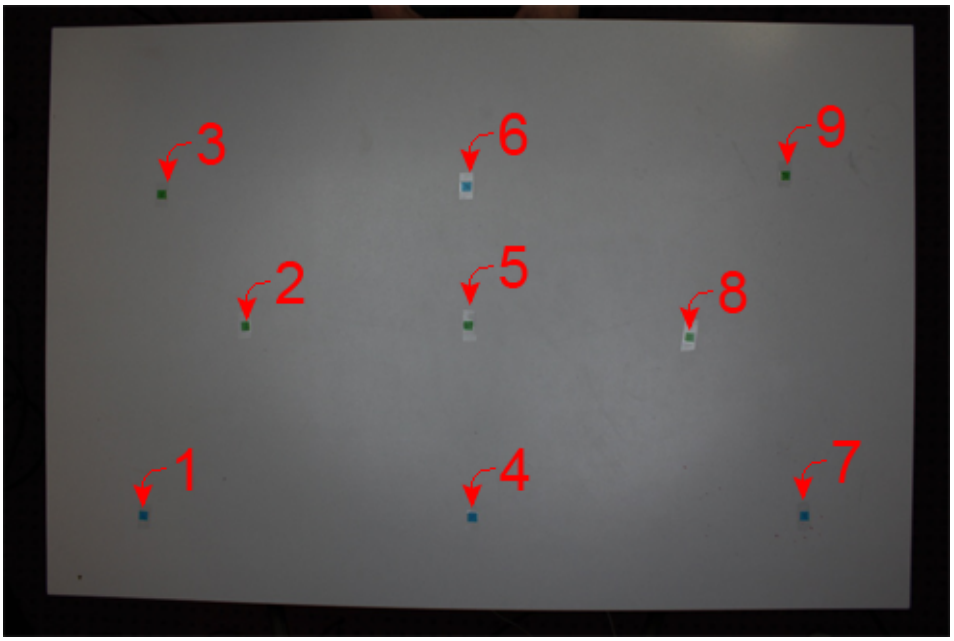


Figure 7.4: Marked touch positions on the table.



Figure 7.5: 1. The handsdown posture. 2. The pointing posture. 3. The free hand pointing posture.

Each participant needed to perform 5 touches while adopting the handsdown posture, 9 touches in the pointing posture and again 5 touches while adopting the free hand pointing posture. For each of these tasks, instructions were displayed on a screen (see Figure 7.6). Users could see, for example, the position at the table that needed to be touched and the required hand posture for the current task. While performing all these tasks, participants were instructed to sit in front of the table to make the test more comfortable and to prevent occlusion of the camera with the head or upper body (see Figure 7.6). Furthermore, when sitting in front of a table, it is less likely that a user touches the surface with the back of his hand entirely vertical to the camera. However, based on observations in the quick-and-dirty study (see Section 7.2) we noticed that when we give participants the task of touching points on a table in a specific posture, they sometimes touch the surface in a very unnatural posture. We think that this is because they focus too much on the touching aspect which is not the case when using an application with touch input. For this reason, we decided to control the overhead camera ourself instead of letting the user commit the input using for example a footswitch. This enables us to correct the posture when the back of the hand is not visible to the camera. Before each participant started with the tasks, we also captured a new picture of the empty table (see Figure 7.4). This image will be used as reference image in the background subtraction step of the hand recognition algorithm (see Section 6.4.3).

7.3.2 Apparatus

For this study, we also used a traditional off-white table as touch surface (see figure 7.6). The digital still camera (Canon EOS 500D) was mounted at a height to capture pictures of the entire table at 100dpi. Because we installed our setup in the lab, we were able to mount four artificial light sources at the ceiling. These lamps are oriented towards the white ceiling. Light will be reflected on the table and this results in an even illumination.

7.3.3 Participants

We recruited a new set of 24 participants (7 female). All participants were between 21 and 35 years old. With this number of participants we are able to collect 456 pictures of hands: 120 for the handsdown and free hand pointing posture, and 216 for the pointing posture.

7.3.4 Results

First of all, the eigenimages technique was tested with images of hands in the handsdown posture, because the skin region can be extracted and orientated most reliably in this posture. However, the results of this technique were not very promising, even when only a small group



Figure 7.6: A display instructed the participants to touch points on the table in specific postures. A digital still camera captures images of the table which is evenly illuminated by artificial light sources.

of users were registered with the system. We think that this is due to the fact that eigenimages requires all images to be almost perfectly aligned. Furthermore, the reliability of the technique drops significantly when there are changes in illumination over the captured images. Even with extra illumination sources, shadows are casted on the captured hands by the participant's body. We think that this technique is more suitable for matching other regions of the hand, such as finger nails, because it is easy to orientate a finger almost perfectly.

While analyzing the results of the local binary pattern technique (see Section 6.3.4), we were surprised that this texture matching technique produced very good results. An extensive discussion of the results of these tests will therefore be provided in this section. These results will not only be used to evaluate the reliability of our user identification technique, but also to get more insight in the optimal values of parameters, to be able to improve our algorithm. We will investigate the effects of 5 independent variables:

1. **Posture:** Each participant was instructed to point at positions on the table in 3 postures. We will investigate the reliability of our identification technique for all these postures.

2. **Resolution:** The extracted images from the back of the users' hands can have different qualities depending on the overhead camera that was used. It is therefore interesting to see how our technique behaves when changing the quality of the images.

3. **Patch size:** In Section 6.3.2, we concluded that the ideal size of the patches used when matching local binary patterns (see Section 6.3.4) can only be determined by analyzing experimental results. To be able to optimize our algorithm, we will therefore evaluate the recognition rate of our identification technique using different patch sizes.

4. **Patch neighborhood size:** In Section 6.3.2, we also discussed that the area in which to search for a patch needs to be restricted. We concluded that the ideal size of this neighborhood depends on the quality of the hand recognition algorithm and the amount of training data. Therefore, we will try to find an ideal value for this variable for our test setup.

5. **Radius of the CLBP-operator:** In Section 6.3.4, we described that when a small radius is used for the CLBP-operator, the technique is able to encode local details. On the other hand, when a larger radius is used, more general features will be encoded. It is interesting to investigate if the extracted features are more discriminative when fine grained details are encoded, instead of more general patterns.

Posture

As we described in Section 7.3.1, each participant needed to touch the surface of the table at 9 marked positions for the pointing posture, 5 positions for the handsdown posture and also at 5 positions for the free hand pointing posture. We are interested in the reliability of our identification technique over these three postures, therefore, we generated 3 scenarios: one for the pointing posture, the handsdown posture and the free hand pointing posture. In all three scenarios we used one picture of the hand of each participant as test data and all other pictures as training data and we did this for all possible combinations. As a result, we have 9 positions x 24 participants = 216 trails for the pointing scenario and 5 positions x 24 particpants = 120 trails for the handsdown and also the free hand pointing posture.

We used the following configuration for this test:

- Resolution: 50 dpi
- Patch size: 20 px
- Patch neighborhood size: 2 cm
- Radius of the CLBP-operator: 2 px

The *recognition rates* for each scenario are shown in Figure 7.7. The *recognition rate* indicates how many users are identified correctly. A recognition rate of 100% (or 1.0) means that all test images are related to the correct user identity. In contrast, a recognition rate of 0% (or 0.0) means that all test images are related to the wrong user identity. We also computed the *match rate* for each scenario (see Figure 7.8). The blue boxplots visualize the *positive match rates* for each scenario. The *positive match rate* indicates how similar a test image is to the most similar positive training image (with positive we mean that the identity of the training image corresponds to the identity of the test image). For example, an average positive match rate of 30% (or 0.30) means that the features in the tested images are on average 30% similar to the highest positive match found in the training data. In Figure 7.8, green boxplots visualize the *highest negative match rate* for each scenario. This rate indicates how similar a test image is to the most similar negative training image (with negative we mean that the identity of the training image does not correspond to the identity of the test image). For example, an average highest negative match rate of 20% or (0.20) means that the features in the tested images are on average 20% similar to the highest negative match found in the training data (the second best match). We excluded all trails in this graph in which the correct identity was not selected from the training data, otherwise this graph is not meaningful.

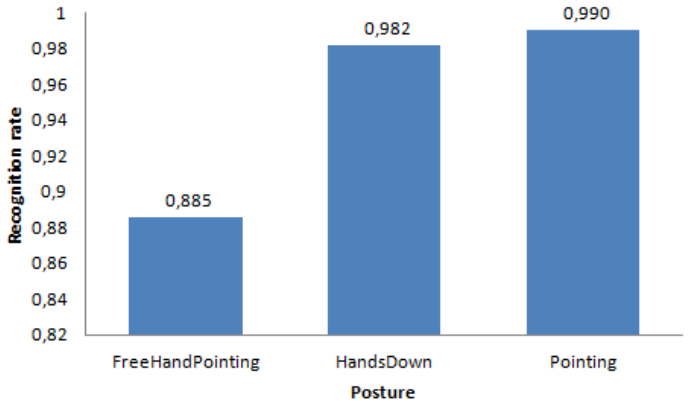


Figure 7.7: The recognition rate of the identification technique over multiple postures.

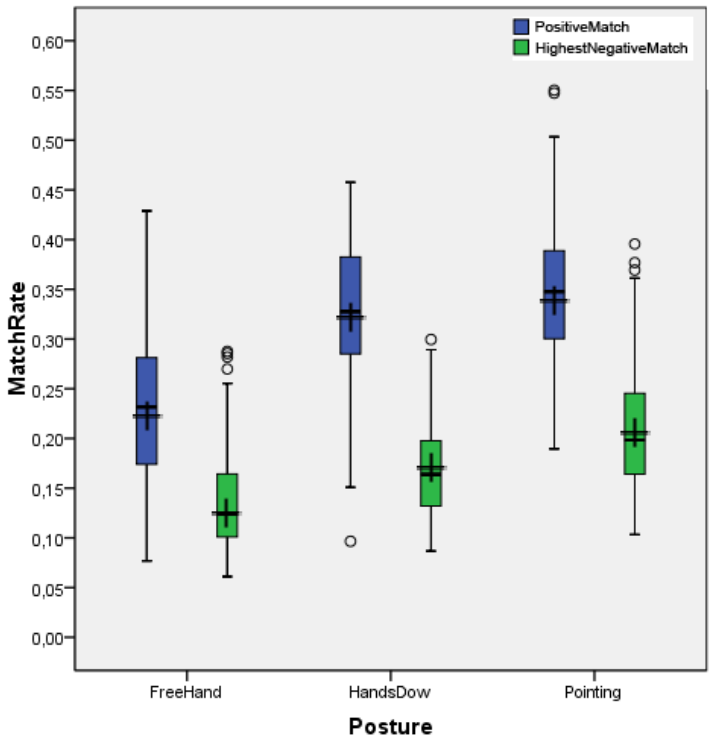


Figure 7.8: The match rate over multiple postures.

Figure 7.7 shows that for the pointing posture, our identification technique relates the correct user identity to the tested images 99% of the time. For the handsdown posture we get almost the same recognition rate as the pointing posture. However, for the free hand pointing posture, the recognition rate is significantly lower. When interpreting the match rates for this scenario (see Figure 7.8), we can see that the average positive match rate is much closer to the highest false match rate for the free hand pointing posture. This means that the similarity rate of the features of the tested images with the highest match in the training data is almost identical to the similarity rate with the second highest match in the training data. When there is not much difference between the two highest matches in the training data, the features are less discriminative, and it is more likely that the system will relate a tested hand to the wrong user identity. This causes the recognition rate to drop, as we saw in the free hand pointing scenario. After analyzing the images of the hands in the free hand pointing posture, we could find the cause for this reduction of the discriminative power of the features in this specific posture. We noticed that our hand recognition algorithm (see Section 6.4) was not able not extract the back of the hands in this posture as precisely as for the other postures. We observed that in some situations parts of the fingers and the table were still visible. These errors can be the result of shadows around the hand that are more present in this posture, because a larger region of the hand floats above the table. Another reason for the errors in the hand recognition step is that finger valleys are sometimes partially occluded by other fingers in the free hand pointing posture. Because of these two errors, more shadow and noise regions are visible around the extracted hands. When our algorithm creates patches for these regions, they will match with many other regions in which this error is present. Therefore, the features will have less discriminative power in this situation and the recognition rate will drop.

Resolution

The extracted images from the back of the users' hands can have different qualities depending on the overhead camera that was used. It is therefore interesting to see how the recognition rate of our identification technique varies when we change the resolution of the images. This investigation will only be done for the pointing posture because this posture has the most trails. We will therefore be able to produce more reliable results. In the previous test, we saw that the recognition rates for the three postures were very similar apart from the free hand pointing posture. We explained that this was only because of errors in the hand recognition algorithm. Therefore, we expect that changing the resolution of the images will have the same consequences for all the postures.

We used the following configuration for this test:

- Patch size: 20 px

- Patch neighborhood size: 2 cm
- Radius of the CLBP-operator: 2 px

In this test, we have six scenarios: 10, 15, 25, 50, 75 and 100 dpi. For each scenario, there are 9 positions x 24 participants = 216 trails. Figure 7.9 shows the recognition rates for each scenario. The recognition rate increases when more detail is visible in the images of the hands. However, at 100 dpi the recognition rate drops significantly. When we take a look at the similarity graph (see Figure 7.10), we can explain this remarkable behavior. In scenarios of 10 dpi and 100 dpi the patches include respectively very general information and very detailed information. In these situations it is very likely that the system can find corresponding patches of images in the training set with the right user identity. However, it is also very likely that the system can find corresponding patches of images in the training set with the wrong user identity. In Figure 7.10 this behavior is visible: both the positive match rate and the closest false match rate are relatively high and very close to each other at 10 and 100 dpi. As a result, in these situations the extracted features are less discriminative, and it is more likely that the system will relate a tested hand to the wrong user identity.

Patch Size

In Section 6.3.2, we concluded that the ideal size of the patches when matching local binary patterns can only be determined by analyzing experimental results. In this section we will evaluate what effect the size of the patch has on the recognition rate of the identification technique. We will also only test this for the pointing posture.

We used the following configuration for this test:

- Resolution: 50 dpi
- Patch neighborhood size: 2 cm
- Radius of the CLBP-operator: 2 px

In this test, we have five scenarios: patch sizes of 15, 20, 25, 30 and 40 px. For each scenario there are 9 positions x 24 participants = 216 trails. In Figure 7.11, we can see that the recognition rate drops when the patch size is larger than 25 px or smaller than 20 px. It was also easy to predict this behavior: when the patch size is very large the information inside becomes to general because we compare the histograms. On the other hand, when the patch size is to small, the information inside is very specific and the algorithm cannot find a similar patch in the training set. In the similarity graph (see Figure 7.12) the same findings

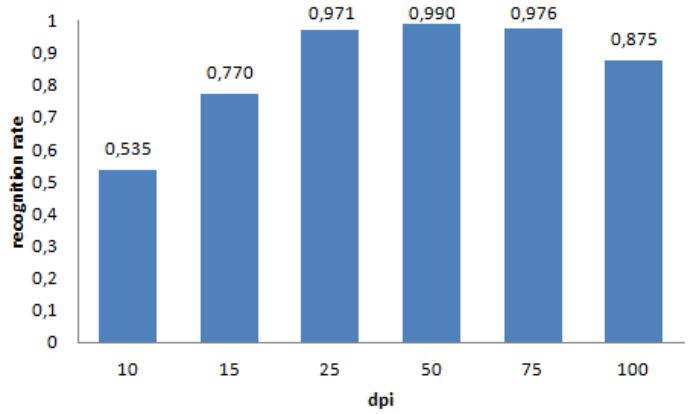


Figure 7.9: The recognition rate of the identification technique when changing the resolution of the overhead camera.

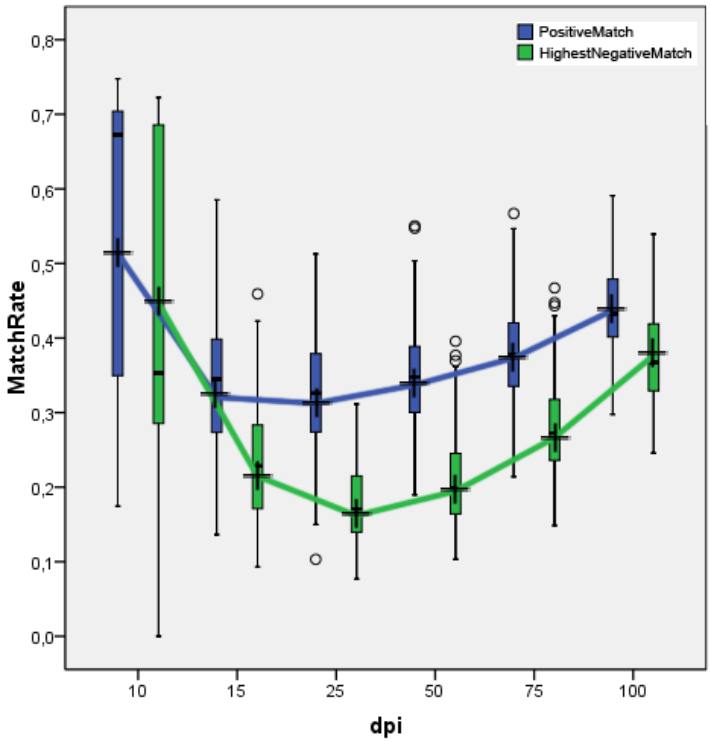


Figure 7.10: The match rate when changing the resolution of the overhead camera.

are present. Patch sizes 20 and 25 have almost identical recognition rates. However, the results indicate that patches of size 20 have more discriminative power because the difference between the positive match rate and the highest negative match rate is larger at this position. As a result, 20 px is the ideal size for the patches in our test case.

Patch Neighborhood Size

When finding a similar patch in the training set, the neighborhood in which to search for potential patches can be restricted as we discussed in Section 6.3.2. We concluded that the ideal size of this neighborhood depends on the quality of the hand recognition algorithm. Instead of defining an area in which to search for similar patches, we can also provide the system with so much training data so that the entire touch surface is covered (see Section 6.3.2). In this section, we will investigate the neighborhood size of a patch in depth and try to find the ideal value for this variable in our test setup. This test will also only be performed on the data of the pointing posture.

We used the following configuration for this test:

- Resolution: 50 dpi
- Patch size: 20 px
- Radius of the CLBP-operator: 2 px

In this test, we have 6 scenarios: neighborhood sizes of 0.25, 0.5, 1, 2, 3 and 4 cm. This size will always be expressed in centimeters because the value depends on the resolution of the images. For example, when the hand recognition algorithm can orientate a hand 10 px accurately, the neighborhood size can be set at a value to compensate this error. However, when the resolution changes, the error will also change and as a result, the neighborhood size in which to search for similar patches needs to change correspondingly. For each of these six scenarios there are 9 positions x 24 participants = 216 trails.

Figure 7.13 shows the recognition rate for each scenario. We can see that the recognition rate significantly drops when the neighborhood size in which to search for similar patches is smaller than 1 cm. This is because our hand recognition algorithm (proposed in Section 6.4) cannot orient the hands 100% precisely and therefore, a certain neighborhood size is aways needed (with our hand recognition algorithm, at least 1 cm). When the neighborhood size is very large (e.g 4 cm) on the other hand, the system can match a patch to a similar patch on the other side of a hand. However, it is remarkable that this does not cause a significant drop in the recognition rate (see Figure 7.13). After investigating this situation in depth, we concluded that our algorithm was still able to find the most similar patches in a very close

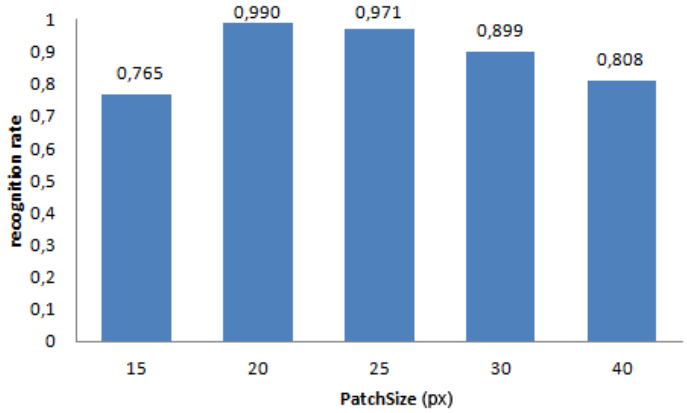


Figure 7.11: The recognition rate of the identification technique when changing the patch size.

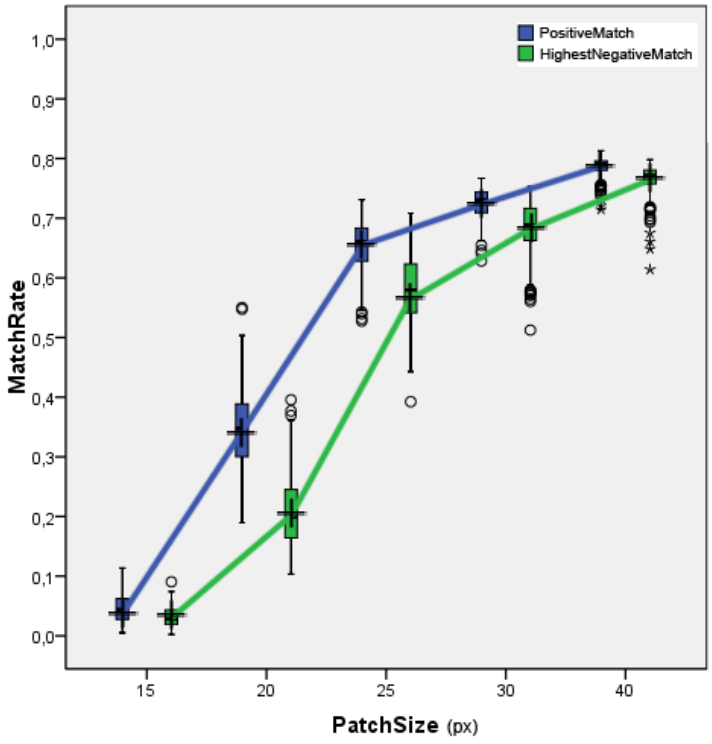


Figure 7.12: The match rate when changing the patch size.

neighborhood of the original patch. This indicates there are many differences in the features that are extracted from a single hand.

Apart from these findings, it is still recommended to use a relatively small neighborhood size because the execution time increases tremendously as more histograms of patches need to be matched. We select 2 cm as the ideal size for the patch neighborhood in our test setup, because this value provides a high recognition rate and also the greatest difference between the positive match rate and the closest false match rate in the similarity graph (see Figure 7.14). The differences between the positive match rate and the closest false match rate can be analyzed in more detail in Figure 7.15. For the neighborhood size of 2 cm, the difference in matching rate is 0.138%. This means that there is on average a difference of 13.8% between the best match and the second best match for an image of a hand. If one would prefer a better system performance, a patch neighborhood size of 1 cm can also be used in this situation.

Radius of the CLBP-operator

In Section 6.3.4, we mentioned that when a small radius is used for the CLBP-operator, more local details in an image are encoded. On the other hand, when a larger radius is used, more general features are encoded. It is interesting to investigate if the extracted features are more discriminative when fine grained details are encoded, instead of more general patterns. This test will also only be performed for the pointing posture.

We used the following configuration for this test:

- Resolution: 50 dpi
- Patch size: 20 px
- Patch neighborhood size: 2 cm

In this test, we have 5 scenarios: a radius of 1, 2, 3 and 4 px and the basic LBP-operator. For each of these 6 scenarios there are 9 positions x 24 participants = 216 trails. The recognition rates for all these scenarios are shown in Figure 7.16. We can conclude that CLBP-operator has more discriminative power when more local details (on a small radius) are encoded. However, the recognition rate also drops when using a radius of 1 px. We believe this is the result of the bilinear interpolation technique that is used when a point on the circle does not correspond to image coordinates. When using a radius of 1 px, the four corners of the neighborhood do not correspond with a single coordinate pair and will therefore get interpolated with two values on the circle and two other adjacent pixels. When afterwards all pixels are compared to the central value, the resulting pattern will not have much discriminative power because the values of four points on the radius depend on other values on this radius.

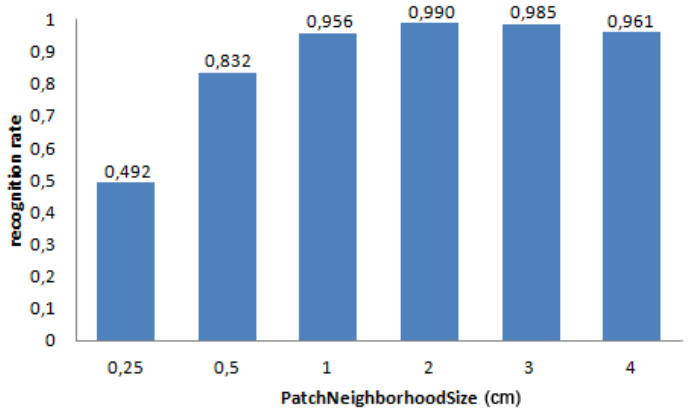


Figure 7.13: The recognition rate of the identification technique when changing the area in which to search for similar patches.

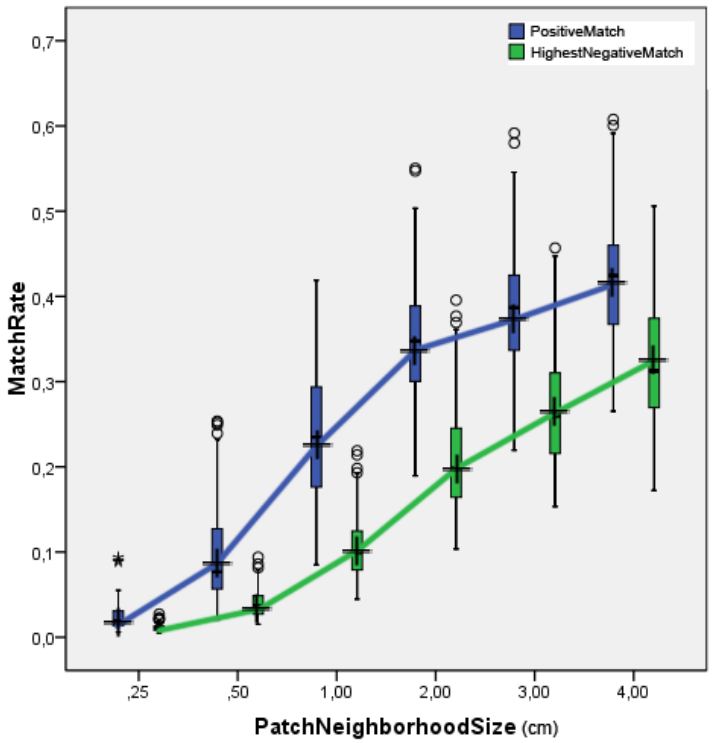


Figure 7.14: The match rate when changing the area in which to search for similar patches.

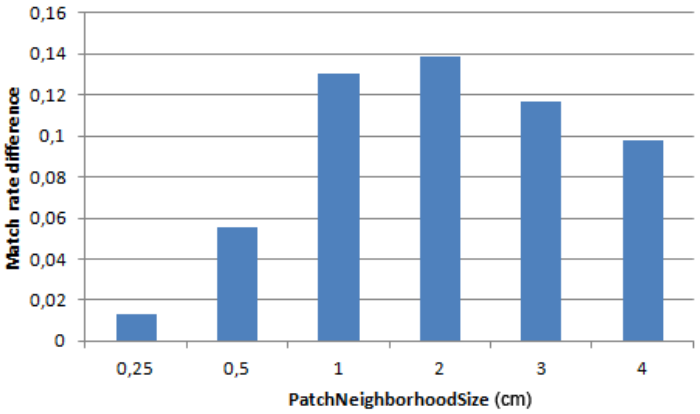


Figure 7.15: The difference in match rates between the positive match rate and the highest negative match in Figure 7.14.

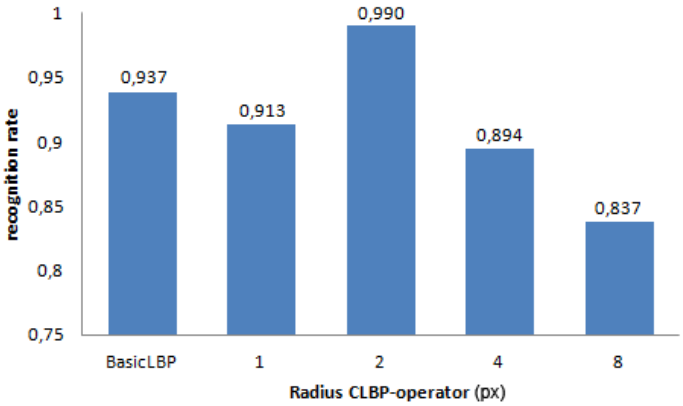


Figure 7.16: The recognition rate of the identification technique when changing the radius of the CLBP-operator.

We also tested our algorithm with different sample points on the radius of the CLBP-operator. The results of this test suggested that when less than 8 sample points where used, the patterns were not very discriminative and the recognition rate dropped. When using the uniform pattern encoding technique (see Section 6.3.4) and more than 8 sample points on the circle, the recognition rate dropped significantly. This was because the uniform patterns provided much less unique features than the original local binary patterns.

7.3.5 Discussion

From this study, we learned that when the eigenimages technique is used as the basis for our feature extraction technique, the system is not very reliable. However, local binary pattern encoding turns out to be a very powerful means for extracting unique information from the users' hand. Based on the statistical results that we presented in Section 7.3.4, we can come to the following conclusions:

- Our identification technique can discriminate between multiple users very reliably. The results indicate that our system selects the right identity up to 99% of the time, even when 24 users are registered. This suggests that our technique can even be used when a large group of users are working together around an interactive display.
- The technique works under multiple postures of the hand. This is a very important characteristic because it ensures that the non-intrusive aspect of our envisioned system is realized (see Section 6.1.1).
- An optimal resolution exists, for which the technique has the highest recognition rate. When the resolution is higher or lower than this optimum, the recognition rate will drop.
- There is also an optimal patch size for the patches used in the identification technique. When the patches are larger or smaller than this optimum, the extracted features are less discriminative.
- A threshold exists for the area in which to search for similar patches. If this neighborhood is very small, the system cannot find the most similar patch and the recognition rate will drop. However, the performance of the system will not drop when this area is relatively large, because the patches close to the original patch will still be the most similar. This suggests that our hand recognition algorithm does not need to be able to orientate the hands very precisely. However, the performance of the system can drop significantly when searching for similar patches in a large neighborhood.
- The recognition rate is higher when a small radius is used for the CLBP-operator (see Section 6.3.4).

- In all the similarity rate graphs (see Figures 7.8, 7.10, 7.12 and 7.14), we can see that the highest false match rate is always relatively close to the positive match rate. Therefore, we conducted an additional test to see if this phenomenon also occurs when only a small group of users is registered with the system. In this test 100 unique sets of 4 users where randomly generated which resulted in 4 users x 9 positions x 100 groups = 3600 trails. The resulting recognition rate was 99.85%. However, the positive match rate was also very close to the highest negative match rate (see Figure 7.17). These results suggest that our technique can only be used to differentiate between users and not for authentication, because it is not possible to define a threshold to detect the users that are not registered with the system.

7.4 Conclusion

In this chapter, we have evaluated the four feature extraction techniques that we presented in Chapter 6. First of all, we conducted a quick-and-dirty test (see section 7.2). From the results of this test, we could already exclude the hand metrics and color matching technique. The results indicated that these techniques were not suitable for extracting features from the users' hands. After this initial test, we conducted a more extensive study in which 24 subjects participated. After the evaluation of the *eigenimages* technique, we concluded that this technique could not discriminate different users very reliably. However, the local binary pattern technique produced very good results. We provided an extensive evaluation of this technique, and concluded that our system is able to identify users correctly up to 99% of the time even when large groups of users are registered with the system. However, our results suggest that this biometric identification technique can only be used to discriminate between users and not for authentication. Further investigation is needed to see if our technique can be extended to support this.

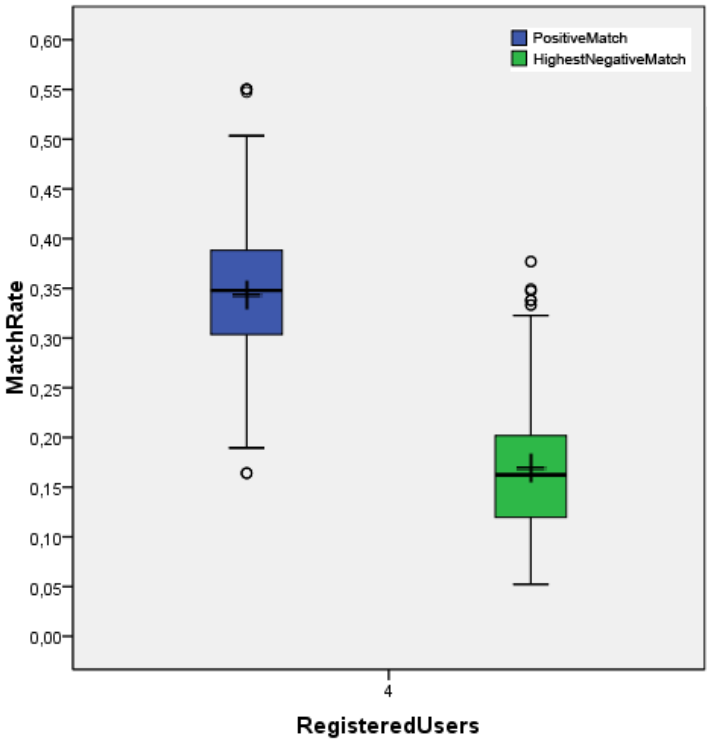


Figure 7.17: The matching rate when only 4 users are registered with the system.

Chapter 8

The Identity Server

8.1 Introduction

In this chapter, we will present a prototype of a system that integrates all findings from Chapter 6 and 7 to support real-time identification of users that are working together on an interactive display. The user identification algorithm is implemented as a separate server so that it is easy to integrate identity-awareness in existing applications. This separation also ensures that there is no restriction in the programming languages, operating systems and toolkits that can be used to implement identity-aware applications.

8.2 System Architecture

The system architecture of the our prototype is off course a combination of hardware and software. Specific hardware ensures that identity information can be extracted in real-time. The software part on the other hand, ensures that identity information is extracted from these frames and applications can easily use this identity information.

8.2.1 Hardware

For this prototype, we used the Microsoft LifeCam Cinema webcam as overhead camera to capture the frames of the back of the users' hands (see Figure 8.1). This webcam has a resolution of 1280 x 720 px and can stream these frames at 50 Hz. An FTIR multi-touch table was used to detect touches and run both the application and identity server. We also used artificial light sources to ensure that the interactive surface is evenly illuminated.

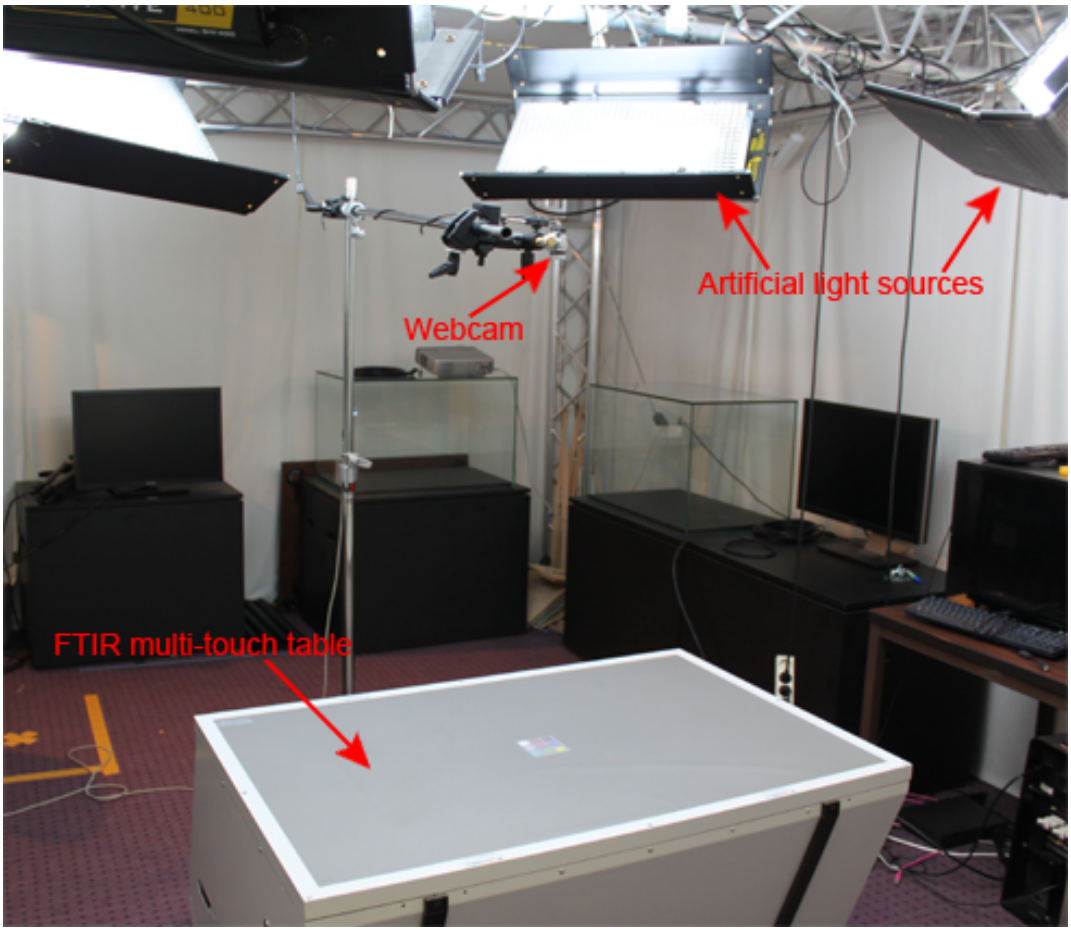


Figure 8.1: The setup of the real-time identity server.

8.2.2 Software

The identity server is implemented in C++ and uses OpenCV (version 2.3.1) [92] for the image processing and computer vision algorithms. To implement the hand recognition and user identification parts, we used the algorithms explained in respectively Section 6.4 and 6.3.4.

Figure 8.2 shows a conceptual visualization of the system. Applications can communicate with the identity server using a TCP-connection. At each registered touch, the application can decide if this touch point needs to be added as a new training sample to the identity server, or if the corresponding user identity needs to be requested from the server. In the first case, the user can specify a user identity and send this information together with the touch position, the time of the touch event and a unique identifier to the identity server. The identity server will inform the client application if the training sample is successfully saved, by sending a response message. The client application needs to be sure that the right user identity is send to the server. Therefore, it is recommended that a client application only sends this command to the server during the authentication phase, in which each user touches the screen at predefined positions. In the second case, the application sends the position of the touch point, the time of the touch event and a unique identifier to the server. The server will then send the corresponding user identity to the client application.

Before the identity server can do something meaningful with these coordinates, the overhead camera connected to the server needs to be calibrated. This can be done using a primitive user interface integrated in the identity server. In this interface, the captured images of the overhead camera are visible. To calibrate the camera, the user needs to click at the four corners of the table that are visible in the captured images. With this information, the server can map the coordinates received from the client application to coordinates in the images captured by the overhead camera.

The identity server continuously extracts the back of the hands that are visible in the captured images. These extracted hands are then stored in an *input buffer* together with the current time and the original position of the hand in the image. Images of hands that are added to the input buffer will be deleted after 3 seconds. In the meantime, the identity server can receive a request from an application to store a recognized hand as training data or to identify a hand. The identity server can now use the time and the coordinates in this received packet to find the closest recognized hand in terms of both time and position. Based on the type of the command received from the client, the server can save the hand in the training set or identify the hand using the training data.

In previous chapters, we explained that there are some variables in our our hand recognition

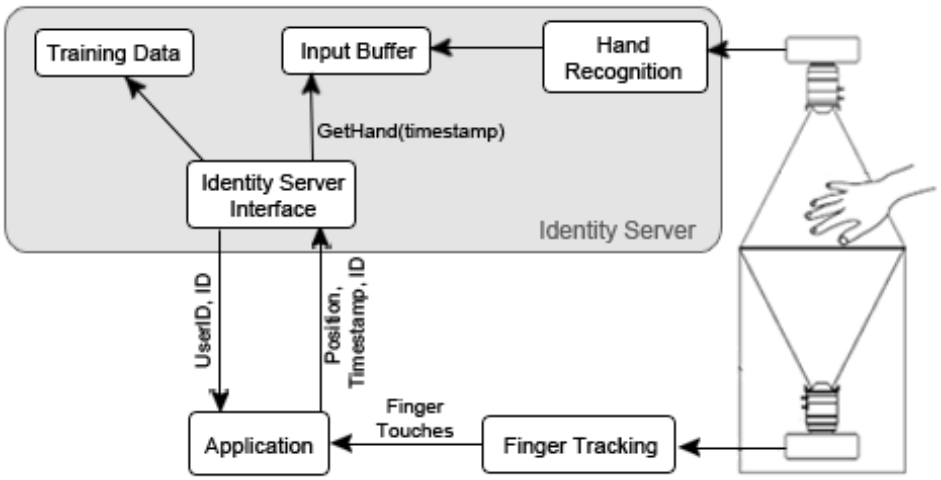


Figure 8.2: A conceptual visualization of the most important components of our system.

algorithm and identity extraction technique that depend on the dpi of the captured images (e.g. the minimum/maximum size of a hand (see Section 6.4.3) and the neighborhood size of a patch (see Section 7.3.4)). The dpi of the images of the detected hands does not only depend on the quality of the overhead camera, but also on the distance between the multi-touch table and the camera. To make our system easy to deploy in different configurations, we printed a pattern on a sheet of paper that is known to the system. When the pattern is placed on the multi-touch table, our identity server can recognize this pattern and automatically measure the current dpi. Another option is to display this pattern directly on the multi-touch screen.

The presented architecture enables our system to process frames at a high rate. This is primarily due to the fact that the most time consuming operation, the binary pattern encoding (requires a significant amount of pixel access), is executed only when a user requests an identity.

8.3 Demo Application

We implemented an identity-aware painting application to demonstrate our identity server. This application is implemented in C# using the .NET/WPF and Microsoft Surface toolkit. A TCP-connection is used to communicate with the identity server, as we described in Section 8.2.2. When starting this application, all users need to register with the system one by one.

At the start of the registration process, the user needs to create a new profile by entering his name. In the second step, this user needs to touch 9 buttons in succession that are displayed

at different positions on the screen. At each touch, the application will inform the user if the training sample is successfully saved (see Figure 8.3). When all users are registered, the application can start.

In the middle of the painting canvas, a color picker widget is displayed from which each user can select their preferred color. When the user selects a color, the application will inform the user that this color is now assigned to him, by displaying the user's name (see Figure 8.4). The application supports that multiple users can draw and change their active colors simultaneously (see Figure 8.4). When multiple users are touching the display simultaneously, there can be a small delay before the identity server has recognized all users. However, this will not affect the user experience, because in the meantime, the user can finish his stroke in a neutral color (gray). When the application received the response from the identity server, the color of this anonymous stroke will change to match the color selected by the user.

8.4 Conclusion

In this chapter, we presented the identity server, a service in which our research findings from Chapters 6 and 7 are integrated to provide a real-time user identification solution. The system can be integrated into existing applications and is easy to deploy. We demonstrated the use of the identity server with an identity-aware painting application in which multiple users can make sketches simultaneously using different colors.



Figure 8.3: A user that registers with the system by providing training samples of his hand.

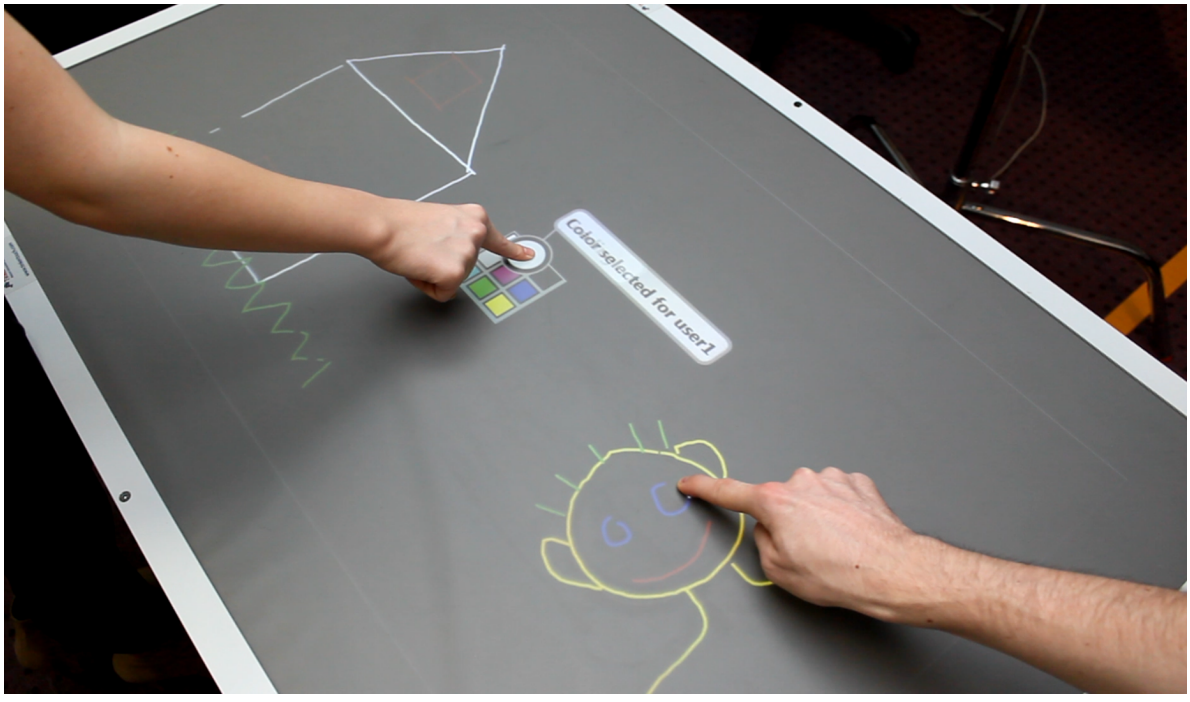


Figure 8.4: Two users that are working simultaneously with our identity-aware painting application.

Chapter 9

Discussion

9.1 Introduction

In this penultimate chapter, we will first of all reflect back on our study and try to connect the dots that were laid out in the different chapters. We will also evaluate our contributions and discuss the future work that can be done in this research area.

9.2 Enriching Touch

The analysis of interactive displays (see Chapter 2) revealed that touch input is almost always modeled as 2-dimensional information. In this thesis, we proposed that touch interaction is much higher dimensional and that this information can be used to make these technologies more natural and easy to use. First of all, we investigated the new dimensions of information that can be extracted from touch interaction and the technology that is required. We also provided an overview of new interaction possibilities that are enabled when all these new dimensions of information are available. Finally, we developed a system that extracts additional information from touch input. In this section, we will reflect back on these parts.

9.2.1 Additional Touch Information

We primarily concentrated on two new sources of information: the identity of a touch point and positional information of the user. A touch contact can be identified on 3 levels. First of all, the identity of the user can be revealed. This user identity is especially useful when the application needs to support parallel-moded interactions or when a user tries to invoke a potentially destructive action. In the latter situation, the system can require that all participants try to reach consensus before the action will be executed. Secondly, the two hands of a user

can be distinguished from each other. This provides the system with information to realize applications that can, for example, differentiate between a dual stretch gesture performed by fingers of the same hand and a dual hand duplicate gesture where a copy of an object is produced. Thirdly, each finger can be identified to provide custom functionalities at the level of finger contacts.

Positional information of the user can be useful on 3 levels. First of all, the extracting position of the user in front of a large display makes it possible to position content in the right direction to face the user. Furthermore, orientation sensitive gestures and handwritings can be interpreted correctly. Secondly, the position of the hand can provide useful information. For example, information can be repositioned to minimize the occlusion of information with the hand. Thirdly, more detailed information of the shape of a touch contact can be used to create, for example, orientation sensitive dials.

Recent research has investigated even more new interaction techniques that can enrich the touch experience when this information is available (see Chapter 5). However, these promising applications and interaction techniques have not yet hit the mainstream because the technology to extract all this information is not readily available.

9.2.2 Non-Intrusive High-Accuracy User Identification

In the introduction of this thesis we defined our envisioned system:

> "We envision a collaborative multi-touch system that can be used by multiple users at the same time. The system needs to be able to identify every touch point with high reliability while preserving the natural interaction that is inherent to multi-touch displays. As a result, custom functionalities can be provided, such as, multiuser undo buttons and parallel moded interactions. Furthermore, the system can even provide support for customized content (e.g. translating text, display extra help, etc.). Another option is to give users privileged access to their private documents, contact lists, bookmarks, etc.´´

To realize this envisioned system, we can take a look at the techniques that are available for identifying users on a multi-touch display. These techniques can be classified in 4 groups: software approaches, techniques using external tokens/devices, tracking of users and biometric techniques. However, they have all their strengths and weaknesses as shown in Table 9.1. As a result, none of these techniques is able to fulfill all our requirements. Therefore, we proposed a new approach for user identification on interactive displays that does not require extra instrumentation or specific hand postures. We presented the idea of extracting biometric

features from the back of the user's hand using an overhead camera. This idea was based on the observation that the back of the hand is the largest, least deformable region of the hand and is also visible to an overhead camera most of the time. Furthermore, the hand is far from being smooth, it is highly textured, multicolored and deeply etched with lines, grooves, folds and furrows which makes it very likely to express unique features. Based on these characteristics, we hypothesized that the back of the hand is a suitable skin region for extracting unique features.

We presented 4 potential techniques that could be used for extracting features from the human hand. Furthermore, we discussed an algorithm to extract the back of the hand in an image captured by an overhead camera. In a preliminary study, two of these techniques turned out to extract features that were not very unique. The two other techniques provided promising results. Therefore, we conducted an extensive study with 24 participants to collect a large dataset of hands in multiple postures. The results indicated that especially the local binary patterns technique was able to encode very unique information from the back of the users' hands.

The results of the studies proved that even when a large group of 24 users are registered with the system, our technique is still able to identify hands under different postures up to 99% of the time. The results also revealed that there is an optimal resolution for which the technique has the highest recognition rate. When the resolution is smaller than this optimum, the recognition rate will drop because there is not enough unique information visible. On the other hand, when the resolution is to large, to much detail is present and the features will become less discriminative. We also analyzed the consequences of changing the area in which to search for similar patches. These results suggested that there is only a lower threshold, more specifically, the minimal area needed because of the imperfections of the hand recognition algorithm. However, when searching for patches in a very large area, the reliability of the system remained relatively constant. This implies that our hand recognition algorithm does not need to be able to orientate hands very precisely, because the recognition rate of the system remains the same when we search for similar patches in the entire hand. However, the performance of the systems drops significantly when matching all combinations of patches. In contrast, we noticed that the quality of the skin segmentation has a major influence on the recognition rate. During the segmentation of the hands in the free hand pointing posture, we noticed some artifacts due to additional shadows and occlusion of finger valleys by other fingers. This caused the recognition rate to drop to 88.5%.

Based on all these tests we can conclude that our technique has a very high recognition rate even for large groups of users. However, in all our results there was a very clear trend visible: the similarity rate between a hand and the best match in the training set was very close to the similarity with the second best match. As a result, it is not possible to define

	Strengths	**Weaknesses**
Software approaches [59]	Self-contained system	Not reliable (social protocols needed)
External tokens/devices [125, 79, 105, 75, 116]	Relatively reliable	Extra instrumentation
Tracking of users [33, 40, 30]	Non-intrusive	Not very reliable
Biometric techniques [112, 144]	Reliable, non-intrusive	Specific postures required

Table 9.1: Strengths and weaknesses of existing user identification technologies.

a threshold for excluding users that are not registered with the system. Consequently, our technique can only be used to differentiate between users and not for authentication purposes. Nevertheless, the fact that our technique is able to identify users non-intrusively is a major benefit compared to all other techniques that are currently available.

In Section 6.2, we defined the back of the hand as an ideal region for the extraction of unique features. However, this skin region is not as persistent and immutable as, for example, fingerprints. It is possible for the skin to get a tan and wounds can result in permanent scars. Furthermore, we ignored the existence of tension lines, which provide the basis for wrinkling of the skin. Tension lines are present all over the hand but it isn't until the skin loses its elasticity with advancing years that they form permanent wrinkles. As a result, it could be possible that our technique is less reliable for elderly people, because the skin can deform much more during the interaction with a touch display.

Our system can be used to implement the following multi-touch interaction patterns that we presented in Chapter 5:

- **Customizing Functionalities:** multi-user undo, parallel-moded interaction, personalized content, etc.
- **Cooperative Operations:** reaching consensus in conflicting situations that are inherent to computer supported collaborative work.
- **Customizing Content:** extra help for novice users, translation of text, hide information, etc.
- **One and Two Handed Gestural Interaction:** differentiating between gestures performed by one or two hands.

- **Automatic Content Orientation:** automatically orientate documents while passing them between users, etc.
- **Orientation-Sensitive Widgets:** orientation-sensitive button, orientation dial, etc.
- **Orientation Invariant Input:** orientation invariant handwriting recognition, etc.
- **Hover States:** support for tooltips, feedforward, etc.
- **Occlusion Problems:** occlusion-aware pie-menus, dynamic placement of content, etc.

Only the first three interaction patterns are directly supported by the current implementation of the identity server. When the server also sends the positional information of the extracted contours and the orientation of the hand to the client applications, the other six interaction patterns mentioned above, can also be implemented.

We can conclude that our presented technique makes it possible to identify users non-intrusively with high accuracy on interactive displays. We showed that our system can even be used for large groups of users. In this thesis, we primarily focused on the traditional touch tables. However, the technique that we presented is very flexible because it only requires an overhead camera. This makes it possible to integrate our identification technique into less traditional setups, such as Microsoft's PlayAnywhere [150]. Furthermore, it is even possible to deploy our technique on a traditional table when investigating, for example, territoriality in collaborative tabletop workspaces [119].

9.3 Future Work

There are still unexplored possibilities to improve our identification technique. We also believe that the discussed applications in this thesis are only the tip of the iceberg. In addition, we think this identification technique has a much broader application domain than identifying users on touch displays.

First of all, we plan to improve our technique to support authentication of users. To realize this, our extracted features need to be more discriminative. This can be realized, for example, by assigning a weight to each patch. Patches that are located in the center of the hand need to have a higher weight than patches on the edges, because the latter are more distorted and are therefore likely to expose less reliable information. Furthermore, the training set can gradually be improved by changing the weight of these patches. A patch that is very similar to a patch in another captured image of the same user, and at the same time, very different from all patches of other users, can be assigned a high weight because this patch contains

very unique information. When using this approach, patches in which parts of, for example, jewelery are visible will have a higher weight because of the highly discriminative information that they contain. It is also an option to extract additional features from other body parts, such as fingernails or parts of the arm that are visible. However, it cannot be guaranteed that these body parts are always visible when touching the display. Therefore, it can be useful to use information from multiple frames.

A possible weakness of our identification technique can be the use of the overhead camera. This camera can be occluded by the head or upper body when bending over the table. Furthermore, the back of the hand will not be visible to the camera when interacting with the hand entirely vertical according to the surface. We believe that these situations will not occur when sitting around the table. In addition, we think that this occlusion problem can certainly be reduced in future versions of our system by using, for example, multiple cameras or a clever setup of mirrors.

Lastly, we think that related research areas, such as tangible interaction, can also benefit from our work. The basis of our identification technique is based on pattern matching. It is therefore possible to use this technique to recognize and track marker-less tangibles on an interactive display.

9.4 Contributions

The contribution of this thesis is fourfold. First of all, we provided an extensive classification of existing technologies that can be used to extract additional information from touch input. Secondly, we presented a catalog of interaction patterns that are applicable when touch technologies are able to provide extra information with each touch (i.e. identity and positional information). Thirdly, we developed a new technique to identify users on an interactive display, this was still a challenging issue in tabletop research. Our technique is non-intrusive, this means that the natural interaction with these displays is still preserved. Studies revealed that our system is able to identify users very reliably (99%), even for groups with 24 users. Finally, we have integrated all our findings in the *Identity Server*. This is a services that makes it possible to identify each touch on an interactive display in real-time. Furthermore, the architecture ensures that the service can easily be integrated into existing applications, and poses no restriction on the programming languages, toolkits or operating systems that can be used.

9.5 Conclusion

In this chapter, we first of all reflected back on our work. We started with a discussion of the limitations of existing user identification techniques for tabletop displays. We clearly mentioned the strengths and weaknesses of these techniques and proposed our user identification technique to overcome these issues. We concluded that the high recognition rate in combination with the non-intrusive character are the major strengths of our technique. We also took a step back and identified the limitations and possible improvements that can be made to our system. We finished this chapter with an overview of our contributions.

Chapter 10

Conclusion

The work in this thesis is based on the observation that almost all touch displays model touch contacts as 2 dimensional information: a pair of coordinates. However, touch interaction can benefit greatly from other information sources, such as identity information and positional information of the user. Therefore we presented 13 multi-touch interaction patterns that can be realized when this information is readily available.

After an extensive investigation of existing techniques to extract identity information from touch points, we concluded that each of them has strengths and weaknesses but none of them is superior. It was not possible to create our envisioned application (presented in the introduction of this thesis) using only existing techniques. This was primarily due to the fact that none of the systems could reliably identify users while preserving the natural interaction that is inherent to multi-touch displays. Therefore, we presented a new technique that makes it possible to identify touch points on an interactive display non-intrusively with high reliability. Our technique extracts features from the back of the hand because this region can be easily captured by an overhead camera under different postures of the hand.

The conducted study revealed that our identification technique is able to identify users correctly up to 99% of the time. The results showed that this recognition rate remained constant even under different postures and when large groups of users are registered with the system. After further investigation of our results, we noticed that besides this high recognition rate, the similarity rate between the hands of multiple users are not very different from each other. It was therefore not possible to define a threshold to exclude users that are not registered with the system. As a result, the current version of our identification technique can only be used to differentiate between users and not for authentication purposes.

Bibliography

[1] 5DT. Data gloves for the fifth dimension, Feb 2000. http://www.5dt.com/downloads/dataglove/old/5DTDataGlove5Manual.pdf.

[2] A. Agarwal, S. Izadi, M. Chandraker, and A. Blake. High Precision Multi-touch Sensing on Surfaces using Overhead Cameras. In *Horizontal Interactive Human-Computer Systems, 2007. TABLETOP '07. Second Annual IEEE International Workshop on*, pages 197–200, 2007.

[3] T. Ahonen, A. Hadid, and M. Pietikainen. Face Recognition with Local Binary Patterns. pages 469–481. 2004.

[4] Apple. iphone, 2007. http://www.apple.com/benl/iphone, last accessed Mar. 10, 2011.

[5] Apple. ipad, 2010. http://www.apple.com/benl/ipad, last accessed Mar. 10, 2011.

[6] Atmel. Manchester coding basics, Mar 2005. http://www.atmel.com/dyn/resources/prod_documents/doc9164.pdf.

[7] D. B. Chasing the colour glove: Visual hand tracking. Master's thesis, Simon Fraser University, 1994.

[8] R. Balakrishnan. "beating" fitts' law: virtual enhancements for pointing facilitation. *Int. J. Hum.-Comput. Stud.*, 61:857–874, December 2004.

[9] R. Ballagas, J. Borchers, M. Rohs, and J. G. Sheridan. The smart phone: A ubiquitous input device. *IEEE Pervasive Computing*, 5:70–, January 2006.

[10] H. Benko, A. D. Wilson, and P. Baudisch. Precise selection techniques for multi-touch screens. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 1263–1272, New York, NY, USA, 2006. ACM.

[11] X. Bi, Y. Shi, and X. Chen. upen: A smart pen-liked device for facilitating interaction on large displays. In *Proceedings of the First IEEE International Workshop on Horizontal*

Interactive Human-Computer Systems, pages 160–168, Washington, DC, USA, 2006. IEEE Computer Society.

[12] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: the see-through interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 73–80, New York, NY, USA, 1993. ACM.

[13] M. R. Bill Buxton. Multi-touch systems that i have known and loved, jan 2007. `http://www.billbuxton.com/multitouchOverview.html`, last accessed: May 10, 2010.

[14] J. O. Borchers. A pattern approach to interaction design. In *Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques*, DIS '00, pages 369–378, New York, NY, USA, 2000. ACM.

[15] G. Boreki and A. Zimmer. Hand geometry: A new approach for feature extraction. In *Proceedings of the Fourth IEEE Workshop on Automatic Identification Advanced Technologies*, pages 149–154, Washington, DC, USA, 2005. IEEE Computer Society.

[16] M. Brain. How upc bar codes work, Mar 2008. `http://electronics.howstuffworks.com/gadgets/high-tech-gadgets/upc.htm, last accessed Mar. 27, 2011`.

[17] P. Brandl, M. Haller, M. Hurnaus, V. Lugmayr, J. Oberngruber, C. Oster, C. Schafleitner, and M. Billinghurst. An adaptable rear-projection screen using digital pens and hand gestures. In *Proceedings of the 17th International Conference on Artificial Reality and Telexistence*, ICAT '07, pages 49–54, Washington, DC, USA, 2007. IEEE Computer Society.

[18] P. Brandl, J. Leitner, T. Seifried, M. Haller, B. Doray, and P. To. Occlusion-aware menu design for digital tabletops. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI EA '09, pages 3223–3228, New York, NY, USA, 2009. ACM.

[19] H. Brignull and Y. Rogers. Enticing people to interact with large public displays in public spaces. In *In Proceedings of the IFIP International Conference on Human-Computer Interaction (INTERACT 2003*, pages 17–24, 2003.

[20] A.-M. Burns and B. Mazzarino. Finger tracking methods using eyesweb. In *Gesture Workshop*, pages 156–167, 2005.

[21] J.-C. Chien, H.-Y. Lu, Y.-S. Wu, and L.-C. Liu. Artoolkit-based augmented reality system with integrated 1-d barcode: combining colorful markers with remote servers of 3d

data for product promotion purposes. In *Proceedings of the Second international conference on Computational collective intelligence: technologies and applications - Volume Part III*, ICCCI'10, pages 200–209, Berlin, Heidelberg, 2010. Springer-Verlag.

[22] J. Choi, B.-K. Seo, and J.-I. Park. Robust hand detection for augmented reality interface. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*, VRCAI '09, pages 319–321, New York, NY, USA, 2009. ACM.

[23] M. Corp. Kinect, 2011. `http://www.xbox.com/kinect, last accessed May. 1, 2011`.

[24] M. Corp. Surface, Mar. 2011. `http://www.microsoft.com/surface, last accessed Apr. 29, 2011`.

[25] T. Cuypers, Y. Francken, C. Vanaken, F. Van Reeth, and P. Bekaert. Smartphone localization on interactive surfaces using the built-in camera. In *In Proc. Procams*, 2009.

[26] C. T. Dang, M. Straub, and E. André. Hand distinction for multi-touch tabletop interaction. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 101–108, New York, NY, USA, 2009. ACM.

[27] P. L. Davidson and J. Y. Han. Extending 2d object arrangement with pressure-sensitive layering cues. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, pages 87–90, New York, NY, USA, 2008. ACM.

[28] P. Dietz and D. Leigh. Diamondtouch: a multi-user touch technology. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, pages 219–226, New York, NY, USA, 2001. ACM.

[29] DNP. Dnp holoscreen, 2004. `http://www.rearpro.com/files/dnp Holo Screen Product Sheet June 2008.pdf, last accessed May. 6, 2011`.

[30] K. C. Dohse, T. Dohse, J. D. Still, and D. J. Parkhurst. Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. In *Proceedings of the First International Conference on Advances in Computer-Human Interaction*, ACHI '08, pages 297–302, Washington, DC, USA, 2008. IEEE Computer Society.

[31] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 10(2):112–122, Oct. 1973.

[32] P. Dragicevic and Y. Shi. Visualizing and manipulating automatic document orientation methods using vector fields. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 65–68, New York, NY, USA, 2009. ACM.

[33] F. Echtler, M. Huber, and G. Klinker. Hand tracking for enhanced gesture recognition on interactive multi-touch surfaces. *submitted to Horizontal Interactive Human-Computer Systems*, 2007.

[34] 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*, CHI '92, pages 599–607, New York, NY, USA, 1992. ACM.

[35] R. Feynman. The messenger series. lecuture 7: Seeking new laws. bbc sience&nature.

[36] K. Fishkin, B. Jiang, M. Philipose, S. Roy, K. P. Fishkin, B. Jiang, M. Philipose, and S. Roy. I sense a disturbance in the force: Unobtrusive detection of interactions with rfid-tagged objects. pages 268–282. Springer-Verlag, 2004.

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

[38] G. W. Fitzmaurice, H. Ishii, and W. A. S. Buxton. Bricks: laying the foundations for graspable user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 442–449, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[39] C. Forlines, D. Wigdor, C. Shen, and R. Balakrishnan. Direct-touch vs. mouse input for tabletop displays. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '07, pages 647–656, New York, NY, USA, 2007. ACM.

[40] B. W. Franks, L. Schwarten, J. Teichert, M. Krause, and M. Herrlich. User Detection for a Multi-touch Table via Proximity Sensors. In *IEEE Tabletops and Interactive Surfaces 2008*. IEEE Computer Society, 2008.

[41] FRGC. Face recognition grand challenge, sept 2005. `http://www.frvt.org/FRGC, last accessed May. 6, 2011`.

[42] C. Ganser, A. Steinemann, and A. Kunz. Infractables: Multi-user tracking system for interactive surfaces. In *Proceedings of the IEEE conference on Virtual Reality*, VR '06, pages 253–256, Washington, DC, USA, 2006. IEEE Computer Society.

[43] T. Grossman and R. Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 281–290, New York, NY, USA, 2005. ACM.

[44] J. Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 115–118, New York, NY, USA, 2005. ACM.

[45] J. Y. Han. Perceptivepixel, 2008. `http://www.perceptivepixel.com, last accessed May. 28, 2011`.

[46] J. Y. Han. Various resistive touchscreen technologies, 2009. `http://www.touchscreenguide.com/touchscreen/res.html, last accessed May. 28, 2011`.

[47] R. Hardy and E. Rukzio. Touch & interact: touch-based interaction of mobile phones with displays. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, MobileHCI '08, pages 245–254, New York, NY, USA, 2008. ACM.

[48] T. Harris. How fingerprint scanners work, Mar 2010. `http://computer.howstuffworks.com/fingerprint-scanner.htm, last accessed May. 7, 2011`.

[49] R. Hofer, D. Naeff, and A. Kunz. Flatir: Ftir multi-touch detection on a discrete distributed sensor array. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, TEI '09, pages 317–322, New York, NY, USA, 2009. ACM.

[50] S. Hollister. Apple granted patent for handheld that recognizes your hands, sep 2010. `http://www.engadget.com/2010/09/25/apple-granted-patent-for-handheld-that-recognizes-your-hands`.

[51] C. Holz and P. Baudisch. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 581–590, New York, NY, USA, 2010. ACM.

[52] C. Holz and P. Baudisch. Understanding touch. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 2501–2510, New York, NY, USA, 2011. ACM.

[53] E. M. Huang and E. D. Mynatt. Semi-public displays for small, co-located groups. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '03, pages 49–56, New York, NY, USA, 2003. ACM.

[54] 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*, UIST '03, pages 159–168, New York, NY, USA, 2003. ACM.

[55] S. Izadi, S. Hodges, A. Butler, A. Rrustemi, and B. Buxton. Thinsight: integrated optical multi-touch sensing through thin form-factor displays. In *Proceedings of the 2007 workshop on Emerging displays technologies: images and beyond: the future of displays and interacton*, EDT '07, New York, NY, USA, 2007. ACM.

[56] S. Izadi, S. Hodges, S. Taylor, D. Rosenfeld, N. Villar, A. Butler, and J. Westhues. Going beyond the display: a surface technology with an electronically switchable diffuser. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, UIST '08, pages 269–278, New York, NY, USA, 2008. ACM.

[57] A. Jain and S. Pankanti. Fingerprint classification and matching. In *HANDBOOK FOR IMAGE AND VIDEO PROCESSING*. Academic Press, 2000.

[58] A. Jain, A. Ross, and S. Pankanti. A prototype hand geometry-based verification system. In *2nd Int'l Conference on Audio- and Video-based Biometric Person Authentication, Washington D.C.*, 1999.

[59] N. E. K. Kim, T. D. Kulkarni. Interaction workspaces: Identity tracking for multi-user collaboration on camera-based multi-touch tabletops. *In Proceedings of the IEEE VisWeek Workshop on Collaborative Visualization on Interactive Surfaces*, 2009.

[60] Y. Kakehi, M. Iida, T. Naemura, and M. Mitsunori. Tablescape plus: upstanding tiny displays on tabletop display. In *ACM SIGGRAPH 2006 Emerging technologies*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[61] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recogn.*, 40:1106–1122, March 2007.

[62] M. Kaltenbrunner and R. Bencina. reactivision: a computer-vision framework for table-based tangible interaction. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, TEI '07, pages 69–74, New York, NY, USA, 2007. ACM.

[63] L. R. Kasday. Touch position sensitive surface, 1984. `http://www.google.com/patents?vid=USPAT4484179`.

[64] D. Kim, P. Dunphy, P. Briggs, J. Hook, J. Nicholson, J. Nicholson, and P. Olivier. Multi-touch authentication on tabletops. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 1093–1102, New York, NY, USA, 2010. ACM.

[65] S. G. Kong, J. Heo, B. R. Abidi, J. Paik, and M. A. Abidi. Recent advances in visual and infrared face recognition: a review. *Comput. Vis. Image Underst.*, 97:103–135, January 2005.

[66] I. Labs. Intelligent interactive billboard technology, 2010. `http://www.immersivelabs.com, last accessed May. 6, 2011`.

[67] E. Larson, G. Cohn, S. Gupta, X. Ren, B. Harrison, and D. Fox. Heatwave: Thermal imaging for surface user interaction. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, Shwetak Patel, University of Washington, 2011. ACM.

[68] H. C. Lee and R. E. Gaensslen. Advances in fingerprint technology. 1991.

[69] J. Leitner, J. Powell, P. Brandl, T. Seifried, M. Haller, B. Dorray, and P. To. Flux: a tilting multi-touch and pen based surface. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI EA '09, pages 3211–3216, New York, NY, USA, 2009. ACM.

[70] J. Letessier and F. Bérard. Visual tracking of bare fingers for interactive surfaces. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 119–122, New York, NY, USA, 2004. ACM.

[71] C. Liao, F. Guimbretière, and K. Hinckley. Papiercraft: a command system for interactive paper. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 241–244, New York, NY, USA, 2005. ACM.

[72] K.-C. Liao, W.-H. Lee, M.-H. Sung, and T.-C. Lin. A one-time password scheme with qr-code based on mobile phone. In *Proceedings of the 2009 Fifth International Joint Conference on INC, IMS and IDC*, NCM '09, pages 2069–2071, Washington, DC, USA, 2009. IEEE Computer Society.

[73] T. M. Mahmoud. A new fast skin color detection technique.

[74] J. B. Mallos. Touch position sensitive surface, 1982. `http://www.google.com/patents?vid=USPAT4346376`.

[75] N. Marquardt, J. Kiemer, and S. Greenberg. What caused that touch?: expressive interaction with a surface through fiduciary-tagged gloves. In *ACM International Conference*

on Interactive Tabletops and Surfaces, ITS '10, pages 139–142, New York, NY, USA, 2010. ACM.

[76] J. Matejka, T. Grossman, J. Lo, and G. Fitzmaurice. The design and evaluation of multi-finger mouse emulation techniques. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1073–1082, New York, NY, USA, 2009. ACM.

[77] M. Matsushita, M. Iida, T. Ohguro, Y. Shirai, Y. Kakehi, and T. Naemura. Lumisight table: a face-to-face collaboration support system that optimizes direction of projected information to each stakeholder. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, CSCW '04, pages 274–283, New York, NY, USA, 2004. ACM.

[78] N. Matsushita and J. Rekimoto. Holowall: designing a finger, hand, body, and object sensitive wall. In *Proceedings of the 10th annual ACM symposium on User interface software and technology*, UIST '97, pages 209–210, New York, NY, USA, 1997. ACM.

[79] T. Meyer and D. Schmidt. Idwristbands: Ir-based user identification on multi-touch surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 277–278, New York, NY, USA, 2010. ACM.

[80] J. Moeller, A. Kerne, and S. Damaraju. Zerotouch: a zero-thickness optical multi-touch force field. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, CHI EA '11, pages 1165–1170, New York, NY, USA, 2011. ACM.

[81] K. A. Mohamed, S. Haag, J. Peltason, F. Dal-Ri, and T. Ottmann. Disoriented pen-gestures for identifying users around the tabletop without cameras and motion sensors. In *Proceedings of the First IEEE International Workshop on Horizontal Interactive Human-Computer Systems*, pages 43–52, Washington, DC, USA, 2006. IEEE Computer Society.

[82] M. R. Morris, A. Huang, A. Paepcke, and T. Winograd. Cooperative gestures: multi-user gestural interactions for co-located groupware. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 1201–1210, New York, NY, USA, 2006. ACM.

[83] M. R. Morris, K. Ryall, C. Shen, C. Forlines, and F. Vernier. Beyond "social protocols": multi-user coordination policies for co-located groupware. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, CSCW '04, pages 262–265, New York, NY, USA, 2004. ACM.

[84] T. Moscovich and J. F. Hughes. Indirect mappings of multi-touch input using one and two hands. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1275–1284, New York, NY, USA, 2008. ACM.

[85] B. A. Myers, H. Stiel, and R. Gargiulo. Collaboration using multiple pdas connected to a pc. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, CSCW '98, pages 285–294, New York, NY, USA, 1998. ACM.

[86] J. R. Napier. *Hands*. Princeton sience library, 1993.

[87] L. O'Gorman. Document image analysis. chapter Curvilinear feature detection from curvature estimation, pages 129–132. IEEE Computer Society Press, Los Alamitos, CA, USA, 1995.

[88] Y. Ohta, T. Kanade, and T. Sakai. Color information for region segmentation. *Graphical Models /graphical Models and Image Processing /computer Vision, Graphics, and Image Processing*, 13:222–241, 1980.

[89] T. Ojala. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(l):51–59, 1996.

[90] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 24(7):971–987, 2002.

[91] A. Olwal and A. D. Wilson. Surfacefusion: unobtrusive tracking of everyday objects in tangible user interfaces. In *Proceedings of graphics interface 2008*, GI '08, pages 235–242, Toronto, Ont., Canada, Canada, 2008. Canadian Information Processing Society.

[92] OpenCV. Opencvwiki, Aug. 2011. `http://opencv.willowgarage.com/wiki/, last accessed Dec. 30, 2011`.

[93] OpticsPlanet. Thermal imaging cameras, 2011. `http://www.opticsplanet.net/heat-seekers-termal-imagers.html, last accessed May. 1, 2011`.

[94] P. Peltonen, E. Kurvinen, A. Salovaara, G. Jacucci, T. Ilmonen, J. Evans, A. Oulasvirta, and P. Saarikko. It's mine, don't touch!: interactions at a large multi-touch display in a city centre. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1285–1294, New York, NY, USA, 2008. ACM.

[95] R. L. Potter, L. J. Weldon, and B. Shneiderman. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '88, pages 27–32, New York, NY, USA, 1988. ACM.

[96] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, pages 32–43, New York, NY, USA, 2000. ACM.

[97] Y. Qin, C. Yu, H. Jiang, C. Wu, and Y. Shi. ppen: enabling authenticated pen and touch interaction on tabletop surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 283–284, New York, NY, USA, 2010. ACM.

[98] R. Raskar, P. Beardsley, P. Dietz, and J. van Baar. Photosensing wireless tags for geometric procedures. *Commun. ACM*, 48:46–51, September 2005.

[99] R. Raskar, P. Beardsley, J. van Baar, Y. Wang, P. Dietz, J. Lee, D. Leigh, and T. Willwacher. Rfig lamps: interacting with a self-describing world via photosensing wireless tags and projectors. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[100] E. Raymond. Latent print examiner, Aug 2010. `http://www.onin.com, last accessed May. 7, 2011`.

[101] D. Reilly, M. Rodgers, R. Argue, M. Nunes, and K. Inkpen. Marked-up maps: combining paper maps and electronic information resources. *Personal Ubiquitous Comput.*, 10:215–226, March 2006.

[102] J. Rekimoto. Smartskin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, CHI '02, pages 113–120, New York, NY, USA, 2002. ACM.

[103] C. Remy, M. Weiss, Ziefle, and J. Borchers. A pattern language for interactive tabletops in collaborative workspaces. In *Proc. EuroPLoP '10*. ACM, 2010.

[104] Y. Rogers, W. Hazlewood, E. Blevis, and Y.-K. Lim. Finger talk: collaborative decision-making using talk and fingertip interaction around a tabletop display. In *CHI '04 extended abstracts on Human factors in computing systems*, CHI EA '04, pages 1271–1274, New York, NY, USA, 2004. ACM.

[105] V. Roth, P. Schmidt, and B. Güldenring. The ir ring: authenticating users' touches on a multi-touch display. In *Proceedings of the 23nd annual ACM symposium on User*

interface software and technology, UIST '10, pages 259–262, New York, NY, USA, 2010. ACM.

[106] G. Roussos and V. Kostakos. rfid in pervasive computing: State-of-the-art and outlook. *Pervasive Mob. Comput.*, 5:110–131, February 2009.

[107] D. M. Russell, C. Drews, and A. Sue. Social aspects of using large public interactive displays for collaboration. In *Proceedings of the 4th international conference on Ubiquitous Computing*, UbiComp '02, pages 229–236, London, UK, UK, 2002. Springer-Verlag.

[108] D. M. Russell, J. P. Trimble, and A. Dieberger. The use patterns of large, interactive display surfaces: Case studies of media design and use for blueboard and merboard. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4*, HICSS '04, pages 40098.2–, Washington, DC, USA, 2004. IEEE Computer Society.

[109] K. Ryall, A. Esenther, C. Forlines, C. Shen, S. Shipman, M. R. Morris, K. Everitt, and F. D. Vernier. Identity-differentiating widgets for multiuser interactive surfaces. *IEEE Comput. Graph. Appl.*, 26:56–64, September 2006.

[110] K. Ryall, K. Ryall, A. Esenther, A. Esenther, K. Everitt, K. Everitt, C. Forlines, C. Forlines, M. R. Morris, M. R. Morris, C. Shen, S. Shipman, S. Shipman, F. Vernier, and F. Vernier. idwidgets: Parameterizing widgets by user identity. In *Tenth IFIP TC13 International Conference on Human-Computer Interaction*, 2005.

[111] D. Schmidt, F. Chehimi, E. Rukzio, and H. Gellersen. Phonetouch: a technique for direct phone interaction on surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, UIST '10, pages 13–16, New York, NY, USA, 2010. ACM.

[112] D. Schmidt, M. K. Chong, and H. Gellersen. Handsdown: hand-contour-based user identification for interactive surfaces. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, NordiCHI '10, pages 432–441, New York, NY, USA, 2010. ACM.

[113] D. Schmidt, M. K. Chong, and H. Gellersen. Idlenses: dynamic personal areas on shared surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 131–134, New York, NY, USA, 2010. ACM.

[114] J. Schöning, P. Brandl, F. Daiber, F. Echtler, O. Hilliges, J. Hook, M. Löchtefeld, N. Motamedi, L. Muller, P. Olivier, T. Roth, and U. von Zadow. Multi-Touch Surfaces: A Technical Guide. Technical report, Oct. 2008.

[115] J. Schöning, M. Rohs, and A. Krüger. Spatial Authentication on Large Interactive Multi-Touch Surfaces. In *IEEE Tabetop 2008: Adjunct Proceedings of IEEE Tabletops and Interactie Surfaces*, Oct. 2008.

[116] J. Schöning, M. Rohs, and A. Krüger. Using Mobile Phones to Spontaneously Authenticate and Interact withMulti-touch Surfaces. In *PPD'08. Workshop on designing multi-touch interaction techniques for coupled public and private displays*, 2008.

[117] K. A. Schöning J, Rohs M. Using mobile phones to spontaneously authenticate and interact with multi-touch surfaces. *in PPD'08, Workshop on designing multi-touch interaction techniques for coupled public and private displays*, May 2008.

[118] S. D. Scott, K. D. Grant, and R. L. Mandryk. System guidelines for co-located, collaborative work on a tabletop display. In *Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work*, pages 159–178, Norwell, MA, USA, 2003. Kluwer Academic Publishers.

[119] S. D. Scott, M. Sheelagh, T. Carpendale, and K. M. Inkpen. Territoriality in collaborative tabletop workspaces. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, CSCW '04, pages 294–303, New York, NY, USA, 2004. ACM.

[120] C. E. Shannon. Communication in the presence of noise. In *Proc. IRE, 37(1):10-21*, Jan 1949.

[121] C. Shen, K. Everitt, and K. Ryall. Ubitable: Impromptu face-to-face collaboration on horizontal interactive surfaces. In *In Proc. UbiComp 2003*, pages 281–288, 2003.

[122] C. Shen, F. D. Vernier, C. Forlines, and M. Ringel. Diamondspin: an extensible toolkit for around-the-table interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '04, pages 167–174, New York, NY, USA, 2004. ACM.

[123] A. S. Shirazi, T. Döring, P. Parvahan, B. Ahrens, and A. Schmidt. Poker surface: combining a multi-touch table and mobile phones in interactive card games. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '09, pages 73:1–73:2, New York, NY, USA, 2009. ACM.

[124] J. Shlens. A tutorial on principal component analysis. In *Systems Neurobiology Laboratory, Salk Institute for Biological Studies*, 2005.

[125] P. R. Siddalinga. Sensor Augmented Large Interactive Surfaces. Master's thesis, Iowa State University, Ames, Iowa, 2010.

[126] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *J. Opt. Soc. Am. A*, 4(3):519–524, Mar. 1987.

[127] C. Sousa and M. Matsumoto. Study on fluent interaction with multi-touch in traditional gui environments. In *TENCON 2007*, pages 1–4, 2007.

[128] J. Stewart, B. B. Bederson, and A. Druin. Single display groupware: a model for co-present collaboration. In *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, CHI '99, pages 286–293, New York, NY, USA, 1999. ACM.

[129] N. A. Streitz, J. Geissler, T. Holmer, S. Konomi, C. Muller-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*, CHI '99, pages 120–127, New York, NY, USA, 1999. ACM.

[130] 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*, CHI '99, pages 120–127, New York, NY, USA, 1999. ACM.

[131] M. Sugimoto, F. Kusunoki, and H. Hashizume. Supporting face-toface group activities with a sensor-embedded board. cscw workshop on shared environments to support face-to-face collaboration, 2000.

[132] A. Sugiura and Y. Koseki. A user interface using fingerprint recognition: holding commands and data objects on fingers. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, UIST '98, pages 71–79, New York, NY, USA, 1998. ACM.

[133] S. Suzuki and K. Be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, Apr. 1985.

[134] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.

[135] C. Systems. Cyberglove, Feb 1990. `http://www.cyberglovesystems.com`.

[136] C. Theobalt, I. Albrecht, J. Haber, M. Magnor, and H.-P. Seidel. Pitching a baseball: tracking high-speed motion with multi-exposure images. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 540–547, New York, NY, USA, 2004. ACM.

[137] J. Thumpson. Beyond words: nonverbal communication in the classroom. New York, NY, USA, 1973.

[138] C. Tomasi, A. Rafii, and I. Torunoglu. Full-size projection keyboard for handheld devices. *Commun. ACM*, 46:70–75, July 2003.

[139] P. Tuddenham and P. Robinson. Territorial coordination and workspace awareness in remote tabletop collaboration. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 2139–2148, New York, NY, USA, 2009. ACM.

[140] B. Ullmer and H. Ishii. The metadesk: Models and prototypes for tangible user interfaces. pages 223–232. ACM Press, 1997.

[141] U. View. Sharp vga touchscreen/fingerprint reader, Feb 2007. `http://www.unwiredview.com/2007/02/26/sharp-vga-touchscreenfingerprint-reader, last accessed May. 7, 2011`.

[142] C. von Hardenberg and F. Bérard. Bare-hand human-computer interaction. In *Proceedings of the 2001 workshop on Perceptive user interfaces*, PUI '01, pages 1–8, New York, NY, USA, 2001. ACM.

[143] S. W. Heat check: Swine flu, body heat, and airport scanners, apr 2009. `http://www.slate.com/id/2217148, last accessed May. 1, 2011`.

[144] F. Wang, X. Cao, X. Ren, and P. Irani. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST '09, pages 23–32, New York, NY, USA, 2009. ACM.

[145] F. Wang and X. Ren. Empirical evaluation for finger input properties in multi-touch interaction. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1063–1072, New York, NY, USA, 2009. ACM.

[146] R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Trans. Graph.*, 28:63:1–63:8, July 2009.

[147] G. Welch and G. Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 1995.

[148] S. Wiedenbeck, J. Waters, L. Sobrado, and J.-C. Birget. Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '06, pages 177–184, New York, NY, USA, 2006. ACM.

[149] A. D. Wilson. Touchlight: an imaging touch screen and display for gesture-based interaction. In *Proceedings of the 6th international conference on Multimodal interfaces*, ICMI '04, pages 69–76, New York, NY, USA, 2004. ACM.

[150] A. D. Wilson. Playanywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 83–92, New York, NY, USA, 2005. ACM.

[151] A. D. Wilson. Using a depth camera as a touch sensor. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 69–72, New York, NY, USA, 2010. ACM.

[152] A. D. Wilson and R. Sarin. Bluetable: connecting wireless mobile devices on interactive surfaces using vision-based handshaking. In *Proceedings of Graphics Interface 2007*, GI '07, pages 119–125, New York, NY, USA, 2007. ACM.

[153] M. Wu and R. Balakrishnan. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, UIST '03, pages 193–202, New York, NY, USA, 2003. ACM.

[154] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35:399–458, December 2003.

[155] R. Ziola, S. Grampurohit, N. Landes, B. Harrison, J. Fogarty, and D. Fox. Applying a sensing-based interaction framework using an object-aware siutated interaction system (oasis). Submitted for publication in CHI 2011.

List of Figures

2.1 Resistive touch technology [46]. 6
2.2 The mutual capacitance sensor configuration in SmartSkin[102]. 6
2.3 The DiamondTouch setup [28]. 8
2.4 Frustrated Total Internal Reflection (FTIR) setup [114]. 10
2.5 Captured images with an infrared camera in a (1) FTIR setup, (2) rear DI setup, (3) front DI setup, (4) DSI setup, (5) DI setup without diffuser. 10
2.6 A rear diffused illumination setup [114]. 11
2.7 A front diffused illumination setup [33]. 12
2.8 A Diffused Surface Illumination setup [114]. 12
2.9 The basic construction of ThinSight [55]. 14
2.10 The Microsoft Surface 2 in which infrared sensors are integrated in every pixel of the LCD display [24]. 14
2.11 ZeroTouch: a frame which can detect the position of all fingers and objects inside [80]. 15

3.1 The Dynamo system provides a single mouse and keyboard for each user [54]. . 18
3.2 Group members need to form a chain of hands to exit the CollabDraw application [82] and feel socially awkward. 20
3.3 The public and private areas in the IBM BlueBoard system [108]. 21
3.4 The Dynamo workspace arrangement: a personal palette for each user, at the top of this figure a carved region is shown shared by the two users [54]. 22
3.5 The Table for N created with the DiamondSpin SDK [122]. 22
3.6 Covering a personal object with a hand [153]. 23
3.7 Two different users executing a pulling away gesture [26]. 28

4.1 Classification of some important touch technologies based on their multi-touch and multi-user capabilities [59]. 31
4.2 Without assigning fingers to hands, the gesture on the left cannot be distinguished from the gesture on the right [26]. 31

4.3 Two users interacting on their interaction workspaces to give input in a bus route map application [59]. 32
4.4 Extending a traditional optical-based touch table with RFID identification capabilities [91]. 34
4.5 Determining the position of an RFID tag using an integrated photo-sensor called RFIG tag [99]. 35
4.6 Cricket based infrastructure for touch identification on a multi-touch table [125]. 37
4.7 A cricket listener attached to a user's hand [125]. 37
4.8 1. Basic allocation through nearest neighbour, 2. advanced allocation using the wristband's orientation [79]. 40
4.9 The P5 dataglove. 41
4.10 A glove with a color patterns to track postures of a hand very accurately [146]. 42
4.11 A fiduciary-tagged glove[75]. 42
4.12 Authentication scheme of users interacting with a multi-touch display: 1. The user touches the wall with the phone. 2. The mobile phone sends a light flash to indicate the region the user wants to interact with and at the same time sends his identifier to the system. .3 The system combines the touch data, light flash and identifier to check the access rights of this user and responds to the device [116]. 45
4.13 The design of the uPen [11]. 46
4.14 The Anoto pattern printed on a special rear-projection foil[17]. 47
4.15 The Anoto pen used on a optical-based touchscreen to identify the users [69]. . 47
4.16 1. An odd frame in which the finger contacts are visible. 2. An even frame in which hand contours are visible [33]. 49
4.17 Integrating proximity sensors in multi-touch screens to know which places are occupied around the table [40]. 50
4.18 A depth camera can provide positional information of users around an interactive surface [151]. 52
4.19 Thermal heatwave of two users interacting on a surface [67]. 53
4.20 Hand geometry sensing device [58]. 54
4.21 Hand contour feature extraction [15]. 54
4.22 Two ways of finger touch. (1) oblique touch. (2) vertical touch. [144] 56
4.23 Finger contact region fitted to an ellipse [144] 57
4.24 Finger contact deformation over time. The crosshair shows the center of the contact region. [144] 57
4.25 Two touch points originated from the same hand [26]. 60
4.26 Two situations in which both touch contacts are originated from the same hand: 1. Performing a grap gesture. 2. Performing a pump gesture. [26]. 60

4.27 Levels of feature extraction in the analysis of fingerprints [100]. 62

5.1 PressurePIN: a pressure-based authentication mechanism for inputting PIN-codes [64]. 67
5.2 Reaching consensus by performing the same gesture simultaneously [82]. 69
5.3 Two users simultaneously touch a photo to transfer ownership. [82]. 69
5.4 Personalizing the interface appearance by translating text to the user's preferred language [112]. 71
5.5 An example of customizing content: moving a lens over a login area automatically fills in credentials [113]. 71
5.6 The SDMouse to emulate the functionality of a 3-button mouse with multi-finger input [76]. 73
5.7 A colored visualization of the vector fields on the table to provide automatic orientation of documents [32]. 74
5.8 The DiamondSpin software [122]. 75
5.9 Orientation-sensitive controls: (1) Orientation-sensitive button. (2) Orientation dial. [144] 75
5.10 Orientation of the input is needed to discriminate between these four gestures [81]. 75
5.11 Ambiguous gestures caused by different hand orientations [144]. 76
5.12 Design of a menu that avoids occlusion caused by the user's hand [18]. 77
5.13 (1) Regular bubble cursor. (2) Directed bubble cursor. (3) Distance weights according to finger orientation. [144] 79
5.14 (1) Aim finger to select an object. (2) Move finger along the orientation vector to switch between multiple objects. [144] 80
5.15 The finger cross selection technique to select objects from a distance [145]. . . . 80

6.1 Extracting geometry features from the human hand. 86
6.2 The histograms for the three color spaces of a single patch of an image of the back of the hand. 86
6.3 Matching a patch of an image to valid patches of an image in the training set based on the location of the patch. 88
6.4 1. The average image 2. An example of an eigenimage. 88
6.5 The basic local binary pattern operator. 89
6.6 The local binary pattern representation of a single channel (in this case the red channel) of two images under different illumination. The local binary patterns are identical. 90
6.7 Examples of neighborhoods that can be captured with the circular local binary pattern operator. 90

6.8 If there are 8 sample points on the circle with the CLBP-operator, all patterns can be categorized under 10 labels using uniform patterns. 91
6.9 A patch on a hand of which each of the three color channels are encoded as a binary pattern. 93
6.10 1. An image of a human hand in the YCrCb color space. 2. Result of the skin segmentation. 3. Result of the opening and closing filter. 4. The extracted contour visualized in the original image. 95
6.11 Calculating the DOS of a curve. 96
6.12 1. A flat hand extracted from an image (skin segmentation). 2. The DOS values of the contour in each point. 3. The cumulative DOS values. 4. The extracted finger tops and valleys obtained by thresholding the cumulative DOS values. 5. The final result. 97
6.13 1. A hand in a pointing posture extracted from an image (skin segmentation). 2. The DOS values of the contour in each point. 3. The cumulative DOS values. 4. The extracted finger tops and valleys obtained by thresholding the cumulative DOS values. 5. The final result. 98
6.14 1. An image in which the sleeve of a shirt is visible. The skin region is already highlighted (see Section 6.4.1). 2. The reference image. 3. Background subtraction of image (1) with image (2). 4. The resulting image in which we can now detect the base line of the arm. 100
6.15 1. An image in which the skin region and the baseline of the arm is detected. 2. Extracting the final direction of the arm. 101
6.16 Determining the position of the wrist. 103
6.17 Blue function: the lengths of all potential wrists in Figure 6.16. Red function: the valid region for the wrist. 103
6.18 Blue function: the backward difference (neighborhood size of 5) for all the values of the blue function in Figure 6.16. Red function: the valid region for the wrist. 104
6.19 Blue function: the forward difference (neighborhood size of 5) for all the values of the blue function in Figure 6.16. Red function; the valid region for the wrist. 104
6.20 The back of the hand extracted from an image of a hand on a table captured by an overhead camera. 104

7.1 Setup of the quick-and-dirty test. 107
7.2 1. Images of the hand of a single participant pointing at different regions of the surface in the same posture. 2. The hands extracted using the hand recognition algorithm discussed in Section 6.4. 108
7.3 Differences in the illumination of the hands because of the shadows casted by the body of the participant. 108

7.4 Marked touch positions on the table. 110
7.5 1. The handsdown posture. 2. The pointing posture. 3. The free hand pointing posture. 110
7.6 A display instructed the participants to touch points on the table in specific postures. A digital still camera captures images of the table which is evenly illuminated by artificial light sources. 112
7.7 The recognition rate of the identification technique over multiple postures. . . . 115
7.8 The match rate over multiple postures. 115
7.9 The recognition rate of the identification technique when changing the resolution of the overhead camera. 118
7.10 The match rate when changing the resolution of the overhead camera. 118
7.11 The recognition rate of the identification technique when changing the patch size. 120
7.12 The match rate when changing the patch size. 120
7.13 The recognition rate of the identification technique when changing the area in which to search for similar patches. 122
7.14 The match rate when changing the area in which to search for similar patches. 122
7.15 The difference in match rates between the positive match rate and the highest negative match in Figure 7.14. 123
7.16 The recognition rate of the identification technique when changing the radius of the CLBP-operator. 123
7.17 The matching rate when only 4 users are registered with the system. 126

8.1 The setup of the real-time identity server. 128
8.2 A conceptual visualization of the most important components of our system. . 130
8.3 A user that registers with the system by providing training samples of his hand. 132
8.4 Two users that are working simultaneously with our identity-aware painting application. 132

List of Tables

3.1 Coordination policies grouped along the dimensions of conflict type (rows) and initiative strategy (columns) [83]. 28

5.1 The structure of the pattern language. 65

9.1 Strengths and weaknesses of existing user identification technologies. 136