2011

Doctorate thesis

Leveraging Finger Properties for Natural Interaction with Direct-Touch Surfaces

1126012 Feng Wang

Advisor Xiangshi Ren

Feb 2011

Course of Information Systems Engineering

Graduate School of Engineering, Kochi University of Technology

Abstract

Leveraging Finger Properties for Natural Interaction with Direct-Touch Surfaces

Feng Wang

Interactive techniques for touch and multi-touch surfaces are novel human computer interaction techniques and have attracted considerable attention from general public due to their inherently natural affordances. One main reason for this naturalness is derived from the ability to let users employ their bare fingers and directly manipulate the system without intermediary devices. Researchers have demonstrated that direct-touch interactive displays offer a more compelling method to interact with a system than working indirectly with a mouse or with other types of pointing devices.

However, there are a few distinct drawbacks which limit the application of direct-touch technology. Albinsson and Zhai [1] reported that the occlusion of screen data caused by the fingers and the hand, very low selection precision and arm fatigue were significant limitations of touch devices. These drawbacks need to be overcome by new sensitive surface techniques.

Based on a survey of earlier studies, we concluded that current multi-touch techniques do not fully exploit the characteristics of the human hand or fingers. The finger's functions in currently available multi-touch interfaces are merely to position the cursor and to click events. Actually, the human hand is a complex mechanism. When one finger contacts the touch panel, the multi-touch device can only generate the contact area coordinates (x, y) of the flat (2-D) surface. Due to the limitations of fixed 2-D touch screen surfaces, the actual number of Degree Of Freedom (DOF) is reduced to ten under the condition that five fingers are used simultane-

ously. This decrease in the DOF number seriously reduces the number of gestures available to interface designers.

To enhance the usefulness of interfaces incorporating sensitive surface techniques and to overcome the drawbacks mentioned above, in this dissertation, after implementations of multi-touch system and its low-level algorithms, we empirically explore a wide range of finger input properties that are capable of controlling targets, i.e., it is likely that we can control computers with more natural and more comfortable gestures. Based on the results of our experiments, the shape of the finger contact area, the size of the contact area and the orientation of the contact finger are effective finger properties that are useful for the design of natural multi-touch gestures. We then deeply investigate the fundamental principles of all the fingers in tapping task and discuss the operational availabilities of fingers. Our results indicate that the fingers have different abilities and potentials for target selection. Four fingers (i.e., left and right index fingers and middle fingers) are more suitable for multi-touch tapping manipulation.

In particular for the orientation, we explore user interface designs that leverage this finger orientation information, as well as further inferences that can be made from finger orientations. These designs and inferences can be useful for interaction with a variety of direct-touch devices that generate finger orientation information, either using our general algorithm or other more specialized sensing technologies. Our work shows that finger orientation is a feasible and valuable input dimension that can be utilized for novel interactions on interactive surfaces.

In summary, this dissertation has established an valuable point that the exploitation of finger's properties allows users to communicate information to a computer faster and more smoothly. New generation user interface can fully use the natural properties to answer the challenges that lie beyond the realm of todays mouse-and-keyboard paradigm. We believe that the usage of natural input properties of human body would bring a new world to future interfaces design.

key words direct-touch, multi-touch, natural user interaction, input properties, touch interaction technique, contact area, contact shape, orientation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fig. 1.1 Apple iPhone is a mobile phone that allows us to make a call by simply pointing your finger at a name or number in address book, a favorites list, or a call log.

Not everyone really believes that a small mobile phone would change the life-style for millions of people. iPhone, a miracle of information technology, does it. The very core of the iPhone is not only the high-tech integration and beautiful figure, but the multi-touch technique and Human Computer Interaction (HCI) technology. Due to the advances of computer and network technologies, drastic changes are taking place in the human society in which relationship between computer and personnel is becoming more important and we rely increasingly on computers for help/to help us. As an interdisciplinary subject of computer science, psychics, physics, mathematics and education, human computer interaction technology is being treasured by many universities, institutions and companys. Generally, current HCI research [3, 4] is advanced in a direction of more natural, more effective, more intelligent and freer.

User Interface (UI) technique is typically treated as one of the hottest questions in HCI research. UI is responsible for information input and output of computer systems in human-computer interaction, produces possible feedbacks, and influences the manipulations of end users [4, 5]. In general, there are three stages in the development of user interface techniques: command line interfaces (CLI), graphicalal user interfaces (GUI) and natural user interfaces (NUI) (see Fig. 1.2). Due to the rapidly growth in popularity of MacOS of Apple and Windows operation system of Microsoft, GUI is the most widely used interface in public life today. GUI uses desktop as metaphor and WIMP as paradigm. However, many drawbacks of GUI still seriously limit users' manipulation on computer. The users have to make frequent menu selections, button operations and command inputs with keyboard, which will result in operation procedure discrete. Even though a task can be implemented in a single-pass, current softwares always break the implementation into a sequence of steps.

Touch technology, a general name of single-point and multi touch technique, is typically regarded as an important part of natural human computer interaction because people manipulate computer devices with their fingers directly without any intermediary media. More importantly, people do not need specific training on how to use touch devices when they face to them. Based on our observation in daily life, many people, even never used touch devices before, can easily and naturally get used to manipulation of those devices without any instructions.

The goal of our research is to investigate the significance of natural user interface and make our interaction more simple, more facile, more natural on direct touch environments. In fact, we have accustomed to use multi-finger interaction in daily computer manipulations for typing keyboard, executing keyboard shortcuts. This dissertation will focus on the introduction of more natural input properties and the interactions that are continuous and coordinated. Multi-touch makes this type of interaction possible. These devices detect every point wherever their surface is in contact with the user's hands or fingers. Thus, they can be used to control many more properties than traditional pointing devices.

1.1 Research Motivation

Fig. 1.2 The development of user interface from Command Line Interface, Graphical User Interface to current Natural User Interface. The pictures of CLI and NUI are cited from Apple Corporation (http://www.apple.com).

Natural user interface is a universal goal pursued by all people around the world. Shneiderman [6] argued that the new computing mode is about what people can do. To accelerate the transition from old to new computing mode, we should reduce computer user frustration, promote universal usability and envision a future in which human need more directly shape technology evolution. The natural instincts and the demands of the people will never change, no matter how the computer technology is inventing and is developing. Alan Cooper, a famous HCI scientist, profoundly described a “stream state” in natural user interaction from users’ psychological level [7]. In the stream state, users are awfully contemplative, often generate minor happyness, and are unconscious of the lapse of time. This state is the real portrait of the natural user responses. The major theme of NUI is to demand the interaction design to provide more stream state to the users. The users can use voices, actions and other natural expressions to communicate with computer seamlessly so as to apparently decrease the difficulties of interaction and promote the interactive performance [8, 9].

This thesis is concerned with natural direct-touch interaction techniques and interrelated design and input properties that influence human performance. The need for this research has emerged from the development of a variety of touch based devices such as mobile phones, PDAs, e-Ink readers and tablet computers. Touch-based interaction is an attractive user-computer interface paradigm. The touch-based interaction is one of natural computer interactions that can enable the ultimate ubiquitous computing.

Although touch-based interactions have much potential to facilitate computer users, there is still dilemma that cumbers advantages of touch. Today's touch sensing devices are typically designed to underly the principle of WIMP (windows, icons, menus and pointing). Therefore, the pointing and target selection are the fundamental actions in device's manipulation. Using bare finger to select targets with touch sensing devices carries both benefits and limitations. By bare finger, touch sensing devices allow user to direct interact with the devices easily and freely. Due to their usability, touch sensing devices are widely used in public installation, such as automated teller machines, telephone kiosks and video surveillance. However, as presented in Albinsson and Zhai [1], three limitations, i.e., the occlusion of screen raised by fingers, low precise target selection and arm fatigue, critically decrease the usability of touch sensing techniques.

Current touch-based interactions mainly rely on the coordinates of fingers. Although users is gradually growing into the main part of the interaction between human and computer, the interactive utilities of us used for interaction are limited. While interacting with modern graphical user interface, we only use our single index finger to click the button in order to tell computers where the mouse pointer is. It seems that the current information society heavily depend on our single index finger and finger's position (x, y). However, the mouse is only an input device with only 2 degrees of freedom input device, therefore it is hard for people to fully apply the hand operating skills learned in their natural life to human-computer interaction to reduce cognitive burden of the interaction, and improve the efficiency of computer operations. On the contrary, our hand has very high degree of freedom (with 23 degrees of freedom [10]). People are accustomed to do their work with their natural behavioral habits in real life. For example, people typically use one finger to touch buttons, use index finger and thumb to insert a card into ATM or vendor machine, use one hand to fetch a book and use two hands to move a heavy object. It is scarcely imaginable that people can use one index finger to enter information era.

To solve the problems mentioned above, especially design more natural user interface for novices, we begin to address this issue by investigating more human input properties. The key research point of natural user interface is to fully exploit the natural input properties of human's body.

Aiming at improving performance and subjective usability for direct-touch systems, we seek to:

- Investigate the characteristics of bare finger, and identify and quantify the influential design factors that make direct-touch interaction techniques more efficient and more natural.
- Design and develop interaction techniques that are suitable for direct-touch systems.

1.2 Thesis Statement

This dissertation is intended to support the following thesis:

No matter how the information technology is developing, the leveraging of finger's natural properties such as contact position, contact area, area size, contact shape, orientation and so on are capable of allowing users to communicate information to a computer faster, more natural and more smoothly in a wide variety of tasks.

1.3 Dissertation Overview

The goal of this dissertation is to show how interaction techniques that use natural input properties of human finger improve the quality of human-computer interaction through enlarge bandwidth and increase input degree of freedom. The structure of this dissertation is shown in Fig. 1.3.

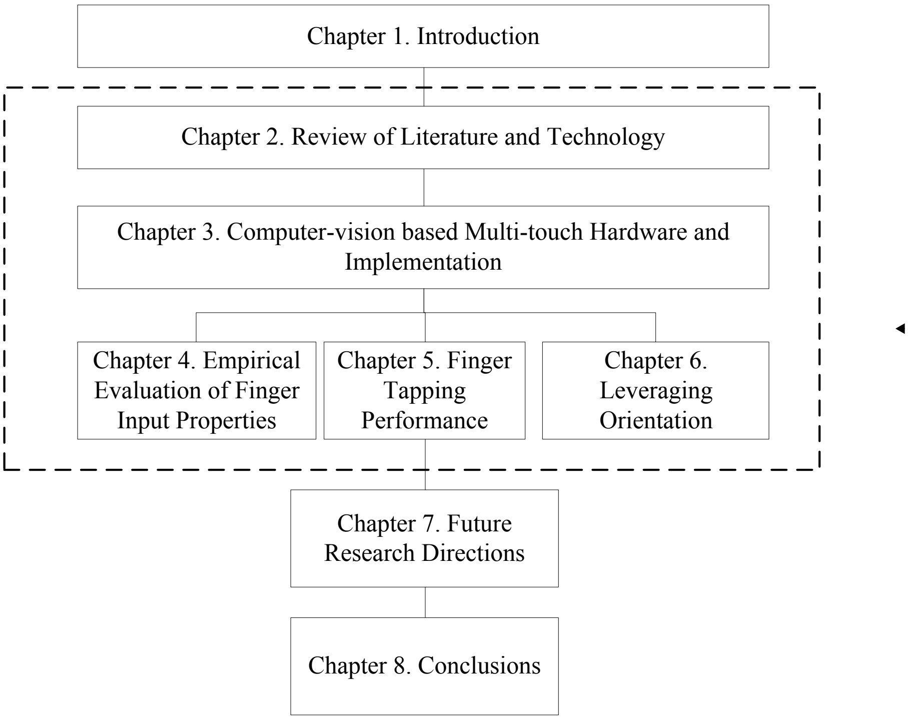

Fig. 1.3 The dissertation structure.

After a brief instruction, we give a survey on direct-touch surface technology including hardware design, input properties and their uses, principles of bare fingers, bimanual manipulation and finger's orientation in depth (Chapter 2). We then begin by discussing a computer vision technique for collecting finger input properties (Chapter 3). This includes the development of computer vision multi-touch experimental platform and related low-level techniques such as blob detection, blob tracking and events. Concentrating on the high-resolution touch device, we implement an algorithm on barycentric coordinates and an image splicing algorithm to support high-performance large screen multi-touch devices. With the support of this device, we empirically evaluate many potential input properties of human finger. Based on the results of our experiment, the shape of the finger contact areas, the size of the contact area and the orientation of the contact finger are effective finger properties that are useful for the design of natural multi-touch gestures (Chapter 4). To help interface designers develop future multi-touch interaction techniques, we investigate the fundamental principles of all the functions of fingers in tapping task and discuss the availabilities of fingers. Our investigation promote the understanding of such methods, and allow us to formulate guidelines for their design (Chapter 5). Finally, we demonstrate that finger orientation information is crucial in the design of orientation-aware interactions and widgets, for example to facilitate target selections, or to optimally orient elements in the workspace to adapt to the user's position. Additional information about the user can also be inferred from finger orientation, such as hand occlusion region or the position of the user. These cues can in turn be leveraged to further enrich the interaction on touch-surfaces (Chapter 6). The contributions of this work are discussed in Chapter 8.

1.4 Contributions

This dissertation contributes to the field of human computer interaction in several ways. It implements a robust computer vision based multi-touch device and two algorithms such as barycentric coordinates conversion and panel splicing are presented. The device can output a series of significant information such as finger's touch area, touch area shape, coordinates of touch position, orientation of the touch finger and so on. It extends the human's input bandwidth and provides more useful information for interface design.

We further empirically evaluate all the availability of each input properties and the characterizations of touch input properties of the human hand in tapping, rocking and orientation gestures. We propose a set of design recommendations based on their measures and a set of ideas for new interaction techniques based on rarely used properties of touch. This form of detailed analysis is appealing to the multi-touch community as it can benefit from this work.

To present design principles of human's fingers, we further investigate the tapping performance of ten fingers of human's body via Fitts' law. The established results form a general frame - work act for the design space of multi-touch interaction. This work also profoundly improves our understanding of interaction using multiple fingers of two hands. Our experiments show that people can effectively use index and middle finger to implement tapping task "as accurately and quickly as possible". To a certain degree, 4 points multi-touch devices can meet the requirements of users enough.

In order to leverage more natural input properties and to provide more flexible graphical interaction in multiple domains, we investigate the detection and the use of finger orientation for multi-touch surfaces. Importantly, we present a technique based on the dynamics of finger deformation on the sensed surface to determine orientation. This is worthwhile and offers a richer variety of interaction techniques for orientation.

In summary, this dissertation enhances the understanding of leveraging finger's natural properties such as contact position, contact area, orientation, directional moving ability and so on to communicate information to a computer faster, freer, more naturally and more smoothly in a wide variety of tasks.

1.5 Terminology

Degree-of-freedom or DOF, is referred to as independent displacements and/or rotations that specify the orientation of the body or system.

Natural User Interface or NUI, is the common parlance used by designers and developers of computer interfaces to refer to a user interface that is effectively invisible, or becomes invisible with successive learned interactions, to its users. The goal is to make the user feel like a natural.

Natural User Interaction In this thesis, natural user interaction is defined as the interaction based on natural user interface.

Bimanual interaction technique Interaction technique involving the use of both of a users hands.

Coordination Control of two or more parameters simultaneously so as to attain a goal more effectively than by independent control.

Multi-touch interaction technique Unless otherwise stated, this term refers to methods that use multiple contact points on a multi-point touch pad for continuous control of multiple parameters in a software system.

Multi-touch, multi-point, multi-finger Relating to the use of multiple finger contacts on a multi-point touchpad.

Direct-Touch Relating to the touch or multi-touch devices, users use finger to directly touch on the surface.

Property In this thesis, the term “property” is referred as the measurable properties that a user may want to control. These may be properties of digital or physical objects (e.g., object layout) or properties of the users body (e.g., finger’s orientation).

Chapter 2

Review of Literature and Technology

2.1 Direct-touch Technology

As mentioned in Chapter 1, WIMP (windows, icons, menus, pointing) is the current main human-computer interaction paradigm in graphical user interface. Mouse is the most widely used pointing device in graphical user interface to provide 2-D positional information. Due to the use of mouse, it is hard for people to fully apply the hand operating skills learned in their natural life to human-computer interaction to reduce cognitive burden of the interaction, and improve the efficiency of computer operations. Multi-touch equipments allow one or more than one user with multiple fingers to interact with computers through graphical user interfaces. Compared with the traditional input devices, one of the biggest advantages of multi-touch capable devices is its ability to accommodate multiple user's simultaneous operations. Actually, our hand and fingers are of very high degree of freedom (with 23 degrees of freedom [10]), and can touch directly without any intermediary media, which greatly enhances the efficiency of our interaction with computers.

Although as early as 1982, Nimish Mehta of Toronto University, has invented the first Multi-touch display based on the pressure of fingers [11]. However, this system has been limited by its availability and extremely high price. Such situation has been changed even with the release of Apple's iPhone, more people are beginning to know and get access to multi-point touch technology. In 2005, Jefferson Y. Han [12], from New York University, proposed a FTIR-based low-cost multi-touch equipment, which has greatly reduced the research cost of multi-touch technology, so that its research has been launched in all over the world, and many new multi-touch technologies [13–16] have come available. Moreover, some products have been successfully commercialized [17].

Various technologies with respective characteristics have been introduced to develop multi-touch capable devices. These technologies are divide into two categories (i.e., sensor based system and computer vision based system) by their approaches to the problem of recognizing and interpreting multiple simultaneous touches.

2.1.1 Sensor based Systems

Many multi-touch devices adopt sensor technology [18–20], can simultaneously detect multiple touch points to identify the multiple points of input. At present, the sensor is always integrated into the screen's surface to be utilized for mobile phones, PDAs and other small-screen handheld because the sensors exist serious impact of environment temperature and humidity on system performance. This determines the systems based on sensor technology are quite expensive.

In 1985, Lee et al. [20] made FMTSID (Fast Multiple Touch Sensitive Input Device), one of the first multi-point touch sensor based devices. This device consists of a sensor matrix panel, ranks of select register, an A/D converter and a control CPU component. It can detect finger touch points by measuring the changes in capacitance. FMTSID can accurately detect multiple finger touch position and finger contact pressure.

The Diamond Touch [18] developed by Dietz et al. at Mitsubishi Electric Research Laboratories (MERL) in 2001 is a multi-user touch-sensitive surface which supports multiple users and a front multi-touch camera. The desktop is a projection screen and a touch-screen as well. A large number of antennae are set below the touch screen. Each antenna transmits a specific signal and each user has a separate receiver. The user's conductivity accommodate signal transmission through his or her seat. When a user touch the panel, the antenna around the touch point transmits weak signals between the user's body and the receiver. This unique feature not only supports multiple contacts of single user (for example bimanual interaction), but also distinguishes between the simultaneous inputs of different users (up to 4) without interfering with each other. The system also can detect the pressure of touch point and allow rich gestures without the interference of foreign objects. On the other hand, DiamondTouch cannot, like other multi-touch technologies, identify multiple touch locations by the same user. Diamond Touch suffers the following disadvantages: it can not recognize objects placed on the surfaces but "touch" movement [21]; Diamond Touch projects images from above down to the desk, so when used, the human body would shadow the display, which hinders the operation.

On the basis of the FMTSID principle proposed by Lee et al. [20], Rekimoto et al. [19] created Smart Skin at Sony Computer Science Laboratory in 2002. Smart Skin is a multi-touch system with higher resolution ratio. The system consists of grid-shaped transmitter and receiver. It can not only identify the number of hand contact position and their shape, but also calculate the distance between the hands and contact surfaces through capacitive sensing and grid antennas. When compared with Diamond Touch, Smart Skin manifests higher capability in returning more abundant contact information (i.e., the finger contact shape). This has inspired Cao et al. [22] who have designed novel interactions by using the shape of contact fingers.

The Apple iPhone released in 2007 is the first mobile device with access to multi-touch technology. iPhone uses capacitive coupling technique to sense multiple touch points. iPhone can achieve multi-touch with limited dimensions, allow people to operate by mere hands, and allow typing through a virtual keyboard, the dial of telephone numbers and the "pinching" technique introduced by Krueger [23] (with the thumb and index finger of the same hand to zoom the map and the photo). These cannot be achieved by traditional input methods like a mouse and a keyboard. Those features of iPhone refresh the common people. iPhone SDK attracts much interest of researchers in the applied research of multi-touch technology in handheld devices.

2.1.2 Computer Vision Based Systems

Due to the falling cost and improved performance of computers, computer vision technology has been greatly developed, which enables us to process real-time signals and high-speed video to meet the requirements of real-time human-computer interaction. Thus researchers have put forward a number of multi-touch systems based on computer vision techniques [12, 23–29].

2.1.2.1 Vision Only Systems

Vision only multi-touch systems rely solely on image processing techniques to identify touches and their positions. Multi-touch systems which employ this technique can be used on any flat surface without the need for a dedicated display device and are of very high portability [24, 30]. However, the flexibility of vision only systems comes at the cost of precision.

Pinhanez et al. [30] have created a computer vision based system called "Everywhere Display".The system uses a camera and projector to turn a common touch screen into an interactive display screen through image processing technology. Despite Pinhanez et al. [30] did not provide any data about the accuracy of the detection algorithm in their paper, it is clear that they have chosen portability at the expense of choice accuracy. Compared with other multi-touch technologies, it is difficult for Everywhere Display to accurately determine the time and finger touch-screen duration.

Microsoft's PlayAnywhere is a relatively compact and well mobile desktop interactive system with a front camera. Wilson [24] has developed many image processing techniques for the desktop interactive system with a front camera based on computer vision. Most notable of his work is the shadow-based touch detection algorithm, which can accurately and reliably detect touch events and their contact position. However, Agarwal et al. [31] pointed out that the algorithm could achieve the best result only when the point of finger is vertical, which limits the system in a collaborative environment application.

Agarwal et al. [31] has developed a computer vision algorithm to improve computer vision based multi-point interactive desktop choice accuracy (accuracy 2-3mm) in terms of the three-dimensional imaging and machine learning technology, which can accurately detect the fingertips touch. The precision, which is up to 98.48% accuracy in comparison with previous technical-level (the choice of precision is generally centimeter level), has been greatly improved.

2.1.2.2 Computer Vision and Optical Hybrid System

Devices based on computer vision and optical multi-touch technology has good scalability and relative low cost, but they have a larger volume. Here are two kinds of computer vision and optical-based multi-touch systems.

Frustrated Total Internal Reflection(FTIR). Frustrated Total Internal Reflection is a kind of optical phenomenon. Beams of LEDs (light-emitting diode) will reflect once they reach the surface of the screen from the touch-screen cross-section. However, if there is a relatively high refractive index material (such as a finger) suppressing the acrylic materials of the touch panel, the conditions of total reflection will be broken. Some of the beams would be projected onto the surface of fingers through the screen surface. The tough finger surfaces cause scattering (diffuse reflection), and the scattered light would be read by the infrared camera set under the acrylic board through the touch screen. In fact, FTIR principle has long been used to produce a number of input devices, such as a fingerprint reader. Jefferson first used FTIR principle to build a low-cost multi-point touch screen [12], which greatly reduced the multi-touch technology research cost.

The corresponding touch information can be detected through corresponding software, *Touchlib* [32]. *Touchlib* is a set of software library developed by NUI Group for the multi-touch system development, which implements the majority of computer vision algorithms. This technique can detect multiple touch points and the location of exposure by using only a simple Blob detection algorithm [12, 33].

Diffused Illumination(DI). DI-based multi-touch technology refers to infrared radiation which reaches the touch screen from the bottom of, and places the diffuse reflection surface on or unearth touch screen. When objects touch the screen, the screen will reflect more infrared light than the diffuse reflectance does, and then the camera would read and the corresponding touch information would be detected through the *Touchlib* [32]. With this diffuse reflection screen objects hovering and on the surface can also be detected.

Compared with FTIR, DI technology has certain advantages. DI system can detect objects' hovering state (the system can recognize hand or fingers moving across the screen, or closer to the screen, without having to actually touch). In addition, the DI-based systems rely on "see" what is on the screen, rather than detect touch, and so, DI is able to identify and detect objects and object tags. But, compared with the simple use of Blob tracking and detection algorithm of FTIR, DI uses more complex image processing technology. In addition, DI system is vulnerable to external light effect.

Microsoft's Surface [25] is the multi-touch system based on the back of the DI technology. Surface built-in camera can not only sense input of users such as the touch and gestures (finger moving across the screen), but also be able to identify and capture the required information of objects placed on the above. This information is sent to the common type of Windows PC for processing, and the results from the digital light processing (DLP) projector are sent back to the Surface. Microsoft Surface is able to sense multiple fingers and hands, and can identify a variety of objects and their location on the surface.

There are other computer vision based and optics-based multi-touch systems such as: laser plane multi-touch technology proposed by Alex (LLP); light-emitting diodes planar multi-touch technology (LED-LP) made by Nima; the scattered light plane multi-touch technology (DSI) presented by Tim Roth. These technologies can be used to build multi-touch devices. For more information, one can visit the Natural User Interface Group (NUI Group) open-source community website (http://www.nuigroup.com). NUI Group provides an environment for mutual exchange for developers interested in human-computer interaction and its members have collected and shared a lot of information and valuable experience about the building of multi-point touch system.

2.2 Input Properties of Current Direct-touch Technology

We refer to earlier literature and investigate the use of finger input properties in touch and multi-touch techniques. We then analyze and sort all the input properties of fingers into four aspects and illustrate them in Table 2.2. The four aspects are position, motion, physical and event properties.

2.2.1 Position Property

The important advantage of touch techniques is that the bare finger can directly operate on the touch screen without other intermediary devices. The contact position of the finger is the first property considered in widget design. Most commercially available touch screen devices in use today are capable of detecting and tracking a single point on the touch panel of the device. With the recent emergence of many multi-touch prototype devices [12, 17–20, 24, 29], research on multi-finger and multi-hand touch interactions has increased [24, 29, 40, 41]. In order to ensure compatibility with traditional GUIs and to permit the sharing of the same interfaces (e.g., a cursor, drag and drop technique and click action), the center point of each contact area is often used as the cursor position.

While touch screens offer direct manipulation, they do have their limitations. The user's

Table 2.1 Classification of human finger properties.

Input Property	Finger Property	Application State
Position property	Coordinate value (x, y)	Widely used, firstly studied by Buxton [11] [34] and Lee [34]
Motion property	Velocity	First adopted by Tuio protocol [35]
	Acceleration	
Physical property	Size of Contact Area	Partially used, i.e., SimPress [36] in the study of Benko
	Shape of contact area	Rarely used [37]
	Orientation	Never used
	Pressure	Used, Pressure Widget [38]
Event property	Tap	Commonly used
	Flick	Used [39]

finger, hand and arm can occlude a significant area of the screen that may lead to low selection precision when pointing at targets that are smaller than finger width [1]. Hall et al. [42] investigated the effects of various factors on touch-screen performance. They reported that accuracy varied from 66.7% for targets of 10 mm per side, to 99.2% for targets of 26 mm per side, and that accuracy was maximized once targets were approximately 26 mm per side. But in that study, only the index finger was measured.

Due to the lack of precision [1], there have been significant studies in the area of precise selection. Potter et al. [40] explored a set of strategies for high-precision touch-screen pointing and presented the "Take-Off" technique. The user is capable of controlling a cursor which is located slightly above the finger. In this method the target is selected by releasing the finger from the surface. Albinsson and Zhai [1] compared Potter's approach with the traditional zoom-pointing method and two new interaction techniques: cross-keys and precision-handle. Vogel and Baudisch [43] presented “Shift”, a technique for single-touch displays that addressed the problems of the Take-Off technique, not by offsetting the cursor, but by showing a small offset callout that displays a copy of the area under the finger with its cursor. The callout is presented automatically when the finger is determined to occlude a sufficiently small potential target, and, in some variants, the small portion of the display in the callout is zoomed for easier selection.

2.2.2 Motion Property

The movement properties of the human finger have been deeply studied in the area of gesture recognition. Apple has filed a patent called “Multi-touch Gesture Dictionary” for iPhone 3G. The dictionary entries include a variety of motions and may take the form of a dedicated computer application. Kaltenbrunner et al. [35] presented the Tuio protocol to meet the requirements of tangible user interfaces for table-top devices. Tuio is a simple yet versatile protocol that defines the common properties of controller objects on the table surface as well as of finger and hand gestures performed by the user. The movement vector and motion acceleration are adopted by the Tuio protocol.

2.2.3 Physical Property

The initial investigation of the use of pressure in user interfaces was presented by Herot and Weinzapfel [44]. They explored the ability of the human finger to apply pressure and torque to a computer screen. Pressure Widget [38] and the subsequent studies of Ramos explored the use of the continuous pressure sensing capabilities of styluses to operate multi-state widgets. Contact area and pressure were studied by Benko et al. [36]. They used the rocking and pressing gestures of the tracked finger to trigger “click” events on a vision based tabletop.

Forlines et al. [45] indicated that two different finger contact postures, vertical contact and oblique contact, generate different contact area shapes. These differences cause different selection error rates. Their study only reports the difference between the two gestures but there is no follow-up discussion about target selection precision or the usability of the two gestures.

2.2.4 Event Property

Finger tapping is typically adopted to simulate the mouse "click" event. In recent studies about natural gestures, Reetz et al. [39] presented the "Superflick" technique for long-distance object placement on digital tables. The Superflick technique simulated the natural object sliding gesture of the human hand. They designed and evaluated two tabletop interaction techniques that closely mimic the sliding of an object across a table.

In brief, our review indicates that while there is a rich body of literature on finger input properties, there has not been a systematic investigation into the full range of human finger input properties, especially regarding contact shape and finger orientation. Even the currently adopted finger properties, such as coordinate values and contact area, still contain a lot of unexplored issues. Thus, this is an area that is ripe for further research.

2.3 Principles and Applications of Bimanual and Unimanual

2.3.1 Two-handed Interaction and Motor Control

Buxton and Myers [46] first proved by experiments that using both hands input in HCI can result in high performing efficiency and performance. Since then, two hands interaction begin to be extensively studied. However, Kabbash et al. [47] reported that if the two-handed input technique is not designed properly, two-handed interaction is not better than one-handed.

We review earlier literature of two-handed interaction techniques in multi-touch and sort all the interaction methods into three aspects, including two-handed asymmetric and symmetric interaction, unimanual interaction, and dual fingers and multi-fingers interaction.

2.3.1.1 Two-handed Asymmetric, Symmetric and Interchangeable Interaction

Research in two-handed interaction field generally focuses on asymmetric interaction techniques where the hands are used asymmetrically, such as peeling a potato. Representative asymmetric techniques include Bier's Toolglass [48], Shaw's THRED [49], Fitzmaurice's Bricks [50] and so on. The design and evaluation of such techniques are usually guided by Guiard's Kinematic Chain (KC) model [51]. The basic characteristics of the KC model are as follows: non-dominant hand sets the frame of reference in which the dominant hand works; the granularity of action, both spatial and temporal, of non-dominant hand is coarser than that of the dominant hand; the action of the non-dominant hand precedes that of the dominant hand. Lots of everyday activities such as handwriting and sweeping can be characterized as asymmetric according to this model. These all illustrated the importance of asymmetric interaction and far-reaching impact on two-handed interaction techniques in multi-touch.

In contrast to asymmetric interaction, some activities such as folding sheet, skipping rope, where the both hands serve the same work at the same time, can be characterized as two-handed symmetric interaction. Some studies have shown that symmetrical interactive technique has its own potential advantages. Casalta et al. [52] found that symmetric techniques performed well for rectangle creation. Balakrishnan and Hinckley [53] studied a symmetric target-following task and found that objects being manipulated in a symmetric interaction should be visually connected. Latulip et al. [54] evaluated some symmetry interaction manipulations for image layouts and image aligning. While in another study, they proposed a symmetric spline manipulation: symSpline [55]. These two studies both showed that symmetric bimanual technique outperforms asymmetric bimanual technique which in turn outperforms unimanual technique for the image adjustment tasks. These studies all shows that symmetric bimanual techniques have significant advantages for image adjustment tasks. What should be noted is that such techniques requires high visual integration of the operation objects, best for a single operation object or an entire group of operation objects.

In brief, many researchers focus on distinguishing between symmetric and asymmetric interaction in the area of two-handed interaction. None mentions the third situation: two-handed interchangeable interaction where user can change two hands to perform tasks as they like, without worrying about the performance and progress of the work. For example, we can use one hand to turn the key to open the door with the other hand holding the door hander, or conversely. In two-handed interchangeable interaction, user can exchange both hands with only a little change in efficiency. It is a good approach to relieve physical fatigue. Besides, it can increase users' selectable space.

2.3.1.2 Unimanual Interaction and Application

Unimanual interaction generally falls into two categories: one is selection or drag with only one finger, the other is cooperative interaction with two or more fingers. For single target selection, Sears and Shneiderman [56] have showed that direct-touch outperforms the mouse. Forlines et al. [45] found that in a single-target selection task with targets of size 1.92 cm and larger, direct-touch offered modest speed advantages over the mouse, but the error rate was twice than mouse. Kin et al. [57] proved that in the multi-target selection tasks, direct-touch with two fingers (one from each hand) outperformed one finger. For unimanual interaction tasks with two or more fingers, Moscovich et al. [58] showed that such interaction applied to visual rotation task of images. It is especially for the translation and rotation of overall targets, two fingers from one hand outperformed two fingers from both hands. Therefore, they indicated that control of orientations may be performed with one hand. In addition, their study also showed that two hands performed better than one at tasks that require separate control of two points when the controlled points are within the range of motion of one hand's fingers.

2.3.1.3 Dual Fingers and Multi-finger Interaction

Dual fingers interaction can be sorted as interaction with two fingers from one hand and interaction with two fingers from both hands (one finger from each hand), the latter is common for two-handed interaction. Dual fingers interaction refers to the index finger and thumb, which are able to demonstrate main DOF of hands. Moscovich et al. [58] investigated constraints of fingers, fingers and hands, hands and how these constraints impact on the frame of reference and hands' motor coordination. In addition, Moscovich et al. [58] and Malik [59] carried out preliminary studies for the symmetry of the fingers' motor. Although most two-handed asymmetric interaction technologies are based on the KC theory, the theory's applicability to fingers control is uncertain.

Multi-fingered input can increase input bandwidth, but it doesn't mean the more fingers the better. For multi-target selection tasks, Multi-fingers can increase the potential for overlapping, or simultaneous selection of targets, but it can also produce a very high error rate. Besides, fingers in the same hand subject to the hand, which limits the fingers' touch area in multi-target selection tasks. Hancock et al. [60] found using three fingers, including two fingers of dominant hand and one finger of non-dominant hand, offers the best performance for image adjustment tasks. Kin et al. [57] investigated that users prefer to use one or two fingers. Few use more than two and never use more than four fingers to touch the desktop at the same time. These studies suggest that the tracking of two simultaneous contacts may be enough to support multi-target selection.

2.3.2 Hand Input And Gesture Recognition

2.3.2.1 Hand Input Property and Input Form

With the recent emergency of multi-touch techniques, more hand potential and interaction properties gradually are explored except for the coordinates of contact points. Moscovich [58] proposed that hand gestures, finger layout, finger joint angle, and so on, could be considered as available interaction properties. For researchers, it will become a hot topic on how to select and integrate these useful hand input properties to the design of multi-touch interaction techniques in purpose of improving the performance of interaction.

For direct-touch technologies in HCI, configuration and gesture information of the hand and the interaction information represented by hand information are the key to the research of direct-touch interaction and the design of corresponding product. Gesture input includes points, track, hand posture and cooperation gestures input. Points and track are widely used and adapted to user's custom better, the diversity of track input is also conducive to the design of various commands. Cooperation gesture input has some advantages in certain applications, but multi-users cooperation will increase the cost of the input commands and decrease interaction efficiency. The hand posture input is not applicable to the prevalent touch device which only identifies the contact points.

2.3.2.2 Gesture Recognition

Gesture recognization techniques presented in this dissertation only refer to the gesture recognition based on computer vision. Current gesture recognition techniques are divided into the following four categories:

Pattern matching. This is the most simple recognition technique, which matches feature points of inputted gestures with feature points of the standard gestures, and then recognizes inputted gesture by measuring the similarity between the two gestures.

Artificial Neural Network. Artificial Neural Network consists of many simple processing units and copies human neurological system to process information.

Hidden Markov Model (HMM). HMM has been used successfully in continuous speech recognition, handwriting recognition and other areas, which is widely used in the field of dynamic recognition. Using HMM for modeling gesture semanteme can increase the possibility of dealing with gesture-based behavior in a high randomness.

Gesture recognition based on geometric features. This recognition technique makes use of edge and regional features of inputted gestures to recognize gestures.

Gesture dictionary is a new gesture recognition technique first proposed by Elias [62], from which we can query the obtained gesture track and get the corresponding gesture semanteme by matching and analyzing the algorithm in the process of gesture recognition. The use of gesture dictionary can increases the possibility of user-defined semantic library and gestures for gesture-based interaction.

To sum up, many researches have been devoted to bi-manual manipulation and principles of bimanual interaction. These previous researches present a series of important establishments and marked guidelines for the future research. However, only a few studies have evaluated finger's touch performance in direct-touch tasks.

2.4 High Precision Selection Techniques

Difficulties with precise interactions on touch screen devices have been addressed before in a vast number of literature. To improve the feasibility of bare hand-based interactions, there have been a number of methods, which can be divided into four categories as follows.

2.4.1 Direct touch

In 1988, Potter et al. [40] initially discussed the precision of direct touch and presented two direct touch screen strategies: "Land-On" and "First-Contact". In these two strategies, cursor is directly under the finger. Sears and Shneiderman [56] evaluated the precision of direct touch and reported that the accuracy was maximized when targets were 32 pixels per side (13.8 × 17.9 mm) in generic touch technique.

Area cursor technique was initially presented by Kabbash and Buxton [63] in order to solve the difficulties of the elder people. Comparing with the point cursor, area cursor is defined an active area or "hot spot" which is larger than the single pixel of standard cursors. Worden et al. [64] proposed an enhanced area cursor technique that has two hot spots: the square area encompassed by the whole area cursor, and a second single point hot spot within the area cursor. When targets are far apart, the cursor behaves like the default area cursor. However, when more than one target is within the area cursor, the point hot spot is used to discriminate between those targets.

In order to improve the selection performance and simplify the user's operation, many researches have been performed on bimanual interaction in user interfaces. In their respectful pioneering work, Buxton and Myers [46] first showed the basic ability of users to parallelize interaction tasks between hands. And then Kabbash et al. [47] further proved that bi-manual operation could improve performance but must in the condition that the user can coordinate actions of their hands.

2.4.2 Cursor offset

Potter et al. [40] presented "Take-Off" technique to provide a cursor with a fixed offset above the tip of a finger while the user touching the screen. This method is effective for most targets sizes but ineffective when the target size is smaller than 4 pixels.

Benko et al. [36] presented "Dual Finger Offset" and "Dual Finger MidPoint" target selection techniques. Dual finger offset technique provides a user triggered cursor offset. The cursor offset is dynamic and trigger by placing a secondary finger anywhere on the surface. The cursor is subsequently offset with respect to the primary finger by predefined fixed amount.

Vogel and Baudisch creatively implemented Shift technique [43], a technique for single-touch displays that addresses the problems of the Take-Off technique. This technique avoids offsetting the cursor and introduces a small offset callout that displays a copy of the area under the finger with its cursor. The callout is presented automatically when the finger is determined to obscure a sufficiently small potential target, and, in some variants, the small portion of the display in the callout is zoomed for easier selection. In contrast, rubbing and tapping do not rely on the properties of target objects, but can operate on the current position alone (e.g., to zoom the entire scene). Shift technique is suitable for the application in PDAs, however, when the end users touch on a large size multi-touch sensitive device, the find movement and the target selection task in callout of non-preferred hand are quite difficult [51].

2.4.3 Target Zoom-in and control-display rate

For user interface designer, magnifying the interested area or adjusting the control-display rate are two common options while dealing with small targets. Increasing the relative size of screen targets has also been explored by scaling the display space [65] or scaling the motor space [66, 67].

Olwal and Feiner [65] experimented to activate various levels of fish-eye distortion by hand gestures in the interface to facilitate target selection. Blanch et al. [67] and Baudisch et al. [66] developed the techniques that adaptively increase the motor space while leaving the displayed image unchanged respectively. These techniques are capable of showing promising results without introducing screen distortions, but require that the system know all target locations.

Benko et al. [36] created Dual Finger Stretch (DFS) technique and Dual Finger X-Menu (DFX) widget. In DFS technique, zooming and selection are not decoupled into two separate actions. Instead they can happen concurrently which results in a fluid interaction. Second, the interface scales in all directions from the original primary fingers location. Consequently, DFS allows users to adaptively adjust the control-display ratio as well as obtain cursor offset while looking at an un-zoomed user interface. DFX is a circular menu and invoked whenever the secondary finger establishes contact with the surface. It is positioned so that the finger is located at its center. The user can select a particular assistance mode by moving the secondary finger to any of the desired regions of the menu. X-Menu provides 4 kinds of speed of the cursor.

Olwal et al. [68] introduced two families of techniques, rubbing and tapping, that use zooming to make precise interaction on passive touch screens possible. Rub-Pointing uses a diagonal rubbing gesture to integrate pointing and zooming in a single-handed technique. In contrast, Zoom-Tapping is a two-handed technique in which the dominant hand points, while the non-dominant hand taps to zoom, simulating multi-touch functionality on a single-touch display. Rub-Tapping is a hybrid technique that integrates rubbing with the dominant hand to point and zoom, and tapping with the non-dominant hand to confirm selection.

2.4.4 On-screen widget

Bier et al. [48] presented Toolglass and Magic Lenses system, allowed users to control the transparent tool palette with the non-dominant hand, while the primary cursor is under the control of the dominant hand with the mouse. Albinsson and Zhai [1] explored several on-screen widgets for increasing precision while selecting small targets on a touch screen. Their interactions were designed so as to be used with touch screens capable of reporting only a single contact point. Therefore the users were required to execute multiple discrete steps before selecting the target. These steps were delimited by the user lifting their finger from the screen, thus impeding the overall interaction performance. Interestingly, they observed that even though their baseline zooming technique (Zoom Pointing) performed best out of the four techniques compared, its main drawback of losing overview or context can be a significant problem in many applications. In order to enhance the ability of precise touch, a huge number of studies have been carried out. However, all existing studies, in varying degrees, have been limited to low performance, operational difficulty, unnatural gesture and so on.

2.5 User Interface Design in Direct-touch technology

With the advances in hardware functionality and improved features with multi-touch systems, researchers are expending parallel efforts in designing new techniques and leveraging upon novel hardware designs. We concentrate our work on two general categories of interaction techniques: techniques that replicate mouse features on direct-touch surfaces; and techniques that leverage upon additional hand and finger properties.

2.5.1 Adapting Mouse Interactions

There exist several motivations for adapting mouse-based interactions on direct-touch surfaces. It is well known that touch-based interactions result in imprecise selections. Researchers have proposed numerous solutions to improve the precision of bare finger interactions [1, 36, 40, 43, 69] and these solutions can be categorized as follows: direct touch improvement [40], cursor offset [36, 40, 43], target zoom-in or control-display ratio adjustment [65, 67] and on-screen widget [1] to precisely select a target. Furthermore, researchers have explored the benefits of using multi-point input to interact with traditional GUI elements [58, 69, 70].

To ensure compatibility with traditional legacy applications, researchers have studied cursor control and mouse simulation techniques. The DiamondTouch mouse [71] supports a right-click by tapping with a second finger. DTMouse [72] further enhances the functionality of the DiamoundTouch-mouse by addressing issues such as mouse-over, smooth toggling of left mouse button, ergonomics and precise input. In DTMouse, states of the mouse were determined based on timeout intervals of holding a finger down.

Matejka et al. [73] presented SDMouse to emulate the functionality of a conventional mouse, including a tracking state, three buttons and chording. The first finger down is used as a tracking finger, and the combination of two or three fingers are explored to trigger left-click, right-click or scrolling events based on the side and the distance of finger touch points.

However, these systems are based solely on extracting the coordinates of the finger contact points on the screen. Such systems can be unstable or need redefinition if the user triggers these states in a different orientation.

2.5.2 Leveraging Additional Finger Properties

In addition to using the center coordinates of the contact region, researchers have proposed techniques that use finger or hand properties for new interactions. Benko et al. [36] proposed the use of contact size to simulate pressure input on the tabletop. They introduced rocking and pressing gestures to define various states, including a “click” event. Wilson et al. [37] used the contact contours to emulate physical reactions between touch input and digital objects. Cao et al. [22] presented ShapeTouch that leverages the contact shape to enable richer interactions similar to those in the real world. Davidson et al. [74] demonstrated a pressure-based depth sorting technique using a pressure-sensitive surface, which extends standard two-dimensional manipulation techniques, particularly those controlled by multi-touch input.

Finger orientation was firstly adopted by Malik et al. [75] in the Visual Touchpad system. The system uses a pair of overhead cameras to track the entire hand of users, and infers finger orientations accordingly. Microsoft Surface [25] determines full finger orientations by leveraging additional hover information enabled by the DI technology. Both these approaches rely on a specific sensing technology, and are therefore not generally applicable to other systems. These systems also did not investigate interaction designs that specifically utilize finger orientation.

Chapter 3

High Performance Computer Vision Based Multi-touch System

3.1 Motivation

In Chapter 1, many low-cost multi touch techniques (i.e., frustrated total internal reflection (FTIR) [12] and Diffuse Illumination (DI)) were investigated. We follow the instructions of NUIGroup [32] and make a FTIR-based multi-touch experimental device. An open source software, *TouchLib* [32], was installed and tested. However, many limitations of the prototype system seriously restrict our research of direct-touch technique.

For most purposes, we need video images with higher resolution so that we can investigate user's manipulations in depth. *Touchlib* can only support the image with maximum resolution of 640×480 pixels. Besides the resolution, acquisition speed of more than 30 frames per second is another important issue for designing high performance and smoother manipulation multi-touch devices. Current software outputs only coordinate values (x, y) of touch points. No additional touch information can be retrieved.

In order to implement a high performance multi-touch system, we have to re-design a software system with C++ language including all essential functions such as images acquisition, touch area analysis, center coordinate calculation, corresponding touch area tracking and event generator of finger movement and tapping. More important, the software can output finger properties information such as contact area, contact area shape and orientation. This brings more fundamental supports for our further research in the following chapters.

3.2 Crucial Techniques

Image processing is the base of computer vision based touch device. The goal of image processing is to detect and track the finger's movement and the actions of finger's up and down. Fig. 3.1 shows a flowchart of image processing procedure. Firstly, the original image should be converted into a binary image. We adopt a pre-defined fixed threshold value for image binarization processing. Noises in the image are filtered by a smoothing processing. Then the background is subtracted from the original image, only contact area's information remain. After rectification, image contour processing is used to detect the existence of blobs. If at least one blob exists, the information (i.e., size, coordinate value of center point of each blob) can be calculated respectively. Through the analysis of blobs, we track the blob's movement and detect all possible touch events by means of a blob tracking technique.

In order to deeply discuss the algorithm implementation of finger touch area recognition and tracking, we informally define "blob" as a finger contact area in images. Blob means a region of connected pixels which is the same as the fields of image processing. Blob analysis is used to identify these regions from the image. The algorithm discerns pixels by their values and places them into one of two categories: the foreground (typically pixels with a non-zero value) or the background (pixels with zero value).

3.2.1 Blob detection

When a finger touches on the panel of the FTIR multi-touch device, contact area and the non-contact area appear at different gray levels or in different colors on the image captured by a web-camera. Fig. 3.2 shows a image captured from a web-camera in multi-touch device: the upper image is the original one and the image below is the white/black conversion result of the

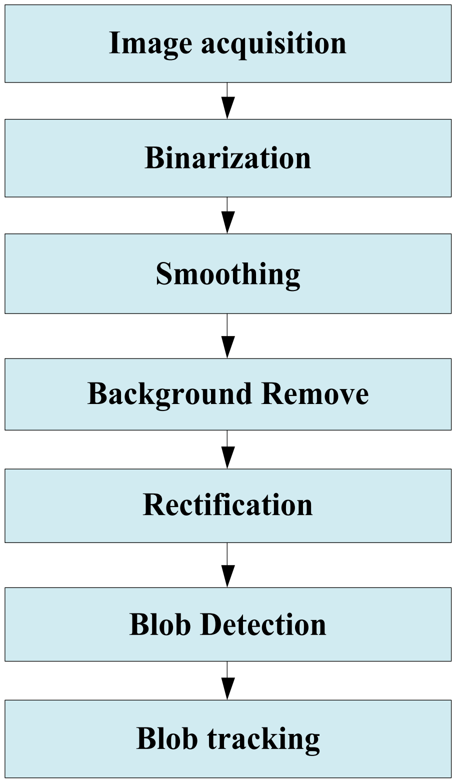

Fig. 3.1 Image Processing Flow Chart.

upper image. The range of pixel's value is between 0 and 255. In our system, the gray value of each pixel in contact area is typically less than 10.

Blob detection is designed to identify the contact area from images. In image processing, blob detection is mainly used to identify pixels with same value from images. And these pixels are separated into different blobs based on relationship of inter-connection. In common, the characteristics of blob such as blob's area, diameter, shape, location and perimeter are typically used to detect the possible blob. Blob detection is a difficult task in image processing indeed. However, blob detection in FTIR multi-touch device is relatively simpler because the background of image is simple and clear. It is easy to remove all background noise. After the image binaryzation, the blob can be clearly identified from the background. A crucial problem of blob detection is that the blob which can be easily identified by the human eye as several distinct is often be interpreted by software as a single blob. The software is difficult to recognize blob with

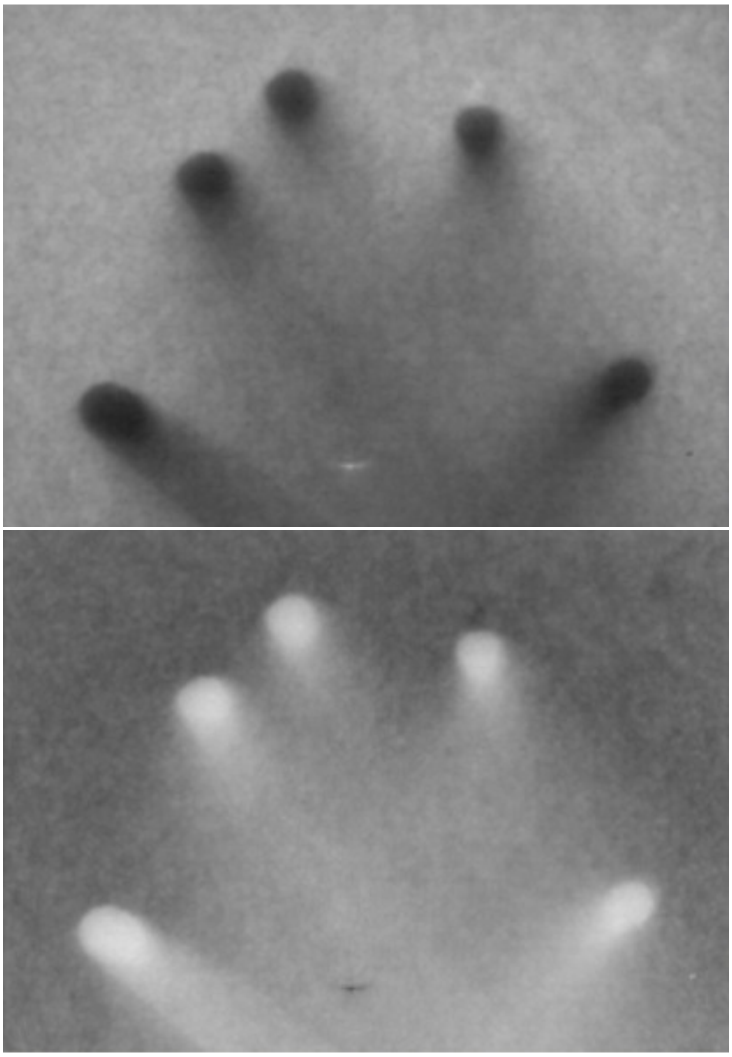

Fig. 3.2 Images captured from a FTIR multi-touch device. Upper one is the original image and image below is a color conversion of the upper image.

weak-signal from images. This will cause the identification mistakes in the multi-touch device.

To enhance the image processing performance, a contour function of OpenCV [76] is invoked to generate the contour map of image. Fig. 3.3 shows the images before and after being contoured. Left image shows an enhanced image and it is easy to find an highlight area on the image. This is a possible blob of the finger. The right one shows the result of image contour processing. The performance of contour function is quite good and the processing time of one 640×480 pixels image is less than 4ms.

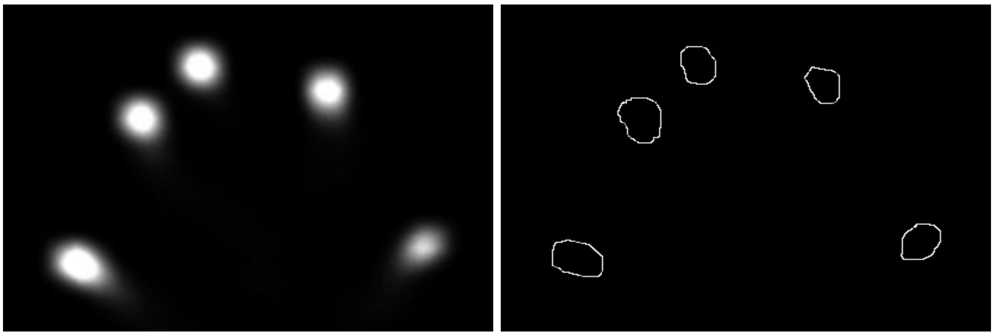

Fig. 3.3 The left image shows the result after image enhancement. The right one is a contour map.

3.2.2 Blob Coordinates Calculation

After blob detection, contact blobs are separated from the image. However, in order to meet the requirement of current WIMP interface, we need to calculate position values (x, y) of each blob so as to provide coordinate values to user interface. A centroid calculation algorithm is used in the study to calculate the coordinate values of center point of the blob. Fig. 3.4 shows the position of centroid. (x_{center}, y_{center}) is the coordinate values of centroid of the object.

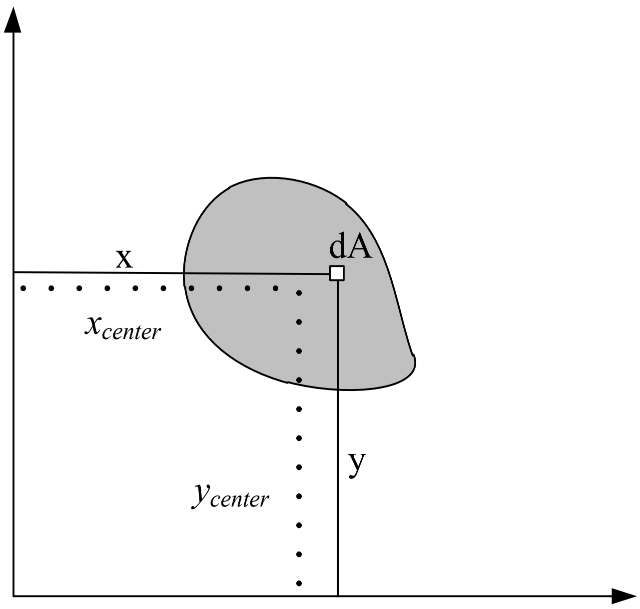

Fig. 3.4 Centroid computing.

Equation3.1 is used for centroid coordinate calculation. x_{center}, y_{center} are the centroid coordinates of the blob.

$$x_{center} = \frac{\int Axda}{A}$$
$$y_{center} = \frac{\int Ayda}{A} \tag{3.1}$$

3.2.3 Corresponding Blob and Blob Tracking

Blob tracking is used to track the finger's movement and the possible action(s). It includes the recognition of the corresponding blobs, the detection of touch events, and the transferring from low-level driver to high level applications. Blob tracking is the most important task in image process of multi-touch device because users' gesture is composed of movements of one or two even more blobs and the touch events (e.g., finger's tapping and lifting up). The design of the gesture is fully based on the results of blob tracking.

Corresponding blob recognition and tracking. Corresponding blob (see Fig. 3.5) means the same finger's blob but in two contextual images. Fig. 3.3 and Fig. 3.5 are two sequential images. In Fig. 3.3, five fingers touches on the panel. Compared two images on Fig. 3.3 and Fig. 3.5, two fingers lifted up from the panel and the other three fingers' positions change. Obviously, three colored blobs are the corresponding blobs of the three original whites (see Fig. 3.5).

To recognize corresponding blobs from many blobs in a series of images, we present a Minimum Distance First algorithm (MDF). We assume there are M blobs in the first image and N blobs in the second image. In the first image, all blobs coordinate values are storied in a two-dimension array variable $(xold_i, yold_i)$, where $1 \leq i \leq M$. In the second image, blobs coordinate values are storied in a two-dimension array variable $(xnew_j, ynew_j)$, where $1 \leq j \leq N$. A dynamic two-dimension array $D(i, j)$ is created to store the distance values of each blob respectively. The distance between two blobs can be calculated through equation3.2.

$$D(i, j) = \sqrt{(xold_i - xnew_j)^2 + (yold_i - ynew_j)^2} \tag{3.2}$$

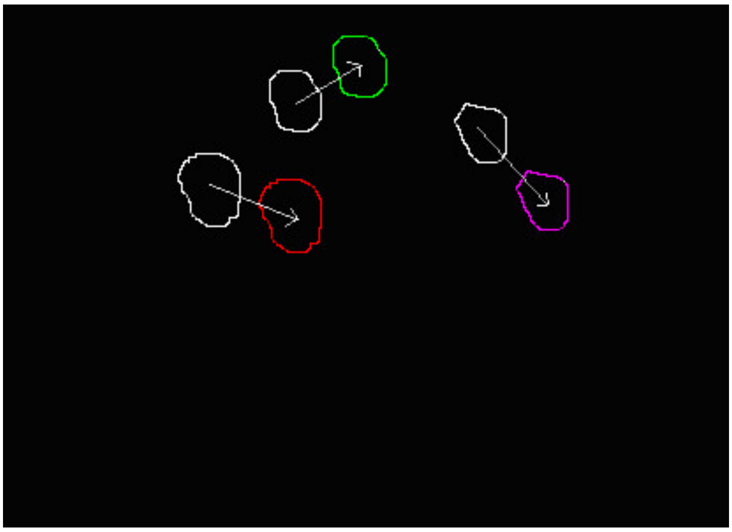

Fig. 3.5 Corresponding blobs.

Base on the results of distance calculation, two blobs which have minimum distance are usually treated as the corresponding blob. And then, a recursive method is used to qualify the validation of the corresponding blobs.

Touch events detection. The multi-touch software system should have the ability to detect touch events, such as the finger's pressing down and lifting up. It is quite simple to detect touch events on the basis of the corresponding blobs available. All the events can be analyzed through the changes of the blob's number. Three conditions are presented in Table 3.1. M is the number of blobs in the first image and N is the number of blobs in the second image. If M equals N, no event happens. If M is less than N, the event of finger's lifting up from the touch screen happens. If M is great than N, the event of more fingers' pressing down happens.

Table 3.1 Trigger Conditions of Touch Event.

No.	Condition	Event	Description
1	$M = N$	none	
2	$M < N$	Finger up	(M-N) finger(s) up
3	$M > N$	Finger down	(M-N) finger(s) down

3.3 Coordinate System and Calibration

3.3.1 Coordinate system

A vision based multi-touch device typically consists of three parts: a projector for display, a camera for action capturing and a touch sensing panel. The displayed area (see red frame of touch panel in Fig. 3.6) must be enclosed within the camera's field of view (FOV). Therefore, all the touch actions can be monitored and processed from the images that are captured from the camera. Obviously, there are two coordinate systems in the vision based multi-touch devices. One is the screen's coordinate system which projected by a projector. Another is the coordinate system of the camera. In common, two coordinate systems cannot coincide because of the installation and debugging error. Meanwhile, the resolutions in two systems are regularly different. The display's resolution is typically 1024×768 pixels. The camera's resolution is typically 640×480 pixels.

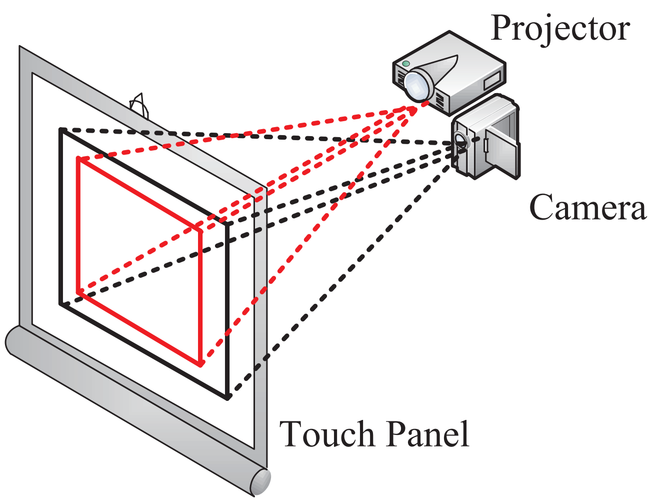

Fig. 3.6 The diagram of two coordinates in multi-touch devices.

It is necessary to map two coordinate systems together in order to transform the coordinate values between two coordinate systems. For example, when a user touches on the panel, the coordinate values of the touch point should be captured by the camera first. And then, the camera's coordinate value must be mapped into display coordinate system. Therefore, we can determine the location on the display screen where the user touches. The touches events, i.e., click and double click, can be further triggered.

3.3.2 Coordinate system mapping

To map coordinate system quickly, refer to the *touchlib* [32], we implement an algorithm on barycentric coordinates.

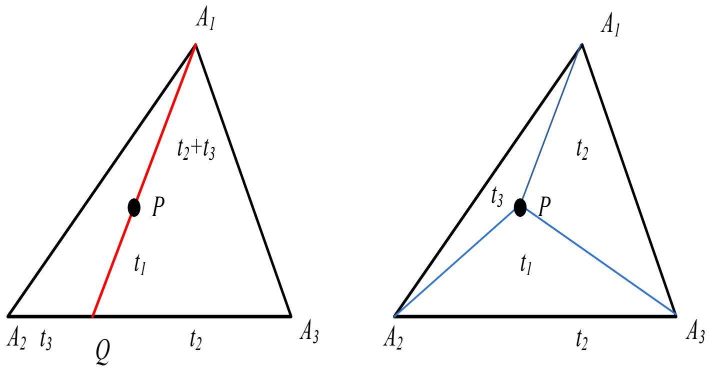

Fig. 3.7 The diagram of barycentric coordinates [2].

Barycentric coordinates [2] are triples of numbers (t_1, t_2, t_3) corresponding to masses placed at the vertices of a reference triangle $\delta A_1A_2A_3$. These masses determine a point P, which is the geometric centroid of the three masses and is identified with coordinates (t_1, t_2, t_3). The vertices of the triangle are given by (1,0,0), (0,1,0), and (0,0,1). To find the barycentric coordinates for an arbitrary point P, find t_2 and t_3 from the point Q at the intersection of the line A_1P with the side A_2A_3. And then determine t1 as the mass at A1 that will balance a mass $t_2 + t_3$ at Q, thus making P the centroid (see left figure of Fig. 3.8). The areas of the triangles ΔA_1A_3P, and ΔA_2A_3P are proportional to the barycentric coordinates t_3, t_2 and t_1 of P (see right figure of Fig. 3.8). Obviously, in whatever any kind of coordinate system, if A_1, A_2, A_3 and P are determined, t_1, t_2 and t_3 are invariable.

Based on barycentric coordinates, it would be easy to map one coordinate value to another

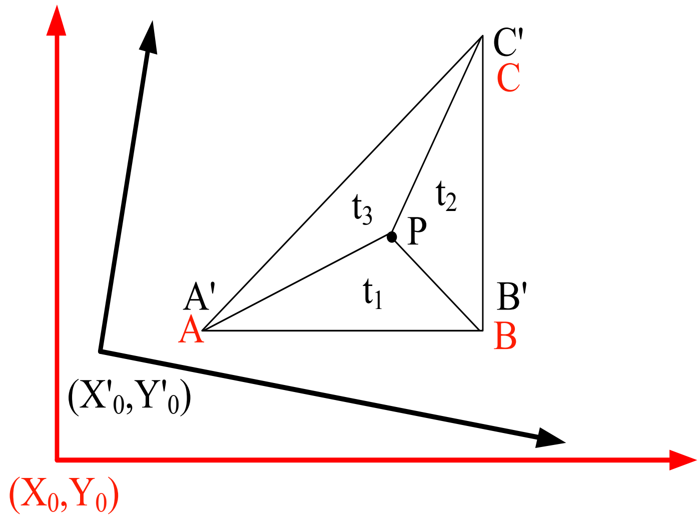

Fig. 3.8 The diagram of coordinate mapping in two coordinate systems.

coordinate. Fig. 3.8 shows a diagram of two coordinates system. Red coordinate system is for display. Black coordinate system is for the camera. Due to the system installation, two coordinate systems cannot coincide. Two issues, the offset of two origins and the rotation of the coordinate system, should be considered in coordinate mapping. ΔABC and $\Delta A'B'C'$ are same triangles but in different coordinate system. At first, calculate the values of t_1, t_2 and t_3 by Equation 3.3.

$$\begin{aligned} t_1 &= \frac{S(PA'B')}{S(A'B'C'} \\ t_2 &= \frac{S(PA'C')}{S(A'B'C')} \\ t_3 &= \frac{S(PB'C')}{S(A'B'C')} \end{aligned} \tag{3.3}$$

And then, the coordinate value of P in display coordinate system can be calculated by Equation 3.4.

$$P = t_1 A + t_2 B + t_3 C \tag{3.4}$$

To construct a relationship between two coordinate system, we display marking points and capture the coordinate values from the camera. Then create a mapping between display and capturing coordinates. With either touchlib [32] or our system, a 4×3 grid is displayed on the screen. Tag 0 is on the top left of the screen and the tag 19 is at the bottom right of the screen. In system calibration, grids are displayed on the screen. Users are asked to touch each of tag respectively from 0 to 19. The coordinate values in two coordinate systems are recorded and processed by equation 3.4.

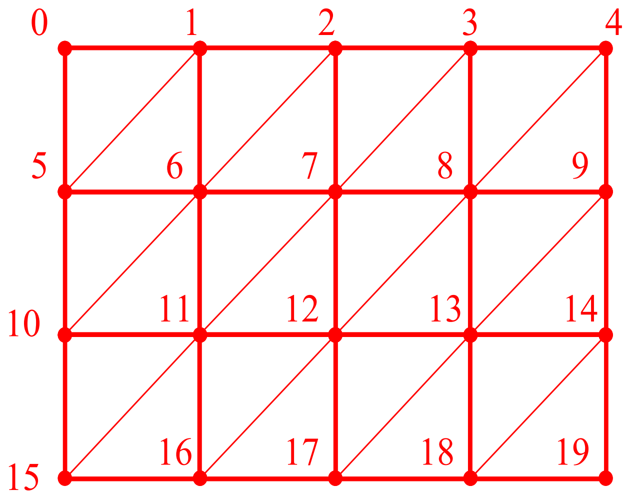

Fig. 3.9 The diagram of calibration grid in multi-touch device.

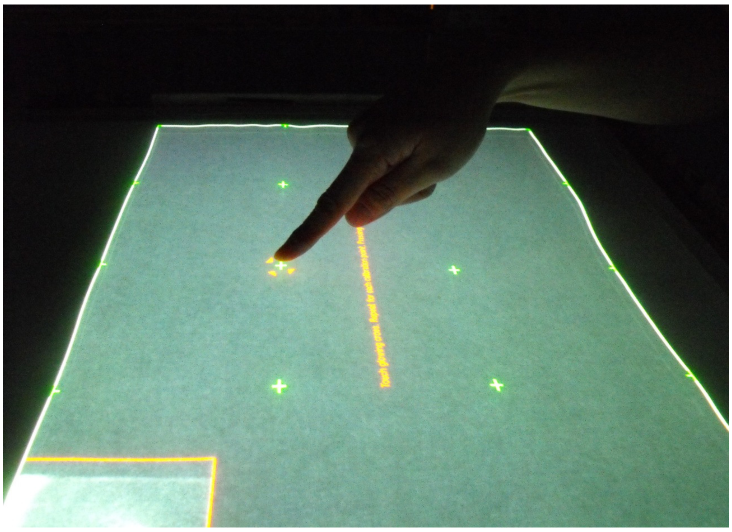

Fig. 3.10 The screenshot of device calibration.

3.3.3 Determining triangle of touch point

There are many display standards on width height ratio such as 4:3, 16:9 and 16:10. In a common 4:3 projector system, the grid should be considered as 5×4 tags, totally 24 triangles. When a user touches on the panel, the software should determine which triangle would include the touch point. Actually, it is not necessary to setup more than 5×4 tags because the performance and the precision in resolution 5×4 is enough for real time multi-touch system.

Barycentric coordinates are a linear transformation of Cartesian coordinates, they vary linearly along the edges and over the area of the triangle. If a point lies in the interior of the triangle, all of the barycentric coordinates lie in the open interval (0,1) [2]. If a point lies on an edge of the triangle, at least one of the area coordinates $t_{1\ldots3}$ is zero, while the rest lie in the closed interval [0,1].

According to Mathworld [2], Point P lies inside the triangle if and only if $0 < t_i < 1 \forall i$ in 1, 2, 3. Otherwise, P lies on the edge or corner of the triangle if $0 \leq t_i \leq 1 \forall i$ in 1, 2, 3. Otherwise, P lies outside the triangle.

3.4 Image splicing for multi-panel splicing

To setup high resolution multi-touch system, the multi-panel splicing technique should be used in vision based multi-touch techniques. Although wide-angle lens (i.e., fish-eye lens) could be used to enlarge FOV, the images with strong distortion in shape would bring us many troubles in image processing especially the part in the border.

Using multi-camera is an easy way in multi-panel splicing. Multiple cameras should symmetrically arrange under the panel. Fig. 3.11 shows the diagram of four cameras splicing. According to our investigation and our experiences, the camera's location and the panel's position should be carefully calibrated. In an optimal situation, the included angle between two coordinate systems should be zero or toward zero degree. Considering the FOV of the camera, in order to ensure the touch at any positions can be effectively monitored, the FOV of each camera should overlap. Especially the FOV in the center of the touch panel, four cameras can monitor that area.

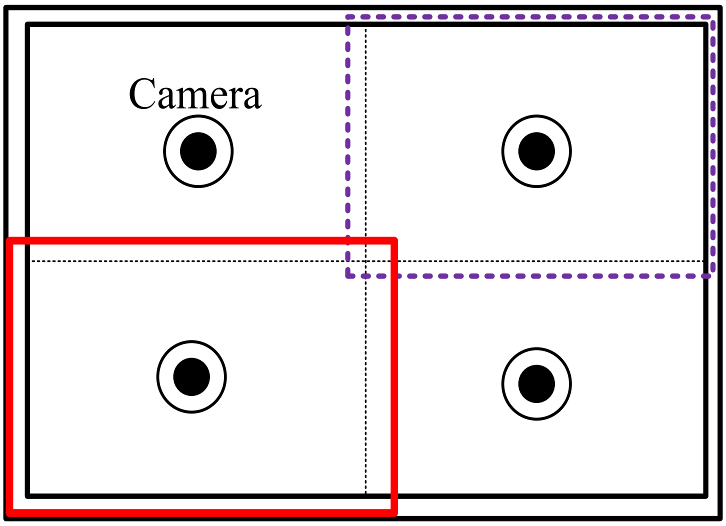

Fig. 3.11 The diagram of 4 cameras splicing multi-touch system. The red area shows FOV of one camera. Obviously, the FOV of the camera should be larger than the FOV of projector's display.

The barycentric coordinates algorithm discussed in the previous section can be effectively used in image splicing of multi-touch technique. In order to meet the need of four cameras' splicing, the calibration grid should be enlarged to 9×7. And the mechanic installation must have enough accuracy to ensure the tags in overlapping area (see the black tags in Fig. 3.12). Total these 15 tags can be detected by at least two cameras.

When a user touches on the panel, the coordinate values of the touch point can be calculated by at least one camera. Three cases can be processed respectively.

(1) Only one camera detects the touch point. Direct invoke the location determination algorithm.

(2) Two cameras detect the touch point. It means the point locates in the overlapping FOV of two cameras. In such case, we have to determine the point belongs to which triangle by

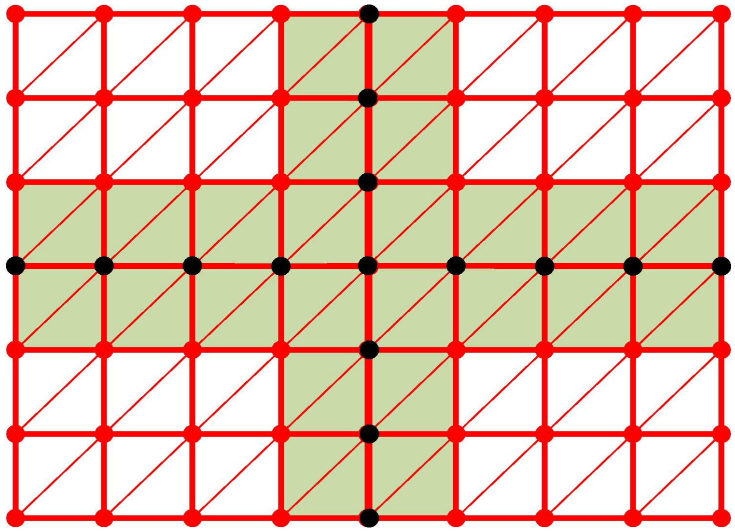

Fig. 3.12 The diagram of calibration grid in splicing multi-touch system. The green zones mean the overlapping zone of two or four camers' FOV.

searching all the possible triangles in overlapping zone and determining the point's ascription. After that, transform coordinate value by the algorithm.

(3) Four cameras detect the touch point. It means the point must be located in the center of touch panel. Search the triangles and determine the point's ascription, transform coordinate value.

3.5 Discussion

3.5.1 Noise and interference reduction

Noise and interference are serious problems during the period of image processing. Poor illumination will cause poor quality and strong noise. The shadow of people's palm also produces the interference. In this study, two methods for binarization processing have been tested to reduce noises and interferences. One uses a fixed threshold value. Another uses an adaptive threshold value: in each image, the average background value and the noise value are computed and we used the value (threshold value = average background value +3 $\times$ noise value) as dynamic threshold value. The result of the adaptive threshold method is not satisfying because the calculation speed is too low to meet the demand of real time system.

3.5.2 Corresponding blobs validation

In this chapter, MDF algorithm is discussed for analyzing the corresponding blobs. But MDF is not robust in some special occasions. For example, if two blobs are close enough to each other, recognition error may occur. In our system, we use multi parameters such as finger's contact area and the finger's orientation to improve the recognition accuracy of corresponding blobs. But if the number of blobs is more than 10 and more parameters are used, the performance of the system will be a critical problem.

3.5.3 Evaluation and limitation of the splicing algorithm

Two methods can set up large screen image splicing. In traditional image processing, image combination technique can be used to combine four respective cameras together and generate one full image to recognize and track touch blobs. The advantage of this method is that one only need to modify the codes of image capturing. However, the combination of images is a high CPU-load task. In order to ensure the "smoothness" of the user's manipulation, the image capturing rate should at least be 30 frames in one second. The image combination cannot meet the requirements indeed.

Using barycentric coordinates algorithm, the performance of the algorithm is satisfactory. The algorithm was evaluated in a desktop PC (Intel Core2 2.2Ghz CPU, 4G memory), the time consumed in coordinate transformation of one touch point is less than 5 ms. Even if in the situation of multi cameras splicing, the performance of the algorithm does not decrease distinctly. For example, if a touch point is in the FOV of left-top and left-down cameras, only 16 triangles should be checked (see the green zone in Fig. 3.12).

3.5 Discussion

The splicing technique in this chapter has some limitations. One of the limitation is the splicing algorithm seriously depends on the installation accuracy. The tags in the overlapping zone must be detected by 2 or 4 cameras. The projector and the camera(s) are the key components of the multi-touch device. In order to ensure the tags' position, the parallelism between the projector and the camera is critically important. However, the parallelism is difficult to adjust manually.

Chapter 4

Empirical Evaluation for Finger Input Properties In Multi-touch Interaction

4.1 Instruction

As mentioned in previous chapters, there are a few distinct drawbacks which limit the application of multi-touch technology. Albinsson and Zhai [1] reported that the occlusion of screen data caused by the fingers and the hand, very low selection precision and arm fatigue were significant limitations of touch devices. These drawbacks need to be overcome by new multi-touch techniques.

Based on the survey of Chapter 2, we concluded that current multi-touch techniques do not fully exploit the characteristics of the human hand or fingers. The finger's function in currently available multi-touch interfaces is merely to position the cursor and to click events. But, in fact, the human hand is a complex mechanism. A total of 23 degrees of freedom (DOF) have been identified through medical and anatomical analysis [10]. When one finger contacts the touch panel, the multi-touch device can only output the contact area coordinates (x, y) of the flat (2-D) surface. Due to the limitations of fixed 2-D touch screen surfaces, the actual number of DOF is reduced to ten in the condition where five fingers are used simultaneously. This decrease in the DOF number seriously reduces the number of gestures available to interface designers.

Moreover, arm fatigue is caused by two factors: one is continual tapping; the other is long-distance movement of the hand across the touch screen. Users are required to tap and move their hands on a screen to generate relevant events such as movement and click events in order to guarantee compatibility with traditional graphical user interfaces (GUIs). Additionally, current multi-touch techniques only adopt variations in the points of contacts to generate recognizable events. For example, the movement of two fingers in opposing directions triggers the "Zoom" function in some common applications.

4.2 Motivation

To enhance the usefulness of interfaces incorporating multi-touch techniques and to overcome the drawbacks mentioned above, obviously, the expansion of input bandwidth is an effective way. We explore a wide range of finger input properties that are capable of controlling targets, i.e., it is likely that we can control computers with more natural and more comfortable gestures. This kind of study will, in turn, need to be guided by a thorough understanding of finger input properties and abilities as they relate to touch. Fig. 4.1 shows the properties of the fingers in multi-touch techniques: contact area, shape, orientation. Considering the initial discussion of Forline et al. [45], we can further investigate these properties, which may help to overcome current drawbacks and improve the interactive ability of end users, in vertical touch and oblique touch gestures.

4.3 Experiment Design

4.3.1 Goals

The objective of this study is to investigate the potential of human finger input properties. This includes determining the real precision of target tapping in vertical touch and oblique touch gestures, variations in the finger's contact position when tapping, variations in the center

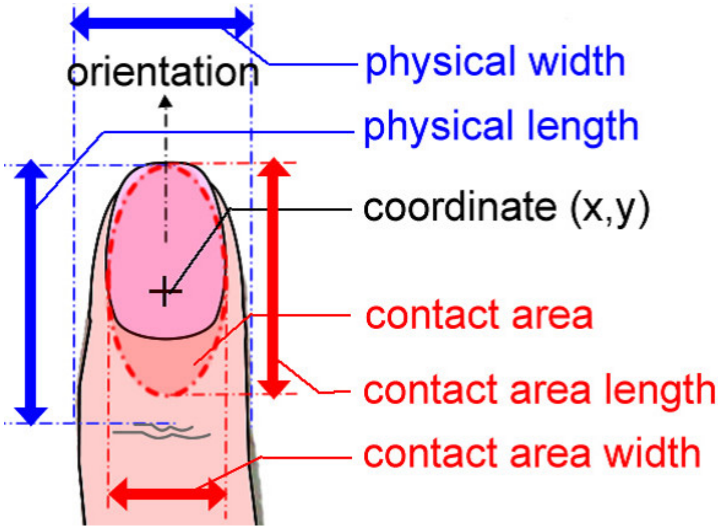

Fig. 4.1 Available finger input properties that may be adopted by multi-touch designers.

point of the finger's contact area in different gestures and variations in finger orientation. An evaluation of the utility of these properties will determine whether they can be integrated into the design of natural gestures in multi-touch techniques.

4.3.2 Apparatus

In order to observe all possible finger input properties on a two dimensional flat panel, we adopted the technology of Han [12] and made a Frustrated Total Internal Reflection (FTIR) based on a multi-touch widget. The touch panel was a transparent acrylic panel, which internally reflects the IR-light. Infrared LEDs were installed along the edge of the acrylic, and infrared light was introduced edge-wise into a platen waveguide.

A standard A4 (210 mm $\times$ 297 mm) sheet of white paper which was printed as an operational interface (see Fig. 4.2) by color laser jet was firmly pasted on the surface of the touch panel and one camera was fixed at the base of the device, vertically beneath the center of the panel, to detect the finger contact area. The camera was a Philips SPC900NC with VGA CCD Sensor and USB 2.0 interface. The default lens of the camera was replaced with a 4.3 mm CCTV Camera board Lens which did not have an IR-block filter. When a finger makes contact

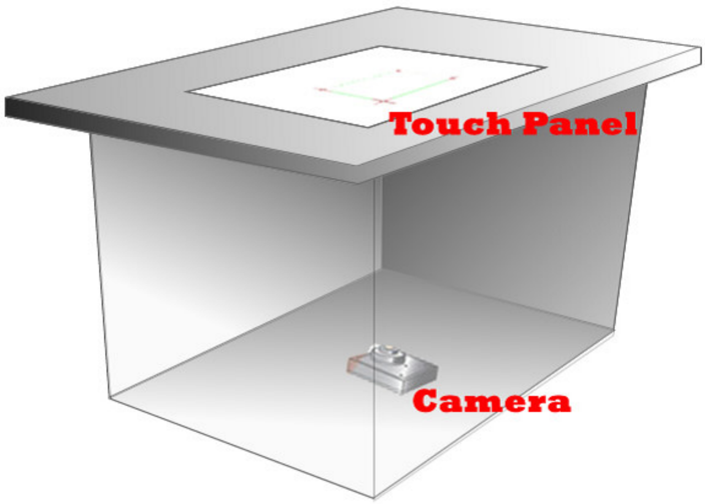

Fig. 4.2 FTIR based multi-touch prototype.

with the touch panel, infrared light escapes from the acrylic so that the camera can detect the finger contact action through variations in the infrared light. The camera was operated at a resolution of 640 × 480 pixels. As an important parameter of the apparatus, the scale of the camera was carefully measured, and the scale of the system in both the x and y axes was 0.4 mm/pixel.

The software was modified and redesigned based on TouchLib [32], an open source multi-touch package. The experimental software was run on a 2.4GHz Core 2 PC with the Windows 2003 Server operating system.

To obtain as much information from each finger touch as possible, we optimized all possible program codes to improve the processing performance. In the current experimental prototype, 30 frames can be processed in one second. That means we can collect 30 pairs of coordinate values in one second to meet the requirements of data analysis. No visual feedback was given to subjects but they could hear a beep as audio feedback when contact was made with the panel.

4.3.3 Tasks

The experimental task consisted of two parts. One was target tapping and the other was finger rocking and pointing. The multi-touch prototype was placed on the floor. The subject sat in front of the touch panel.

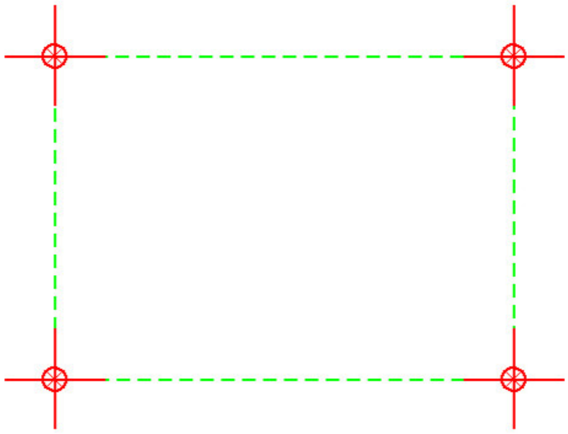

Fig. 4.3 Experimental user interface. The center points of each red circle are the touch targets. The diameter of each circle is 4 mm. The width of the rectangle is 100 mm and the length is 60 mm (cross hair center to cross hair center).

Target Tapping. A tapping task was used. Tapping is a primary natural finger gesture used in most current multi-touch interfaces. We designed a simple tapping task to investigate the precise tapping ability of the five fingers of the dominant hand. The difference from other studies is that the size of the target was fixed. We referred to the experiment design proposed by Sears and Shneiderman [56]. Fig. 4.3 shows the experimental interface and the locations of the four targets. This task includes two sub-tasks: one is vertical touch and the other is oblique touch (see Fig. 4.4).

Finger rocking and orientation. A rocking task was used to investigate the variations in the finger contact area for both gestures and the rotatable range of finger orientation. At first, the subject used his or her finger to vertically touch the widget's panel and then he/she tilted the

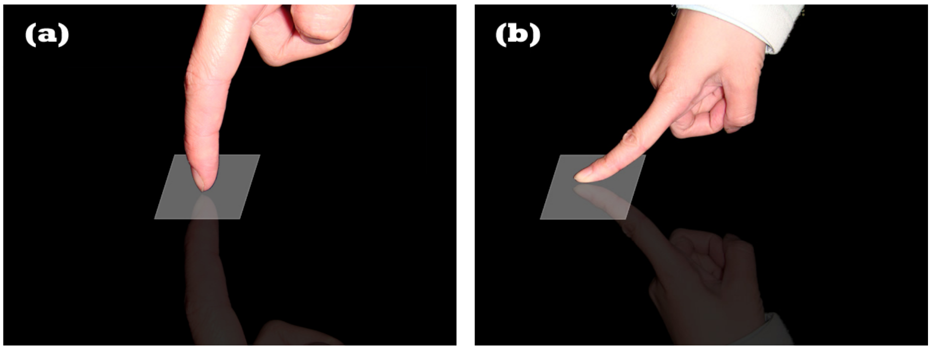

Fig. 4.4 (a) is defined as “vertical touch”. (b) shows the gesture of “oblique touch”.

finger down. And second, when the finger was in an oblique state, the user horizontally rotated the finger clockwise and counter clockwise to change the finger’s orientation to the maximum on the premise of maintaining user comfort (see Fig. 4.5).

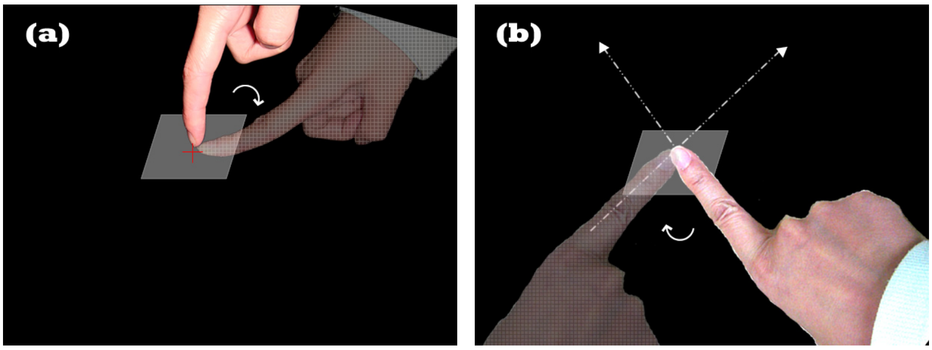

Fig. 4.5 (a) is defined as finger rocking and (b) is defined as finger orientation rotation.

4.3.4 Participants

Eight male and four female volunteers, 26-37 years old, participated in the experiment. All were right-handed and had a little experience using touch devices such as ATMs. The physical sizes of each subject’s finger-tips (end joints) were recorded. The average values of physical width (W) and physical length (L) (see Fig. 4.1) are listed in Table 4.1 in millimeters.

Table 4.1 The physical size of the five finger-tips. W = width, L = length, AVG = average value, SD = standard deviation (unit: mm).

	Thumb		Index		Middle		Ring		Little	
Finger tip	W	L	W	L	W	L	W	L	W	L
AVG	20.1	30.3	16.0	24.8	16.6	25.8	15.0	25.4	13.7	22.6
SD	2.7	3.2	1.9	1.7	1.9	2.4	1.9	2.5	1.7	2.5

4.3.5 Procedure and Design

The participants were instructed to tap six times on each of four targets using two finger gestures with their five fingers in turn. The participants then landed each finger on the touch panel, rocked it, rotated the orientation and lifted it off the touch screen. In summary, the experiment consisted of:

12 participants ×

5 fingers ×

2 tasks ×

2 sub tasks ×

6 repetitions

= 1440 trials.

Prior to performing trials for each task, participants were given a short set of warm-up trials to familiarize themselves with the touch manner. Participants were instructed to perform the tasks as quickly and accurately as possible. The experiment lasted approximately 20 minutes for each participant. A short questionnaire was administered at the end of the experiment to gather subjective opinions. For each trial, we collected all the finger touch data (position, shape, size of contact area, width of contact area, length of contact area). An audible beep provided feedback when each trial was successfully completed.

4.4 Results

4.4.1 Touch Area Center Point Variation

As a flexible motor system, distortion of the finger is inevitable while it is touching the hard screen panel. This distortion is disadvantageous for gathering contact area position measurements. We adopted the traditional centroid algorithm [77] to calculate the coordinate value of the contact area because the precision of such an algorithm is capable of attaining sub pixel level accuracy and is enough to guarantee the results of our analysis.

Snapshots of the full finger tapping procedure, from initial landing on the screen to lifting from the screen, were captured (see Fig. 4.6). The blue cross represents the center of the contact area. It is easy to see that the center point of each image varies during the tapping procedure.

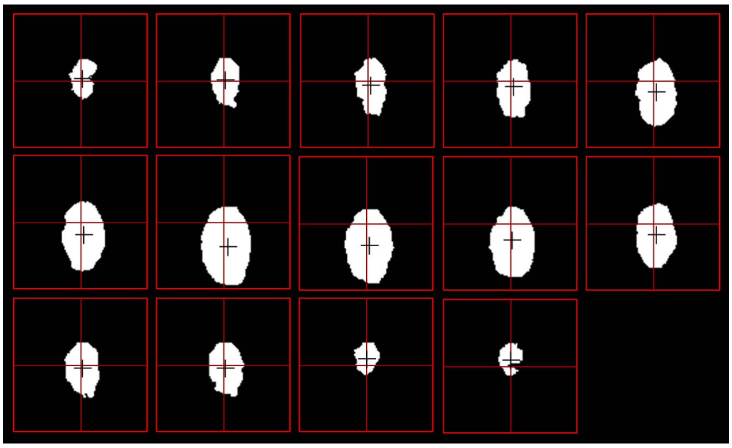

Fig. 4.6 Snapshots showing the finger's contact with the screen over the full procedure.

In order to deeply study the finger touch procedure and to obtain more precise position coordinates, we divided the tapping procedure into three states: Land On, Stable and Lift Up, in each of the touch gestures.

Land On refers to the state in which the user first contacts the screen and the moment when the multi-touch device detects the finger's initial contact. Stable refers to the state during which the finger is stably in contact with the screen surface. Lift Up is the final state where the finger lifts up from the touch screen. We used the first recorded data of each touch as the Land On coordinate value and the last data of the procedure as the Lift Up coordinate value. We adopted the data showing the contact area with maximum size as the center point of the Stable state.

As mentioned above, four targets are located on the paper and the accurate coordinates of targets are determined in advance. The distance from the center point of the Land On state to the center point of the Target (Land On-Target), the distance from the center point of the Stable state to the center point of the Target (Stable-Target) and the distance from the center point of the Lift Up state to the center point of the Target (Lift Up-Target) were calculated by a two-point formula (see Table 4.2).

Table 4.2 The average distances of the three sets of data (Land On-Target, Stable-Target, Lift Up-Target) of five fingers in two gestures, SD = standard deviation (unit: pixels, scale = 0.4 mm/pixel).

	Thumb	**Index**	**Middle**	**Ring**	**Little**
Veritical touch					
Land On-Target	6.03	6.44	6.75	6.72	7.15
(SD)	(2.95)	(3.00)	(3.32)	(3.68)	(3.79)
Stable-Target	5.56	5.83	6.37	6.38	6.67
(SD)	(2.69)	(3.00)	(3.19)	(3.21)	(3.31)
Lift Up-Target	5.70	5.56	6.48	6.35	6.61
(SD)	(2.72)	(2.67)	(3.28)	(3.32)	(3.34)
Oblique touch					
Land On-Target	8.14	6.29	7.02	6.71	7.36
(SD)	(4.03)	(2.82)	(3.49)	(3.30)	(4.22)
Stable-Target	7.68	5.85	6.08	6.08	6.99
(SD)	(3.76)	(3.09)	(3.13)	(3.15)	(4.12)
Lift Up-Target	8.11	5.87	6.58	6.45	6.99
(SD)	(4.04)	(2.80)	(3.34)	(3.17)	(4.13)

Table 4.2 shows that the values of Land On-Target are larger than the values of Stable-Target and Lift Up-Target. The values of Stable-Target and Lift Up-Target are closer. There are significant differences between Land On and Stable ($F_{1,8}$ = 9.56, $p < .05$), and between Land On and Lift Up ($F_{1,8}$ = 5.75, $p < .05$). However, no significant difference was found between Stable and Lift Up.

This result suggests that in multi-touch widget design, the coordinates of the Land On state are not accurate enough to be adopted as the cursor position. A comparison of the values of Stable-Target and Lift Up-Target for the five fingers reveals that the center point of the Stable state represents the cursor position more accurately. We adopt the Stable state as the default state in the subsequent sections if without additional comment.

In order to investigate variations in coordinate values in the tapping procedure, the average distance deviation for Land On-Stable (from the center point of the Land On state to the center point of the Stable state) and Stable-Lift Up (from the center point of the Stable state to the center point of the Lift Up state) in two touch gestures were calculated and listed in Table 4.3. The maximum average deviation in distance for Land On-Stable is 3.06 pixels (1.22 mm) and for Stable-Lift Up is 3.46 pixels (1.37 mm). Due to the natural increase in the contact area, the distance deviation in the oblique touch gesture is greater than in the vertical touch gesture.

4.4.2 Tapping Precison

Though the finger which is relatively stubby cannot obtain the same selection precision as a stylus, the finger's fundamental target selection ability is worthy of study. In multi-touch techniques, all the fingers of the two hands have the potential to work together to affect events more efficiently.

From the basic analysis of the data, the results for the five fingers show the same trends. Due to the limitation of space, we only present the scatter diagrams and distribution diagrams for the index finger (see Fig. 4.7). Fig. 4.7a and Fig. 4.7b show that the center points of the

Table 4.3 The average distances of the two sets of data (Land On-Stable, Stable-Lift Up) of five fingers in two gestures, SD = standard deviation (unit: pixels, scale = 0.4 mm/pixel).

	Thumb	**Index**	**Middle**	**Ring**	**Little**
Veritical touch					
Land On-Stable (SD)	1.99 (2.08)	1.99 (2.17)	1.44 (1.30)	1.36 (1.36)	1.54 (1.78)
Stable-Lift Up (SD)	1.70 (1.63)	1.71 (2.40)	1.05 (0.99)	1.14 (0.97)	1.04 (1.17)
Oblique touch					
Land On-Stable (SD)	2.87 (3.05)	3.06 (3.08)	2.79 (2.65)	2.25 (2.20)	2.74 (2.57)
Stable-Lift Up (SD)	3.20 (3.23)	3.46 (2.96)	2.46 (2.34)	1.93 (1.88)	2.20 (2.24)

touch area are evenly distributed around the coordinates of the target within a definable range. The histograms in Fig. 4.7c and Fig. 4.7d show that the distribution of data is approximately in accord with the normal distribution in vertical and oblique touch gestures. In such premises, upper level of 95% confidence interval can be considered to be the effective target size.

Table 4.4 lists all the tapping deviation data for the five fingers (thumb, index, middle, ring and little) with two touch gestures. All the data of the vertical and the oblique touch events are calculated respectively. Average Values (AVG), Standard Deviation (SD), Lower Level of 95% Confidence Interval (LLCI) and Upper Level of 95% Confidence Interval (ULCI) are listed in the Table 4.4.

Regarding touch precision, there is no significant difference between the vertical touch gesture and the oblique touch gesture, when using the index finger, middle finger, ring finger, or little finger. However, there is a significant difference when using the thumb ($F_{1,22}$=12.5, $p < 0.05$).

Two results can be analyzed from Table 4.4, Fig. 4.7 and relevant ANOVA results.

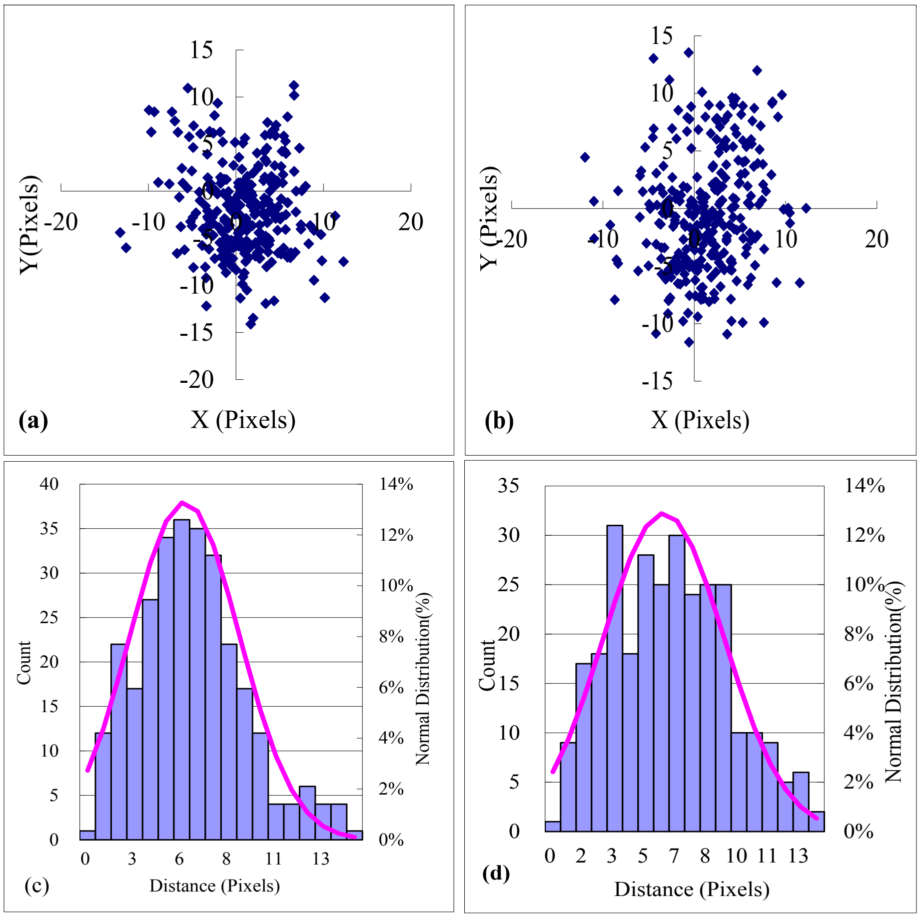

Fig. 4.7 Scatter diagrams and normal distributions diagrams of the index finger in the vertical touch (a)(c) and oblique touch gestures (b)(d). The origin of the coordinate system (zero) in (a)(b) represents the position of the target. The blue point is the position of each tap. The distance value in (c)(d) is the value of Stable-Target.

First, the precision for target selection of the index finger, the middle finger and the ring finger is relatively better than the precision of the thumb and the little finger. The average value and upper level of 95% confidence interval of "All Data" in Table 4.4 show that the index, middle and ring fingers are more accurate than the thumb and little fingers. In the subjective investigation, 12 subjects all reported that the little finger was difficult to use in the tasks. This is consistent with the experimental data.

Second, the radius of circular targets needs to be greater than 14.38 pixels (5.76 mm) and square targets need to be at least 28.76 pixels (11.52 mm) per side to maintain direct touch

Table 4.4 The tapping data for the five fingers in three conditions: the average values (AVG), standard deviation (SD), lower level of 95% confidence interval (LLCI) and upper level of 95% confidence interval (ULCI) (unit: pixels, scale = 0.4 mm/pixel).

		Thumb	**Index**	**Middle**	**Ring**	**Little**
All Data	**AVG**	6.63	5.84	6.22	6.23	7.57
	SD	3.44	3.05	3.16	3.18	4.14
	LLCI	0.97	0.83	1.02	0.99	0.75
	ULCI	12.28	10.85	11.43	11.46	14.38
Vertical Touch Data	**AVG**	5.56	5.83	6.37	6.38	6.67
	SD	2.69	3.00	3.19	3.21	3.31
	LLCI	1.14	0.89	1.12	1.09	1.23
	ULCI	9.99	10.76	11.62	11.66	12.11
Oblique Touch Data	**AVG**	7.68	5.85	6.08	6.07	6.99
	SD	3.76	3.09	3.13	3.15	4.12
	LLCI	1.49	0.76	0.92	0.89	0.21
	ULCI	13.87	10.94	11.23	11.26	13.76

precision. According to the statistical theory, each upper level of 95% confidence interval value of five fingers in Table 4.4 can be regarded as the effective target size of each finger under the prerequisite that the distance data is in accord with the normal distribution. This effective target size also indicate finger touch accuracy. In order to meet the requirements of five fingers with two gestures, the minimum optimal size of targets is determined from the value of maximum upper level of 95% confidence interval (little finger of “All Data”) in Table 4.4. It is obvious from the data (see Table 4.4) and the scatter diagrams (see Fig. 4.7) that the value is the optimal radius of the target. If we consider the square target, the size must be 14.38 $\times$ 2 (28.76 pixels, 11.52 mm) per side to guarantee a 95% confidence level.

4.4.3 Finger Touch Area Shape, Size and Orientation

Shape. Fig. 4.8 shows the shape of the finger contact area. The shape of the contact area can be approximately represented by the equation of a rectangle or an ellipse. Three parameters, i.e., width (minor axis), length (long axis), slant angle, can describe one touch area of a finger. Table 4.5 presents the average statistical width and length for the two touch gestures. The real size of the contact area is calculated directly from the finger imprint (see Fig. 4.8) and is different from the value of width $\times$ length.

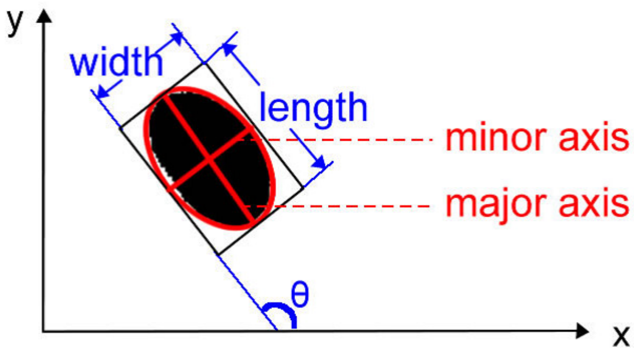

Fig. 4.8 Shape of the contact area of the finger. The area with the black color shows the finger imprint.

Based on the data of Table 4.5 regarding the physical size of the finger, the width and length of the contact area in vertical touch is approximately 30% - 40% of the physical width and length of the full finger-pad (end-joint). In the oblique touch state, the average width is approximately 90% of the physical finger tip size, while the average length is approximately 70% - 80% of the physical finger tip.

Orientation and area size. The human finger has the ability to indicate direction in common life. In multi-touch techniques, the finger has the same ability to indicate direction on a 2D touch panel. When the finger touches the panel in the oblique gesture, the finger's pointing direction can be defined as "finger orientation".

The average size of the contact area in the vertical touch state (VA), the average contact

Table 4.5 The average width and length of the contact area for the two touch gestures. Physical width and length of the finger-tip (see Table 4.1 but with different unit) (unit:pixels, scale = 0.4 mm/pixel).

	Thumb	**Index**	**Middle**	**Ring**	**Little**
Physical Width (PW)	50.0	40.0	40.0	37.5	35.0
Physical Length (PL)	75.0	62.5	65.0	62.5	57.5
Veritcal touch					
Width	14.9	13.8	14.6	15.3	13.8
Length	18.0	16.7	18.1	18.5	17.9
Width/PW (%)	30	35	37	41	39
Length/PL (%)	24	27	28	30	31
Oblique touch					
Width	42.9	36.0	36.9	34.3	31.2
Length	58.3	50.9	47.1	47.3	44.5
Width/PW (%)	86	90	92	91	89
Length/PL (%)	78	81	72	76	77

area in the oblique touch state (OA), the proportional relation between VA and OA and the maximum orientation rotatable range are listed in Table 4.6.

Table 4.6 presents the finger's ability to control according to the finger's orientation property. The orientation of the finger can comfortably vary by more than 100 degrees. All subjects reported that it is easy to perform such actions.

In addition, the size of touch area has significant difference between two gestures. The area of oblique touch is at least 5.5 times the area of vertical touch.

Table 4.6 The average size of the contact area for the two touch gestures. Range of orientation represents the horizontal rotation ability (unit: VA and OA = $pixels^2$, Range of orientation = degrees).

	Thumb	Index	Middle	Ring	Little
Vertical Touch Area (VA)	194.0	178.1	196.3	209.5	179.7
Oblique Touch Area(OA)	1831.2	1375.3	1301.2	1158.0	1031.6
OA/VA	9.4	7.7	6.6	5.5	5.7
Range of Orientation	106.3	127.3	128.9	132.1	130.6

4.5 Discussion

4.5.1 Device Stability and Measurement Precision

This study is based on our own prototype FTIR multi-touch widget. The stability of the device is obviously a crucial factor in the study. In our experiment, we have verified the target location 6 times. The variation of the target coordinates, which is less than 0.5 pixels (0.2 mm), fully proves the stability of the device.

Another issue that could possibly affect the precision of the set-up is the indirect display of the interaction interface on the touch screen. The experimental interface is printed on plain paper instead of being displayed by a projector. Whether the current device can meet the requirements of precise measurement is the most important issue in the current study. In order to guarantee measurement precision, the device was carefully calibrated before the formal experiment. The camera was fixed vertically beneath the center of the touch panel in order to minimize distortion of the image. We checked the image scales of the four corners and found that the difference in these scales is less than 0.02 pixels (0.008 mm). In the environment where a projector is used, the camera cannot be fixed beneath the panel because the influence from the strong lights of the projector would significantly distort the test environment. Furthermore, images captured from a camera in a tilted position would lead to barrel distortion and significantly affect the accuracy of the results.

The results of this study suggest that the optimal minimum radius for a circular target is 5.76 mm. This result is significantly different from that of the study by Hall et al. [42], who reported that accuracy varied from 66.7% for targets of 10 mm per side, to 99.2% for targets of 26 mm per side, and that accuracy was maximized once targets were approximately 26 mm per side. But the result of the study of Sears and Shneiderman [56] is very close to the current result in our study. In their study, they reported that the accuracy was maximized when targets were 32 pixels per side (13.8 $\times$ 17.9 mm). We analyzed the differences among the three studies and concluded that differences in the devices are the main cause of the different results. Variation in measurement resolution is also a possible cause of such significant differences.

4.5.2 The Physical Size of Fingers and Relevant Questions

Each person's fingers are different in size. Whether this difference will change the result of the current study is a key question. We investigated the physical size of the fingers of the participants. Based on this investigation, we noticed that there is a relationship of scale between the physical width and physical length of the end joint of the human finger. The physical length of the human finger-tip is about 1.5 times its width. Even when pressing on the touch screen with strong pressure, the finger width is only 10% larger than the original width, i.e., the width without strong pressure. This degree of distortion does not significantly affect the measurement of the width, length or area.

The second question needing to be discussed is whether the current study's results are only relevant to adults but not to children. We therefore measured the fingers of 6 children. The result shows that the width and length of the fingers produce the same results by scale.

As mentioned in the previous sections, the resolution of the video image is 640 $\times$ 480 pixels. The physical properties of five fingers are precisely evaluated in the study. Of course, the measured values in different devices are possibly different because of the different physical characteristics of each device, e.g., resolution and sensitivity. Though most current commercial multi-touch products cannot support this high resolution, we believe that the results in our study, especially the proportional relations between each property, always exist and will be useful for all kinds of multi-touch devices.

4.5.3 The Contact Area and Relevant Questions

The contact area of a finger can be used in computer control as event trigger. Forlines et al. [45] discussed the finger contact area and its effect in target selection. Benko et al. [36] adopted the finger contact area to trigger “click” events. We further explored the significance of variations in the finger contact area for both gestures. Table 4.6 clearly shows the result of contact area variations for both gestures. The size of the contact area can be used to trigger an event because of three factors: (1) The area in the oblique touch state is at least 5.5 times larger than the area in the vertical touch state; (2) The results of 6 repetitions in the task show that the values of area size in vertical touch and oblique touch gestures (see Table 4.6) are stable; (3) All subjects reported that it was easy to perform such actions.

4.5.4 The Determination of Stable State

Table 4.2 shows that the center point of the Stable state may represent the cursor position more accurately. The first touch coordinate cannot be treated as the final touch position. In the real system, the method for determining the Stable state is a problem that requires more consideration. Based on the current investigation, the estimate of the Stable state coordinates is the simplest and most accurate way to determine the coordinates of the finger’s contact area. When the width of the contact area is greater than a predetermined threshold and the length is greater than the width, the state can be considered to be the Stable state. Of course, the empirical value of the threshold should be tested under experimental conditions.

In low-resolution multi-touch sensitive devices, variations of coordinates in different states are not a critical problem. However, in high-resolution devices, such variations must be considered in the system design. For example, in traditional touch techniques where the finger's lift-up action triggers a "click" event, any deviation in the coordinates will possibly cause a wrong target selection. With the advent of new high-resolution multi-touch sensitive panels, further consideration of the Stable state are necessary.

4.5.5 The Limitations of Using Finger Properties

We investigated all the available finger input properties. However, there are a few limitations when we try to use these properties simultaneously because of anatomical limitations. For example, the contact area for one finger cannot be used as an event trigger when multiple fingers of one hand are simultaneously involved in the interaction. If one finger is rocked from the vertical state to the oblique state to change the contact area, the area of the other fingers will also change. Similar limitations exist in the orientation property. When the user rotates a finger horizontally to change the finger's orientation, the orientation of all the fingers will change. However, these limitations do not influence the adoption of these properties in multi-touch techniques because tasks can be deployed to different hands. For example, the preferred hand can be used for target selection and the non-preferred hand can be used to trigger menu and select menu item.

4.6 Implications for Design

4.6.1 Guidelines

The results of our experiment suggest several guidelines for the design of multi-touch widgets:

Choose the coordinates according to the most precise of the three touch states, i.e., use the coordinates derived from the Stable state rather than from the Land On or Lift Up states. Distortion of the finger will affect the precise center point of the touch area. Table 4.2 presents the deviation of distance for contact of the five fingers' during the three touch states. Compared with the Stable state and Lift Up state, the deviation of the Land On state is larger. In the design of multi-touch devices, the first contact position is not accurate for consideration. Especially in high resolution multi-touch devices, the coordinate data should be derived from the coordinates of the Stable state.

Direct touch targets should be greater than 11.52 mm per side for square targets (or 5.76 mm radius for circular targets) in GUI design. In the user interface design of multi-touch widgets, the size of square targets must be larger than 11.52 mm (28.76 pixels) per side. When we design circular targets, the radius should be greater than 5.76 mm (14.38 pixels). These design paradigms can ensure a high touch precision with all fingers including the thumb and the little finger and with all tapping gestures.

Decrease arm movement and tapping actions in gesture design. Arm fatigue is the main drawback of multi-touch techniques. Constant tapping and the movement of the wrist or arm between points on the screen cause fatigue. The larger the touch screen, the more fatigue the arm will feel. To decrease the effect of arm fatigue, the movement of the hand and the tapping action should be kept to a minimum in gesture design. Based on the results in the study, more natural gestures (i.e.,contact area, orientation) can be designed than those used in currently available widgets. For example, variations in the size of the contact area can be used to trigger an event in an application. The rocking of fingers can control the movement of cursor. The arm movement can be decreased.

Decrease the influence of the occlusion of the display by the fingers. The occlusion of the interface display by the fingers and the hand further increases the difficulty of target selection. The user cannot determine the precise position of the target under such circumstances. In order to improve the usability of multi-touch techniques, finger orientation and finger contact area can be used to determine the position of the area obscured. Based on this determination, GUI designers can avoid improper layouts in user interfaces.

4.6.2 Widget Designs

Based on our experimental results and our findings from the previous section, the design space of finger input properties is explored here. To support our exploration, it is useful to define certain parameters of the design space.

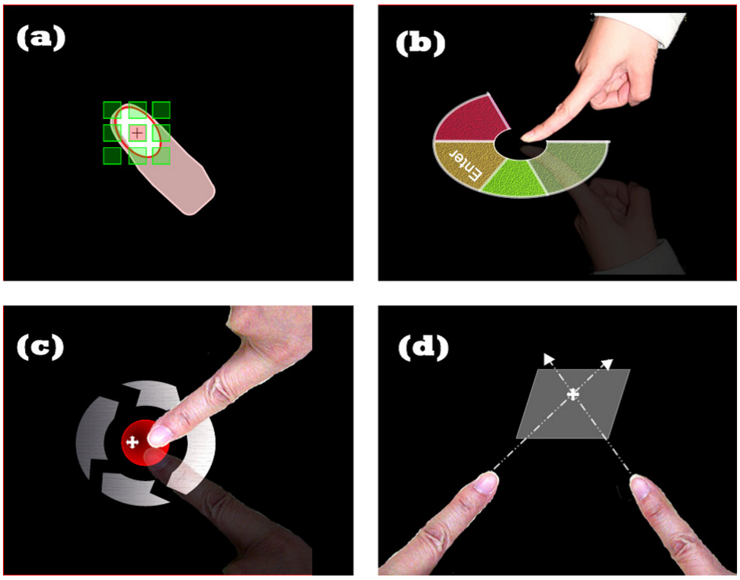

Fig. 4.9 Widget design demo, (a) finger combination cursor, (b) finger sector menu, (c) finger pointing stick, (d) finger cross selection.

Finger Combination Cursor. We informally define "Combination Cursor" as the combination of one area cursor and one point cursor. The shape of the finger imprint is treated as an area cursor and the center point of the contact area is a point cursor. The area cursor is a cursor that has a larger than normal activation area. The area cursor simply has a larger hot spot. There is evidence that performance with area cursors is better than performance with regular cursors for some small target acquisition tasks [63].

The human finger is a natural area-cursor input device. Based on the previous section, the shape of the finger's contact area can be described by an elliptical equation. The center coordinate value of the contact area can be treated as the position of the common point cursor. Fig. 4.9a shows the combination cursor. The finger combination cursor is capable of improving GUI performance in target selection tasks. When the finger touches the screen, two strategies are adopted to determine the selected target: (1) If only one target is covered by the tapped area, the target can be selected directly by the area cursor technique; (2) If there are more than 2 targets in the contact area, the target nearest to the center coordinates is the target. At the same time, we can adopt a "Shift" technique [24], where a callout can be used to display the finger touch area. The finger combination cursor combines the advantages of area cursor and point cursor.

Finger Sector Menu. Pie menu use is widespread in GUI design. In multi-touch techniques, a pie menu is often triggered by a finger touch. But some menu items are always obstructed by the finger. We present a new sector menu technique to resolve the occlusion of the finger. Incorporation of the newly defined finger properties improves the usability of the pie menu and makes the operation more natural. Fig. 4.9b shows the finger sector menu in use. The finger sector menu is triggered by variations in the finger contact area. When the contact area is greater than a predetermined threshold, a finger sector menu is triggered and displayed following the orientation of the finger. The position of the hand can be determined by variations in the direction of the contact coordinate(s): when a user touches the panel with the vertical touch gesture, the coordinates of the touch point (x1, y1) are obtained; the user then tilts the finger down; when the finger is in an oblique state, the second touch point (x2, y2) is obtained; the direction from the (x1, y1) to (x2, y2) is considered to be pointing towards the position of the hand. In the premise of knowing the finger physical position, the occlusion of the display menu item by the finger can be avoided. The user can select one menu item by rocking the finger or changing the finger's orientation. With the support of the finger sector menu technique, the user can pop-up the finger sector menu and select one menu item in a natural gesture without any additional finger movement.

Finger Pointing Stick. The pointing stick is an isometric joystick used as a pointing device that is used in notepad computers such as the IBM/Lenovo Thinkpad series. The finger looks and operates like a joystick while in vertical contact with the screen. The finger can simulate most functions of the pointing stick naturally. From the vertical to oblique positions, the center of the touch area can be used to move the cursor. The user can rotate the finger horizontally to fine-tune the cursor. With a proper setup of the control display ratio, the rocking of the finger can control the cursor movement in one direction (see Fig. 4.9c). The finger pointing stick is also capable of controlling the pop-up menu's selection.

Finger Cross Selection. Multi-touch screens are usually used in out-door information displays and interactions. The size of the screens is increasing in order to satisfy the requirements of special applications. The "Finger Pointing Stick" technique is a good way to control cursor movement by simply rocking the finger. The finger cross selection technique is an extension of the finger pointing stick. Fig. 4.9d shows the concept of finger cross selection. In order to select a distant target, the orientation of two fingers can present two radial lines. We can select the target by controlling the position of intersection of the two lines. The Finger Cross Selection technique is especially useful in wall-size display technology if the position of target is out of the user's reach.

Chapter 5

Empirical Evaluation of Tapping Performance on Direct-touch Surfaces

5.1 Introduction

Direct touch interaction increases the richness of interaction between the computer and the user. The concept of direct interaction system was first proposed by Shneiderman [78], which referred that one way to improve direct-manipulation systems is to extend the richness of operation objects and the directness of operation method. That means direct-manipulation should make the best use of the user's potential coordinating bimanual hands to implement complex tasks. In addition, direct touch interaction can be applied in some special circumstances, such as mobile, agravic and so on.

Bimanual interaction techniques have been discussed by many researchers. Obviously, bimanual interaction has some remarkable advantages. For example, two-handed input can increase the parallelism of manipulations and reduce the time of task switching, especially two-handed input techniques are in line with the everyday bimanual skills used in the physical world.

5.2 Motivation

Two-handed input is not always better than one-handed. Kabbash et al. [47] reported that for complex tasks that require the actions highly coordinated, two-handed input increased the user's cognitive load. In addition, users rarely use all fingers of two hands during interaction process. Moreover, the problems of hand occlusion, hand fatigue and low selection precision cannot be avoided in the research of two hands interaction techniques [1]. Finally, although bimanual interaction can increase the richness of gestures, it also makes it more difficult to design and to extend these gestures. Therefore, bimanual interaction technique must be designed carefully in order to make two-handed input effectively.

Based on the investigations in Chapter 2, we conclude that current multi-touch techniques treat all fingers as input devices with same weighted and the functions of each finger are the same. We do not fully exploit the characteristics of the human hand or fingers. In light of these previous researches, the investigation of the performance, manipulation complexity and comfort assessment of ten fingers is a valuable study.

5.3 Touch Performance of Human fingers

As mentioned in the previous chapters and sections, WIMP is a current mature paradigm for human computer interaction. In this paradigm, pointing is most important because all the events are based on the pointing actions. In WIMP paradigm, “windows”, “icon” and “menu” are the forms of expression of user interface, but “pointing” represents the control style in the manipulations of users. WIMP paradigm is wildly adopted by today's mainstream operation system, the improvement of “pointing” will enhance the availability of WIMP.

Finger is a natural input device of human body. Total 10 fingers maximally extend the manipulation and control ability of people. In common life, users typically regard that the index finger is the most flexible finger and the thumb is the most powerful finger. Many previous researches focus on gesture design, asymmetric, symmetric and interchangeable interaction, however, few discussed the manipulation performance of each finger. Actually, the manipulation performance of each finger of dominant or non-dominant hand is significant for interface design when users design user interface for multi-touch technique. After all, each finger is a fundamental element for input manipulation and gesture design.

5.4 Experiment 1

5.4.1 Goal

The goal of experiment is to empirically investigate the touch performance of each finger of dominant or non-dominant hand. This includes determining the touch performance in vertical touch gesture, touch error rate in “as accurate and quick as possible” condition. Such an evaluation will determine how to design natural gestures in multi-touch techniques.

5.4.2 Apparatus

A self-made FTIR multi-touch device is used. The touch panel is a 70×50 cm transparent acrylic panel, which internally reflects the IR-light. Infrared LEDs are installed along the edge of the acrylic, and infrared light is introduced edge-wise into a platen waveguide. The camera is a Philips SPC900NC with VGA CCD Sensor and USB 2.0 interface. When a finger makes contact with the touch panel, infrared light escapes from the acrylic so that the camera can detect the finger contact action through variations in the infrared light. The camera is operated at a resolution of 640 $\times$ 480 pixels. A projector is connected to the PC and is used to display screen on the panel. The resolution of screen is set up to 1024 $\times$ 768 pixels.

The experimental program is written by Action Script Language of Adobe Flash. TouchLib [32] is the background multi-touch processing program and TUIO protocol [35] is used to transfer protocol data between Flash and Touchlib. The experimental software runs on a 2.4GHz

Core 2 PC with the Windows XP SP3 operating system.

5.4.3 Task

The task of the experiment follows the instruction of Fitts' law. Fitts' law is a model of human psychomotor behavior developed in [79, 80]. By extending Shannon's theorem in information theory (a formulation of effective information capacity of a communication channel), Fitts discovered a formal relationship that models speed/accuracy tradeoffs in rapid, aimed movement (not drawing or writing). According to Fitts' Law, the time spent to move and point to a target of width W at a distance A is a logarithmic function of the spatial relative error ($\frac{A}{W}$), that is:

$$MT = a + b\log_2(\frac{A}{W} + 1) \tag{5.1}$$

where, MT is the movement time. a and b are empirically determined and device dependent constants. A is the distance (or amplitude) of movement from the start to the target center. W is the width of the target, which corresponds to "accuracy".

The term $\log_2(\frac{A}{W} + 1)$ is called the index of difficulty (ID). It describes the difficulty of the motor tasks. 1/b is also called the index of performance (IP), and measures the information capacity of the human motor system. Mathematically speaking, Fitts' Law is a linear regression model.

Fitts' law is an effective quantitative method of modeling user performance in rapid, aimed movements, where one appendage (like a hand) starts at a specific initial position, and moves to rest within a target area. Card et al. [81] reported the first comparative evaluation of the mouse, and is also the first that explored Fitts' Law in Human-Computer Interaction. Fitts' Law is a theory intensively used in Human-Computer Interaction. It can be used for assisting interface design and interface evaluation.

In the experiment, by examining the size of touch panel and the resolution of screen, we design A as 100, 200 and 300 pixels respectively and design W with 20, 40 and 60 pixels respectively. A black bar will be displayed on the left or right side of the touch panel and the subjects are demand to touch on the bar in the condition of “as accurately and quickly as possible”. All subjects stand in front of the multi-touch device and face to the long-edge of the touch panel.

5.4.4 Participants

Twelve volunteers, 20-23 years old (average 21.6), participated in the experiment. All were right-handed and had no prior experience with direct-touch surfaces. Before the experiment, they were prepared with certain amount of training to guarantee the correct manipulation.

5.4.5 Procedure and Design

The participants were instructed to tap six times on each of four targets using two finger gestures with their five fingers in turn. In summary, the experiment consisted of:

12 participants ×

5 fingers ×

2 hands ×

3 A(100, 200, 300 pixels) ×

3 W(20,40,60 pixels) ×

7 blocks ×

6 repetitions

= 7560 trials.

Prior to performing trials for each task, participants were given a short set of warm-up trials to familiarize themselves with the touch manner. For each trial, we collected the finger touch data (time and position, shape). The experiment lasted approximately 60 minutes for each participant include his/her rest time.

5.4.6 Results

5.4.6.1 Results of ID and MT

The data of ten fingers are carefully processed. According to the Fitts'law, we calculate ID, MT and finally obtain the equation of their relationship. Fig. 5.1, Fig. 5.2, Fig. 5.3, Fig. 5.4 and Fig. 5.5 show the results of ID and MT of each finger respectively.

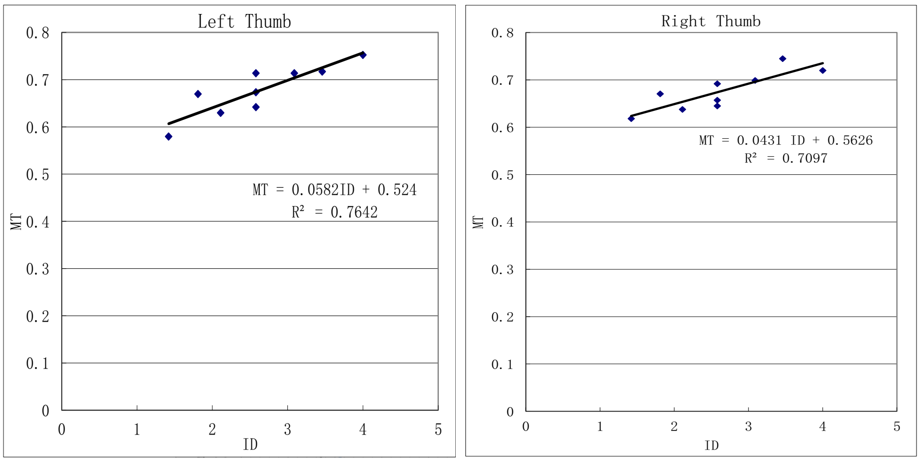

Fig. 5.1 Touch tasks of Left and right thumb under Fitts' Law.

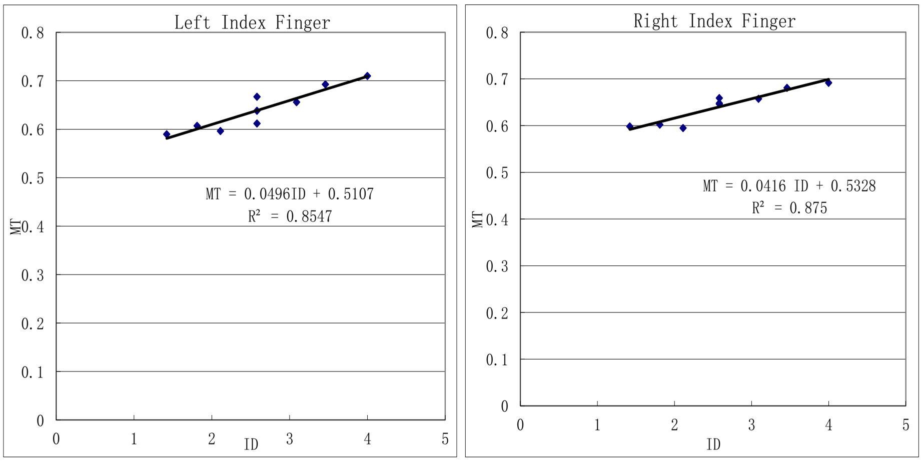

Fig. 5.2 Touch tasks of Left and right index finger under Fitts' Law.

5.4 Experiment 1

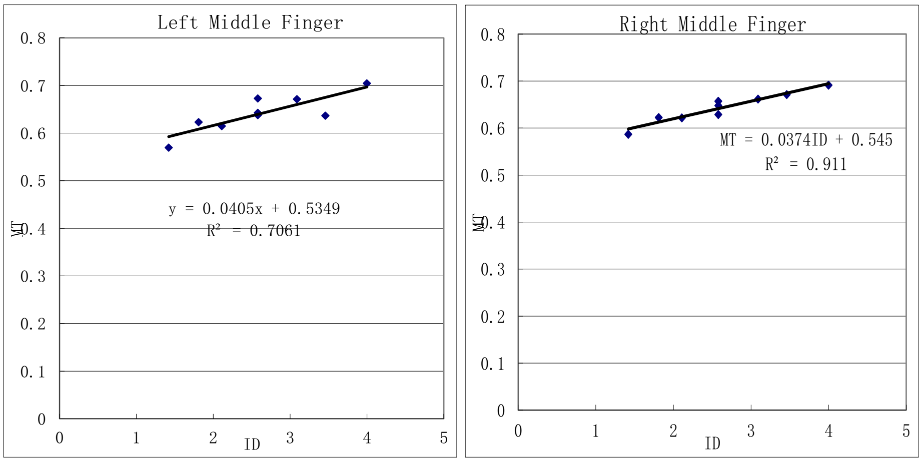

Fig. 5.3 Touch tasks of Left and right middle finger under Fitts' Law.

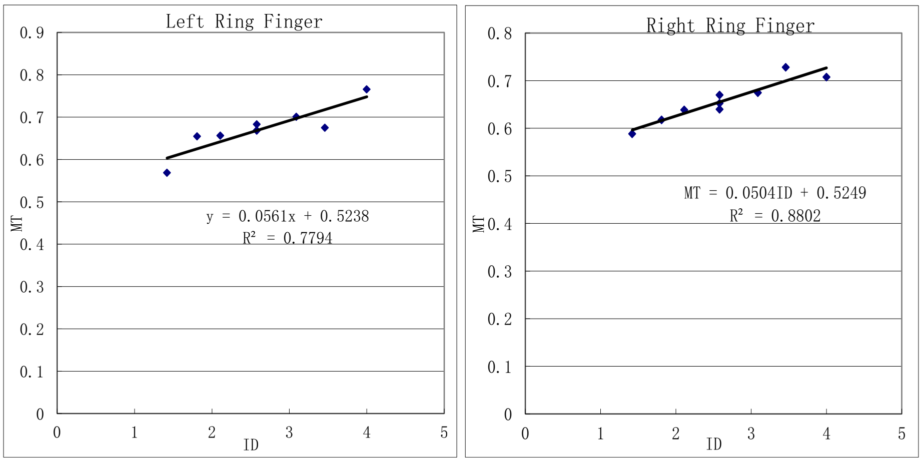

Fig. 5.4 Touch tasks of Left and right ring finger under Fitts' Law.

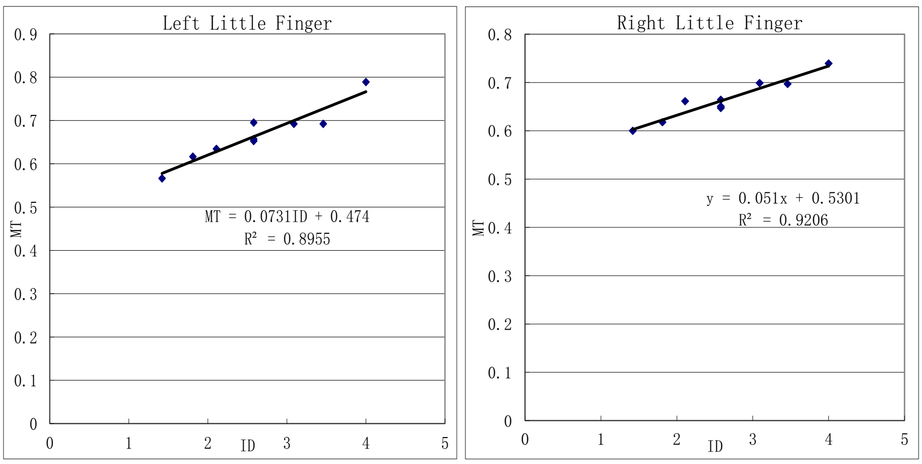

Fig. 5.5 Touch tasks of Left and right little finger under Fitts' Law.

5.4.6.2 Throughput

The throughput (TP) of an input device is a measure of its efficiency and is as 1/b [82]. Throughput is used by ISO 9421 Part 9 when it comes to comparing the efficiency of different input devices [83, 84]. Throughput is calculated in bits per second (bps). ISO 9241-9 proposes that, for a comparisons of input devices, evaluation results should be based on throughput rather than task completion time. Zhai [82] suggest that future studies should use the complete Fitts law regression characterized by (a, b) parameters to characterize an input system. a reflects the non-informational aspect and b the informational aspect of input performance. For convenience, 1/b can be named as throughput which, unlike ID/MT, is conceptually a true constant.

Table 5.1 The a, b and 1/b values of ten fingers in Fitts' law

Finger	a	b	1/b(Throughput)
Left Thumb	0.524	0.0582	17.1821
Left Index Finger	0.5107	0.0496	20.1612
Left Middle Finger	0.5349	0.0405	24.6914
Left Ring Finger	0.5238	0.05611	17.8221
Left Litter Finger	0.474	0.0731	13.6799
Right Thumb	0.5626	0.0431	23.2019
Right Index Finger	0.5328	0.0416	24.0384
Right Middle Finger	0.545	0.0374	26.7380
Right Ring Finger	0.5249	0.0504	19.8413
Right Litter Finger	0.5301	0.051	19.6078

Table 5.1 shows the "throughput" of each finger. Fig. 5.6 shows a bar diagram of each finger's throughput. Fig. 5.6 indicates the throughput of each finger is equal to the finger's length except right thumb. The throughput of left middle finger, left index finger, right thumb,

right index finger and right middle finger are relatively higher. This is consistent with our common sense. People usually think that the index finger as the most flexible finger, however, the experimental results show the middle finger offers the best performance as a tapping device.

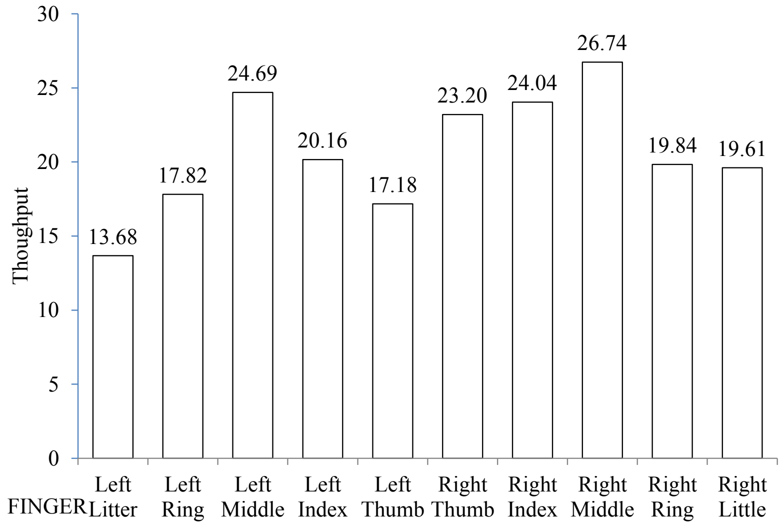

Fig. 5.6 Bar diagram of each finger's throughput (1/b). (unit: bits/s)

We record the duration time of tapping task with each finger respectively. By comparing fingers of dominant and non-dominant hand, we find the task duration time of each finger has no significant difference. Actually, litter fingers of two hands also can implement tapping task opportunely (see Fig. 5.7).

5.4.6.3 Error rate

Another interesting result is the missing rate in experiment. We collected the data and count the number of errors. Fig. 5.8 shows the error rate of the tapping task. We define a "missing" state as a touch position beyond the edge of the target. We treat one missing state as an error.

Obviously, the error rates of index finger and middle finger are also lower than other fingers. And the error-rates of dominant hand are apparently lower than non-dominant hand. It

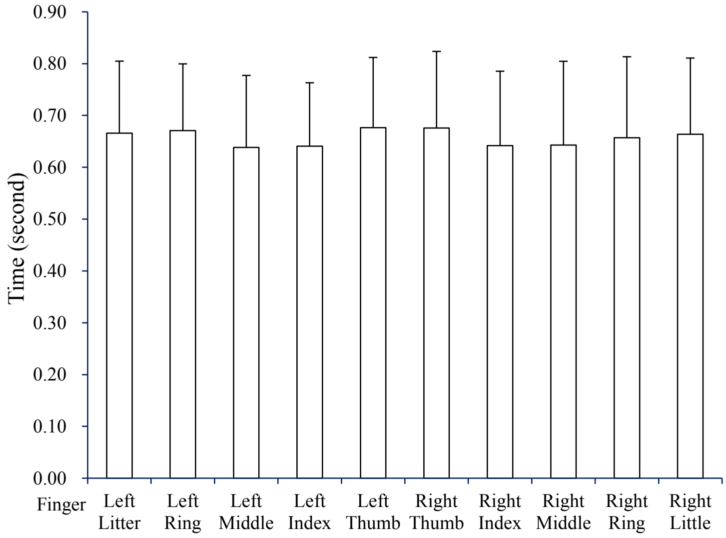

Fig. 5.7 Bar diagram of task duration time and standard deviation of each finger. (unit: time=second)

turns out that dominant hand is more suitable for using in gesture design. It is easy to imagine that the error rates of two thumbs are higher because of their fatness. The results regarding two little fingers also show the error rates would become lower if the finger is slim. Because the users can observe the touch area more carefully with fewer occlusions when they use little fingers.

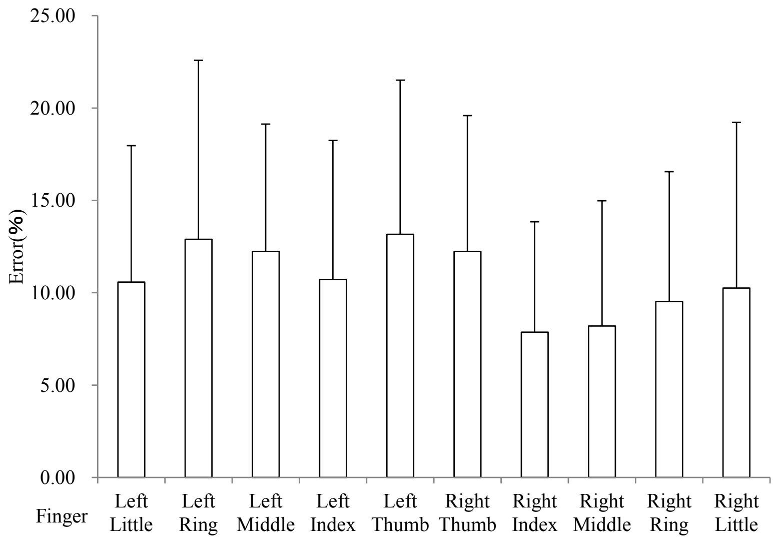

Fig. 5.8 The bar diagram of each finger's error rate in Fitts' law experiment.

5.5 Experiment 2

5.5.1 Description

Two indexes (i.e., complexity index and comfort index) are defined to subjectively evaluate subjective assessments of the participants. “Complexity Index” indicates the ease of direct-touch manipulation of fingers. The participants use complexity index to assess which finger they can easily implement task, which is not. “Comfort index” is defined as comfort level of manipulation. It is a subjective assessment index. After the experiment 1, many participants complain that the task is so boring that they feel very fatigue. Comfort index is used to evaluate the comfort level and fatigue level of the manipulation. The evaluation score of two indexes bases on a seven-point system. 1 of 7 basis means too difficult or too fatigue and 7 of 7 basis means very easy and comfortable.

5.5.2 Results

After the first tapping task which follows the requirements of Fitts’ law, twelve participants are asked to record their subjective assessment on the investigation sheet. The results are showed in Fig. 5.9 and Fig. 5.10.

Fig. 5.9 and Fig. 5.10 show the same pattern. No matter which hand is used, the index finger and middle finger are most suitable for tapping task. Almost all participants report that tapping task can be completed easily and comfortably with their index finger and middle finger. Litter finger and ring finger are not suitable for tapping task. Many participants complain that they feel very tired when using little finger to tap on the surface. The thumb is a special case. Actually, thumb is the most powerful finger on the hand. However, the terrible error-rate of the thumb determines that thumb is not a good choice for tapping gesture.

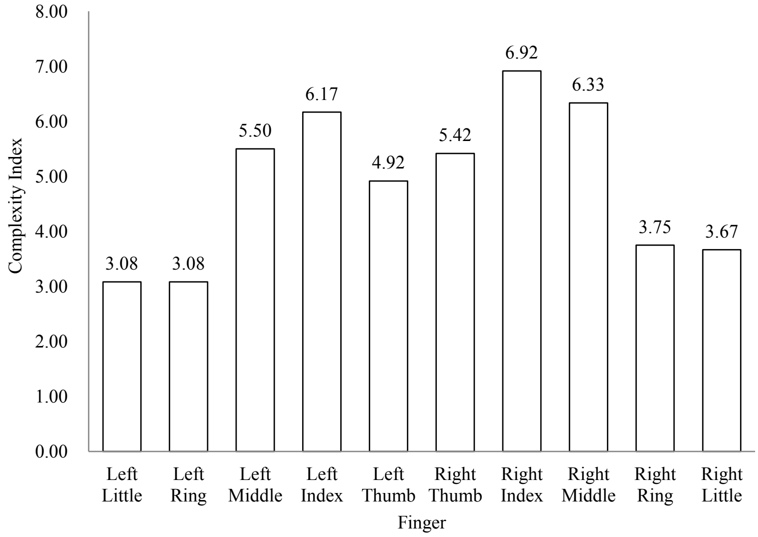

Fig. 5.9 The statistical bar diagram of complexity index.

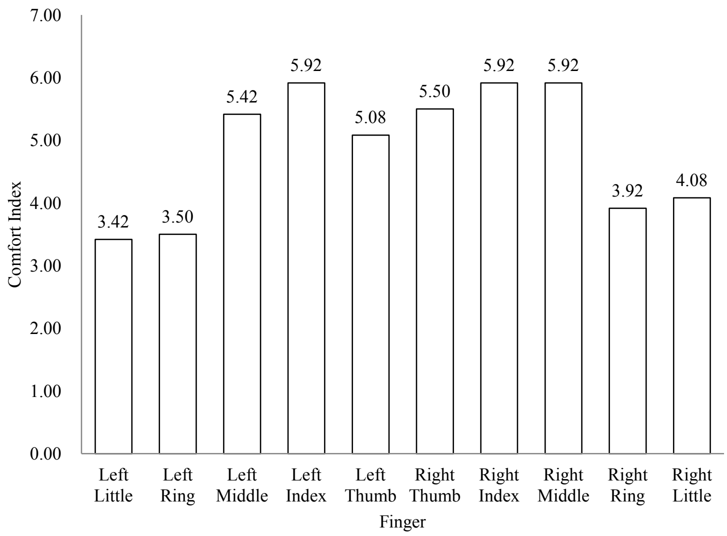

Fig. 5.10 The statistical bar diagram of comfort index.

5.6 Discussion

It should be noted that this study does not explore multi-target touch or multi-user collaboration. Many previous literatures have studied the collaboration of two fingers or more fingers in interface design. Our work build on the results established by the single finger's tapping performance assessment.

5.6.1 The number of touch points and interface design

Many commercial multi-touch systems can support 4 points touch at the same time. Some can support 2 points (e.g., some Wacom products). Of course, multi-touch means at least two touch points. Actually, the keyboard everyday we use is a standard multi-touch device. When we press “shift” key and other alphabet keys simultaneously, a capital letter will appear on the screen. But we have never seen that we press four keys on the keyboard. Considering our experimental results, we can make a rough conclusion that if we design tapping or touch gesture for interaction, two points multi-touch device is enough to meet the requirements. Most common multi-touch gestures (i.e., pinch for zoom in/out) are based on two points.

However, more touch points imply that we can get more useful amount of input. In theory, it can extend the input bandwidth and enhance the interactive ability. How many touch points we can use to design user interface is still an question.

5.6.2 Assessment with Steering Law

In the study, we do not design an experiment for assessment with Steering Law [85]. The steering law is a predictive model of how quickly one may navigate, or steer, through a 2-dimensional tunnel. The tunnel can be thought as a path or trajectory on a plane that has an associated thickness or width, where the width can vary along the tunnel. The goal of a steering task is to navigate from one end of the tunnel to the other as quickly as possible, without touching the boundaries of the tunnel.

However, in current direct-touch devices, due to the drawbacks of “fat finger”, we seldom use our fingers to pull-up a menu and select a submenu item. We typically use fingers to touch an icon or a button to trigger a function. Hence, we think that the finger’s assessment with steering law is important indeed but it is not useful for interface design of current multi-touch system.

Chapter 6

Detecting and Leveraging Finger Orientation for Interaction

6.1 Introduction

In previous chapters, we investigate the finger's properties and deeply study the precision and performance of each finger in order to extend the availability of direct-touch devices. We have discussed that many drawbacks limit the application of direct-touch technology. For example, current multi-touch designs are mainly based on multi-point information, i.e. the touch-sensitive devices primarily use the center coordinates of the human finger's contact region as cursor positions. Therefore, most interactions mainly rely on touch positions or variations in touch movements. Relatively few research demonstrations have used auxiliary information other than touch position, such as the shape [37, 71] or size of the contact region [36] for enhancing the naturalness of the interaction.

6.2 Motivation

One potentially accessible piece of information from a finger's contact point is its orientation. Orientation is a natural cue as it provides the direction for a user to point in and is used in many daily interactions, such as pointing to communicate with others, acquiring or positioning an object, or even leading and directing attention. If we consider an orientation vector consisting of a direction and an angle from a point of reference, very few systems have used the orientation vector to enhance the naturalness of the interactions.

6.3 Finger Orientation

By extracting the longitudinal axis of a finger's contact shape, it is straightforward to detect the undirected angular configuration $(0° - 180°)$ of a straight line that approximates the user's finger. However, this result is ambiguous. The exact orientation vector of the finger could be one of two opposite directions that align with the undirected line, making it difficult to utilize the exact direction of the finger for interaction (see Fig. 6.1a).

To address this limitation, we present a novel and robust algorithm that accurately and unambiguously detects the orientation vector by considering the dynamics in finger contact (see Fig. 6.1b). This algorithm is general enough for any direct-touch surface that generates contact shape information, without resorting to additional sensors. We demonstrate that finger orientation information is a key in the design of orientation-aware interactions and widgets, for example to facilitate target selection, or to optimally orient elements in the workspace to adapt to the user's position. Additional information about the user can also be inferred from finger orientation, such as hand occlusion region and the position of the user. These cues can in turn be leveraged to further enrich the interaction on touch-surfaces.

6.4 Finger Orientation Detection Algorithm

We present an algorithm to detect the directed orientation vector of the user's finger, based on real-time information collected from the shape of the finger contact. Here the finger orientation refers to the 2D orientation of the finger's projection on the surface. Prior literature [45,86] points at two types of finger touch on interactive surfaces: vertical touch and oblique touch (see Fig. 6.2). A vertical contact occurs when the finger is directly pointing downward, toward the

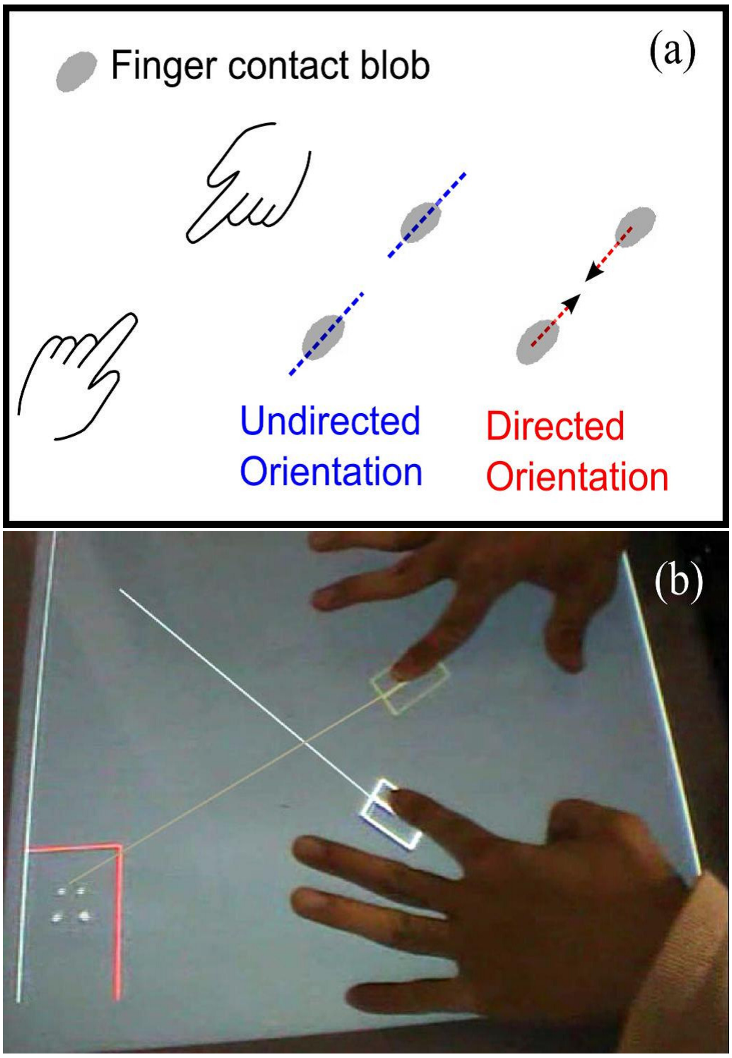

Fig. 6.1 (a) Undirected orientation vs. directed orientation vector of the finger. (b) Orientation detection in action.

surface (see Fig. 6.2a). Obviously this does not provide usable orientation information. Conversely, an oblique touch occurs when the finger lands on the surface at an oblique angle (see Fig. 6.2b). Considering common practices handling physical objects, as well as the necessity to accommodate long fingernails by some people (especially women), it is expected that the oblique touch is more likely to happen when people touch interactive surfaces. A unique finger orientation can be determined from an oblique touch, which is the basis of our algorithm.

For each frame of input that contains all contact pixels on the surface, we first conduct a

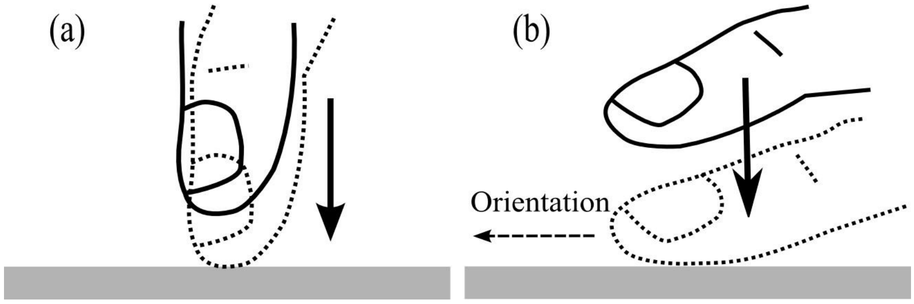

Fig. 6.2 Two ways of finger touch. (a) vertical touch. (b) oblique touch.

connected component analysis to extract all finger contact regions. Then for each finger contact, its orientation is determined by our algorithm. The algorithm has four major steps: fitting the contact shape; detecting the oblique touch; disambiguating the finger direction; and continually tracking the orientation.

6.5 Fitting Contact Shape

When a finger obliquely touches the surface, its contact region appears as an elliptic shape (see Fig. 6.3). This shape can then be fitted into a perfect ellipse described by Equation 6.1 using least-square fitting, as presented in [61]:

$$\left(\frac{(x-x_0)\cos\theta+(y-y_0)\sin\theta}{length/2}\right)^2+\left(\frac{(y-y_0)\cos\theta-(x-x_0)\sin\theta}{width/2}\right)^2=1 \qquad (6.1)$$

The length (magnitude of the long axis), width (magnitude of the short axis), and slant angle $\theta(0 \leq \theta \leq \pi)$ describe the shape of the ellipse; and (x_0, y_0) is the center coordinate of the finger contact. The area of the contact region can be calculated by simply counting the pixels within it. The slant angle θ describes the undirected orientation of the finger.

In order to generate reliable finger orientation, we need to determine whether the finger is currently in an oblique touch state. Two properties of the finger contact region are critical for identifying an oblique touch: area and aspect ratio. Equation 6.2 shows the identification criteria:

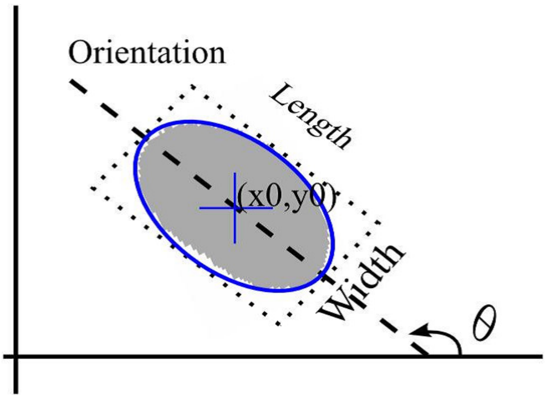

Fig. 6.3 Finger contact region fitted to an ellipse. Width, length and slant angle can be obtained from results of the fit. Identifying Oblique Touch.

$$\begin{cases} area > ta \\ aspect\ ratio = \dfrac{length}{width} > ts \end{cases} \tag{6.2}$$

where ta and ts are empirically determined thresholds. Both criteria need to be satisfied for an oblique touch to be identified, otherwise we consider the finger to be in either vertical or accidental touch and ignore its orientation. These two criteria were determined based on pilot trials and previous literature.

Area Criterion. As found in prior investigations of finger input properties [61], the contact area is significantly different in vertical and oblique touches. The mean contact area in vertical touch is between 28.48 and 33.52 mm^2, whereas the mean contact area in oblique touch is significantly larger (between 165.06 and 292.99 mm^2). After further validation by pilot trials, we set the threshold ta to be 120 mm^2.

Aspect Ratio Criterion. Area alone is not reliable enough to identify an oblique touch because a large contact area can also result from pressing harder in a vertical touch. The undirected finger orientation information is stable only when the finger contact is elongated. The larger the aspect ratio (i.e., the more oblique the finger is), the more accurate is our estimation of the orientation. In our algorithm, based on pilot trials of comfortable manipulations, we set the aspect ratio threshold ts to be 120%.

6.6 Disambiguating Finger Direction

From the contact shape fitting step we have acquired the undirected finger orientation θ. However, the true direction of the finger could be either θ or $180° + \theta$. The key innovation of our algorithm is to resolve this ambiguity by considering the dynamics in the finger landing process.

The human finger has soft and deformable tissues. The distortion of the finger muscle is inevitable upon contact. Since it is difficult to extract full orientation from a single finger contact, we instead closely examine the deformation of the finger's contact region in the process of it landing on the surface. Fig. 6.4 shows the contact region across time when a finger is landing.

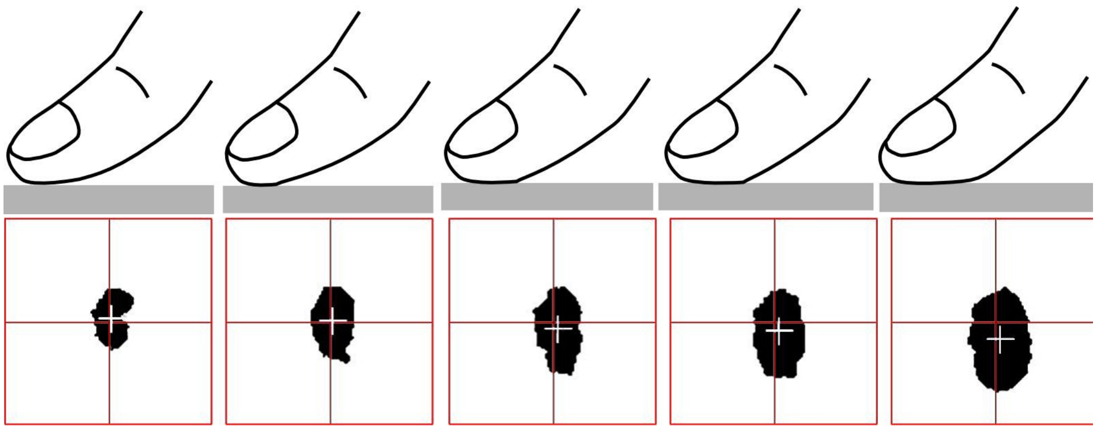

Fig. 6.4 Finger contact deformation over time. The crosshair shows the center of the contact region.

It is apparent that the center point of the finger contact moves inward, towards the user's palm. This movement can be explained by closely examining the landing process in an oblique touch: the finger tip gets in contact with the surface first; as the pad of the finger increases its contact area the center of the contact region shifts inward.

Considering the finger's deformation, by tracking the variation of the contact center during the landing process, we can roughly infer which side the user's palm lies in, and in turn which direction the finger is pointing to. Fig. 6.4 shows the variation of the contact center between two consecutive frames $t-1$(blue) and t (red). Frame $t-1$ is the last frame of non-oblique touch state and frame t is the first frame of oblique touch state. We can then calculate angle as a rough estimation of the directed finger orientation by taking the azimuth angle of vector $(-\Delta x, -\Delta y) = (x(t-1) - x(t), y(t-1) - y(t))$, which points away from the palm.

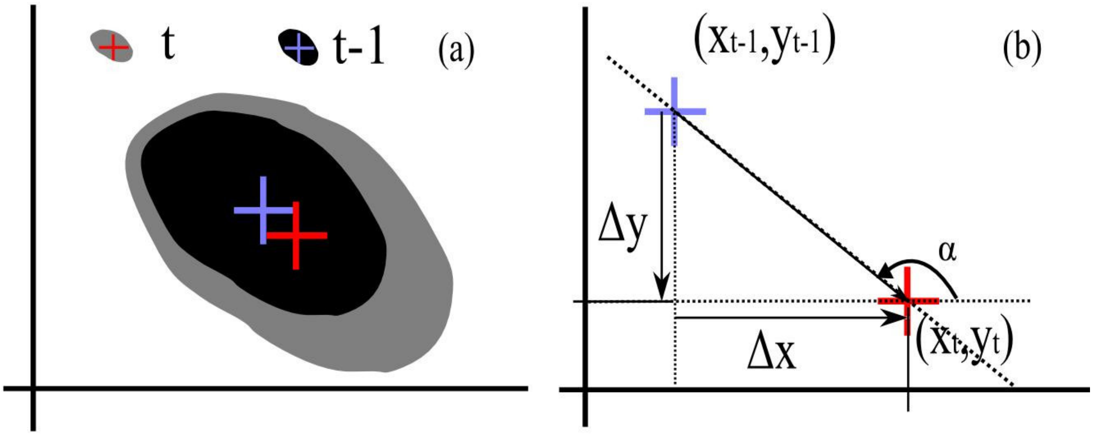

Fig. 6.5 Finger direction disambiguation. (a) Finger contact at frame t-1 and t. (b) Rough estimation of directed finger orientation.

Finally, α is used as the cue to disambiguate the undirected finger orientation θ, so that the final directed finger orientation Φ is consistent with α.

$$\Phi = \begin{cases} \theta & (|\alpha - \theta|) \leq 90^\circ \\ \theta + 180^\circ & (|\alpha - \theta|) < 90^\circ \end{cases} \tag{6.3}$$

6.7 Continual Orientation Tracking

Once the complete finger orientation has been unambiguously determined for one frame using the previous step, this step need not be repeated for the following frames. The orientation disambiguation of the following frames then depends on the fact that no abrupt change of orientation will occur between each two consecutive frames. Due to the finger's range-of-motion and the limitations imposed by the physical anatomy of the finger, the variation in the finger's orientation is likely to be very gradual.

In every subsequent frame $t+1$, the directed finger orientation in the previous frame $\Phi(t)$ is used as the cue to disambiguate the current undirected finger orientation $\theta(t+1)$, i.e.:

$$\Phi(t+1) = \begin{cases} \theta(t+1) & (|\Phi(t) - \theta(t+1)|) \leq 90^\circ \\ \theta(t+1) + 180^\circ & (|\Phi(t) - \theta(t+1)|) > 90^\circ \end{cases} \qquad (6.4)$$

6.8 Performance Evaluation

6.8.1 Goal

We conducted an experimental evaluation to assess the performance of our finger orientation detection algorithm, including the stability and precision in determining orientation of static and dynamic fingers. We note that given the irregular shape of a finger and different viewing perspectives, in practice the finger orientation is largely subject to human interpretation. An objective "true value" does not exist in a strict sense. Instead, to best inform interaction designs, in this evaluation we compare the detected orientation to the finger orientation subjectively perceived by the user; this is what users would rely on for real interactions if no visual feedback is provided.

6.8.2 Apparatus

The apparatus we used in the study is a direct-touch tabletop surface based on Frustrated Total Internal Reflection (FTIR) technology [12]. The tabletop is approximately 27" $\times$ 18" in size, and 0.8m in height. A camera installed beneath the surface captures the input image, working at a resolution of 640×480 pixels and at a capture rate of 30 fps. The experimental software is built upon the Touch-Lib open source API [32], augmented by our orientation detection algorithm. The system runs on a 2.4GHz Duo Core PC with Windows XP SP2 OS.

6.8.3 Participants

Eight volunteers, four male and four female, 26-37 years old, participated in the experiment. All were right-handed and had no prior experience with direct-touch surfaces.

6.8.4 Task

We evaluated the algorithm with four tasks, each examining a different aspect of the algorithm. The participant sat in front of the tabletop and used the right index finger to complete each task. We did not provide any visual feedback concerning the orientation of the finger as detected by the algorithm. As a result, participants had to completely rely on their subjective perception of the finger orientation.

Task 1 - Static Orientation Stability. This task examines the stability of the algorithm when the finger is kept still. The participant touches the surface at an arbitrary position and finger orientation, and dwells in the position for more than 5 seconds. The user lifts the finger when prompted by the experimenter. All the values of finger orientation were recorded and the data up to 5 seconds are used to evaluate the orientation stability during this period.

Task 2 - Static Orientation Precision. This task examines the orientation detection precision for a static finger. A red cross is displayed on the surface. An arrow on the red cross indicates the orientation to point at by the participant. Participants touch the center of the cross while matching the finger orientation as accurately as possible to the direction of the cross (see Fig. 6.6a). We use four directions for the task: 165°, 150°, 135°, and 120° (counterclockwise from east), as these can be comfortably achieved by the right index finger. If necessary, the participant can further adjust the finger orientation after landing on the surface. Once satisfied, the participant presses a key on the keyboard to indicate completion. We record the detection error (detected finger orientation minus actual arrow direction, at the moment of task completion) for each trial.

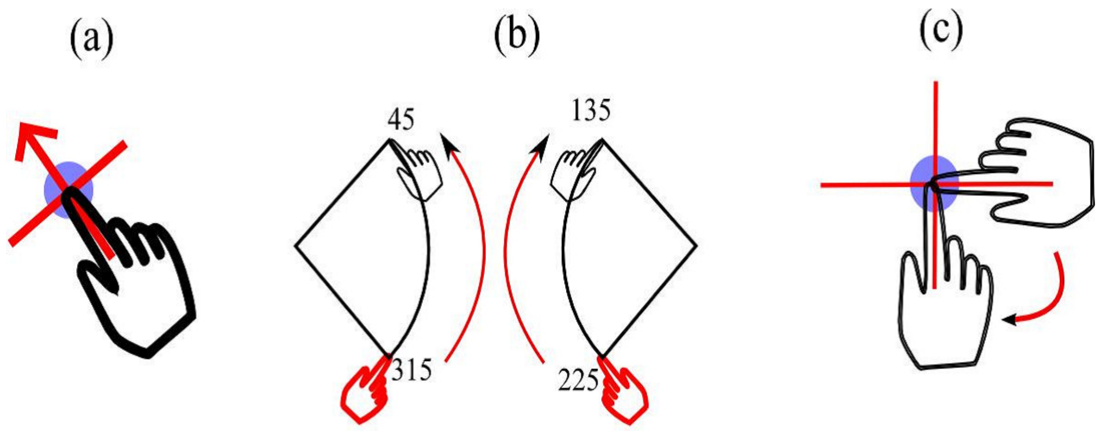

Fig. 6.6 Evaluation tasks. (a) Static orientation precision task. (b) Dynamic orientation precision task. (c) Involuntary Position Variation task.

Task 3 - Dynamic Orientation Precision. This task examines the orientation detection precision when the finger is moving and rotating on the surface. Participants trace the index finger along a circular arc displayed on the surface from a start to an end point (see Fig. 6.6b). At any given point during the movement, the finger orientation needs to be aligned with the arc (i.e. its tangential direction at the point) as precisely as possible. Two arcs are used for the task: counterclockwise from 315° to 45°; and clockwise from 225° to 135°. This arc tracing task effectively enables us to continuously acquire the (perceived) orientation "ground truth" during a dynamic operation. For each point on the arc, we record the detection error (detected finger orientation minus tangential direction at the point).

Task 4 - Involuntary Position Variation in Rotation. This task examines the involuntary variation of finger position coordinates (x, y) associated with a finger rotation. An involuntary variation in position occurs when the user incidentally moves the center coordinate of the finger during a rotation. In this task, the participant placed the finger on a red cross displayed on the surface, and rotated it clockwise from $0°$ to $270°$ while keeping the finger position static (see Fig. 6.6c). We record the range in variation of the finger's center position during the rotation.

For all tasks, the participant was asked to repeat the trial if the orientation disambiguation result was incorrect (i.e., providing the opposite orientation). Within each task, all trials were randomly ordered to prevent practice effects.

6.8.5 Design

Participants performed each task with six repetitions. The full evaluation consisted of:

8 participants $\times$

[Task-1 + (4 orientations in Task-2) + (2 arcs in Task-3) + Task-4] $\times$

6 repetitions

= 384 trials

6.8.6 Result

Disambiguation Success Rate. For all tasks, the disambiguation algorithm generated 13 errors in total. This resulted in a success rate of 96.7% (384 out of 397 trials), indicating good performance of the algorithm.

Static Orientation Stability (Task 1). The average variation range during each finger dwelling period is $0.59°$ (std. dev.= 0.15°). This demonstrates that our algorithm is very stable. The low level of random noise is caused by both the user's unconscious finger jitter, and the imaging noise introduced by the camera. According to this result, in practice we can ignore finger orientation changes that are less than $1°$.

Static Orientation Precision (Task 2). Results of Task 2 show that the detected finger orientation matches closely with the finger orientation perceived by the user. The average detection error (absolute value) is 2.69° (std. dev. = 1.76°). ANOVA showed no significant difference between the detected finger orientation and the perceived orientation, indicating the detection error was not biased towards one specific direction. When considering the signs of the error, we obtained an upper bound of +5.84° and a lower bound of -4.70° at the 95% confidence interval.

Note that this is the detection error when there is no visual feedback and therefore incorporates both the imprecision of the algorithm and variations in user perception. This indicates that for interactions that involve a single touch action without visual feedback, our algorithm can provide a precision within approximately $\pm 5^{\circ}$. Across the complete 360° orientation range, this gives 36 usable orientation levels (each with the tolerance interval of 10°) that can be reliably detected for interaction. However, in the presence of visual feedback, the user can adjust their input accordingly and perform closed-loop actions with much higher accuracy (1°) as suggested by our results on static orientation stability.

Dynamic Orientation Precision (Task 3). The results of task-3 show that the continual orientation tracking algorithm is reasonably accurate across the whole movement range. The average orientation error (absolute value) is 14.35° (std. dev. = 9.53°). Again ANOVA showed no significant difference between the detected finger orientation and the perceived orientation, indicating the lack of systematic detection bias. The upper and lower bound of signed error at the 95% confidence interval was +29.69° and -26.81° respectively. This increased error is partly explained by the difficulty for the user to smoothly and precisely control finger orientation during finger movement. Based on our observation, instead of continuously rotating the finger throughout the trial, most participants made discrete compensation changes of finger orientation when they noticed it deviated from the arc. This observation is likely to hold in other actions of simultaneous finger movement and rotation as well. According to this, in interaction designs we should ideally avoid requiring the user to precisely control the finger orientation while moving the finger, especially if no visual feedback is provided. No significant difference in orientation detection resulted between clockwise and counterclockwise movements.

Involuntary Position Variation in Rotation (Task 4). The average position variation during finger rotation was 2.02mm for x-coordinate (std. dev. = 0.96mm); and 2.00mm for y-coordinate (std. dev. = 1.08mm). Aside from detection noise, this variation can be explained by two factors: the user unconsciously moves the finger during rotation; and the user's perceived rotation center does not precisely match the finger center detected by the system. This variation in finger position needs to be taken into account when designing interactions based on finger rotation. Displacements under 2mm of the finger's position during rotation should be ignored in such interactions.

On the other hand, the detected orientation across each rotation trial shows that it changes continuously and smoothly at a relatively constant rate, different from the discrete jumps observed in Task 3. This confirms that the user is able to finely control finger orientation for interactions when the finger position is kept static. Combining this with results from Task 3, for interactions that involve both finger translation and rotation, the best strategy may be to let the user first move and rotate the finger simultaneously for coarse maneuver, but to also allow the user to "park" the finger for finer orientation adjustment in the end.

6.9 Interactions Using Finger Orientation

Finger orientation information as detected by our algorithm may lead to a set of new interaction designs.

6.9.1 Enhancing Target Acquisition

Finger orientation can be employed to design new target acquisition techniques on interactive surfaces.

Directed Bubble Cursor. Bubble cursor [87] is an efficient target selection technique that dynamically resizes the activation region of an area cursor [63] so that it always select the one target at the shortest distance to the center of the cursor (see Fig. 6.7a). On a direct-touch surface, we can further enhance the bubble cursor by considering the finger orientation, so that the selection is biased towards targets in front of the finger direction, and against targets to the back of it. The shape of the activation region may also become slightly skewed to reflect this (see Fig. 6.7b). This is realized by applying different multiplying weights to the target distances based on the target's relative azimuth angle compared to the finger center and orientation. Targets with an azimuth angle of 0 (i.e. in line with the finger orientation) have the smallest weight, and those with an azimuth angle of 180° (i.e. opposite to the finger orientation) have the largest weight (see Fig. 6.7c). By choosing the target with the shortest weighted distance, the directed bubble cursor displays a behavior that is consistent with real-world conventions when using a finger to refer to objects, where both finger position and direction play a role.

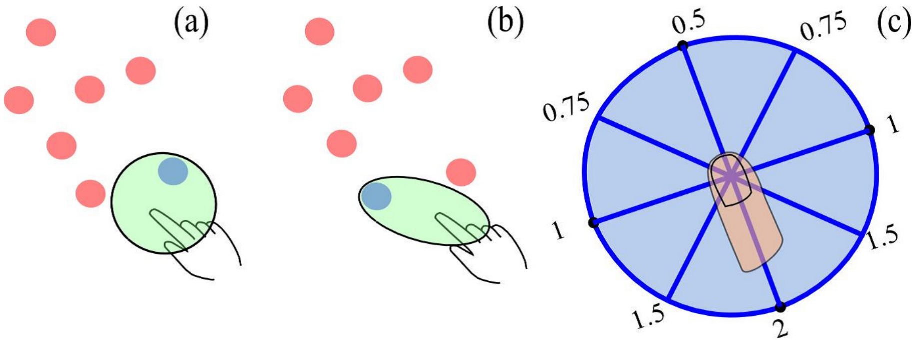

Fig. 6.7 (a) Regular bubble cursor. (b) Directed bubble cursor. (c) Distance weights according to finger orientation.

Aim and grab. The finger orientation can also be utilized to select objects that are far away on the interactive surface by distant pointing. To do so, the user touches the finger on the surface to cast a selection ray aligned to the finger orientation. The first object intersected by the ray gets selected. The user can rotate the finger to aim at different targets. To switch between multiple objects intersected by the selection ray, the user can move the finger forward or backward along the ray. This is similar to the Depth Ray selection technique proposed by [88] for 3D volumetric displays. A finger flick inward brings the selected object to the user (see Fig. 6.8).

As discussed in [61], we can also use the intersection of two selection rays determined by two fingers for precise selection of distant targets (see Fig. 6.1b).

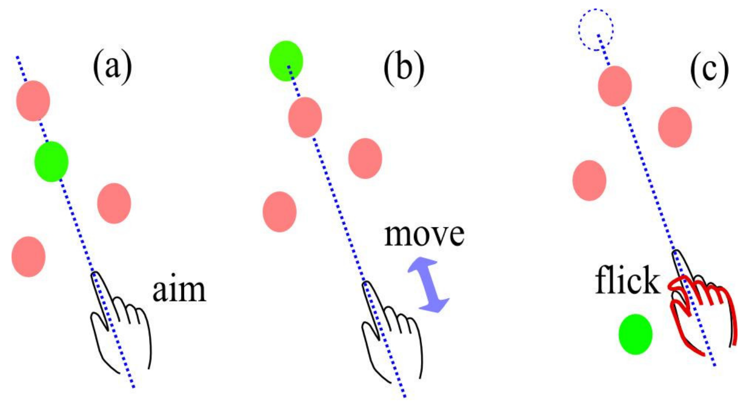

Fig. 6.8 Aim and grab. (a) Aim finger to select an object. (b) Move finger along the orientation vector to switch between multiple objects. (c) Flick finger inward to bring the selected object.

6.9.2 Orientation-Sensitive Widgets

Finger orientation can be treated as an additional direct input dimension for interface widgets. Fig. 6.9 shows two example designs of such orientation-sensitive widgets.

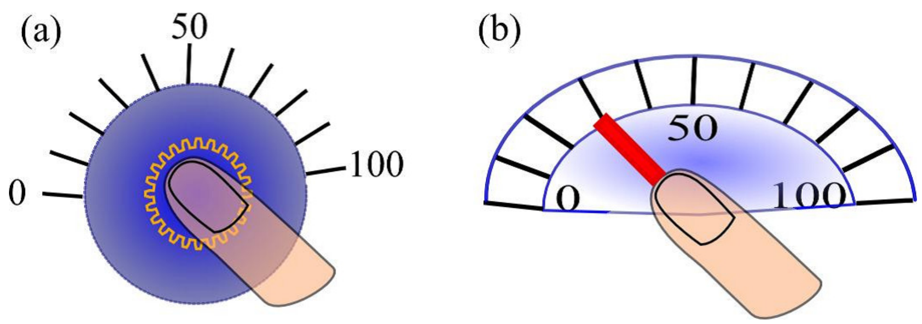

Fig. 6.9 Orientation-sensitive controls. (a) Orientation-sensitive button. (b) Orientation dial.

An orientation-sensitive button (see Fig. 6.9a) allows the user to use the finger orientation to specify the parameter of the button functionality while hitting the button. By doing so, function invocation and parameter specification are combined into a single step (a valuable substitute to widgets such as combo boxes on GUIs). As discussed previously, for such a widget with no continuous visual feedback we can support an orientation resolution of 10°.

An orientation dial (see Fig. 6.9b) allows the user to continuously adjust a parameter with high precision. Compared to other parameter adjustment widgets such as a slider, this supports a large range of parameter values while requiring minimal finger movement and screen estate.

6.10 Inferences From Finger Orientation

In addition to directly utilizing finger orientation for input, we can make further inferences about the user by considering the finger orientations and positions.

6.10.1 Estimating Occlusion Region

For the design of direct-touch interactions, hand and finger occlusion is often a major concern that cannot be entirely avoided. However, based on the position and orientation of the finger touch, we could effectively estimate the hand occlusion region on the fly, and adapt interface layouts to minimize the impact of occlusion.

Considering the anatomy of the human finger and palm, we estimate the occlusion region to be a circular sector opposite to the finger orientation , with the vertex at the center of the finger tip (x, y), and the central angle at approximately $\delta = 60°$(angular value selected from [89]; Fig. 6.10).

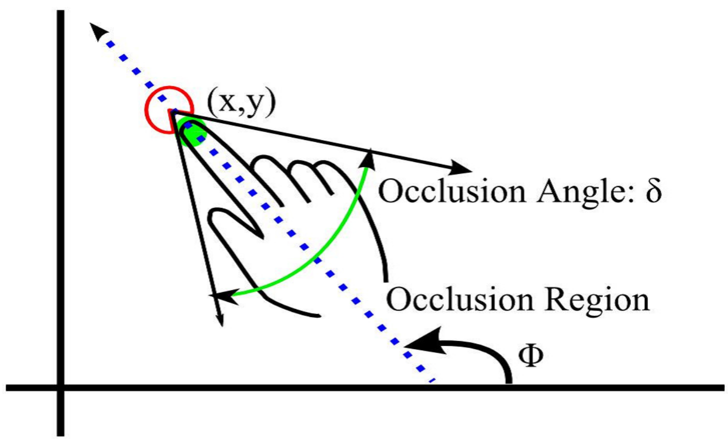

Fig. 6.10 Occlusion region estimation.

With knowledge of the occlusion region, we can dynamically place content and interface elements outside it. In addition, we can design special interface widgets that adapt to accommodate the finger orientation and avoid occlusion. For example, a pie-menu or torus-menu with a gap can dynamically reorient itself so that the gap is always aligned with the body of the finger (see Fig. 6.11) [61]. Brandl et al. [90] explored similar occlusion-aware menu designs, but required the user to use a pen and rest the palm on the surface simultaneously to determine the menu orientation. Comparatively, our finger orientation detection algorithm allows these designs to be broadly applied to various scenarios and technologies.

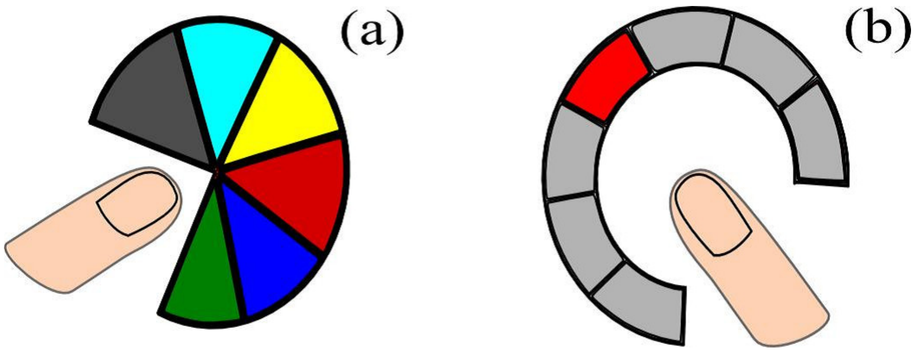

Fig. 6.11 Menus adapting to finger orientation to avoid occlusion. (a) Pie menu. (b) Torus menu.

6.10.2 Inferring User Position

When a user puts a finger on a horizontal surface, typically the finger points away from the user's body (see Fig. 6.13). For tabletop interaction, this provides a simple cue to infer the rough position of the user. Although inherently imprecise, this can be useful enough for simple scenarios where the user can only take a few possible positions. In a typical interactive tabletop usage scenario, the user sits along either one of the two long sides of the tabletop. Knowing the orientation of the finger touch, we can infer that the operating user is sitting at the side opposite to the finger orientation. This information is particularly useful for orienting the interface and content (especially text) to suit the user's perspective.

Another common usage scenario is when two users sit on opposite sides of the tabletop. By applying the same heuristics, we provide a lightweight way of differentiating finger touch inputs from different users without resorting to technologies like the DiamondTouch [18]. Many user-specific operations can then be easily supported, such as the use of interface widgets that function differently depending on who triggers them, or setting different operation privileges for different users (see Fig. 6.13).

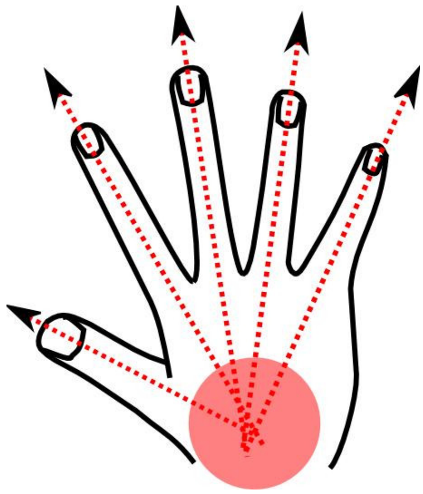

Fig. 6.12 Inferring user position from finger orientation.

6.10.3 Relationship between Multiple Fingers

The anatomy of the human hand imposes certain constraints on the possible orientations and positions of fingers from the same hand. We can exploit this information to infer the relationship between multiple fingers on the surface.

In natural and comfortable positions, the orientation of a finger indicates a departure away from the center of the palm. As a result, the lines of direction based on the orientations of two or more fingers from the same hand will intersect and provide a rough location of the user's palm. This location is usually to the opposite side of the directions pointed by all finger, and within a reasonable distance from the position of the fingertips (see Fig. 6.13).

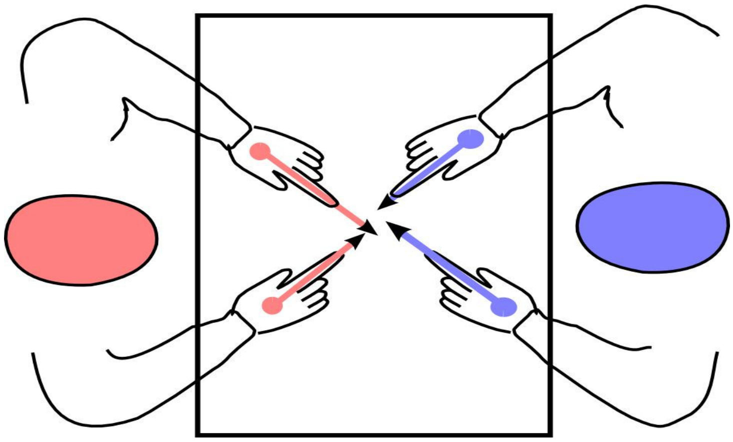

Fig. 6.13 Typical hand configuration.

Based on this information, for a pair of finger touch points, we calculate the intersection point I of the two straight lines aligned with their positions and orientations. For each fingertip position P with orientation Φ, we calculate the orientation angle ΦIP of ray IP (i.e. pointing from I to P). If $|\Phi IP - \Phi| < 90°$ and distance $|IP| < td$ (td is an empirically determined threshold, chosen to be 140 mm in our implementation) for both fingers, we determine that they belong to the same hand (see Fig. 6.14a). Otherwise the two fingers belong to different hands (see Fig. 6.14b). In the latter case we may also infer whether the two hands belong to different users in some simple cases (see Fig. 6.14c). As discussed previously, if assuming that the two users are sitting in fixed positions across a horizontal surface, and the orientations of the two fingers clearly point oppositely to where the users are supposed to be sitting, we can then associate these two hands with each user. When three or more fingers are touching, we may determine their pair-wise relationships, and make higher-level inferences if necessary. Obviously, this approach does not account for atypical cases such as when two hands overlap. However it would be reliable enough for interaction purposes in natural scenarios.

Knowing the relationship between finger touches can be particularly useful for various interactions. Moscovich and Hughes [58] experimentally showed that multi-finger manipulations by one hand and by two hands are suitable for different tasks. Inspired by this, we could assign

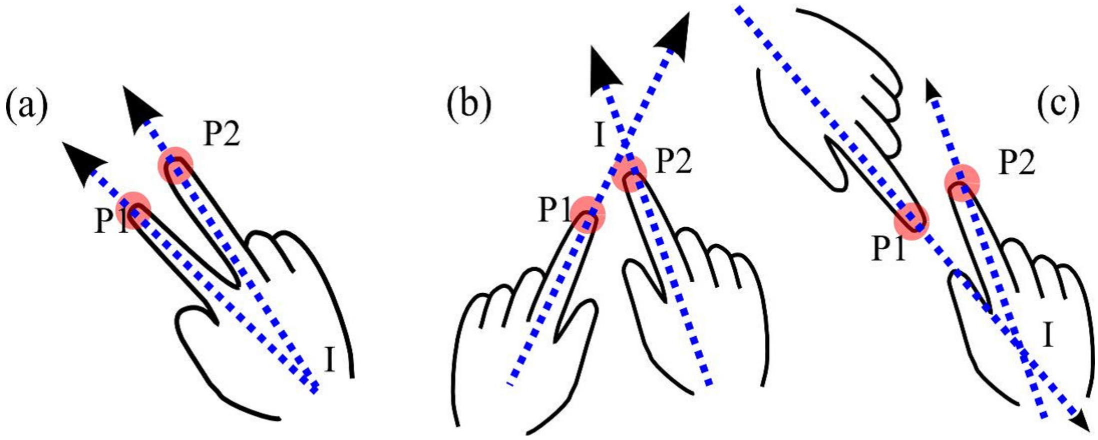

Fig. 6.14 Inferring relationship between fingers. (a) Same hand. (b) Two hands from same user. (c) Two hands from different users.

different functionalities for each case. For example, fingers from the same hand result in multi-finger inking on a digital object; fingers from both hands by the same user result in the classic rotation/scaling manipulation; and fingers from multiple users could “tear apart” the object to create multiple copies.

6.10.4 Enabling Orientation-Invariant Input

The direct-touch input on interactive surfaces naturally affords using the finger to perform trajectory-based gestures, similar to pen gestures that are broadly used on tablet PCs or hand-held devices. Pen gestures performed on those devices usually have an unambiguous upright orientation relative to the input panel. However, for gestures performed on interactive surfaces, especially horizontal tabletops, the orientation of the gesture inputted can be arbitrary depending on the user’s position. This creates a dilemma for unambiguous gesture recognition on interactive tabletops. Either the system has to assume the user is inputting from a fixed orientation, which constrains the usage of the tabletop. Alternately the system has to recognize input in a rotation-invariant way and avoid any orientation-specific gesture, which largely limits the gesture design space. This problem becomes even more prominent if we want to introduce handwriting recognition input on tabletops, since many of the English and numerical characters are inherently orientation-specific (see Fig. 6.15).

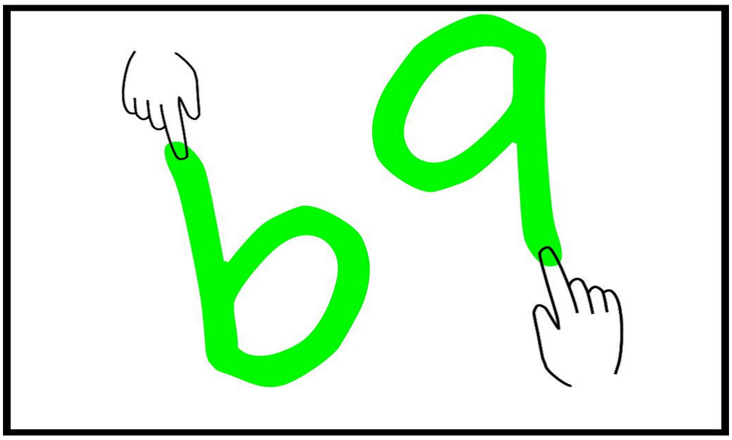

Fig. 6.15 Ambiguous gestures caused by different hand orientations.

By taking finger orientation into account, this problem could be alleviated. Before recognition, the orientation of the input gesture can be normalized by a compensated rotation determined by the average finger orientation while performing the gesture. As a result, the user could perform finger gestures or handwrite unambiguously from any side of the tabletop. Blasko et al. [91] explored similar concepts on a handheld tablet by estimating its own orientation through face tracking or stylus pose.

Another example of orientation-invariant input is to support multi-finger mouse emulation on interactive tabletops. Matejka et al. [73] presented SDMouse, an efficient technique to simulate full mouse functionality by mapping different buttons to different fingers on a multi-touch screen. The technique differentiates “mouse buttons” partly by their directional position with reference to the index finger. For example, the finger on the left side of the index finger is considered the thumb and mapped to the left button. This poses a problem for migrating SDMouse onto a tabletop surface, where the definition of “a side” is ambiguous and varies with the hand orientation. Again, by considering the orientation of the index finger, we can unambiguously associate fingers to buttons located in a reachable location regardless of the user's position (see Fig. 6.16).

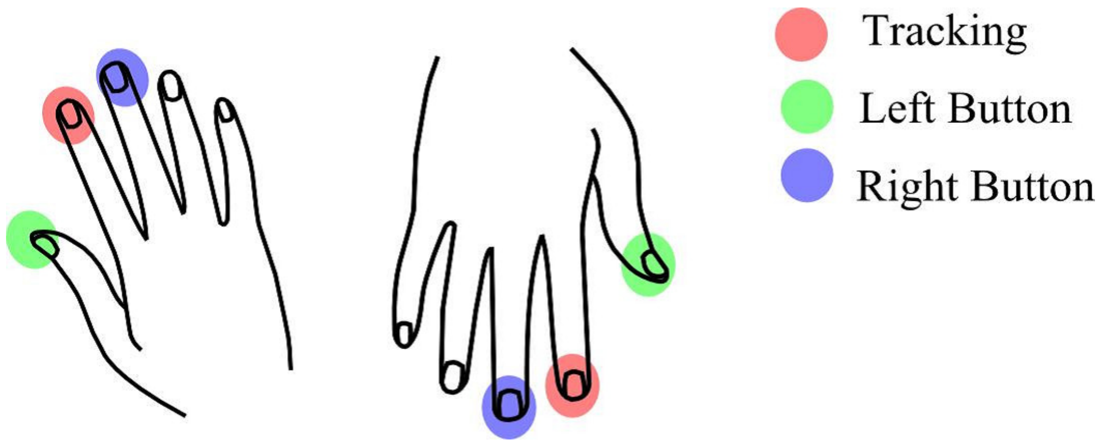

Fig. 6.16 Orientation-invariant mouse emulation.

6.11 Discussion

6.11.1 Algorithm Limitations

We have shown that our finger detection algorithm is effective and accurate. However a few limitations do exist: The algorithm assumes an oblique touch from the user, which is the case for most common interaction scenarios. However, as Forlines et al. [45] discussed, such an oblique touch may be less obvious when the user touches a vertical surface, or in areas of a tabletop that are very close to themselves. For these scenarios, the interaction designs would need to afford supplemental ways for the user to indicate the concept of orientation.

The orientation disambiguation step relies on the finger center displacement during the finger landing process and assumes that this displacement is caused solely by the deformation of the finger. This requires that the landing action should consist of a downward vertical movement of the finger only, which is typically true in regular cases. In the less frequent case that the finger landing is accompanied by a concurrent horizontal finger movement (i.e. "sliding down"), the finger disambiguation algorithm could be biased. However, the concurrent horizontal movement during the short period of finger landing is unlikely to be significant enough to reverse the disambiguation result.

Although our experimental evaluation only examined the algorithm performance with the index finger, in practice we observed that the algorithm also works well with most other fingers, with the only exception being the thumb. Restricted by the anatomy of the human hand, a user would typically touch the surface with the side face of the thumb, instead of its pad. This does not display the usual center displacement pattern as with other fingers, and usually results in an incorrect detection by our algorithm. On the other hand, when users do touch the surface with the pad of the thumb, they usually do so with the thumb pointing towards themselves, as opposed to it being away from themselves as with other fingers. Our algorithm can correctly detect the thumb orientation in this case. However this action is less informative for inferring the user's position.

It should be noted that using finger orientation alone is not the final answer for novel interaction designs on interactive surfaces. By combining finger orientation with other input properties of the hand such as size, shape, or pressure, the limitations of our algorithms would be overcome in interaction scenarios.

6.11.2 Technology Compatibility

We have tested our algorithm on a representative computer vision based technology. Our algorithm requires solely the contact shape of the finger to work. This input requirement makes our algorithm largely compatible with a variety of other sensing technologies, such as capacity-based sensing or embedded optical sensor arrays [14]. However given the nature of the different technologies, the parameters of the algorithm may need to be adjusted accordingly. For example, FTIR-based devices often require the user to press slightly harder to generate enough touch area for processing. On the other hand, technologies that provide certain additional information such as hover state, as in ThinSight [14], can utilize these to further improve the reliability of our algorithm.

6.11.3 3D Finger Orientation

In this chapter, we focused on the 2D orientation of the finger. However, 3D finger orientation may also be interesting for interacting with digital surfaces. This information could potentially be acquired by using new sensing technologies such as depth cameras (www.3dvsystems.com). For example, with knowledge of the full 3D orientation of the contact finger, we could enable tilt-based interactions such as those explored with the TiltMenu [92] for pen-based interactions. Alternatively, intuitive 3D manipulations of digital objects may be explored on interactive surfaces by using the finger as a proxy.

Chapter 7

Future Research Directions

7.1 Principles and Fundamental Investigation

This thesis investigates human ability of using input properties for improving interactive performance and decreasing interactive difficulties. Future intended work on this subject mainly includes the evaluation of newly proposed designs. Their impact on reducing interference issues should be carefully evaluated, as well as evaluating the discrete control level of these input properties especially the finger contact area and the orientation. We will also investigate the effects of using direct multi-touch devices and indirect multi-touch devices and we will assess how they differ from our current results and observations.

However, there are several directions that can be pursued to extend the current work. As stated in the literature review, there is a lack of effective assessment methods. Current GOMS and other assessment methods cannot describe the natural feeling of users. Meanwhile, this thesis examines basic issues on finger's area, shape and orientation. Actually, hand is a integral input device, we should consider all the combination of five fingers and palm. Considering the finger's touch and contextual multi-channel information, we can obtain more information from the actions of users. Integrating all the possible information, we can design more natural user interface for users.

In addition, due to the quick development of mobile systems, the study of the human natural input properties for mobile systems is a new challenge to researchers. Concerning the small-size touch screen and low computing performance, we find this is a big-cake left for future work. The future work also includes investigations on the combinations of all the input properties. This thesis proposes a series of direct-touch interaction applications and implications. In the future we shall seek to couple these new ideas with new-generation mobile systems such as table-let computers and e-ink readers. It will significantly improve the interactive experience between human and computers.

7.2 Orientation and Detection Technique

Several open questions remain to be explored in the future:

First, we would like to further improve our algorithm to overcome the limitations discussed previously, for example by investigating the detailed geometry of the fingers' touch. This would be led by a deeper investigation of the properties of finger contact, including less typical scenarios such as when the user touches with the side of a finger. Particular attention would be given to the thumb, which has several unique properties compared to other fingers. This is especially important as some multi-touch manipulations often involve the movement of both thumb and index finger.

Additionally, based on the inference made from the structural interrelationship of the fingers, we are interested in experimenting with extracting higher-level information to cluster touch points into congruent hand configurations. Continuous tracking of the user's full hands may also be made possible by considering the dynamics of the fingers even when they are not always touching the surface.

At present, most of our designs have been implemented as proof-of-concepts, while "aim and grab" and "orientation-sensitive widgets" are in design stage. In the future we will iterate on these prototypes to improve the design details. We have not yet implemented an orientation-invariant gesture/handwriting recognition engine as we proposed. We plan to develop and experimentally evaluate such an engine, and also explore its applications in other scenarios such as using handheld devices.

7.3 Feedback and Reality Experience

Although the drawbacks of direct-touch devices have been discussed by many previous studies, the effective feedback approaches and modes have not been studied in depth. There are no formal definitions or conclusions on what is the best feedback mode in direct-touch systems. Many commercial direct-touch devices have provided some interesting feedback modes already. For example, Android ($http://www.google.com$) operation system can support tactile feedback. When a user touches on the panel, the mobile phone with Android system will generate a tremor to notice the user that the touch manipulation is accepted by the system. In traditional direct-touch devices, visual feedbacks are typically provided. In common, the background color of targets will change while users are touching. However, the high error-rate of tapping gesture determines that the current feedback modes cannot meet the needs of the users.

In spite of the existence of these feedback studies, only a few studies discuss the reality experience on direct-touch devices. No matter what feedback modes are provided, users always complain that the touch feeling on a flat panel is lack of reality experience because all the targets displayed on the surface are actual 2D images. Some studies have presented some new techniques in which a real 3D object can be displayed on the touch panel. These new techniques can conspicuously improve the reality experience in touching. However, the expensive price limits the application of the techniques. It is worth to have a deep study on a proper feedback modes with more real experience in common direct-touch devices.

Chapter 8

Conclusions

8.1 Summary of Contributions

8.1.1 Hardware and low-level algorithm

Multi-touch is a novel human computer interactive technique. The high resolution and large screen multi-touch devices are useful for human's daily life. For example, we urgently need interactive white board system for education. However, the cost of the device should be considered in the implementation of the large screen multi-touch devices. Although many commercial products are invented, how to design and implement low cost large screen multi-touch device, is still worth to study.

In the dissertation, refer to the current FTIR/DI devices, we implement a computer vision based multi-touch device. It is important that we present a practical algorithm for quick location calculation of touch points. A contour-based blob recognition technique recognizes and tracks corresponding blobs robustly and quickly. The performance of recognition and tracking fulfills the requirements of the real time multi-touch system. In addition, we present a robust technique for multi panel splicing. The experimental results prove that the algorithms and techniques are effective.

With the support of this device, we empirically evaluate the potential input properties of human finger. Based on the results of our experiment, the shape of the finger contact areas, the size of the contact area and the orientation of the contact finger are effective finger properties that are useful for the design of natural multi-touch gestures.

8.1.2 Input Properties Evaluation

With the support of a self-made FTIR based multi-touch device, we implement a series of experiments in order to investigate readily available human finger properties. Our results indicate that the five fingers of one hand have different abilities and potentials for target selection. The target selection precision offered by the index finger, the middle finger and the ring finger are better than the thumb and the little finger. Based on the results of our experiment, the shape of the finger contact area, the size of the contact area and the orientation of the contact finger are effective finger properties that are useful for the design of natural multi-touch gestures. This form of detailed analysis is appealing to the multi-touch community as it can benefit from this work.

8.1.3 Orientation and Natural User Interface

Our contribution in orientation and NUI is two-fold. We first presented a simple and generally applicable algorithm to unambiguously detect the directed orientation of user's fingers on interactive surfaces from contact information only. Researchers can apply this algorithm on various direct-touch surfaces to serve their own purposes. We then explored user interface designs that leverage this finger orientation information, as well as further inferences that can be made from finger orientations. These designs and inferences can be useful for interaction with a variety of direct-touch devices that generate finger orientation information, either using our general algorithm or other more specialized sensing technologies. Our work shows that finger orientation is a feasible and valuable input dimension that can be utilized for novel interactions on interactive surfaces.

8.2 Limitations and Open Issues

Several questions regarding our studies and bi-manual interaction in general remain unanswered. We investigate the tapping performance of each finger. It is not clear if these performances can make additional conclusions on user interface design because of the lack of effective evaluation method. Another outstanding issue is the usability of the orientation. In current mobile devices, none can output information of finger's orientation. We have no way to effectively research the fingers' orientation and the application in mobile systems.

8.3 Closing Remarks

Obviously, natural language is the main source of natural user interface. We would like to communicate with computers with spoken language. However, when confronted the reality, we still need our hands for communication. The role of the computer has shifted from number-crunching machine to communication device, but the role of the hand movements in communicating information to a computer and in computer-mediated communication, has largely been limited to that of a single pointing finger [70].

This dissertation has established an valuable point that the exploitation of finger's properties allows users to communicate information to a computer faster and more smoothly than single-point interaction techniques. We designed and implemented a series of experiment to confirm the fact that many natural finger properties can provide new guidelines for future development. New generation user interface can fully use the natural properties to answer the challenges that lie beyond the realm of today's mouse-and-keyboard paradigm. We believe that the usage of natural input properties of human body would bring a new world to future interface design.

Acknowledgement

I want to thank the people who have guided me along the Ph.D. path, in particular, my advisor, Prof. Xiangshi Ren, for his generous support, invaluable advice, and kind assistance he provided since I first arrived at Kochi. He taught me how to ask questions and speak out my ideas. He showed me different ways to tackle a research problem and teach me the importance of being persistent to accomplish any goal. His dedication to work, perfectionism, humor and decency will have a far reaching impact on my life, academic and beyond. Many thanks, my dear professor.

I would like to extend my appreciation to the reviewers of this dissertation: To Dr. Xiang Cao for his invaluable help in providing the initial push that brought this work from an idea to a reality. I am also very grateful to Professor Akihiro Shimizu, Professor Takata Yoshiaki, Professor Shinichi Yoshida and Professor Yukinobu Hoshino for their constructive comments on my thesis.

Meanwhile, I would like to thank my family members, particularly my wife, Mrs. Hui Deng, for their generous love and support to my life all along.

I'd like to thank members of the Ren lab, particularly Tomoki Ooya, Fukutoku Fumiya, Chuanyi Liu, Xiaolei Zhou , Yizhong Xin and Minghui Sun for their friendship and contributions to my work.

I have received support in the form of scholarships, facilities or travel grant from TAF (Telecommunications Advancement Foundation), IISF (International Information Science Foundation) and Kochi University of Technology. This research would have been impossible without these generous supports.

This is a blank page.

References

[1] P.-A. Albinsson and S. Zhai, "High precision touch screen interaction," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Ft. Lauderdale, Florida, USA), pp. 105–112, ACM, 2003.

[2] W. MathWorld, "Barycentric coordinates." http://mathworld.wolfram.com/BarycentricCoordinates.html, 2009.

[3] J. Canny, "The future of human-computer interaction," *Queue*, vol. 4, no. 6, pp. 24–32, 2006.

[4] B. Myers, "A brief history of human-computer interaction technology," *interactions*, vol. 5, no. 2, pp. 44–54, 1998.

[5] B. A. Myers, "User interface software technology," *ACM Comput. Surv.*, vol. 28, no. 1, pp. 189–191, 1996.

[6] B. Shneiderman, "Leonardo's laptop: human needs and the new computing technologies," 2005.

[7] A. Cooper, R. Reimann, and D. Cronin, *About face 3: the essentials of interaction design*. Wiley-India, 2007.

[8] N. Cross, "Natural intelligence in design," *Design Studies*, vol. 20, no. 1, pp. 25–39, 1999.

[9] C. Nass and B. Reeves, "Social and natural interfaces: theory and design," in *CHI '97 extended abstracts on Human factors in computing systems: looking to the future*, (Atlanta, Georgia), pp. 192–193, ACM, 1997.

[10] R. Anderson, "Social impacts of computing: Codes of professional ethics," *Social Science Computer Review*, vol. 10, no. 4, p. 453, 1992.

[11] B. Buxton, "Multi-touch systems that i have known and loved," *Microsoft Research*, 2007.

[12] 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, (Seattle, WA, USA), pp. 115–118, ACM, 2005.

[13] J. Y. Han, "Multi-touch interaction wall," in *ACM SIGGRAPH 2006 Emerging technologies*, (Boston, Massachusetts), p. 25, ACM, 2006.

[14] 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*, (San Diego, California), p. 6, ACM, 2007.

[15] D. Wigdor, C. Forlines, P. Baudisch, J. Barnwell, and C. Shen, "Lucid touch: a see-through mobile device," in *Proceedings of the 20th annual ACM symposium on User interface software and technology*, (Newport, Rhode Island, USA), pp. 269–278, ACM, 2007.

[16] A. Butler, S. Izadi, and S. Hodges, "Sidesight: multi-"touch" interaction around small devices," in *Proceedings of the 21st annual ACM symposium on User interface software and technology*, (Monterey, CA, USA), pp. 201–204, ACM, 2008.

[17] SMART, "Smart technologies inc.." http://www.smarttech.com/, 2010.

[18] 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*, (Orlando, Florida), pp. 219–226, ACM, 2001.

[19] 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*, (Minneapolis, Minnesota, USA), pp. 113–120, ACM, 2002.

[20] S. Lee, W. Buxton, and K. C. Smith, "A multi-touch three dimensional touch-sensitive tablet," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (San Francisco, California, United States), pp. 21–25, ACM, 1985.

[21] S. Do-Lenh, F. Kaplan, A. Sharma, and P. Dillenbourg, "Multi-finger interactions with

papers on augmented tabletops," in *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, pp. 267–274, ACM, 2009.

[22] X. Cao, A. D. Wilson, R. Balakrishnan, K. Hinckley, and S. Hudson, "Shapetouch: Leveraging contact shape on interactive surfaces," in *Horizontal Interactive Human Computer Systems, 2008. TABLETOP 2008. 3rd IEEE International Workshop*, (Amsterdam), pp. 129 – 136, IEEE, 2008.

[23] M. W. Krueger, T. Gionfriddo, and K. Hinrichsen, "Videoplace—an artificial reality," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (San Francisco, California, United States), pp. 35–40, ACM, 1985.

[24] A. D. Wilson, "Playanywhere: a compact interactive tabletop projection-vision system," 2005.

[25] Microsoft, "Microsoft surface." http://www.surface.com, 2009.

[26] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer, "Visual panel: virtual mouse, keyboard and 3d controller with an ordinary piece of paper," in *Proceedings of the 2001 workshop on Perceptive user interfaces*, (Orlando, Florida), pp. 1–8, ACM, 2001.

[27] P. Wellner, "Interacting with paper on the digitaldesk," *Commun. ACM*, vol. 36, no. 7, pp. 87–96, 1993.

[28] N. Matsushita and J. Rekimoto, "Holowall: designing a finger, hand, body, and object sensitive wall," p. 210, ACM, 1997.

[29] A. D. Wilson, "Touchlight: an imaging touch screen and display for gesture-based interaction," in *Proceedings of the 6th international conference on Multimodal interfaces*, (State College, PA, USA), pp. 69–76, ACM, 2004.

[30] C. Pinhanez, R. Kjeldsen, L. Tang, A. Levas, M. Podlaseck, N. Sukaviriya, and G. Pingali, "Creating touch-screens anywhere with interactive projected displays," in *International Conference Multimedia*, pp. 460–461, ACM New York, NY, USA, 2003.

[31] A. Agarwal, S. Izadi, M. Chandraker, and A. Blake, "High precision multi-touch sensing

on surfaces using overhead cameras," (TABLETOP'07), IEEE, 2007.

[32] Nuigroup, "Touchlib." http://www.nuigroup.com/, 2009.

[33] J. Kim, J. Park, H. Kim, and C. Lee, "Hci (human computer interaction) using multi-touch tabletop display," in *Communications, Computers and Signal Processing*, (PacRim), p. 391C394, IEEE, 2007.

[34] W. Buxton, R. Hill, and P. Rowley, "Issues and techniques in touch-sensitive tablet input," *SIGGRAPH Comput. Graph.*, vol. 19, no. 3, pp. 215–224, 1985.

[35] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza, "Tuio: A protokol for table-top tangible user interfaces," in *Proceedings of Gesture Workshop 2005*, Gesture Workshop, 2005.

[36] 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*, (Montréal, Québec, Canada), pp. 1263–1272, ACM, 2006.

[37] A. D. Wilson, S. Izadi, O. Hilliges, A. Garcia-Mendoza, and D. Kirk, "Bringing physics to the surface," in *Proceedings of the 21st annual ACM symposium on User interface software and technology*, (Monterey, CA, USA), pp. 67–76, ACM, 2008.

[38] G. Ramos, M. Boulos, and R. Balakrishnan, "Pressure widgets," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Vienna, Austria), pp. 487–494, ACM, 2004.

[39] A. Reetz, C. Gutwin, T. Stach, M. Nacenta, and S. Subramanian, "Superflick: a natural and efficient technique for long-distance object placement on digital tables," 2006.

[40] 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*, (Washington, D.C., United States), pp. 27–32, ACM, 1988.

[41] 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*, (Vancouver, Canada), pp. 193–202, ACM, 2003.

[42] A. Hall, J. Cunningham, R. Roache, and J. Cox, "Factors affecting performance using touch-entry systems: tactual recognition fields and system accuracy," *Journal of applied psychology*, vol. 73, no. 4, pp. 711–720, 1988.

[43] D. Vogel and P. Baudisch, "Shift: a technique for operating pen-based interfaces using touch," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (San Jose, California, USA), pp. 657–666, ACM, 2007.

[44] C. Herot and G. Weinzapfel, "One-point touch input of vector information for computer displays," *ACM SIGGRAPH Computer Graphics*, vol. 12, no. 3, pp. 210–216, 1978.

[45] 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*, (San Jose, California, USA), pp. 647–656, ACM, 2007.

[46] W. Buxton and B. A. Myers, "A study in two-handed input," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Boston, Massachusetts, United States), pp. 321–326, ACM, 1986.

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

[48] 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*, (Anaheim, CA), pp. 73–80, ACM, 1993.

[49] C. D. Shaw and M. Green, "Thred: a two-handed design system," *Multimedia Systems*, vol. 5, no. 2, pp. 126–139, 1997.

[50] 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, (Denver, Colorado, United States), pp. 442–449, ACM Press/Addison-Wesley Publishing Co., 1995.

[51] Y. Guiard, “Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model,” *Journal of Motor Behavior*, vol. 19, pp. 486–517, 1987.

[52] D. Casalta, Y. Guiard, and M. B. Lafon, “Evaluating two-handed input techniques: rectangle editing and navigation,” 1999.

[53] K. Hinckley, R. Pausch, D. Proffitt, J. Patten, and N. Kassell, “Cooperative bimanual action,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Atlanta, Georgia, United States), pp. 27–34, ACM, 1997.

[54] C. Latulipe, C. S. Kaplan, and C. L. A. Clarke, “Bimanual and unimanual image alignment: an evaluation of mouse-based techniques,” in *Proceedings of the 18th annual ACM symposium on User interface software and technology*, (Seattle, WA, USA), pp. 123–131, ACM, 2005.

[55] C. Latulipe, S. Mann, C. S. Kaplan, and C. L. A. Clarke, “symspline: symmetric two-handed spline manipulation,” in *Proceedings of the SIGCHI conference on Human Factors in computing systems*, (Montréal, Québec, Canada), pp. 349–358, ACM, 2006.

[56] A. Sears and B. Shneiderman, “High precision touchscreens: Design strategies and comparisons with a mouse,” *International Journal of Man-Machine Studies*, vol. 34, no. 4, pp. 593–613, 1991.

[57] K. Kin, M. Agrawala, and T. DeRose, “Determining the benefits of direct-touch, bimanual, and multifinger input on a multitouch workstation,” in *Proceedings of Graphics Interface 2009*, (Kelowna, British Columbia, Canada), pp. 119–124, Canadian Information Processing Society, 2009.

[58] 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, (Florence, Italy), pp. 1275–1284, ACM, 2008.

[59] S. Malik, *An exploration of multi-finger interaction on multi-touch surfaces*. PhD thesis, 2007.

[60] M. Hancock, F. Vernier, D. Wigdor, S. Carpendale, and C. Shen, "Rotation and translation mechanisms for tabletop interaction," p. 8, IEEE, 2006.

[61] 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*, (Boston, MA, USA), pp. 1063–1072, ACM, 2009.

[62] J. Elias, W. Westerman, and M. Haggerty, "Multi-touch gesture dictionary," 2007.

[63] P. Kabbash and W. A. S. Buxton, "The "prince" technique: Fitts' law and selection using area cursors," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Denver, Colorado, United States), pp. 273–279, ACM Press/Addison-Wesley Publishing Co., 1995.

[64] A. Worden, N. Walker, K. Bharat, and S. Hudson, "Making computers easier for older adults to use: area cursors and sticky icons," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Atlanta, Georgia, United States), pp. 266–271, ACM, 1997.

[65] A. Olwal and S. Feiner, "Rubbing the fisheye: precise touch-screen interaction with gestures and fisheye views," in *Supplyment of UIST2003*, pp. 83–84, 2003.

[66] P. Baudisch, E. Cutrell, K. Hinckley, and A. Eversole, "Snap-and-go: helping users align objects without the modality of traditional snapping," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Portland, Oregon, USA), pp. 301–310, ACM, 2005.

[67] R. Blanch, Y. Guiard, and M. Beaudouin-Lafon, "Semantic pointing: improving target acquisition with control-display ratio adaptation," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Vienna, Austria), pp. 519–526, ACM, 2004.

[68] A. Olwal, S. Feiner, and S. Heyman, "Rubbing and tapping for precise and rapid selection on touch-screen displays," in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, (Florence, Italy), pp. 295–304, ACM, 2008.

[69] T. Moscovich and J. F. Hughes, "Multi-finger cursor techniques," in *Proceedings of Graphics Interface 2006*, (Quebec, Canada), pp. 1–7, Canadian Information Processing Society, 2006.

[70] T. Moscovich, *Principles and Applications of Multi-touch Interaction*. PhD thesis, Brown University, 2007.

[71] A. Esenther, C. Forlines, K. Ryall, and S. Shipman, "Support for multi-user, multi-touch applications," tech. rep., MITSUBISHI ELECTRIC RESEARCH LABORATORIES, 2002.

[72] A. Esenther and K. Ryall, "Fluid dtmouse: better mouse support for touch-based interactions," in *Proceedings of the working conference on Advanced visual interfaces*, (Venezia, Italy), pp. 112–115, ACM, 2006.

[73] J. Matejka, T. Grossman, J. Lo, and G. Fitzmaurice, "The design and evaluation of multi-finger mouse emulation techniques," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 1073–1082, ACM, 2009.

[74] 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*, (Monterey, CA, USA), pp. 87–90, ACM, 2008.

[75] S. Malik and J. Laszlo, "Visual touchpad: a two-handed gestural input device," in *Proceedings of the 6th international conference on Multimodal interfaces*, (State College, PA, USA), pp. 289–296, ACM, 2004.

[76] G. Bradski and A. Kaebler, *Learning OpenCV*. O'REILLY, 2008.

[77] A. Patwardhan, "Subpixel position measurement using 1d, 2d and 3d centroid algorithms with emphasis on applications in confocal microscopy," *Journal of Microscopy*, vol. 186,

no. 3, pp. 246–257, 1997.

[78] B. Shneiderman, “Direct manipulation: A step beyond programming languages (abstract only),” *ACM SIGSOC Bulletin*, vol. 13, no. 2-3, p. 143, 1982.

[79] P. Fitts, “The information capacity of the human motor system in controlling the amplitude of movement,” *Journal of Experimental Psychology*, vol. 47, pp. 381–391, 1954.

[80] I. S. MacKenzie, “Fitts’ law as a research and design tool in human-computer interaction,” *Human-Computer Interaction*, vol. 7, no. 1, pp. 91 – 139, 1992.

[81] S. Card, W. English, and B. Burr, “Evaluation of mouse, rate-controlled isometric joystick, step keys, and text keys for text selection on a crt,” *Ergonomics*, vol. 21, no. 8, pp. 601–613, 1978.

[82] S. Zhai, “Characterizing computer input with fitts’ law parameters–the information and non-information aspects of pointing,” *International Journal of Human-Computer Studies*, vol. 61, no. 6, pp. 791–809, 2004.

[83] R. Soukoreff and I. MacKenzie, “Towards a standard for pointing device evaluation, perspectives on 27 years of fitts’ law research in hci,” *International Journal of Human-Computer Studies*, vol. 61, no. 6, pp. 751–789, 2004.

[84] S. Douglas, A. Kirkpatrick, and I. MacKenzie, “Testing pointing device performance and user assessment with the iso 9241, part 9 standard,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, p. 222, ACM, 1999.

[85] J. Accot and S. Zhai, “Beyond fitts’ law: models for trajectory-based hci tasks,” in *Proceedings of the SIGCHI conference on Human factors in computing systems*, (Atlanta, Georgia, United States), pp. 295–302, ACM, 1997.

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

[87] 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*, (Portland, Oregon, USA), pp. 281–290, ACM, 2005.

[88] T. Grossman and R. Balakrishnan, "The design and evaluation of selection techniques for 3d volumetric displays," in *Proceedings of the 19th annual ACM symposium on User interface software and technology*, (Montreux, Switzerland), ACM, 2006.

[89] G. Blasko, W. Beaver, M. Kamvar, and S. Feiner, "Workplane-orientationcsensing techniques for tablet pcs," in *Proceedings of the 17th annual ACM symposium on User interface software and technology, Conference Supplement*, 2004.

[90] P. Brandl, T. Seifried, J. Leitner, M. Haller, B. Doray, and P. To, "Occlusion-aware menu design for digital tabletops," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 201–204, ACM, 2009.

[91] G. Blasko and S. Feiner, "Single-handed interaction techniques for multiple pressure-sensitive strips," in *CHI '04 extended abstracts on Human factors in computing systems*, (Vienna, Austria), pp. 1461–1464, ACM, 2004.

[92] F. Tian, L. Xu, H. Wang, X. Zhang, Y. Liu, V. Setlur, and G. Dai, "Tilt menu: using the 3d orientation information of pen devices to extend the selection capability of pen-based user interfaces," in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, (Florence, Italy), pp. 1371–1380, ACM, 2008.

Appendix A

Publications

A.1 Articles in or submitted refereed journals

1. F. Wang and X. Ren, “A survey of human computer interaction technology for disabled persons” in *International Journal of Innovative Computing, Information and Control*, Vol.6, No.6, 2010, pp. 2459-2468

2. F. Wang and X. Ren , “An Investigation of Human Computer Interaction Models for the Disabled” in *Information-An International Interdisciplinary Journal*, 2009, Vol 12, No. 3, pp. 585-591

A.2 Articles in full paper refereed international conference proceedings

3. F. Wang and X. Ren, “Empirical Evaluation for Finger Input Properties In Multi-touch Interaction”, *ACM, 27th Conference on Human Factors in Computing Systems*. ACM SIGCHI, 2009, pp. 1063-1072, Acceptance rate: 25.1%

4. F. Wang, X. Cao, X. Ren, P. Irani, “Detecting and Leveraging Finger Orientation for Interaction with Direct-Touch Surfaces”, *22nd ACM Symposium on User Interface Software and Technology, 2009, ACM SIGCHI and ACM SIGGRAPH*, pp. 23-32, Acceptance rate: 17.4%

5. F. Wang, “Leveraging Fingers for Natural Interaction with Direct-Touch Surfaces”, *Doctoral Symposium, 22nd ACM Symposium on User Interface Software and Technology, ACM*

SIGCHI +SIGGRAPH, Accpetance rate: 7.3%

6. F. Wang, X. Ren, Z. Liu, “A Robust Blob Recognition And Tracking Method In Vision-Based Multi-Touch Technique”, *International Conference on Intelligent Pervasive Computing (IPC-08)*, 2008, IEEE, pp. 634-637

A.3 Articles in abstract refereed international conference proceedings

7. F. Wang, “A widget design and an empirical evaluation for fundamental human finger factors in touch technique”, *International Conference on Next Era Information Networking*, 2008

A.4 Articles in refereed local conference proceedings

8. F. Wang, “High performance image processing implementation in vision-based multi-touch technique”, *Shikoku-section Joint Convention of the Institutes of Electrical and related Engineers*, 2008