Intuitive Interaction Techniques for Mobile Devices with Human Gestures

Department of Frontier Informatics,
Graduate School of Frontier Sciences

47-36333　Koji Yatani

Adviser: Prof. Masanori Sugimoto

31 January, 2005

ジェスチャを用いた携帯端末のための直感的なインタラクション技術に関する研究

新領域創成科学研究科 基盤情報学専攻

47-36333　矢谷 浩司

指導教官: 杉本 雅則助教授

2005年1月31日

DEDICATION

To my family and friends

ABSTRACT

In recent years, mobile devices have rapidly penetrated into our daily lives. Several drawbacks of mobile devices have been mentioned so far, such as their limited computational capability, small screen real estate, and, so on, as compared with notebook or desktop computers. However, if we focus on the most powerful feature of mobile devices, namely mobility, we can explore possibilities for a new user interface of the devices. In this paper, we use PDAs and propose intuitive techniques of information transfer for mobile devices, which could not otherwize be achieved with notebook or desktop computers. Our system, Toss-It, enables a user to send information from the user's PDA to other electronic devices with a "toss" or "swing" action, as the user would toss a ball or deal cards to others. Implementation of Toss-It largely consists of three parts - gesture recognition, location recognition, and file transfer system. In this thesis, we describe these three parts of our implementation, and finally evaluate and affirm practicability and usability of Toss-It.

概　要

携帯電話やPDAに代表されるような、モバイルデバイスが日常の生活に広く普及してきているが、そのインタフェースに関していくつかの問題を抱えている。しかしながら、モバイルデバイス特有の携帯性を生かすことで新たなインタフェースを提案できると考えられる。本研究では、モバイルデバイスとしてPDAを用い、モバイルデバイスにおける情報の移動を直感的に実現する、Toss-Itというインタフェースを提案する。Toss-Itによって、ユーザはボールをトスするように、あるいはカードを配るようにデジタルな情報の移動を実現できる。本論文ではToss-Itの実装方法を、動作の認識、位置の認識、ファイル転送システムの3つに分けて述べる。さらに、認識手法の評価実験、ユーザスタディによる評価実験から、Toss-Itの実現可能性や有用性に関して述べる。

ACKNOWLEDGEMENTS

I humbly express my utmost gratitude to my supervisor, Prof. Masanori Sugimoto, who gave me many helpful suggestions over this research project. He also helped me write academic papers and supported my research activities. I also greatly appreciate Prof. Hiromichi Hashizume of National Institute of Informatics who supported the development of the hardware for gesture recognition and taught me many things on computer science.

I appreciate all the students that contributed to this research project as the project members. Toshiya Kishimura contributed to the development of the algorithms for gesture recognition and the file transfer system. Koichi Tamura was committed to the development of the hardware for gesture recognition. Keiichi Hiroki supported the user study experiment by coordinating the programs in Toss-It. Ayumu Kaneko and Yusuke Sugano were devoted to developing a novel location recognition system for Toss-It.

I show my gratitude to Kosuke Miyaji and Makoto Miwa, who are among my best friends together with whom we often discuss about our research. I am also grateful to Sunil Kandel, who kindly volunteered to proofread this thesis. I give my thanks to my colleagues in Sugimoto laboratory - Kazuhiro Hosoi, Sosuke Miura, Kosuke Miyahara and Yuichiro Takeuchi. I thank all the students that cooperated with the experiments and supported my making demo videos of Toss-It.

Finally, I appreciate my family, especially my father and my mother, who supported my school life.

Contents

List of Figures

List of Tables

Chapter 1

OVERVIEW

This first chapter describes the introduction of my research project that is presented in this thesis. The chapter then presents the organization of the thesis itself and describes the contributions that the thesis makes to the field of Human-Computer Interaction.

1.1 Introduction

In recent years, mobile devices, such as cellular phones or personal digital assistants (PDAs), have rapidly penetrated into our daily lives. Many people, from children to elderly people, use mobile devices in various situations. We can argue that mobile devices have made a significant contribution to the popularization of computers.

As mobile devices have become popular, several problems regarding user interfaces of mobile devices have also surfaced. For example, character input methods which are implemented in current mobile devices are not sufficiently acceptable. Display is another problem. Even though many rich contents, such as high-resolution images or videos, are accessible, mobile devices do not have enough large displays to show these rich contents. Therefore, many research projects on user interfaces of mobile devices have been conducted in order to solve these problems (MacKenzie and Soukoreff, 2002; Yee, 2003).

Among these problems, information transfer in mobile devices is one of the significant problems to be solved. Suppose you want to copy a file in your mobile device to others' mobile devices or electronic devices around you. Although a memory card or an infrared communication is available as information transfer methods, these methods require several steps to complete a task, for example, (1) copy a file to a memory card, (2) move close to a person, (3) remove the card from your device, (4) insert the card into the person's device, (5) copy the file to the person's device, and (6) remove the card and return it to your device. When you want to pass a file to several colleagues, you have to conduct the same procedures repeatedly. In another case, if you want to print out a photo in your mobile device through a printer in front of you, you have to conduct frustrating operations on its graphical user interface (GUI), such as selecting menu items with a stylus pen several times, in order to specify the printer.

On the other hand, when you pass something to another person around you in the real world, all you have to do is just toss it toward the person. If you could send digital information from your mobile device to other devices as you would pass physical objects to others, you will be liberated from bothersome and awkward operations on your device.

In such a context, we propose a system called Toss-It that enables users to transfer information in their mobile devices in an intuitive manner, by utilizing their mobility. In Toss-It, users can transfer digital information by conducting "toss" or "swing" actions with their mobile devices, as they would toss a ball or deal cards to others.

In this thesis, we present the system implementation of Toss-It in three parts - gesture recognition, location recognition, and file transfer system. We then describe the experiments to evaluate the developed technologies. A user study experiment of Toss-It shows the practicability and usability of Toss-It.

1.2 Organization

This thesis is organized as following.

- Chapter 2 describes a review of previous works related to Toss-It. It also clarifies the differences between Toss-It and other works.

- Chapter 3 delivers the concept of Toss-It, and mentions why a “toss” or a “swing” action is one of the appropriate gestures in information transfer operations. This chapter also describes the technical requirements for Toss-It.
- Chapter 4 presents the gesture recognition method in Toss-It. This chapter consists of two sections. One is about the hardware, and the other is about the recognition and estimation algorithm.
- Chapter 5 describes the location recognition method which is used in the current version of Toss-It. This chapter also consists of two sections like Chapter 4.
- Chapter 6 describes the system architecture and the software for transferring files.
- Chapter 7 presents the experiments for the gesture recognition method and the location recognition method. This chapter also describes the user study experiments to evaluate the whole system of Toss-It.
- Chapter 8 presents discussions from the results of the experiments mentioned in Chapter 7. This chapter also mentions how to improve the practicability and usability of Toss-It.
- Chapter 9 mentions the issues to be investigated in the future. This chapter then summarizes and concludes this thesis.

1.3 Contributions

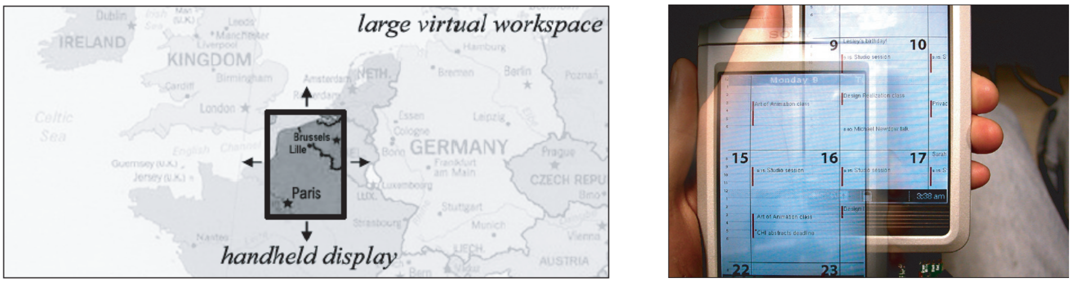
This work makes the following contributions to the field of information technology, especially to the field of Human-Computer Interaction (HCI).

- This work proposes a novel and more intuitive interface for mobile devices.
- The gesture recognition method proposes a more effective recognition algorithm for hand gestures with inertial sensors.
- The location recognition method realizes both location recognition and identification of multiple users at the same time.

Chapter 2

REVIEW OF RELATED WORKS

In this chapter, we present some research works related to Toss-It. We categorize them into three groups; Intuitive interfaces for mobile devices, support for information transfer, and gestural input techniques. These categories are not exclusive and some works may cover two or all categories. However, as a matter of convenience, we categorize them based on the main contribution of each work.



(a) (b)

Figure 2.1. Peephole Display, (a) Concept of a peephole display, (b) Calendar month view with Peephole display (photos were blended but not otherwise edited.) (Yee, 2003)

2.1 Intuitive Interfaces for Mobile Devices

Many research works related to user interfaces for mobile devices have been conducted because mobile devices have been increasingly popular. In this section, we present some of them, especially works which realize intuitive manipulations in mobile devices.

Hinckley et al. (2000) proposed a method which realizes intuitive manipulations in mobile devices with several sensors. The authors attach a tilt sensor (a two-axis accelerometer), a proximity range sensor (an infrared transmitter and receiver), and a touch sensor to a PDA. These sensors can make a PDA guess a user's context. For example, the PDA can guess whether a user is holding it or not with the touch sensor. It is also possible that the system detects whether the PDA is tilted, rotated, shaken or still with the accelerometer. The authors link the guessed user's context to manipulations. One of the good examples is scrolling the display. When a user tilts his PDA vertically, he can scroll the display up/down. A user also can change the display to the portrait/landscape mode by turning the device. Other research projects have proposed intuitive interfaces for mobile devices with similar approaches (Harrison et al., 1998; Rekimoto, 1996).

Yee (2003) suggested two-handed interaction techniques combining pen input with spatially aware displays, which is called Peephole Display. In Peephole Display, the information is spread out on a flat virtual workspace larger than the display, and the display shows a movable window (or "peephole") on the space, as shown in Figure 2.1(a). Although this kind of idea was first proposed by Fitzmaurice (1993), the author integrates other interaction techniques, which were proposed in the Toolglass (Bier et al., 1993) and the zooming UI (Bederson and Hollan, 1994; Perlin and Fox, 1993), into his work. For example, a user can view a larger image on a PDA by physically moving the display around to see different parts of the image (Figure 2.1(b)). A user also can zoom in/out by moving his PDA along the depth axis. A usability study shows that the Peephole technique can be more effective than current methods for navigating information on handheld computers.

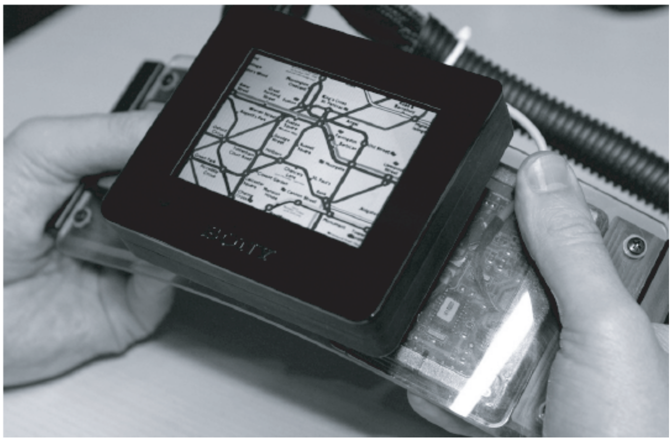


Figure 2.2. Gummi prototype (Schwesig et al., 2004)

Schwesig et al. (2004) developed an intuitive interaction technique and device concept, called Gummi. Gummi consists of a flexible display (e.g., an organic light-emitting display), a bending sensor, a 2D position touch sensor and other flexible electronic components. Users can interact with the Gummi devices by bending them. For example, users can zoom in/out visual information on the display of a Gummi device by bending it upward or downward. The authors built a prototype of Gummi (Figure 2.2), and conducted several evaluation tests for Gummi. They reached the conclusion that Gummi interaction techniques are feasible, effective and enjoyable.

As mentioned above, there are some works which enables users to manipulate mobile devices more intuitively. These works, however, are mainly focused on usage by single person, not on usage between multiple devices, like information transfer. In the next section, we will present several works from the point of view of interfaces for supporting information transfer operations.

2.2 Interfaces for Supporting Information Transfer Operations

In this section, we present several works which are focused on information transfer, especially in mobile devices.

Ayatsuka et al. (2000) proposed a novel technique for information transfer by using "scooping" and "spreading" gestures. A user can project what are shown in his PDA onto the table from the projector, which is equipped on the ceiling, by a "spread" gesture. A user also can bring what are projected onto the table to his PDA by a "scoop" gesture. These gestures and the positions of PDAs are recognized with electromagnetic 3D sensors.

With Pick-and-Drop (Rekimoto, 1997), a user can transfer information between mobile devices in a "pick-and-drop" manner, as shown in Figure 2.3. A special stylus with a limited

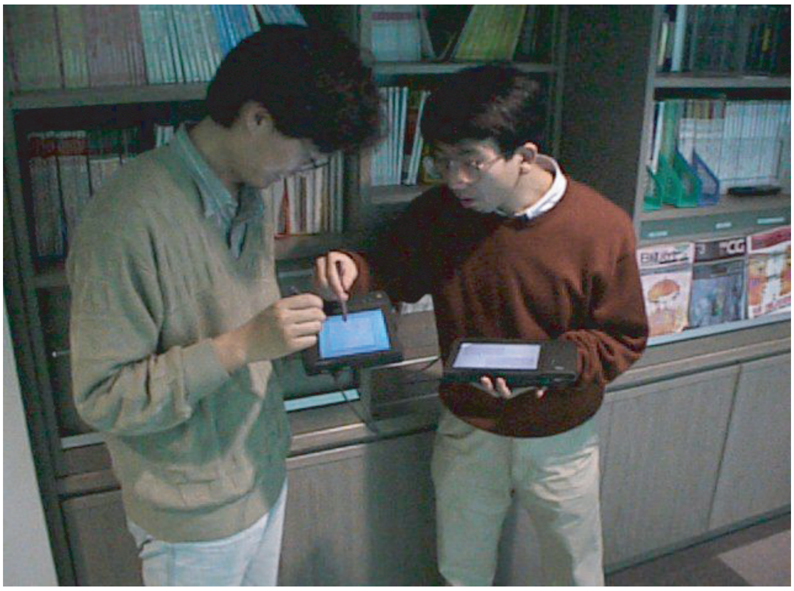


Figure 2.3. Intuitive interaction between PDAs with Pick-and-Drop (Rekimoto, 1997)

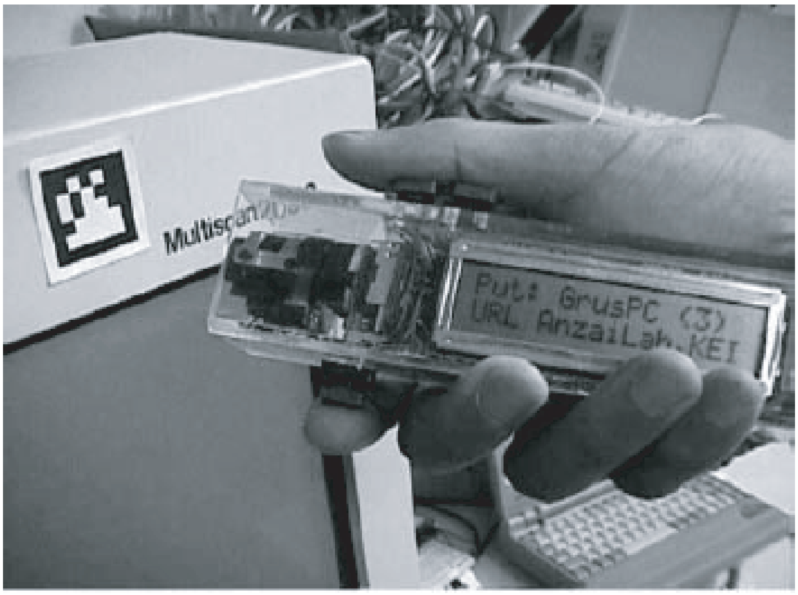


Figure 2.4. A special hand-held device and a visual marker in InfoPoint (Kohtake et al., 2001)

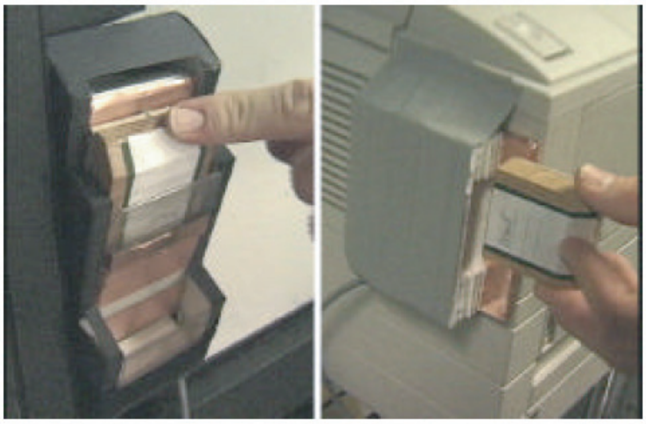


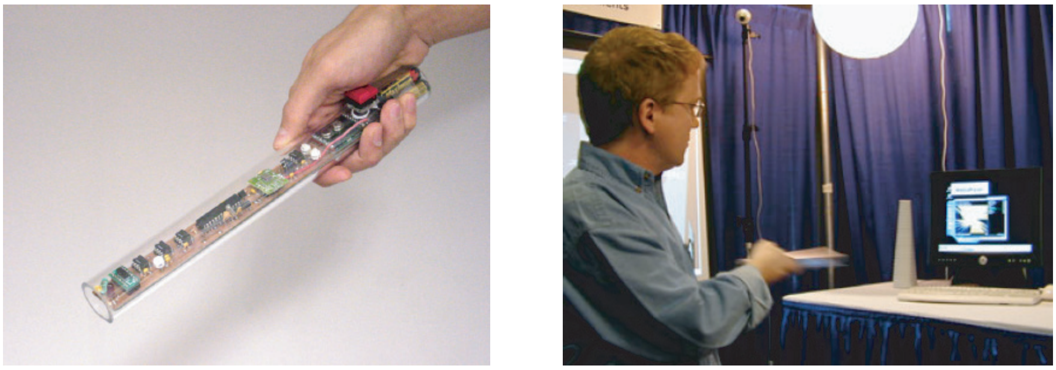
Figure 2.5. Mediablocks at a whiteboard and a printer (Ullmer et al., 1998)

memory is used in Pick-and-Drop. When a user wants to transfer a file in a mobile device to another device, first of all, he selects a file in his mobile device by tapping it with the special stylus, an action called “picking up” action. The special stylus memorizes the object ID of the selected file when the user taps it. Then, he taps on the display of another device. This is the “dropping down” action. The selected file is sent to the device through a network based on the memorized object ID. This series of “pick-and-drop” actions lets users manipulate digital information more intuitively as if they picked up and dropped down physical objects.

A similar work has been conducted by Swindells et al. (2002). In their proposed system, infrared transmitters, called GesturePen, and tags are used as a method for identifying the sender and the receiver of an information transfer transaction. Each GesturePen is associated to a computer device (e.g., a PDA or a laptop). When a user points to a tag with a GesturePen, their system identifies the source device and the destination device by the user’s pointing gesture. Information is transferred from the source device to the destination device over a wireless LAN.

Kohtake et al. (2001) also has built another similar system, called InfoPoint. In InfoPoint, a special hand-held device embedded with a mobile camera and a visual marker (a 2D-matrix code) are used (Figure 2.4). Users can transfer digital information by taking snapshots of visual markers with the hand-held devices as if we transferred information by ‘drag-and-drop’ operations on the GUI in a PC. Suppose a user wants to print out a photo in his digital camera with InfoPoint. He takes a snapshot of the visual marker attached to his digital camera with a hand-held device. The hand-held device gives the user a visual feedback in its LED display. Then, the user moves to a printer, and takes a snapshot of the visual marker attached to the printer. InfoPoint recognizes the marker of the printer and then determines both the source device (the digital camera) and the destination device (the printer). The information is transferred through a network, and the photo is printed out from the printer.

Ullmer et al. (1998) have introduced a tangible user interface for information transfer, called mediaBlocks (Figure 2.5). mediaBlocks have small electrical tags, and users can store digital information which they want to transfer in mediaBlocks. Actually, mediaBlocks stores metadata



(a) (b)

Figure 2.6. XWand, (a) The XWand device, (b) Manipulating a music player in a PC with XWand (Wilson and Shafer, 2003)

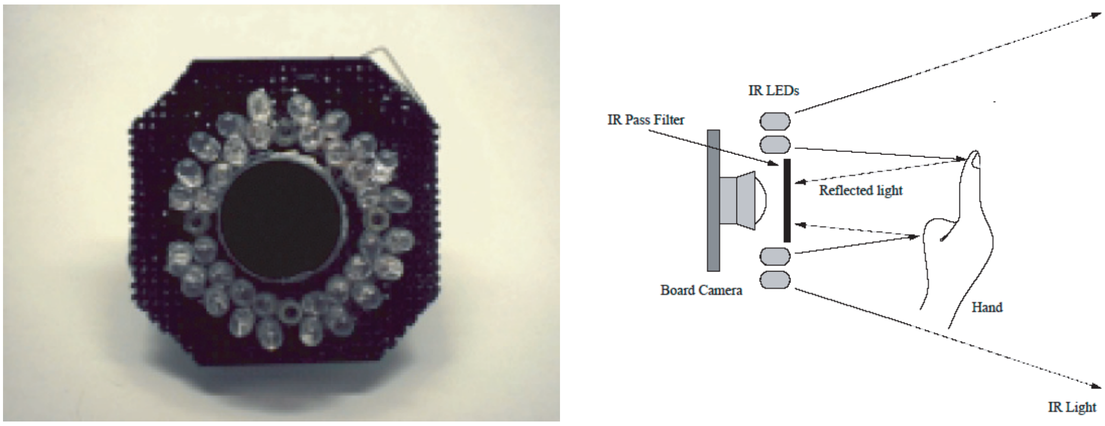
of the information, such as an ID number. Information transfer is done with a similar process to Pick-and-Drop. For example, suppose a user wants to print out the information on a digital whiteboard with a neighborhood network printer. A user inserts a mediaBlock into the slot attached to the digital whiteboard. The mediaBlock stores of the metadata the information on the digital whiteboard. Then, the user inserts the mediaBlock into the slot on a printer. The information is transferred through the network, and is printed out with the printer. As just described, mediaBlocks act as physical "containers" in information transfer.

The proposed systems mentioned above make operations in information transfer more intuitive or more real-world oriented. However, these systems mainly support information transfer between neighborhood devices. Therefore, users have to move enough close to the target device when they want to transfer information. Moreover, these systems don't enable users to transfer information to multiple devices simultaneously, which is one of the characteristic functions of Toss-It (see Section 3.1).

2.3 Gestural Input Techniques

In the last section in Chapter 2, we present related works on gestural input techniques, which utilize human gestures as input methods for computers. In gestural input, hand gestures, that is, gestures with fingers and arms, have been mainly applied to user interfaces in order to facilitate manipulations for computers (Pavlovic et al., 1997). The reason is why a hand is one of the most flexible and usable parts in a human body, and people can express their intentions more easily with hand gestures.

XWand (Wilson and Shafer, 2003) utilizes hand gestures to control computers and home appliances. XWand is a baton-like device, as shown in Figure 2.6(a), which embeds various sensors for supporting pointing and gesture recognition tasks. Suppose that you are in a room



(a) (b)

Figure 2.7. Gesture Pendant, (a) The Gesture Pendant device, (b) Side view of Gesture Pendant with IR reflecting off the user's hand (Gandy et al., 2000)

with some electrical devices and you have XWand. If you want to switch on the television set, you just shake XWand toward it vertically. And you can turn the volume up/down by rotating XWand as pointing to the television set. You also can manipulate a music player in a PC with XWand like Figure 2.6(b). The position and orientation of XWand can be estimated with a magnetometer and an infrared (IR) LED which are embedded in XWand, in combination with two IR cameras on the ceiling. Gestures can be recognized with an accelerometer and a gyroscope in XWand. A similar device also has been proposed as a novel music instrument (Marrin, 1997).

Gestural input techniques also have made a great contribution to interfaces for wearable computers, because input devices and methods which are available or suitable for wearable computers are generally limited. In what follows, we will present several examples of gestural input techniques which are applied to wearable computers.

DataGlove (Zimmerman et al., 1987) is one of the earliest devices of wearable computers. DataGlove is a device like a glove, and a user can put it on in the same way as a usual glove. DataGlove can sense movements or positions of user's fingers with embedded sensors. It also can give him tactile feedback with a vibrator. However, DataGlove has been developed as an input device for desktop PCs, because computers which have mobility, such as what comes to our mind when we hear the term "wearable computers", were not available in those days.

Rekimoto (2001) has proposed a wearable computer system called GestureWrist. GestureWrist looks like a wrist watch. It recognizes hand gestures and forearm movements with an accelerometer and a capacitive sensor. Because we put it on like a wrist watch, not like a glove, it can not detect precise movements of fingers. However, it is more unobtrusive than DataGlove.

Gesture Pendant (Gandy et al., 2000), as its name suggests, is a special pendant which embeds a camera with an IR filter (Figure 2.7(a)). The camera is ringed by IR LEDs. When a user wants to do a manipulation with Gesture Pendant, a user conducts a hand gesture in front of the device. Infrared light is emitted by IR LEDs, and the light reflects off the hand. The camera in Gesture Pendant picks up the reflected infrared light through the IR filter (Figure 2.7(b)). The device can recognize user's hand gestures with images from the camera.

Brewster et al. (2003) also proposed other interesting interaction techniques for wearable computers. One is a 3D audio radial pie menu that uses head gestures for selecting items. The other is a sonically enhanced 2D gesture recognition system for use on a belt-mounted PDA. These two techniques are designed to allow users to manipulate computers in an ‘eyes-free’ manner.

As mentioned above, there are many works which support human-computer interaction with human gestures. However, in Toss-It, we utilize gestural input techniques for supporting interaction between multiple computers and people, which is one of the most different points as compared to other works.

Chapter 3

TOSS-IT

In this chapter, we describe the concept of our proposed system, called Toss-It. We then discuss why a “toss” or “swing” action is appropriate in information transfer with mobile devices. We also mention what are technically required in order to realize Toss-It.

3.1 Concept

With Toss-It, users can transfer digital information from their own mobile devices (PDAs in this work) to others' mobile devices or other equipments in an intuitive manner, as if they would toss a ball or deal cards to others. Followings are examples of how Toss-It can be used:

- Pass a file from a user's PDA to another user's PDA with just a "toss" action toward him as shown in Figure 3.1(a) (unicast transfer).
- Pass a file from a user's PDA to another user's PDA beyond other users in-between with a stronger "toss" action as shown in Figure 3.1(b).
- Print out an image from a user's PDA through a printer with just a "toss" action toward the printer (Figure 3.1(c)), or project a slide onto a screen through a projector with just a "toss" action toward the screen.
- Pass a file from a user's PDA to multiple users with just a "(horizontal) swing" action toward them as shown in Figure 3.1(d) (multicast transfer).

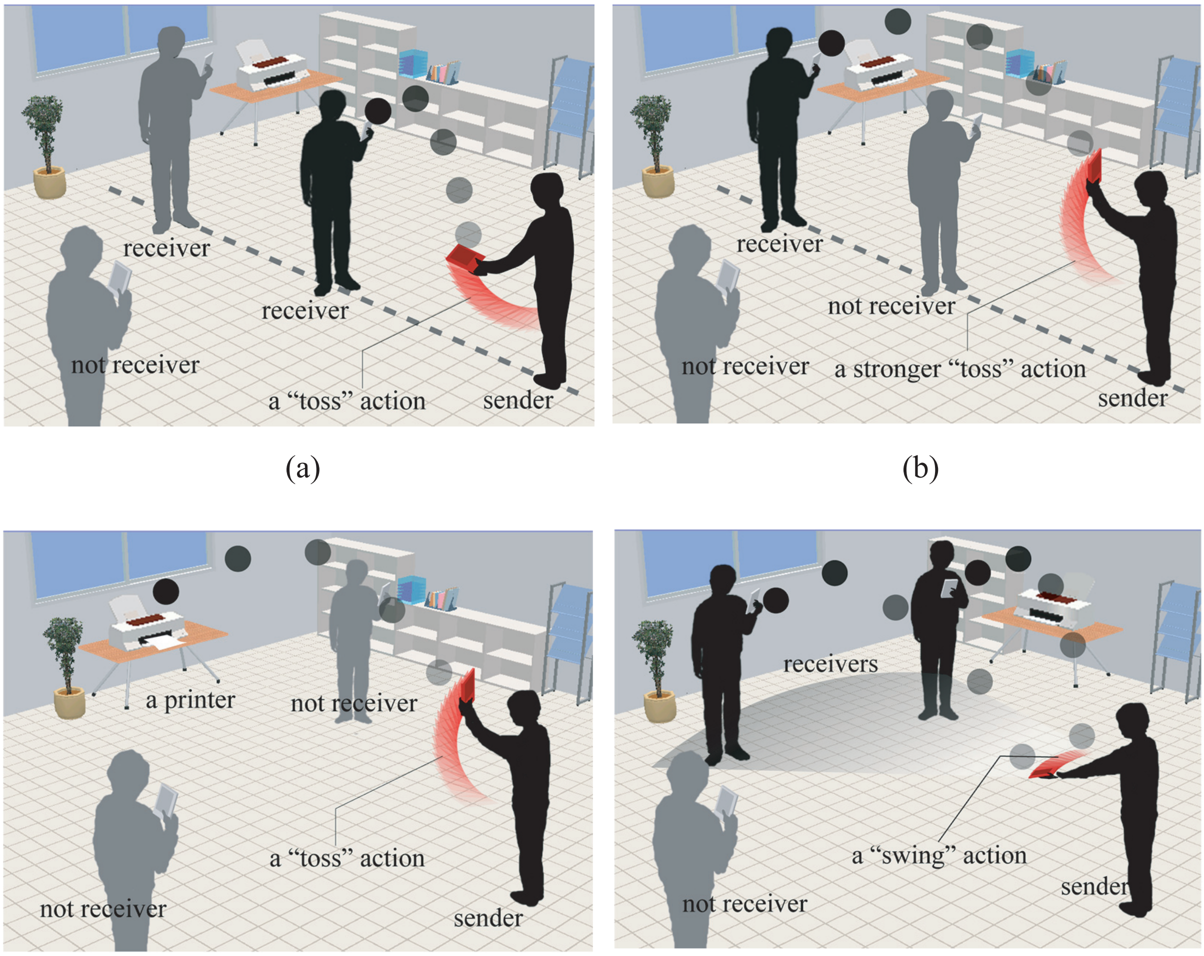
3.2 Implications of "Toss" and "Swing" Actions

For an information transfer transaction to be completed successfully, the following terms need to be determined beforehand.

- **From Who**: Who or which device is the sender in the transaction?
- **To Who**: Who or which device is the receiver in the transaction? And, the receiver is just one or multiple?
- **What**: What data will be transferred in the transaction?
- **How**: What method is used in the transaction, e.g., an infrared communication, a Bluetooth network, or a wireless LAN?
- **When**: When should the transaction start?

In "toss" and "swing" actions, three of the five matters can be determined; That is,

- **From Who**: Those who conduct "toss" and "swing" actions.
- **To Who**: Those who or Devices which the action is conducted toward.
- **When**: At the instant when the action is completed.



(a) (b)

(c) (d)

Figure 3.1. Intuitive information transfer techniques with Toss-It, (a) from a PDA to another PDA, (b) from a PDA to another PDA beyond other users in-between, (c) from a PDA to a printer, (d) from a PDA to multiple PDAs

Therefore, we can claim that “toss” and “swing” actions have three implications in information transfer, even though a “toss” or “swing” action is just a single gesture. This is the reason why we can pass what is in our hand by just tossing it toward others.

Mobility, which mobile devices naturally have, is another factor which can realize the concept of Toss-It. For example, we cannot conduct “toss” and “swing” actions easily with desktop PCs or laptops. We can say mobility in mobile devices makes them have high affinity with hand gestures.

Therefore, a “toss” or a “swing” action is one of the appropriate gestures which are applied to mobile devices in order to make information transfer operations in mobile devices more intuitive.

3.3 Requirements for Toss-It

In order to realize information transfer by users’ “toss” and “swing” actions, Toss-It must satisfy the following requirements.

- **Requirement A**: Toss-It can recognize user’s toss and swing actions conducted with his PDA.
- **Requirement B**: Toss-It can automatically identify the positions and orientations of multiple users’ PDAs and electronic devices.
- **Requirement C**: Based on a user’s action, Toss-It can transfer digital information from his PDA to other users’ PDAs or to the corresponding electronic devices.

It is desirable to satisfy the requirements without any external equipment embedded or installed in an environment where Toss-It is used, in order to make Toss-It available anytime and anywhere. This is the final goal of our project. Our initial goal, however, is to investigate if the proposed idea (information transfer by “toss” and “swing” actions) is possible, practical and usable.

As for Requirement A, we determined to attach inertial sensors to a PDA. This approach has several merits as compared to approaches such as using external equipments. For example, suppose we use a camera-based technology. In this case, multiple cameras and their complicated calibrations for them are necessary in order to capture users’ actions in any location. We describe the details of our approach in Chapter 4.

As for Requirement B, there are several existing location recognition technologies applicable to Toss-It (Hightower and Borriello, 2001). We are now developing a novel location and orientation recognition technology that can acquire relative positions and orientations of multiple devices without any external equipment, such as beacons in the ceiling (金子他, 2005; 菅野他, 2005). In the current implementation, however, we substitute a camera-based technology described in Chapter 5.

As for Requirement C, it may be possible to use peer-to-peer or ad-hoc network technologies. In this paper, however, we use a wireless LAN and a server computer. The server also manages data on the positions and orientations of multiple devices. When a user conducts a “toss” or “swing” action with his PDA to transfer information, the software of Toss-It estimates the strength (in the case of a “toss” action) or the trajectory (in the case of a “swing” action) of the action, identifies target devices, and sends the information to the devices via wireless LAN. We present the details in Chapter 6.

Chapter 4

GESTURE RECOGNITION

This chapter presents the gesture recognition method in Toss-It. This chapter consists of two main sections. The first main section is about the hardware. We describe how to determine which inertial sensors we use in Toss-It and how to design the circuit board for the inertial sensors. The second main section is about the recognition algorithm. We explain not only how to recognize “toss” and “swing” actions, but how to estimate the strength of a “toss” action and the trajectory of a “swing” action.

4.1 Hardware

4.1.1 Designing the prototype circuit board

In order to recognize “toss” and “swing” actions, we use inertial sensors (e.g., accelerometers and gyroscopes). This approach is a popular method for gesture recognition, which other works have adopted (Hinckley et al., 2000; Wilson and Shafer, 2003). It has several merits compared to other possible approaches. For example, if we use a vision-based approach, we have to install many cameras in order to recognize users’ actions anywhere and anytime. Moreover, the calibration of multiple cameras and complicated recognition methods are required. In our approach, users’ actions can be recognized anytime and anywhere without any special external equipment because inertial sensors are attached to users’ PDAs themselves.

When we design a circuit board for inertial sensors, the first thing we have to do is which inertial sensors we use. In Toss-It, inertial sensors are required to sense drastic actions, such as a “toss” or a “swing” action. For example, acceleration is expected to be much bigger than the acceleration of gravity. This is true for angular velocity.

We determined to use Analog Devices ADXL210 as an accelerometer. The detection range is ± 10[g] and the sensitivity is 100[mV/g]. Additionally, it can sense acceleration along two axes. For detecting the rotation, we use Silicon Sensing Systems Japan CRS03 as a gyroscope. The detection range is ± 200[deg/sec] and the sensitivity is 10[mV/(deg/sec)]. It can sense angular velocity along one axis.

We made a prototype circuit board with the sensor mentioned above, which is shown in Figure 4.1. In this board, a H8 microprocessor (H8/3048F) is embedded for processing the output of the inertial sensors and communicating with a host computer. The followings are features of this microprocessor;

- 16[MHz] clock frequency
- 128[Kbytes] ROM and 4[Kbytes] RAM
- 8-channels analog-to-digital converter
- 2-channels serial communication interface

This microprocessor converts the analog output of the inertial sensors to the digital signals, and sends them to a host computer (a PDA or a PC) through the serial communication interface.

We conducted an informal experiment with this prototype circuit board. In this experiment, we observed and analyzed the output of the inertial sensors when we conducted “toss” and “swing” actions with the board. The followings are lessons learned from this experiment;

- The inertial sensors have enough capability for detecting “toss” and “swing” actions: When we conducted “toss” or “swing” actions, the average of the maximum values in the output of the accelerometers was around ± 5[g]. That is within the detection range of the accelerometers. The gyroscope was also proved to have enough detection range. Therefore, these inertial sensors were appropriate for detecting user’s actions in Toss-It.

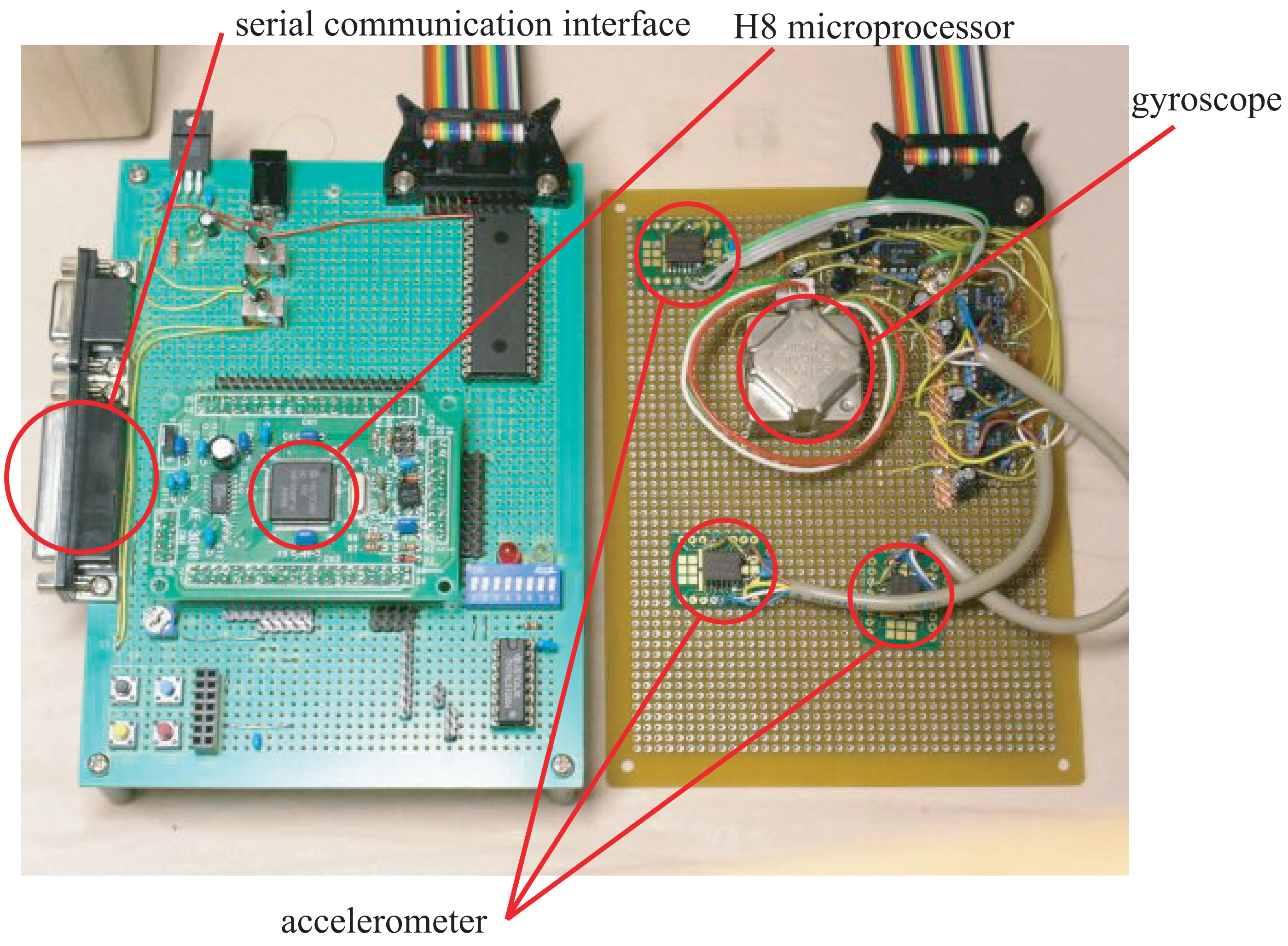


Figure 4.1. The prototype circuit board for inertial sensors

- A noise is always caused in the inertial sensors: We observed relatively big fluctuations in each output of the inertial sensors, even though the board was in the state of rest. After analyzing with a spectrum analyzer, we confirmed that a noise was constantly caused within the frequency range over 1[kHz]. This noise is thought to be caused from the sensors and the board itself. Therefore, a low pass filter, which are designed to set the cutoff frequency to 1[kHz], is required for eliminating the noise.

- The gyroscope is a little too big: In the prototype circuit board, we used CRS03. The size of this gyroscope is 29[mm] × 29[mm] × 10.4[mm]. In order to detect 3D movements precisely, it is better that the board can sense its rotation along the three axes (i.e., x-axis, y-axis, and z-axis). Therefore, we have to design the circuit board to allow to attach three gyroscopes. We also have to make the circuit board as small as possible, because the size of a PDA is limited (about 7[cm] × 12[cm]). Considering these matters, we concluded that it is better that we use a smaller gyroscope which has similar capability as CRS03.

4.1.2 Designing the improved version of the circuit board

Based on the lessons learned, we designed the improved version of the circuit board. In this board, we use four ADXL210 accelerometers and three Murata ENC-03J gyroscopes. The size of ENC-03J is 15.3[mm] × 8.0[mm] × 4.3[mm], which is much smaller than CRS03. The detection range is the ± 300[deg/sec] and the sensitivity is 0.67[mV/(deg/sec)]. There are also low pass filters for each sensor. The cutoff frequency of each low pass filter is set to 1[kHz].

To capture users' "toss" and "swing" actions, it is sufficient for the circuit board to sense six degrees-of-freedom movements. In our current implementation, the board can sense 11 degrees-of-freedom (i.e., four two-axes accelerometers and three one-axis gyroscopes). This redundant implementation is for capturing users' actions as accurately as possible. Although the H8 microprocessor has to deal with 11 output data from the inertial sensors, it has only the 8-channels analog-to-digital converter. Therefore, we have determined to use an analog multiplexer for the output of the accelerometers in order to realize signal processing like Time Division Multiplex.

The H8 microprocessor samples the output of the inertial sensors for every 10[msec]. The sampled data are sent to a host computer through the serial communication interface.

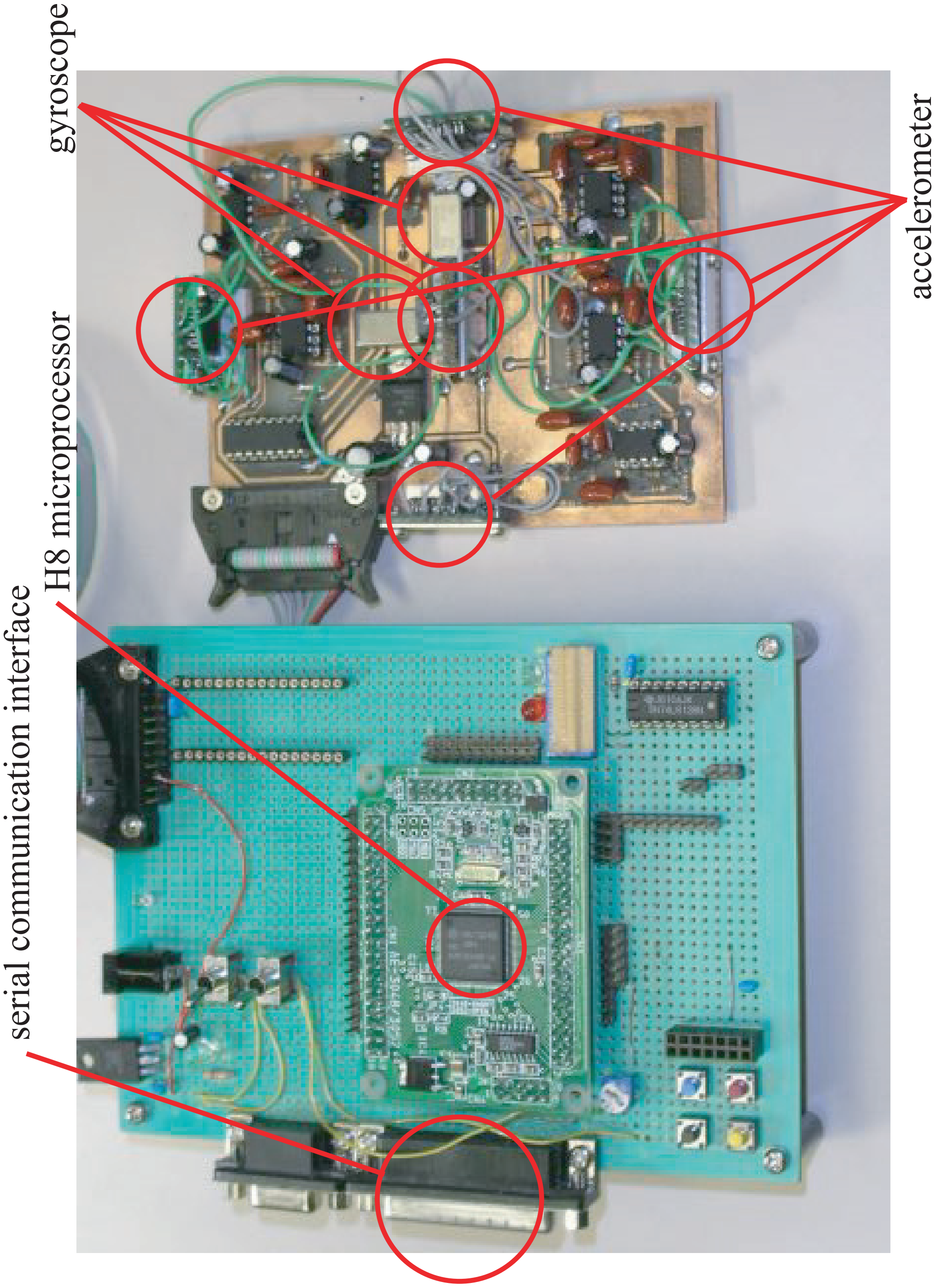


Figure 4.2. The improved version of the circuit board for inertial sensors

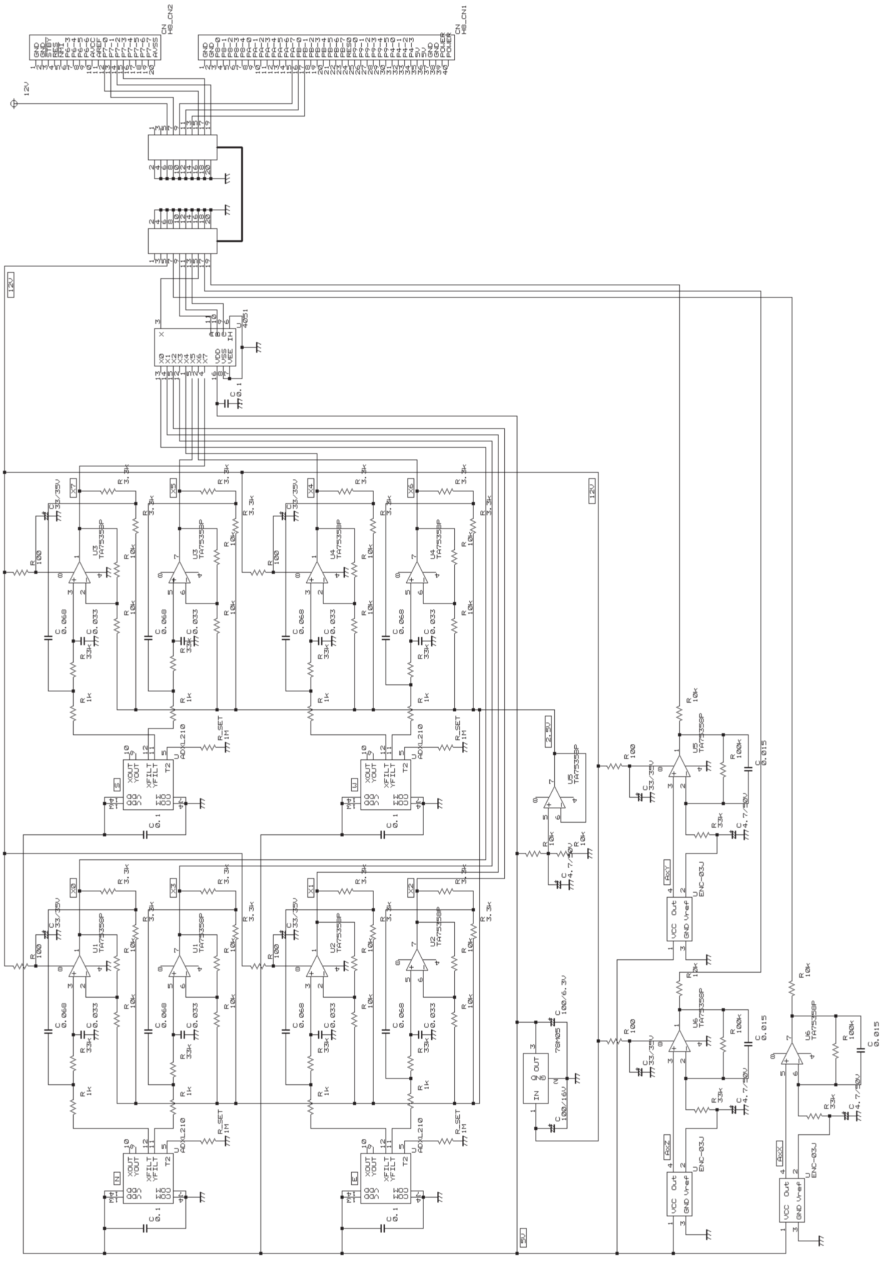


Figure 4.3. The circuit diagram of the improved version of the circuit board

4.2 Recognition Algorithm

Toss-It is required not only to recognize “toss” and “swing” actions, but also to estimate the strength of a “toss” action or the trajectory of a “swing” action. This is why Toss-It needs to distinguish the receiver(s) by the flying distance in “toss” actions, as shown in Figure 3.1(a) and 3.1(b), or the range in “swing” actions, as shown in Figure 3.1(d). In this section, we first present how to calculate the velocity and the trajectory in a “toss” or “swing” action. Consequently, we present the three algorithms for recognizing “toss” and “swing” actions; algorithm for eliciting “toss” or “swing” actions, algorithm for estimating the strength of a “toss” action, and algorithm for estimating the trajectory of a “swing” action.

4.2.1 How to Calculate the Velocity in a “Toss” Action or the Trajectory in a “Swing” Action

Some works which utilize trajectories calculated from output of inertial sensors have been already conducted. Bang et al. (2003) proposed a pen-type input device embedded with accelerometers and gyroscopes. The posture and the movement of the device are recognized with the sensor. The authors achieved success to recognize characters which are written in the air with the device, by calculating the trajectory of its movement. Another application was proposed by Kourogi and Kurata (2003). In their system, a user puts on inertial sensors, a wearable camera, an inertial head tracker and a display. The system calculates both the user’s position and the trajectory of the user’s movement from the sensors and the camera. It allows not only to track walking people, but to detect whether they are going up/down stairs. Toss-It calculates the velocity and the trajectory in a “toss” or “swing” action with a similar approach to what are used in the works mentioned above.

If we know the accurate values of acceleration along three axes in the world coordinate system, we can calculate the trajectory of the movement by twice integral operations of the acceleration. The inertial sensors embedded in the circuit board, however, only can sense accelerations and velocities along three axes in the PDA’s local coordinate system. For this reason, a coordinate transformation method is necessary. The most frequently-used methods for coordinate transformation are the two following methods; a method with Euler angles (宮川他, 2002), and a method with quaternions (Chou, 1992). For several reasons, such as using a more succinct theory, we determined to use a method with Euler angles as a coordinate transformation method.

For the sake of simplicity, we first think the rotation of the coordinate system at the fixed point. As shown in Figure 4.4, we consider the acceleration in the XYZ coordinate system and the acceleration in the $X_1Y_1Z_1$ coordinate system, which can be obtained by rotating the XYZ coordinate system ϕ [rad] along X-axis, θ [rad] along Y-axis,ψ [rad] along Z-axis. Let $\boldsymbol{p}$ be a point in the XYZ coordinate system, and $\boldsymbol{p}_1$ be a point in the $X_1Y_1Z_1$ coordinate system. If $\boldsymbol{p}$ moves to $\boldsymbol{p}_1$ after one sampling time, $\boldsymbol{p}_1$ can be written with $\boldsymbol{p}$ and the Euler angles like Equation 4.1.

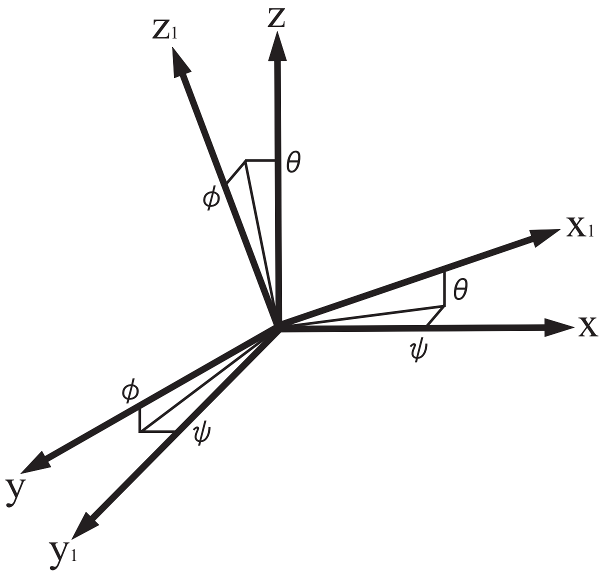


Figure 4.4. A coordinate system XYZ and the rotated coordinate system $X_1Y_1Z_1$

$$\boldsymbol{p}_1 = \begin{pmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{pmatrix} \cdot \boldsymbol{p} \tag{4.1}$$

In Equation 4.1, the matrix is called a rotation matrix, which we represent as $\boldsymbol{\Omega}$. If the rotation is done in a sufficiently short time, that is, the sampling time is small enough, we can approximate $\sin\alpha \sim \alpha, \cos\alpha \sim 1, \alpha\beta \sim 0$ (α, β are minute angles). In this case, we can get a more concise expression of $\boldsymbol{\Omega}$ as

$$\boldsymbol{\Omega} \sim \boldsymbol{I} + \begin{pmatrix} 0 & -\delta\psi & \delta\theta \\ \delta\psi & 0 & -\delta\phi \\ -\delta\theta & \delta\phi & 0 \end{pmatrix} \tag{4.2}$$

where $\boldsymbol{I}$ is the identity matrix and $\delta\phi, \delta\theta, \delta\psi$ are minute rotation angles of ϕ, θ, ψ. The three rotation angles, ϕ, θ, ψ, in Equation 4.1 and 4.2, can be calculated with the output values from the gyroscopes. Therefore, we can obtain $\boldsymbol{\Omega}$ by Equation 4.1 and 4.2.

Consequently, we consider the case in which the PDA's local coordinate system has been rotated n times, as $X_1Y_1Z_1, X_2Y_2Z_2, \cdots, X_nY_nZ_n$ (n is a positive integer). Let $\boldsymbol{R}(n)$ be the transformation matrix in the n-th rotation. And, let $\boldsymbol{\Omega}(n)$ be the rotation matrix between $X_{n-1}Y_{n-1}Z_{n-1}$ and $X_nY_nZ_n$. Moreover, suppose that the PDA's local coordinate system is the same as the world coordinate system at the initial state. Based on this assumption, we can say $\boldsymbol{R}(0) = \boldsymbol{I}$. We can write $\boldsymbol{R}(n)$ with $\boldsymbol{R}(n-1)$ as

$$\boldsymbol{R}(n) = \boldsymbol{R}(n-1)\boldsymbol{\Omega}(n)^{-1} \tag{4.3}$$

In Equation 4.3, $\boldsymbol{\Omega}(n)^{-1}$ can be always obtained, because of the property of the rotation matrix. Therefore, we can derive the expression of $\boldsymbol{R}(n)$ from the above recurrence equation.

$$\boldsymbol{R}(n) = \prod_{i=1}^{n} \boldsymbol{\Omega}(i)^{-1} \tag{4.4}$$

Equation 4.3 and 4.4 indicate that we can calculate the transformation matrix successively from the output of the gyroscopes.

We can calculate the trajectory of the movement by twice integral operations of the acceleration. We define $\boldsymbol{a}(k)$ as the acceleration vector at any one time, k (k is a positive integer). We also define $\boldsymbol{a}_s(k)$ as the vector which consists of the observed values of the acceleration in the PDA. Now, we can obtain the net acceleration of the PDA by subtracting the acceleration of gravity from $\boldsymbol{a}_s(k)$. That is,

$$\boldsymbol{a}(k) = \boldsymbol{a}_s(k) - \boldsymbol{g}(k) \tag{4.5}$$

where $\boldsymbol{g}(k)$ is the acceleration vector of gravity which acts on the PDA at the time k. $\boldsymbol{g}(k)$ can be calculated with the acceleration vector of gravity in the world coordinate system $\boldsymbol{g}_0$, as following.

$$\boldsymbol{g}(k) = \boldsymbol{R}(k)^{-1}\boldsymbol{g}_0 \tag{4.6}$$

Next, we can calculate the velocity vector $\boldsymbol{v}_w(k)$ in the world coordinate system with the rotation matrix and $\boldsymbol{a}(k)$ like below.

$$\boldsymbol{v}_w(k) = \boldsymbol{v}_w(k-1) + T\boldsymbol{R}(k)\boldsymbol{a}(k) \tag{4.7}$$

where T is the sampling time. Finally, we can obtain the position vector $\boldsymbol{p}_w(k)$ in the world coordinate system.

$$\boldsymbol{p}_w(k) = \boldsymbol{p}_w(k-1) + T\boldsymbol{v}_w(k) \tag{4.8}$$

Now, we get the trajectory of the movement as the line connecting all the points that are obtained by Equation 4.8.

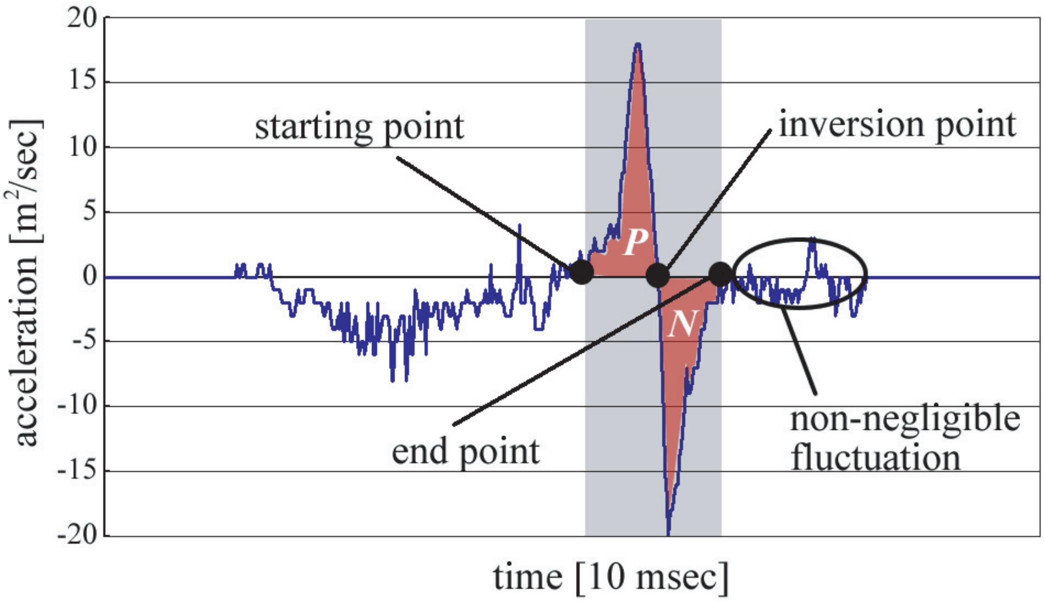


Figure 4.5. A typical example of the output data of an accelerometer when a user conducted a toss action with Toss-It

4.2.2 Eliciting "Toss" or "Swing" Actions

Ideally, Toss-It can recognize a "toss" or "swing" action through the output data of the inertial sensors. Several informal experiments, however, have indicated that non-negligible fluctuation occurs in the output data just after the action has been completed. Figure 4.5 shows a typical example of the output data of an accelerometer when a user conducted a "toss" action with Toss-It.

In order to eliminate this fluctuation, we have devised a new recognition algorithm. To apply this algorithm, an assumption is made that a "toss" or "swing" action is initiated and finished in a state of rest. This assumption justifies the idea that the area of the positive part (P in Figure 4.5) is equal to that of the negative part (N in Figure 4.5).

The recognition process is summarized as follows: First, Toss-It searches an intersecting point of the output data curve and the zero acceleration line as shown in Figure 4.5. When Toss-It has found a new intersecting point, it calculates the integral of the acceleration values between the intersecting point and the previous intersecting point, named the "starting point". If the value is greater than a predefined threshold, Toss-It regards the current intersecting point as the "inversion point" and begins to calculate the integral of the acceleration values from the inversion point. While Toss-It makes the calculation, it evaluates the summation of the two integral values (the integral between the starting and inversion points, and the integral from the inversion point). When the value of the summation becomes approximately zero, Toss-It stops the calculation and regards the current point as the "end point". Finally, Toss-It recognizes that a user's action happened between the starting point and the end point (the highlighted region in

Figure 4.5).

For the sake of simplicity, the calculation mentioned in Section 4.2.1 is started in the current version of Toss-It, when a user selects a file to be sent.

4.2.3 Estimating the Strength of a “Toss” Action

Toss-It estimates the strength of a “toss”, in order to determine how far “tossed” information travels and which devices receive the information. After several informal experiments of a toss action, we have made an assumption for reasonably accurate estimations and less complex calculations: When we toss something, we release it at the maximum speed. A toss action is started at the vertically downward position to the floor and finished without a follow through. Toss-It regards a point of the maximum velocity during the toss action as a release point of “tossed” information. Toss-It also calculates the launch angle by integrating the data from the gyroscopes. After determining the maximum velocity and the launch angle, Toss-It estimates the flying distance of a “toss” action with the equation of motion as

$$distance = \frac{v_0^2 \sin 2\theta}{g} \tag{4.9}$$

where v_0 is the scalar of the maximum velocity, θ is the launch angle, and g is the scalar of the acceleration of the gravity.

4.2.4 Estimating the Trajectory of a “Swing” Action

Toss-It calculates the trajectory of a user’s “swing” action through the second integral of accelerations. To determine how many degrees a user has swung his PDA around him, we assume that a trajectory of a swing is an arc. Toss-It calculates an angle of the arc, and recognizes the devices inside the arc angle as receivers of information as shown in Figure 3.1(d). To simplify the calculation of the angle, we use only horizontal moves and neglect vertical moves of a “swing” action. An arc angle of a user’s “swing” action can be calculated as follows.

$$angle = 2 \sin^{-1} \frac{l}{2r} \tag{4.10}$$

where l is the chord length and r is the radius of the arc, which is decided by each user’s arm length.

Chapter 5

LOCATION RECOGNITION

Chapter 5 describes the location recognition method in Toss-It, which allows to recognize not only positions but orientations of multiple users as well. This chapter also consists of two main sections like Chapter 4. The first main section is about the hardware. We present a special marker and a stereo camera in our developed location recognition system. The second main section is about the recognition algorithm. We describe how the system recognizes positions and orientations of multiple users.

5.1 Approach

From the discussion in Section 3.2, a location recognition method in Toss-It is required to recognize who is where, and whom the action is conducted toward in order to determine the sender and the receivers correctly. Therefore, the location recognition method must recognize positions and orientations of multiple users.

Over the past years, a considerable number of studies have been made on indoor location recognition methods (Hightower and Borriello, 2001). There are two main reasons for this. Firstly, many researchers have been greatly influenced by the concept of ubiquitous computing (Weiser, 1991) and context-aware computing (Moran and Dourish, 2001). In these computing environments, users' location information is regarded as one of the most useful context information. Additionally, technologies to get users' location information are more established and more reliable than technologies to get users' other context, such as users' mental status. Secondly, generally speaking, the global positioning system (GPS), which seems to be the most widely-used technology for location recognition, does not work well on recognizing users' location in the indoor situations. Therefore, researches have explored indoor location recognition methods which have enough accuracy and flexibility.

One of the most popular technologies for indoor location recognition is radio frequency or ultrasonic, as represented by the Bat system (Harter et al., 2002) and RADAR (Bahl and Padmanabhan, 2000). Cricket (Priyantha et al., 2000, 2001) is also one of the good examples in this kind of technologies. Cricket uses special beacons and listeners. Beacons are usually installed in unobtrusive places, such as the ceiling. Listeners are attached to users' devices and receive signals from the beacons. The authors used a combination signal of radio frequency and ultrasound in order to enable a listener to determine the distance to beacons, from which the closest beacon can be more unambiguously inferred.

Electromagnetic sensing offers a classic position-tracking method. This kind of indoor location recognition system generates axial DC magnetic-field pulses from a transmitting antenna. The system computes the position and orientation of a receiving antenna by measuring the response in three orthogonal axes to the transmitted field pulse, combined with the constant effect of the earth's magnetic field. Although this technology provides high-precision location recognition, the equipments are generally very expensive.

Using cameras is another reasonable method. Especially, stereo cameras are very effective and easy to use when we want to recognize 3D positions of objects. In EasyLiving, which was proposed by Brumitt et al. (2000), stereo cameras are used for location recognition in a home environment. EasyLiving realizes a context-aware computing service based on the users' location information.

As mentioned above, there are many possible location recognition methods applicable to Toss-It. However, most of them require special equipments in an external environment (e.g., beacons on the ceiling). This constraint may derogate the merit of mobile devices, that is, we can use them anytime and anywhere we want to. Therefore, a location recognition method which is independent of external environments is more appropriate for Toss-It. We are now proceeding a research project to establish a novel indoor location recognition technology which

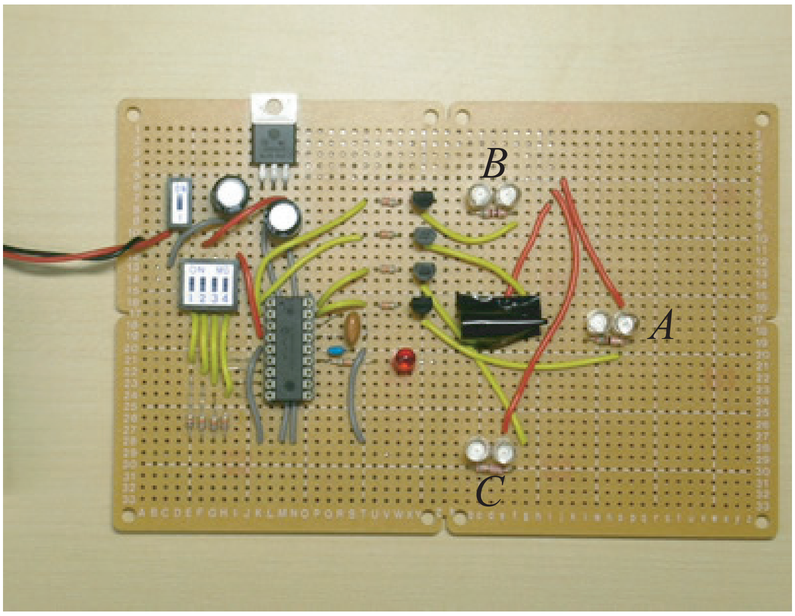


Figure 5.1. A special marker with infrared LEDs

requires no equipment in external environments (金子他, 2005; 菅野他, 2005). However, in this paper, we substitute a vision-based method, which we explain in this chapter.

5.2 Hardware

We determined to use a vision-based method for location recognition. This approach has several merits compared to other possible approaches. For example, a vision-based method can be developed more easily.

In order to recognize 3D positions of users more accurately, we have to use multiple cameras. However, when we use multiple cameras, we have to do some difficult configurations, such as calculating the fundamental matrix in the epipolar geometry (Trucco and Verri, 1998). Therefore, we determined to use stereo cameras, as EasyLiving (Brumitt et al., 2000) does, to avoid the complexity. Moreover, we use a special marker to enable the system to recognize positions and orientations of users more accurately.

Figure 5.1 shows the special marker in our location recognition system. Figure 5.2 is the circuit diagram of this marker. This marker is attached to each user's device. In this marker, IR LEDs are arranged to form an isosceles triangle (A, B, and C in Figure 5.1). The position and orientation of a user's PDA are determined by the shape of the triangle. In other words, the position can be estimated as the center of mass of the triangle. The orientation also can be estimated as the vector sum of $\vec{BA}$ and $\vec{CA}$. For more details, see the next section.

Moreover, the IR LED which is arranged in the vertex A of the isosceles triangle is blinking at a prefixed pattern. The blinking pattern is set to be unique in each marker. This enables the

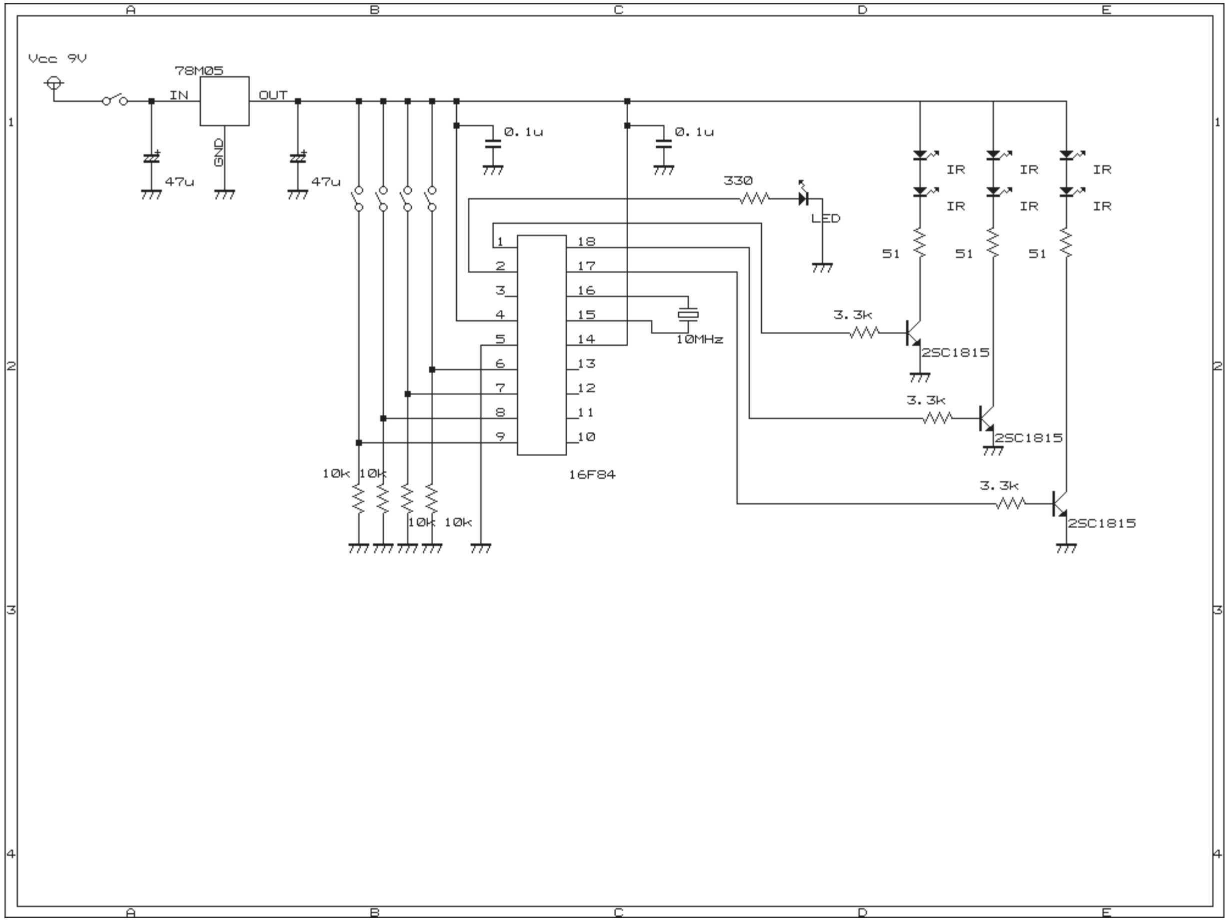


Figure 5.2. The circuit diagram of the marker

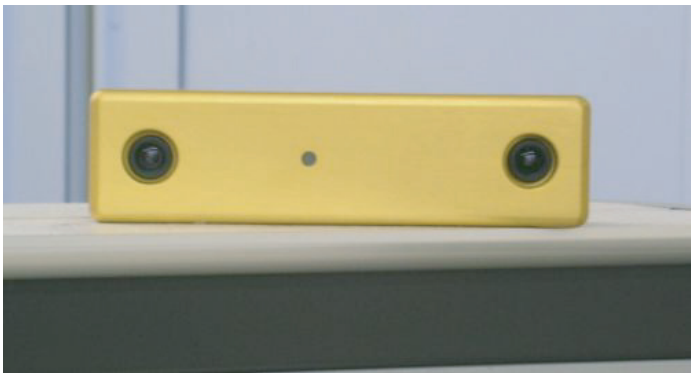


Figure 5.3. A stereo camera

system to identify multiple markers by recognizing the blinking pattern of each marker.

Figure 5.3 is the stereo camera, which is fixed to a ceiling to capture the marker. On the two lenses of this stereo camera, there is an optical filter which eliminates visible light. This optical filter enables the stereo camera to capture only the light of IR LEDs on the marker.

5.3 Recognition Algorithm

In this section, we describe how the positions and orientations of multiple users are recognized with the markers and the stereo cameras which are mentioned in Section 5.2. The recognition algorithm can be divided into five parts like below.

- Extracting the light of IR LEDs
- Calculating the 3D point of each IR LED
- Estimating the position and orientation of each marker
- Tracking each marker
- Identifying each marker

Below, we explain stepwise how each these five steps of the recognition algorithm works.

5.3.1 Extracting the light of IR LEDs

As mentioned in Section 5.2, the stereo cameras capture only the light emitted by the lR LEDs on the markers. In the images of the stereo cameras, the light of the IR LEDs is shown as white circles in a black background as shown in Figure 5.4. The first thing to determine the position of each IR LED is to extract the white circles from the black background.

Each pixel in the camera images is represented by 8-bit grayscale, that is, represented by an integer value from 0 (black) to 255 (white). The algorithm first changes the camera images into the binary images by using the predefined threshold (e.g., 128). In the binary images, the pixels in the white circles are represented by the value $V = 1$. Next, the algorithm groups them into circles. For this purpose, we use a connected components labeling method (谷口, 1996). This method is based on the pixel connectivity, which is a relation between two or more pixels.

We can say two pixels are connected if they have to fulfill certain conditions on the pixel, such as color and spatial adjacency. To formulate the adjacency criterion for connectivity, we first introduce the notation of neighborhood in our method. The set of pixels for a pixel p with the coordinate (x, y), is given by:

$$N(p) = \{(x-1, y-1), (x, y-1), (x+1, y-1), (x-1, y)\} \tag{5.1}$$

In the connected components labeling method, the label of a pixel p is determined by the labels of $N(p)$. Therefore, we can think this equation means the algorithm scans four neighbors of p (i.e. the left of p, above it, and the two upper diagonal terms) in order to determine the label of p. Let $L(p)$ be the label of a pixel p with the coordinate (x, y). On the initial state, $L(p)$ is set to "0" for all p. The algorithm determines the label of each p as the following pseudo code.

```
$label$ = 1;
Until all the pixels have been scanned from top to bottom and from left to right,
    If $V(p)$ equals to 0, that is, the pixel belongs to the background,
        Set $L(p)$ to 0.
    Else
        Check all the labels of the neighborhood of $p$, in other words, check
        $L(p_n)$, where $p_n \in N(p)$;
        If all the values of the four label are 0,
            Set $L(p)$ to $label$;
            $label$++;
        Else if there is only one kind of the non-zero values in the four label,
        which is $label1$,
            Set $L(p)$ to $label1$;
        Else if there is two kinds of the non-zero values in the four label,
        which are $label1$ and $label2$ ($label1 < label2$),
            Set $L(p)$ to $label1$;
            Set $L(p_l)$,where $p_l \in \{p|L(p) = label2\}$ to $label1$;
    Move to the next pixel;
End.
```

After this labeling process, a unique label is assigned to each circle in the camera images. Therefore, we can extract the region of each circle from the images with the labels.

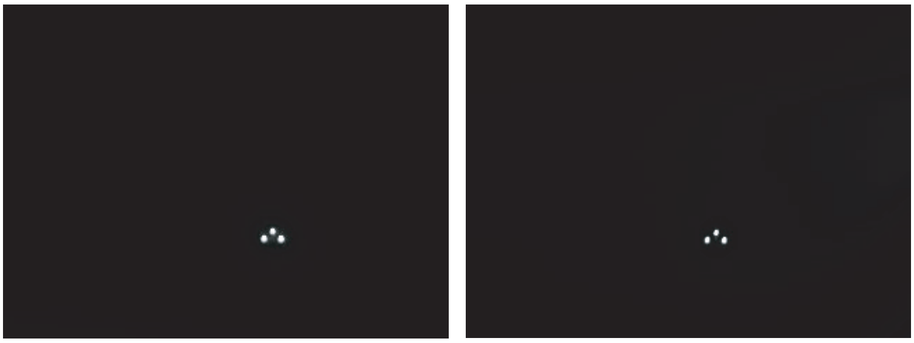


Figure 5.4. IR LEDs captured by a stereo camera (left: the left-eye image, right: the right-eye image)

5.3.2 Calculating the 3D position of each IR LED

After extracting the white circles, the algorithm calculates the center of mass of each circle. The algorithm consequently makes the correspondences between the right-eye image and the left-eye image of the stereo camera. Making the correspondences between two images for the stereoscopic, generally speaking, is a complicated process (Trucco and Verri, 1998). However, we can facilitate this process because we use a stereo camera. Only shift along the horizontal direction is the difference between the two images in a stereo camera. If you shift the left-eye camera image rightward, you can get the same image as the right-eye camera image (Figure 5.4). This makes us get the same result in the two images through the labeling process mentioned above. In other words, we can think the circles which should be the correspondences are assigned to the same label value.

It is no longer difficult to calculate the 3D position of each IR LED, after calculating the center of mass of each circle in the images and making the correspondences between the images. The 3D position can be easily calculated by the perspective model and the triangulation, which are shown in Figure 5.5. Let us consider how to calculate the position of P in Figure 5.5. p_l and p_r are the projection of P in each camera image. T, which is called the *baseline*, is the distance between the center of the cameras, O_l and O_r. Let x_l and x_r be the x-coordinates of p_l and p_r, with respect to the principal points c_l and c_r, f the common focal length, and Z the distance between P and the baseline. From the similar triangles (p_l, P, p_r) and (O_l, P, O_r), we get the following equation.

$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z} \tag{5.2}$$

Solving this equation for Z, then we obtain

$$Z = f\frac{T}{d} \tag{5.3}$$

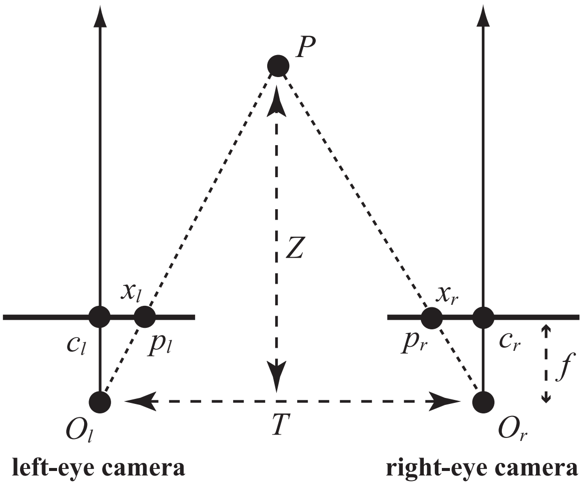


Figure 5.5. Stereo system with the stereo cameras

where $d = x_l - x_r$, which is called the *disparity*. After we obtain the depth (the z-coordinate of P), we can also obtain the x-coordinate and y-coordinate of P with the following equations, which are derived from the assumption of the perspective model (Trucco and Verri, 1998).

$$x = f\frac{X}{Z} \tag{5.4}$$

$$y = f\frac{Y}{Z} \tag{5.5}$$

where X, Y are the x-coordinate and the y-coordinate of P in the real world, and x, y are the x-coordinate and the y-coordinate of the projected point of P in the camera images, that is, p_l and p_r.

In fact, in order to calculate 3D positions of IR LEDs with the process mentioned above, a calibration for each camera is required. This calibration is called *rectification,* which makes conjugate points have the same vertical coordinate. If you want to know more details, please see the reference (Trucco and Verri, 1998). In our implementation, we rectify the images of a stereo camera with APIs for stereo cameras.

5.3.3 Estimating the position and orientation of each marker

The next step of the recognition algorithm is to estimate the position and orientation of each marker. For this purpose, the recognition algorithm at first has to know which points of IR LEDs belong to whose marker. We call this process *grouping*, which groups together the points which belong to a marker.

Let P_{ir} be the 3D point of an IR LED, which is obtained by the process mentioned in Section 5.3.2. The algorithm calculates the Euclidean distances between P_{ir} and all the points of other IR LEDs, and searches points which satisfy the following condition; the distance between P_{ir} and the point is less than the predefined threshold distance. This predefined threshold distance can be determined by the actual lengths between IR LEDs in a marker (i.e., AB, AC, and BC in Figure 5.1). We set this threshold to 10[cm], because the actual length of AB, AC in Figure 5.1 is about 4[cm] and one of BC is about 6[cm]. If one or two such points are found, the algorithm groups P_{ir} and them together, and regards that they all consist of one marker. This process finishes until all the points of IR LEDs have been grouped.

After $grouping$, the algorithm calculates the position of each marker, which is obtained by the center of mass of all the points in each group. Additionally, if a group consists of three points of IR LEDs, the algorithm calculates the orientation of the marker. As described in Section 5.2, the orientation can be calculated as the vector sum of $\vec{BA}$ and $\vec{CA}$ in Figure 5.1. Therefore, the algorithm has to know which two points consist of the edge BC. For this purpose, the algorithm calculates all three possible lengths and regards BC as the maximum length among the three edges. After determining BC, the algorithm estimates the orientation of the marker with the process already mentioned. Otherwise, if a group consists of two points of IR LEDs, the orientation is set to what is taken over by the tracking method described in Section 5.3.4.

5.3.4 Tracking each marker

The tracking method in the recognition algorithm is very simple. The algorithm compares the estimated positions of the markers in the current camera frame and in the previous camera frame, and conjugates the positions which satisfy the condition that the Euclidean distance is less than the predefined threshold (20[cm] in the current implementation). If there is only one conjugate between the two camera frames, the algorithm regards the conjugate as the same marker. In this case, the algorithm considers that the marker moves from the conjugated position in the previous camera frame to the conjugated position in the current camera frame.

If the tracking is successful, the information of the marker, such as the ID number or the orientation, is taken over. Otherwise, the algorithm does not track the marker, and restart to get the information of the marker.

5.3.5 Identifying each marker

In order to identify each marker, the algorithm uses the blinking pattern which is assigned to each marker. As mentioned in Section 5.2, the IR LED which is arranged in the vertex A of the isosceles triangle in Figure 5.1 is blinking at a unique blinking pattern to each marker.

We defined the five blinking patterns for the identification as shown in Table 5.1. In Table 5.1, the symbols, “1” and “0”, represents turning the IR LED on/off during 0.2[sec]. One cycle of the blinking pattern consists of five symbols. In other words, it takes 1[sec] to blink an IR LED at one cycle. We determined the speed of blinking from the frame rate of the stereo cameras (about 10[fps] in the usual case).

User	Blinking Pattern	
1	00001	(0.8[sec] off and 0.2[sec] on)
2	00011	(0.6[sec] off and 0.4[sec] on)
3	00111	(0.4[sec] off and 0.6[sec] on)
4	01111	(0.2[sec] off and 0.8[sec] on)
5	11111	(always on)

Table 5.1. Blinking patterns of markers

The identification process works with the $grouping$ process and the tracking method mentioned above. If there are three points of IR LEDs in one group, the algorithm considers that the IR LED in the vertex A is on. Otherwise, if there are just two points, the algorithm considers that the IR LED in the vertex A is off. Through the tracking method, the history of the status of the blinking IR LED is also taken over. The identification process identifies each marker with the blinking history. Ideally, the algorithm can identify each marker in 1[sec].

Chapter 6

FILE TRANSFER SYSTEM

In this chapter, we present the system for file transfer. We describe the network configuration and the software for file transfer.

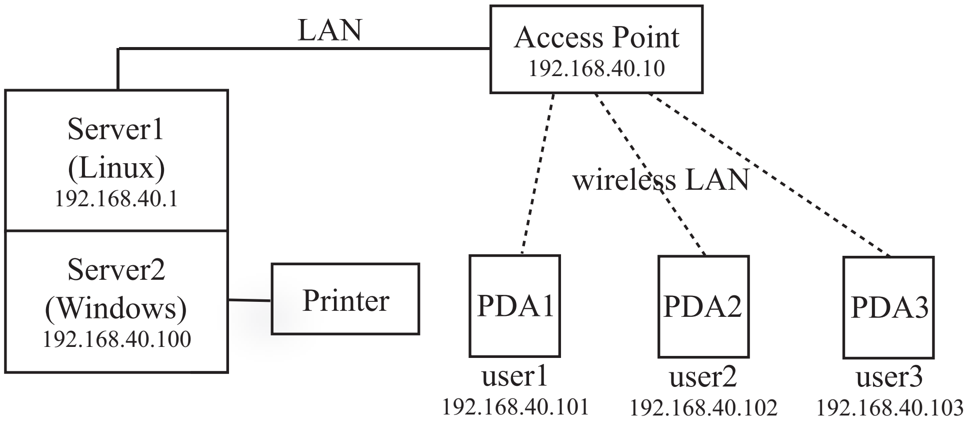


Figure 6.1. The component devices in Toss-It

6.1 System Architecture

As mentioned in Chapter 3, there are several possible technologies, such as peer-to-peer (P2P) or ad-hoc network, as the network technology in Toss-It. However, it does not matter to this thesis which technologies are used in Toss-It. Therefore, we use a wireless LAN because most of recent computer devices support the wireless LAN technology. We also prepare server computers. One is for facilitating configurations for the network, and the other is for processing calculations for gesture recognition and location recognition.

Figure 6.1 shows one example of the system components in Toss-It. The IP addresses are allocated to the devices in advance. A device ID is also allocated to each device. Server2 maintains the database table which can associate devices IDs with IP addresses. A DNS daemon is running on Server1, and Server2 is for processing calculations for gesture recognition and location recognition. In the following sections, we will present software which runs on clients, especially PDAs, and Server2.

6.2 Software for clients

The software which runs on clients, especially PDAs, has two main functions; collecting data from the sensors and sending/receiving files. We will explain the details of these two functions in this section.

6.2.1 Collecting data from the sensors

The microprocessor on the circuit board for inertial sensors, which is mentioned in Section 4.1, transmits data from sensors in packets of a fixed size. As shown in Figure 6.2, the packet is made up of 12 [bytes]. The size of each sensor is 1 [byte]. Therefore, the packet has 8 [bytes] data for accelerations and 3 [bytes] data for angular velocities. There is 1 [byte] dummy data in

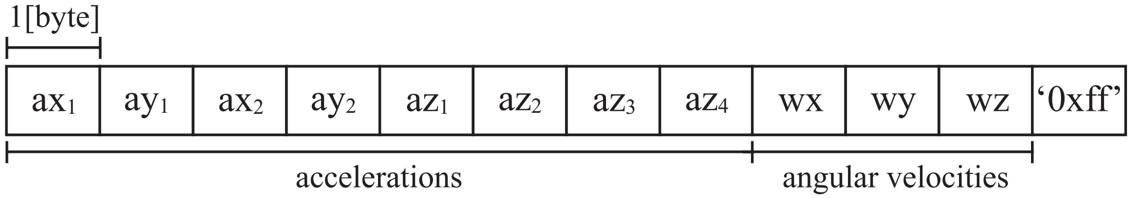


Figure 6.2. A packet of data from the sensors

the packet. This dummy data is used as a packet separator, which is used to separate one data packet from the following data packet.

The software running on PDAs receives packets from the microprocessor via a serial communication and sends them to Server2 through a socket communication. In Server2, calculation for gesture recognition is carried out as described in Section 4.2.

6.2.2 Sending/Receiving files

If Toss-It recognizes a "toss" or "swing" action and determines the receivers, the software on PDAs sends the selected files via Server2. We will explain the details with an example.

Suppose that user1 sends a file to user2. User1 selects a file to be sent on his PDA, and conducts a "toss" action toward user2. If Toss-It has recognized the action, Server2 returns the device ID of the receiver (i.e., the device ID of the user2's PDA) to the software on the user1's PDA. Next, the user1's PDA sends the file to Server2 with the receiver's device ID through a socket communication. Server2 receives the file and searches the receiver's IP address with the database table. Then, Server2 sends the file to the user2's PDA through a socket communication.

This is the case of the unicast transfer (Figure 3.1(a), 3.1(b) and 3.1(c)). In the case of the multicast transfer (Figure 3.1(d)), the process of file transfer is similar. The difference is that Server2 sends the file to multiple devices.

6.2.3 GUI of the software

Figure 6.3 shows the GUI of the software running on PDAs. On this GUI, users can select files to be sent. When a file is received, the file is shown in the center of the GUI. In the current implementation, the software only can deal with image files (e.g., GIF, JPG, or PNG files). However, it is not difficult that the software supports other kinds of files.

6.3 Software for servers

The software which runs on the servers has the following functions.

- **Function 1**: Recognizing users' gestures (mentioned in Section 4.2)

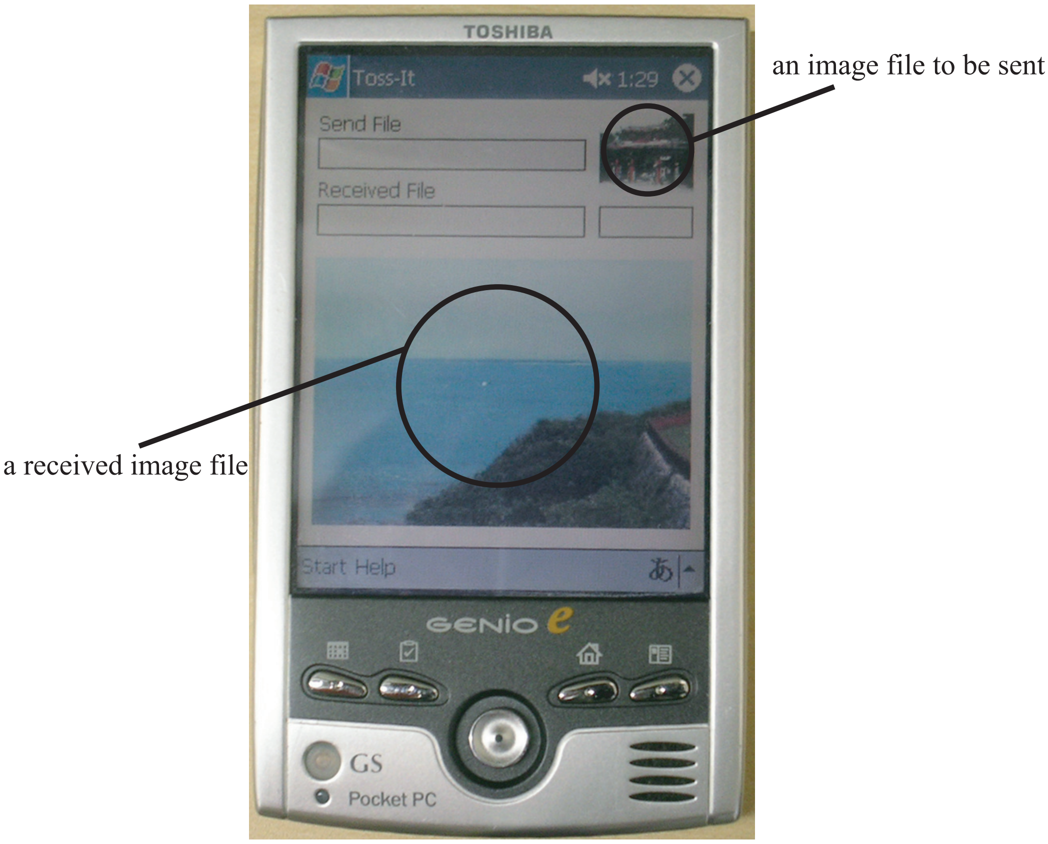
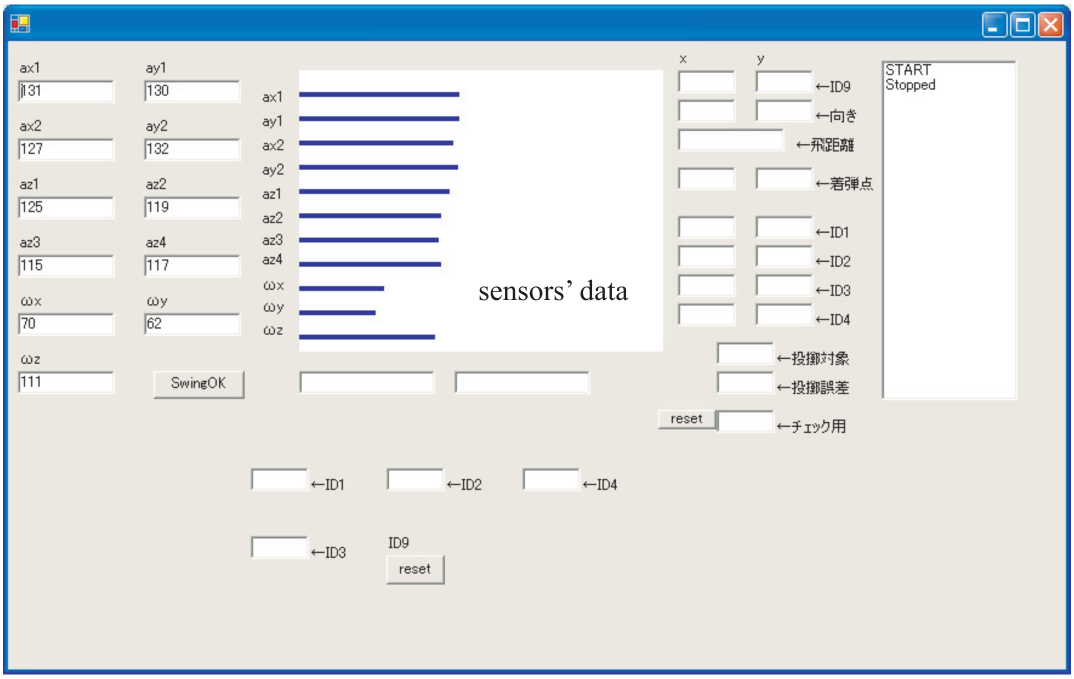


Figure 6.3. GUI of the software on a PDA

- **Function 2**: Recognizing users’ location (mentioned in Section 5.3)
- **Function 3**: Receiving packets of sensors’ data
- **Function 4**: Relaying files
- **Function 5**: Maintaining the database table

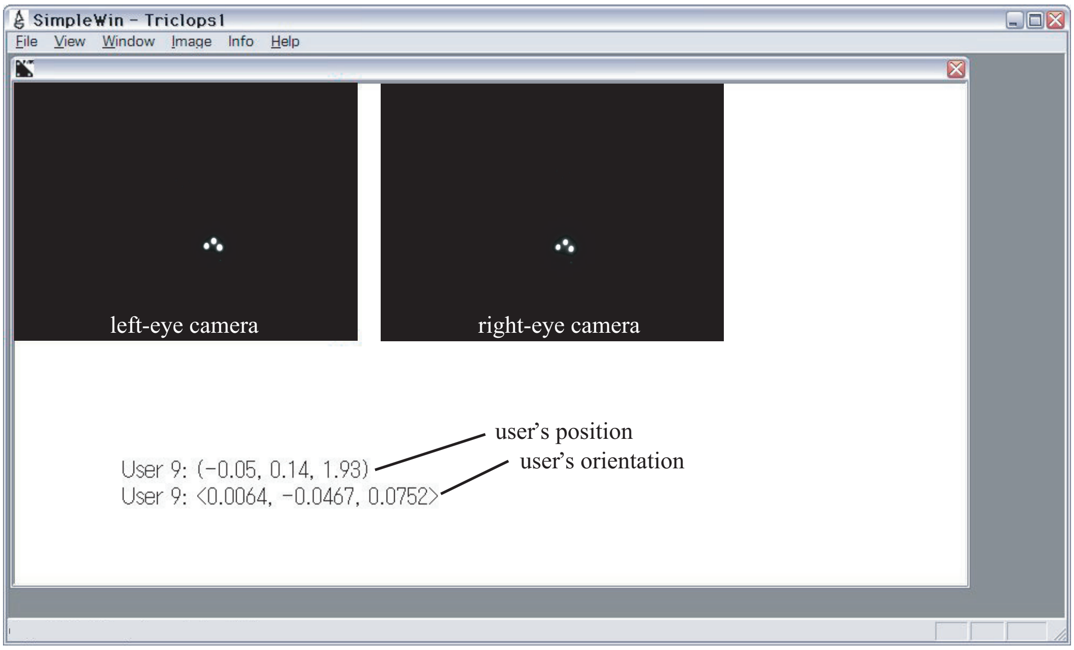
Each function is programmed as one module for flexible developments. Moreover, we have made GUIs for modules of Function 1 and Function 2. The GUIs are shown in Figure 6.4 and 6.5.



ax1 131
ay1 130
ax2 127
ay2 132
az1 125
az2 119
az3 115
az4 117
ωx 70
ωy 62
ωz 111

START
Stopped

Figure 6.4. GUI of the software for gesture recognition



User 9: (−0.05, 0.14, 1.93)
User 9: <0.0064, −0.0467, 0.0752>

Figure 6.5. GUI of the software for location recognition

Chapter 7

EXPERIMENTS AND EVALUATIONS

In this chapter, we describe the experiments for gesture recognition and location recognition in order to evaluate them. This chapter also includes a user study which was conducted to evaluate the whole system of Toss-It.

7.1 Experiments for Gesture Recognition

7.1.1 Experiments

Through several informal experiments, we observed that Toss-It could easily distinguish between a “toss” (vertical move) and a “swing” (horizontal move). We, therefore, evaluated Toss-It on how accurately it could recognize receivers with “toss” and “swing” actions.

Six subjects (all male, five right-handed and one left-handed) participated in the following three experiments. In the first experiment, the subjects were asked to conduct a “toss” action and send information to devices placed at three different locations (1[m], 2[m], and 3[m] away from a subject). In the second experiment, they were asked to do the same things in the first experiment. The difference is that we made another assumption in the estimation algorithm mentioned in Section 4.2.3. The assumption is that θ in the equation (4.9) always equals to 45[deg]. In the third experiment, the subjects were asked to conduct a three different horizontal “swing” with their PDAs (45[deg], 90[deg], and 135[deg]). Each subject repeated “toss” and “swing” actions 25 times for each of the three locations and three angles, respectively.

Additionally, we determined to observe actual distances when users tossed a box of which the size and weight were almost the same as a PDA for the purpose of comparison. We asked two subjects (male, right-handed) to toss the box at three different locations (i.e., 1[m], 2[m] and 3[m] away from a subject). We calculated the average and standard deviations from the result.

7.1.2 Results of the recognition of a “toss” action

Figure 7.1 shows the distribution of the estimated distance of a “toss” by subjects in the first experiment. From this figure, we can find that there are peaks between 0.50[m] to 0.75[m] for subjects’ one-meter-toss trials, those between 1.75[m] to 2.25[m] for their two-meter-toss trials, and those between 2.75[m] to 3.00[m] for their three-meter-toss trials. The average and standard deviation for each target distance are described in Table 7.1. Differences of “toss” actions between two target distances (1[m] and 2[m], or 2[m] and 3[m]) proved to be statistically significant by a Welch’s t-test (two-tailed, $p<.01$).

Figure 7.2 shows the distribution of the estimated distance of a “toss” by subjects in the second experiment. From this figure, we can find that there are peaks between 0.75[m] to 1.00[m] for subjects’ one-meter-toss trials, those between 1.75[m] to 2.00[m] for their two-meter-toss trials, and those between 2.25[m] to 2.50[m] for their three-meter-toss trials. The average and standard deviation for each target distance are described in Table 7.2. The standard deviations are less than ones in the first experiment. Differences of “toss” actions between two target distances (1[m] and 2[m], or 2[m] and 3[m]) also proved to be statistically significant by a Welch’s t-test (two-tailed, $p<.01$).

Table 7.3 displays the average and standard deviation for each target when subjects tossed the box. As compared to these values, the results of the estimation are not good. We will discuss about the reasons in Chapter 8.

Target distance[m]	Average[m]	Standard Deviation
1	0.995	0.683
2	1.91	1.01
3	3.02	1.35

Table 7.1. Average and standard deviation for each target distance in the first experiment

Target distance[m]	Average[m]	Standard Deviation
1	1.06	0.418
2	1.90	0.513
3	2.88	0.806

Table 7.2. Average and standard deviation for each target distance in the second experiment

Target distance[m]	Average[m]	Standard Deviation
1	1.01	0.0631
2	1.97	0.108
3	3.05	0.195

Table 7.3. Average and standard deviation for each target distance when subjects tossed the actual box

Target angle[deg]	Average[deg]	Standard Deviation
45	54.2	14.8
90	89.2	14.3
135	129	8.80

Table 7.4. Average and standard deviation for each target angle in the third experiment

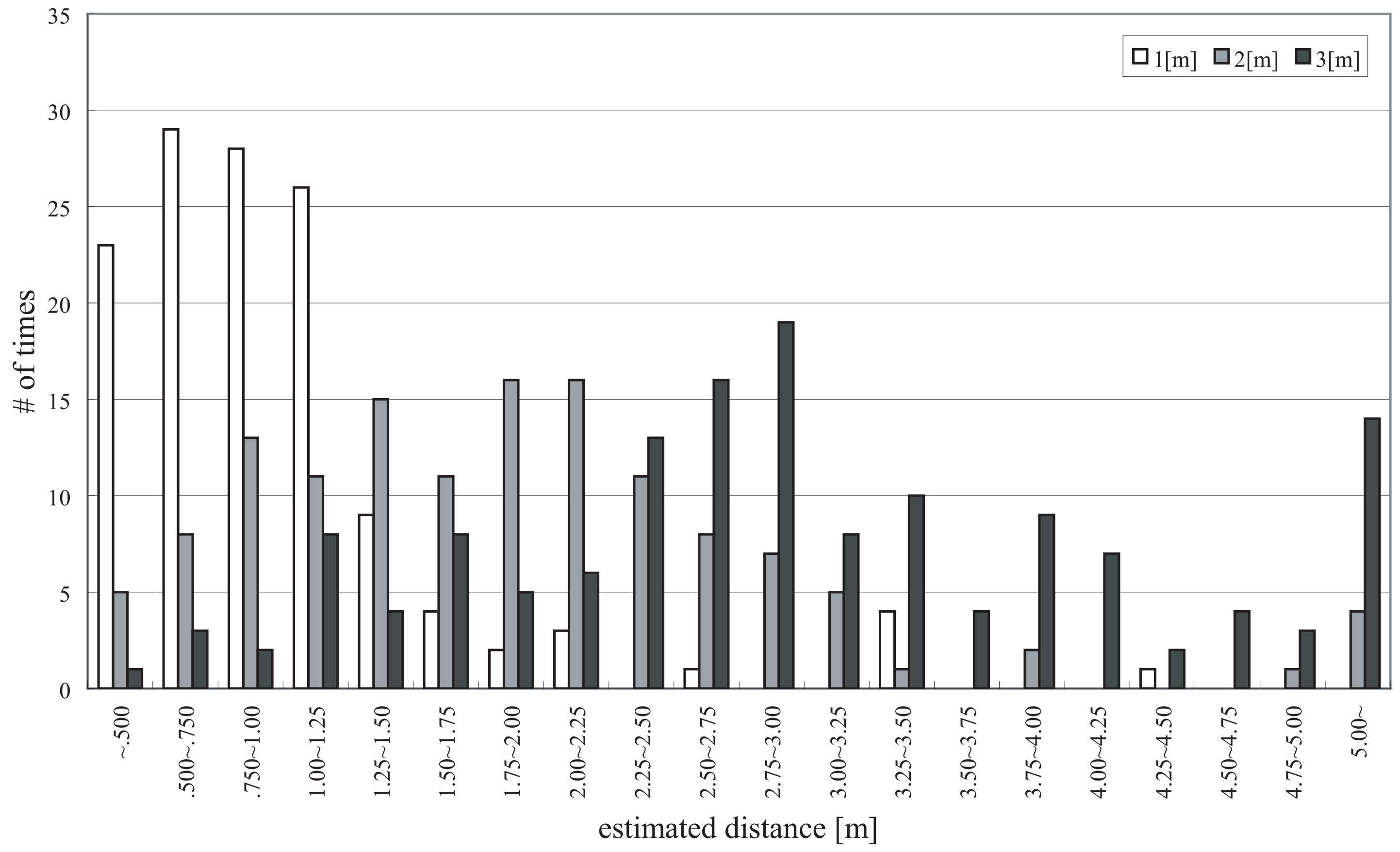


Figure 7.1. A distribution of the estimated distances in the first experiment

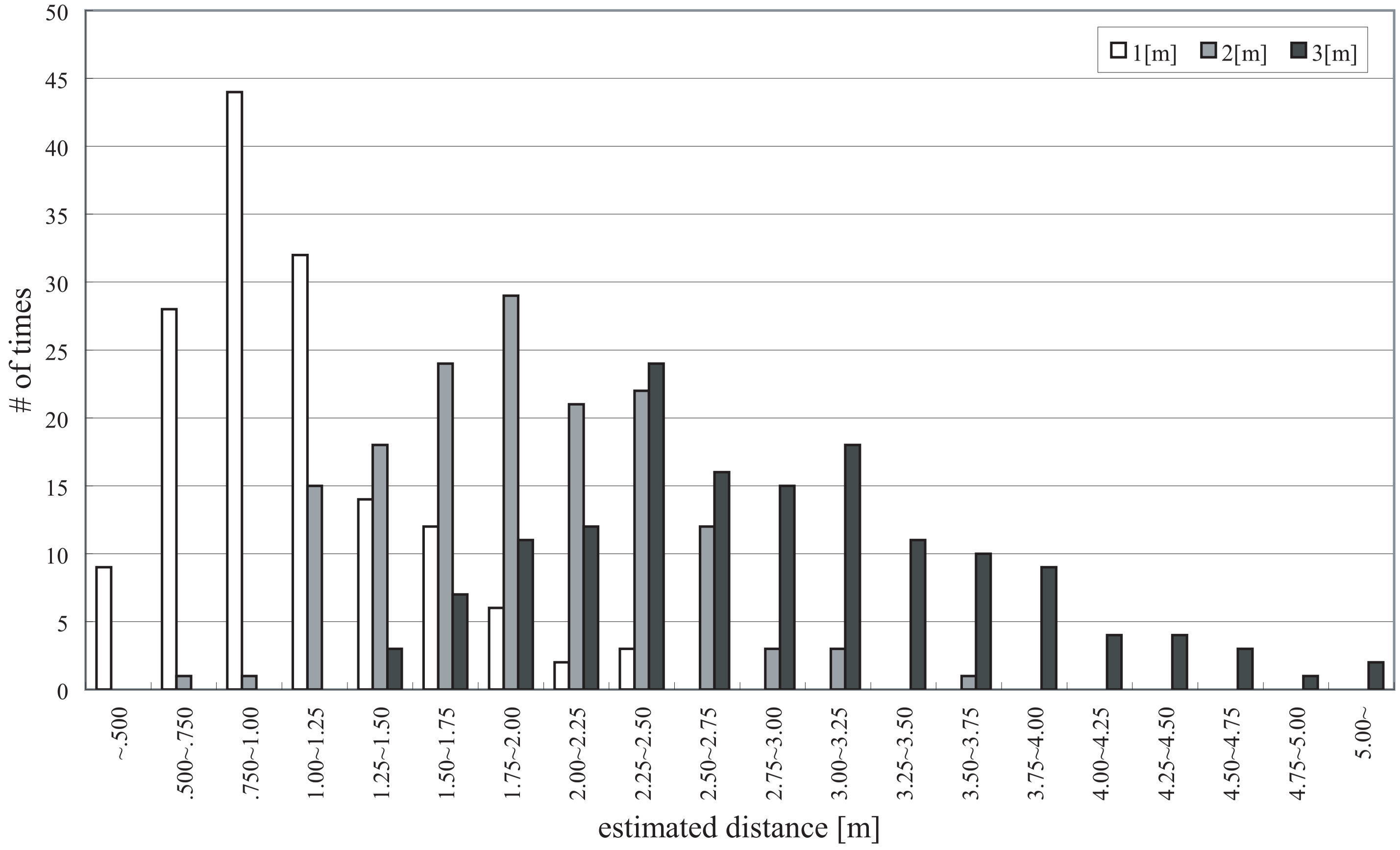


Figure 7.2. A distribution of the estimated distances in the second experiment

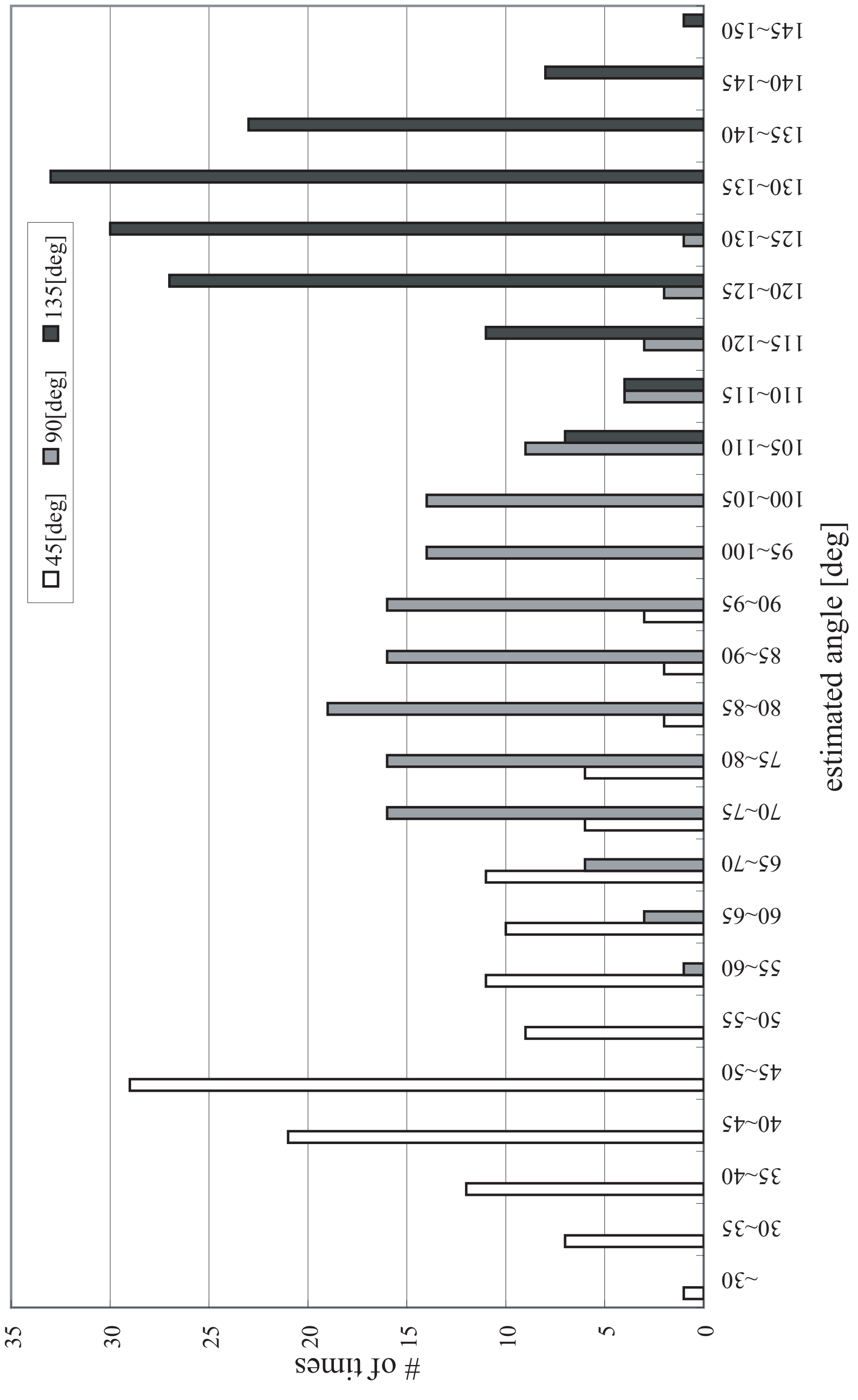


Figure 7.3. A distribution of the estimated angles in the third experiment

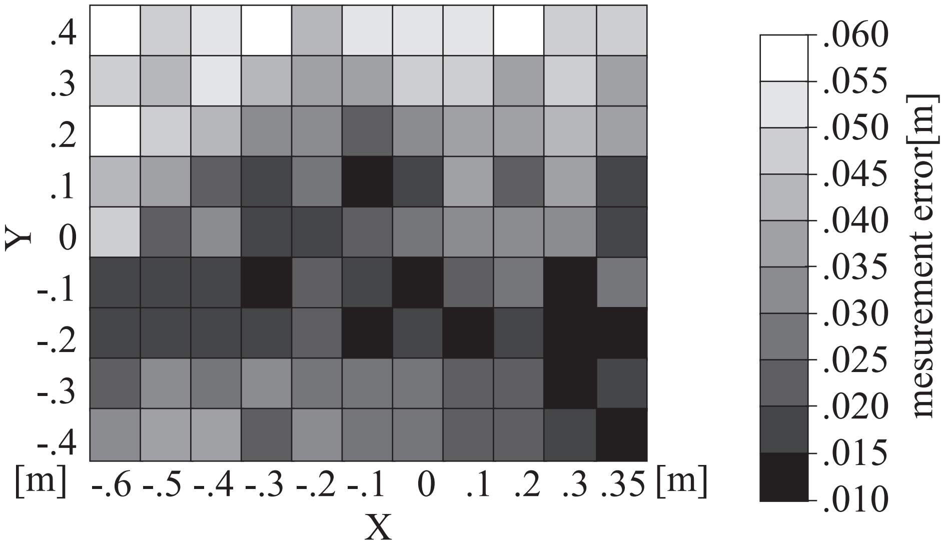


Figure 7.4. The error distribution in the position estimation

7.1.3 Results of the recognition of a “swing” action

Figure 7.3 shows the distribution of an estimated angle of a “swing” by the subjects. From Figure 5, we can find that there are peaks between 45[deg] to 50[deg] for subjects’ 45-degree-swing trials, those between 80[deg] to 85[deg] for their 90-degree-swing trials, and those between 130[deg] to 135[deg] for their 135-degree-swing trials. The average and standard deviation for each target angle are described in Table 2. Differences of “swing” actions between two target angles (45[deg] and 90[deg], or 90[deg] and 135[deg]) proved to be statistically significant by a Welch’s t-test (two-tailed, $p<.01$).

7.2 Experiments for Location Recognition

7.2.1 Experiments

We conducted an experiment for the location recognition method mentioned in Chapter 5. In the experiment, we recorded the estimated values of the position and orientation of a marker at a distance of 1.5 [m] from the camera, and calculated the measurement errors. Because the stereo cameras are installed on the ceiling to look down the floor straightforward, orientations on a plane parallel to the floor are important. Therefore, when we measured orientations, we set the marker on a plane parallel to the floor. We also confirmed how accurately the markers were identified.

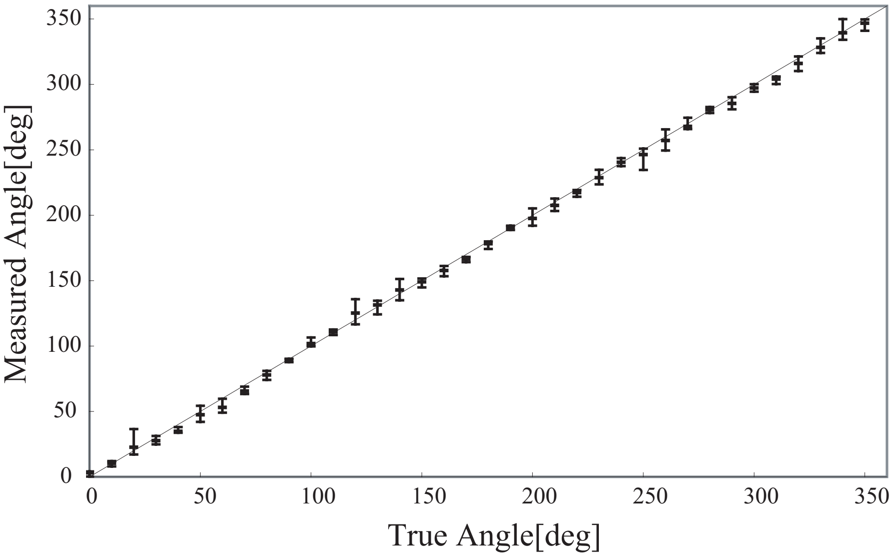


Figure 7.5. The error distribution in the orientation estimation (at the center of the camera image)

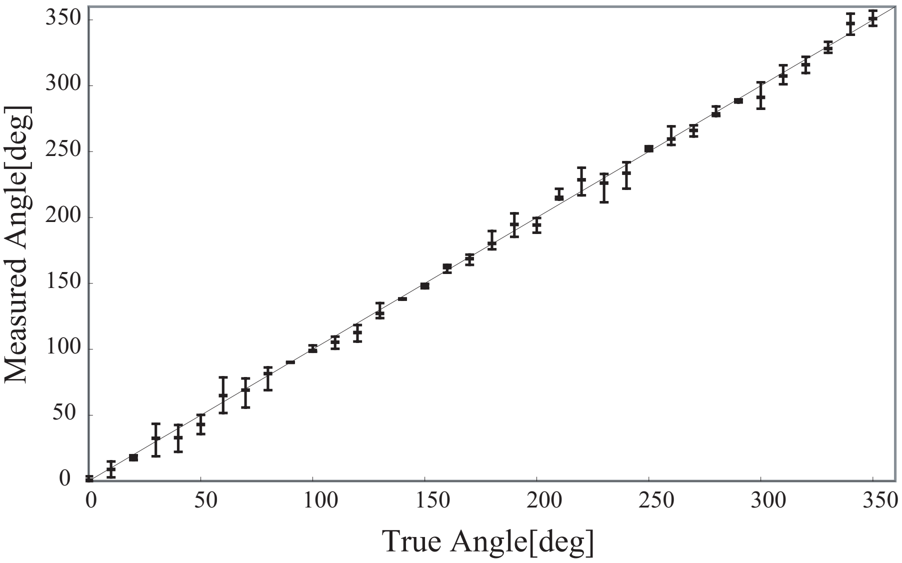


Figure 7.6. The error distribution in the orientation estimation (at the edge of the camera image)

7.2.2 Results of the estimation of positions

Figure 7.4 shows the error distribution in the position estimation. As shown in Figure 7.4, the measurement errors increase when a marker is set further from the center of the camera. The maximum value of the measurement errors is about 6 [cm] at the edge of the camera image.

7.2.3 Results of the estimation of orientations

Figure 7.5 and 7.6 show the error distributions in the orientation estimation. Figure 7.5 shows the distribution at the center of the camera image, and Figure 7.6 shows the distribution at the edge of the camera image. As shown in Figure 7.6, Even though the variation of the measurement errors is maximized at the edge of the center image, it is about 10 [deg].

7.2.4 Results of the identification

We confirmed whether the location recognition method can distinguish the blinking patterns which are shown in Table 5.1 for the identification. As the result of the experiment, we observed that the location recognition method can distinguish the blinking patterns in less than 2 [sec]. When a user conducted a rapid movement, such as a “toss” or “swing” action, the location recognition method couldn’t acquire the blinking pattern. However, when a user stayed still after the movement, it could acquire the blinking pattern again, and could identify the marker again.

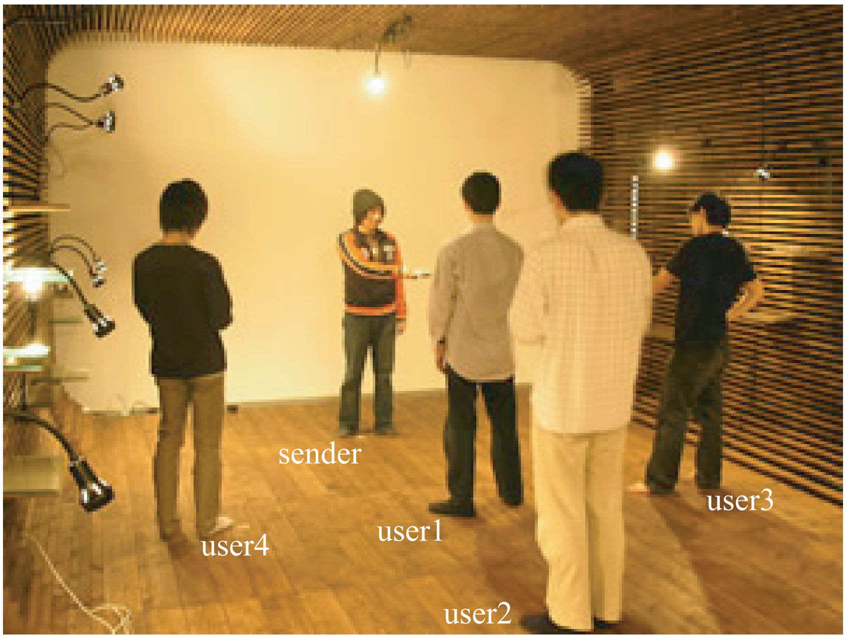


Figure 7.7. Scene in the user study

7.3 User Study

We conducted a user study for evaluating the whole system of Toss-It as shown in Figure 7.7. Table 7.5 and Figure 7.8 show the experimental setting for evaluating Toss-It. Five subjects (male, right-handed) participated in the user studies, and four of them were asked to conduct the following tasks: E1, E2, E3, and E4 to transfer information by "toss" actions to one receiver, and E5, E6, E7, and E8 to transfer information by "swing" actions to multiple receivers. The subjects were asked to conduct each of the eight tasks ten times. The results are summarized as Table 7.6. In the studies, each trial of the tasks by the subjects was judged as "success", if all and only the target users received the information. (e.g. in E6, user1, user2, and user4). In the experiments of "toss" actions (E1, E2, E3 and E4), Toss-It transferred information to the nearest person within a two-meter radius from a landing point estimated through a sender's toss action (If no person existed in the area, no information transfer was done.). The analyses of the user studies clarified the following issues:

- Distance estimation errors: In 10.0[%] of all the trials in E1, Toss-It sent information to user2, because estimated distances of toss actions were between 2.25[m] and 5[m]. In 2.5[%] of all the trials in E2, estimated distances by Toss-It were between 5[m] and 8[m], and about 1[m] in 20.0[%] of the trials. This kind of failures also occurred in E3 (15.0[%]) and E4 (2.5[%]).

- Orientation recognition errors: In 10.0[%] of all the trials in E1, Toss-It sent information to user4. This means that the recognition error of a sender's orientation was more than

Experiment	Action	Target
E1	Toss	user1
E2	Toss	user2
E3	Toss	user3
E4	Toss	user4
E5	Swing	all
E6	Swing	user1, user2, user4
E7	Swing	all
E8	Swing	user1, user2, user3

Table 7.5. Experiments of “toss” and “swing” actions in the user study

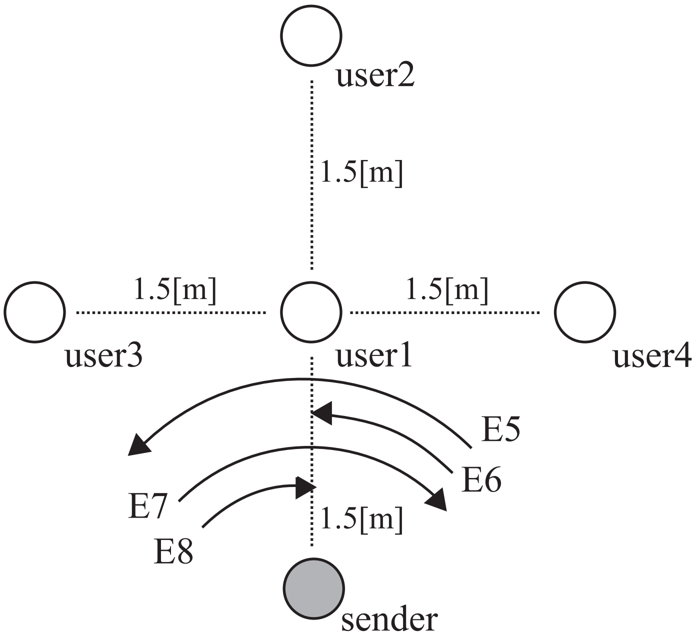


Figure 7.8. The positions of the users in the user study, and the start points and swing angles in the four experiments of “swing” actions

Experiment	Transfer Success Rate[%]
E1	75.0
E2	75.0
E3	80.0
E4	70.0
E5	85.0
E6	60.0
E7	90.0
E8	55.0

Table 7.6. Success rates in each experiment in the user study

20 degrees (about a half of the angle formed by the lines from the sender to user1 and user4), which is much larger than the maximum orientation recognition error (less than 10 degrees). Video analyses of the user studies clarified that although the sender conducted toss actions by exactly facing toward the target receiver, his PDA did not always direct exactly toward the receiver. This type of failures also happened in the trials in E2 (5.0[%]), E3 (10.0[%]), and E4 (20.0[%]).

- Position recognition errors: In some cases, information transfer to the target person failed because of the mixture of distance and position estimation errors: although Toss-It could have identified a correct receiver estimated through a sender's toss action, it failed due to the recognition errors of the sender's and the receiver's positions. However, this kind of failures did not often occur.

- Features in swing actions: The success transfer rates of E5 and E7 was higher than those of the other tasks, because the senders often swung their PDAs around themselves a larger angle than necessary. However, the senders in E6 and E8 also swung their PDAs a larger angle than necessary, which deteriorated its transfer success rate.

Chapter 8

DISCUSSIONS

In this chapter, we make some discussions from the results of the experiments. We discuss about the gesture recognition method, the location recognition method, and the user study experiments.

8.1 Discussions about Gesture Recognition

In the two experiments for recognition of a “toss” action, the method was proved to distinguish adjacent targets (1[m] and 2[m], 2[m] and 3[m]) from the viewpoint of statistics. However, the standard deviations in Table 7.1 and 7.2 were much bigger than ones in Table 7.3. This indicates that the estimated values of flying distance varied much more widely than the actual values. The user study experiment also suggested that more precise estimation of users’ “toss” and “swing” actions are necessary.

From careful observations of “toss” actions, we discovered users usually made snaps in order to throw it fine when they threw a box. The current implementation of the gesture recognition method cannot deal with snap actions well, because snap actions are usually very rapid movements. It may be possible to increase the accuracy of estimation of flying distances, if we improve the hardware and the recognition algorithms in order to recognize snap actions.

8.2 Discussions about Location Recognition

From the experiments for location recognition and the user study, the accuracy of the position estimation was proved to be enough. However, we should make some improvements for the orientation estimation, because an orientation recognition error is one of the major factors which cause information transfer less successful.

In the current implementation, the location recognition method cannot support simultaneous use by more than five people. If we prepare more blinking patterns of markers, the location recognition method could deal with location information of more people. However, the numbers of recognizable blinking patterns are constrained by the blinking speed of IR LEDs and the frame rate of the stereo cameras. In this case, the frame rate is thought the most critical bottleneck point, because it is now limited to about 10[fps]. Therefore, in order to enable this location recognition method to recognize the position and orientations of much more than five people (e.g., 20 people), we must improve the performance of the recognition algorithm.

8.3 Discussions about User Study

It may be effective to improve the transfer success rate by capturing all users’ gestures or utilizing contextual information. Suppose that receivers express their intention to receive information by tilting their own PDA vertically. Toss-It first identifies candidates of the receivers conducting the tilting gesture, and then sends information to some of them determined by sender’s toss or swing actions. It may also be possible to utilize orientations of users’ PDAs. For example, when their PDAs do not face toward a sender, Toss-It judges that they have no intention to receive information from the sender. Capturing users’ gestures or orientations also effective for blocking the users to receive unnecessary information.

By utilizing situational and contextual information, the transfer success rate will be improved. For example, it is reasonable that Toss-It regards user2 as the correct receiver in E2, when the estimated location is much beyond the user2's location. If Toss-It do not restrict candidates of receivers to those within a two-meter radius from a landing point calculated through a sender's toss action, the transfer success rate of E2 increases to 77.5[%].

The current version of Toss-It should be improved by testing the methods described above. We believe that through the improvements, the proposed information transfer techniques will become more practical and usable.

Chapter 9

CONCLUSIONS

We describe some directions for future research in the first section of this last chapter. We then summarize this thesis and make conclusions.

9.1 Future Works

Several issues on Toss-It still remain to be investigated. The improvement of the precise estimation of “toss” and “swing” actions is one of the most important issues. In the current implementations, the estimation values of “toss” and “swing” actions are less precise as compared to the actual values. As discussed in Chapter 8, we have to make some improvements to the hardware and the recognition algorithms.

In order to fit in users’ intention, it is another possible approach that we use an adaptive algorithm for the estimation process. We can think actions of different users have different characteristics. Moreover, mobile devices are usually used for personal use. Therefore, it is reasonable that each mobile device learns the characteristics of the owner’s actions and processes the estimation based on what it learns. In this approach, users have to let mobile devices learn their actions in the beginning. However, the estimation process can be made more adaptive to each owner, which is one of the biggest merits in this approach.

Another important issue is about feedback. The current version of Toss-It gives receivers only visual feedback (i.e., mobile devices show the received information on the displays). In order to enhance the intuitiveness of Toss-It, it must be investigated what kind of feedback (e.g., auditory or tactile) should be given to senders and receivers.

We will also explore possibilities for various applications based on the Toss-It architecture. One example is a universal remote controllers (for manipulating multiple devices simultaneously by a “swing” action, or a device behind an obstacle with a “toss” action) like XWand (Wilson and Shafer, 2003). Another possible application is for entertainment games. Recently, some entertainment games which combine human gestures or motion have been released. These games make players feel more realistic and more immersed. Computer-supported collaborative play (CSCP) is also an interesting concept in the field of entertainment computing (Ishii et al., 1999). With multiple mobile devices and the Toss-It architecture, we can propose novel entertainment computing environments in which multiple players can enjoy games by using their own mobile devices collaboratively.

For these possible applications, we also have to make the Toss-It architecture to recognize more precise and complicated gestures continuously. Calculating the trajectories of gestures is one of the good approaches for this purpose, because systems can recognize not only the gestures themselves but some characteristics of the gestures, such as velocities and sizes. However, output of inertial sensors naturally includes some little errors. Integral calculations accumulate these errors to cause significant recognition errors when the system recognizes trajectories of gestures. Therefore, continuous adjustments in integral calculations are necessary to decrease the effects of these errors. One possible approach is to make appropriate calibrations for inertial sensors with the pose of the device (Kim and Golnaraghi, 2004). For example, if the pose estimated from the inertial sensors is significantly different from ones estimated from other pose recognition methods, the system automatically calibrates the status of the inertial sensors and initializes integral calculations.

We also have a plan to conduct intensive usability studies in order to evaluate Toss-It as a user interface for mobile devices. There are several methods for evaluating Toss-It from the viewpoint of a user interface. For instance, we measure and compare necessary time to complete manipulations for information transfer in Toss-It and conventional techniques (e.g., with a memory card or through an infrared communication). Subjective analysis is another evaluation method. We are going to evaluate Toss-It by combining these evaluation methods together.

9.2 Summary

In this thesis, we described Toss-It, intuitive interaction techniques for mobile devices with human gestures. In Chapter 1, we remarked the introduction and the contributions of this research project.

We showed some related works to Toss-It in Chapter 2. We categorized them into three groups; Intuitive interfaces for mobile devices, interfaces for supporting information transfer manipulations, and gestural input techniques. We compared our Toss-It projects with other researches and clarified the differences.

In Chapter 3, we presented the concept of Toss-It. We also discussed why a “toss” or “swing” action is one of the appropriate gestures in transferring digital information in mobile devices. In the last section of this chapter, we listed up three major technical requirements for realizing the concept of Toss-It.

We presented the gesture recognition method in Chapter 4. In the first section of this chapter, we explained about the hardware for the gesture recognition. We made the prototype circuit board with inertial sensors. Through several informal experiments, we discovered there were several problems to be solved in the prototype circuit board. We then made the improved circuit board in which four accelerometer and three gyroscopes are embedded. In the second section, we explained about the recognition algorithm. The recognition process can be divided into three parts; Eliciting “toss” or “swing” actions, estimating the strength of a “toss” action, and estimating the trajectory of a “swing” action. We described the details of how each part works.

We described the location recognition method in Chapter 5. We substitute a vision-based location recognition method in the current implementation of Toss-It. We first presented a marker with IR LEDs for location recognition. We then explained how to estimate positions and orientations of multiple users with the marker and a stereo camera. We also explained how to identify each marker.

We explained about the system for transferring digital information in Chapter 6. We use a wireless LAN and three kinds of servers. We described the system architecture and showed GUIs for PDAs and servers.

In Chapter 7, we present the experiments for evaluation. We conducted experiments for the gesture recognition method in order to evaluate how accurately the method recognized actions. We also evaluate the accuracy of the estimation of a user’s position and orientation in the location recognition method. We then conducted a user study for evaluating the whole system.

In Chapter 8, we made some discussions about the gesture recognition method, the location recognition method and the results of the user study experiments. From these discussions, although several improvements would be needed, we are sure that the proposed information transfer techniques are practical and usable.

9.3 Conclusions

Our initial motivation of this thesis is to propose intuitive interaction techniques for mobile devices in information transfer. For this purpose, we applied “toss” and “swing” actions to information transfer operations. Toss-It enables users to transfer digital information by “toss” and “swing” actions as if they would toss a ball or deal cards to others. Toss-It is successful to combine real-world movements with electric-world operations.

Mark Weiser who introduced an interesting concept of a computing environment, named ubiquitous computing, wrote at the start of his famous article (Weiser, 1991) as following.

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

We also think computers are more deeply interwoven into the real world in the future. In such environments, we think it is very important that computers have user interfaces with which people can manipulate them as they act in the real world. This makes people less aware of the existence of computers. We are sure that user interfaces like what Toss-It provides will play an important role in realizing computer-woven everyday life.

LIST OF PUBLICATIONS

International conferences (reviewed)

Koji Yatani, Koiti Tamura, Hiroki Keiichi, Masanori Sugimoto, and Hiromichi Hashizume, “Toss-It: Intuitive Information Transfer Techniques for Mobile Devices”, *the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI 2005)* (to appear).

Koji Yatani, Koiti Tamura, Masanori Sugimoto, and Hiromichi Hashizume, “Information Transfer Techniques for Mobile Devices by Toss and Swing Actions”, in *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications (WMSCA 2004)*, pp. 144–151, December 2004.

Domestic conferences (reviewed)

矢谷 浩司, 田村 晃一, 廣木 桂一, 杉本 雅則, 橋爪 宏達, “Toss-It: モバイルデバイスにおける「トス」や「振り」の動作を用いた情報の移動を直感的に実現するインタフェース技術”, インタラクション 2005 (to appear).

矢谷 浩司, 岸村 俊哉, 田村 晃一, 杉本 雅則, 橋爪 宏達, “モバイルデバイスを用いた「振る」ことによる情報の移動を実現するインタフェース”, インタラクション 2004, pp. 229–230, March 2004.

Domestic conferences (non-reviewed)

矢谷 浩司, 田村 晃一, 杉本 雅則, 橋爪 宏達, “ユーザの位置、方向、ジェスチャを認識することによるモバイルデバイスのための情報移動手法”, ヒューマンインタフェース学会研究会, vol. 6, No. 4, pp. 31–36, November 2004.

矢谷 浩司, 岸村 俊哉, 田村 晃一, 杉本 雅則, 橋爪 宏達, “Toss-It: モバイルデバイスにおける情報の移動を直感的に実現するインタフェース”, 電子情報通信学会 技術研究報告, vol. 104, No. 169, pp. 19–24, July 2004.

岸村 俊哉, 田村 晃一, 矢谷 浩司, 杉本 雅則, 橋爪 宏達, “人間の自然なジェスチャを利用した実世界指向端末”, 情報処理学会第 66 回全国大会, pp. 4-251–4-252, March 2004.

田村 晃一, 岸村 俊哉, 矢谷 浩司, 杉本 雅則, 橋爪 宏達, “複数センサを用いることによるジェスチャ入力可能な携帯情報端末の設計”, 情報処理学会第 66 回全国大会, pp. 4-253–4-254, March 2004.

REFERENCES

Ayatsuka, Y., Matsushita, N., and Rekimoto, J. (2000) “HyperPalette: A Hybrid Computing Environment for Small Computing Devices”, in *Extended Abstracts of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 133–134.

Bahl, P. and Padmanabhan, V. N. (2000) “RADAR: An In-Building RF-Based User Location and Tracking System”, in *Proceedings of the Annual Conference on Computer Communications*, pp. 775–784.

Bang, W. C., Chang, W., Kang, K. H., Choi, E. S., Potanin, A., and Kim, D. Y. (2003) “Self-contained Spatial Input Device for Wearable Computers”, in *Proceedings of the IEEE International Symposium on Wearable Computers*, pp. 26–34.

Bederson, B. B. and Hollan, J. D. (1994) “Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics”, in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 17–26.

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. (1993) “Toolglass and Magic Lenses: the See-through Interface”, in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pp. 73–80.

Brewster, S., Lumsden, J., Bell, M., Hall, M., and Tasker, S. (2003) “Multimodal ’Eyes-Free’ Interaction Techniques for Wearable Devices”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 473–480.

Brumitt, B., Meyers, B., Krumm, J., Kern, A., and Shafer, S. A. (2000) “EasyLiving: Technologies for Intelligent Environments”, in *Proceedings of International Symposium on Handheld and Ubiquitous*, pp. 12–29.

Chou, J. C. K. (1992) “Quaternion Kinematic and Dynamic Differential Equations”, *IEEE Transactions on Robotics and Automation*, Vol. 8, No. 1, pp. 53–64, February.

Fitzmaurice, G. W. (1993) “Situated Information Spaces and Spatially Aware Palmtop Computers”, *Communication of the ACM*, Vol. 36, No. 7, pp. 39–49.

Gandy, M., Starner, T., Auxier, J., and Ashbrook, D. (2000) “The Gesture Pendant: A Self-illuminating, Wearable, Infrared Computer Vision System for Home Automation Control and Medical Monitoring”, in *Proceedings of the IEEE International Symposium on Wearable Computers*, pp. 87–94.

Harrison, B. L., Fishkin, K. P., Gujar, A., Mochon, C., and Want, R. (1998) “Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 17–24.

Harter, A., Hopper, A., Steggles, P., Ward, A., and Webster, P. (2002) “The Anatomy of a Context-aware Application”, *Wireless Networks*, Vol. 8, No. 2/3, pp. 187–197.

Hightower, J. and Borriello, G. (2001) “Location Systems for Ubiquitous Computing”, *IEEE Computer*, Vol. 34, No. 8, pp. 57–66, August.

Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. (2000) “Sensing Techniques for Mobile Interaction”, in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 91–100.

Ishii, H., Wisneski, C., Orbanes, J., Chun, B., and Paradiso, J. (1999) “PingPongPlus: Design of an Athletic-tangible Interface for Computer-supported Cooperative Play”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 394–401.

Kim, A. and Golnaraghi, M. F. (2004) “Initial Calibration of an Inertial Measurement Unit Using an Optical Position Tracking System”, in *Proceedings of the IEEE Position Location and Navigation Symposium*, pp. 96–101.

Kohtake, N., Rekimoto, J., and Anzai, Y. (2001) “InfoPoint: A Device that Provides a Uniform User Interface to Allow Appliances to Work Together over a Network”, *Personal and Ubiquitous Computing*, Vol. 5, No. 4, pp. 264–274.

Kourogi, M. and Kurata, T. (2003) “Personal Positioning Based on Walking Locomotion Analysis with Self-Contained Sensors and a Wearable Camera”, in *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 103–112.

MacKenzie, I. S. and Soukoreff, R. W. (2002) “Text Entry for Mobile Computing: Models and Methods, Theory and Practice”, *Human-Computer Interaction*, Vol. 17, No. 8, pp. 147–198, August.

Marrin, T. (1997) “Possibilities for the Digital Baton as a General-Purpose Gestural Interface”, in *Extended Abstracts of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 311–312.

Moran, T. P. and Dourish, P. (2001) “Introduction to This Special Issue on Context-Aware Computing”, *Human-Computer Interaction*, Vol. 16, No. 2–4, pp. 87–95.

Pavlovic, V., Sharma, R., and Huang, T. S. (1997) “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 677–695.

Perlin, K. and Fox, D. (1993) “Pad: An Alternative Approach to the Computer Interface”, in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pp. 57–64.

Priyantha, N. B., Chakraborty, A., and Balakrishnan, H. (2000) “The Cricket Location-support System”, in *Proceedings of the Annual International Conference on Mobile Computing and Networking*, pp. 32–43.

Priyantha, N. B., Miu, A. K., Balakrishnan, H., and Teller, S. (2001) “The Cricket Compass for Context-aware Mobile Applications”, in *Proceedings of the Annual International Conference on Mobile Computing and Networking*, pp. 1–14.

Rekimoto, J. (1996) “Tilting Operations for Small Screen Interfaces”, in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 167–168.

Rekimoto, J. (1997) “Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments”, in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 31–39.

Rekimoto, J. (2001) “GestureWrist and GesturePad: Unobtrusive Wearable Interaction Devices”, in *Proceedings of the IEEE International Symposium on Wearable Computers*, pp. 21–27.

Schwesig, C., Poupyrev, I., and Mori, E. (2004) “Gummi: A Bendable Computer”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 263–270.

Swindells, C., Inkpen, K. M., Dill, J. C., and Tory, M. (2002) “That One There! Pointing to Establish Device Identity”, in *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, pp. 151–160.

Trucco, E. and Verri, A. (1998) *Introductory Techniques for 3-D Computer Vision*: Prentice Hall PTR.

Ullmer, B., Ishii, H., and Glas, D. (1998) “mediaBlocks: Physical Containers, Transports, and Controls for Online Media”, in *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, pp. 379–386.

Weiser, M. (1991) “The Computer for the 21th Century”, *Scientific American*, Vol. 265, No. 30, pp. 94–104.

Wilson, A. and Shafer, S. (2003) “XWand: UI for Intelligent Spaces”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 545–552.

Yee, K. P. (2003) “Peephole Displays: Pen Interaction on Spatially Aware Handheld Computers”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1–8.

Zimmerman, T. G., Lanier, J., Blanchard, C., Bryson, S., and Harvill, Y. (1987) “A Hand Gesture Interface Device”, in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 189–192.

金子 歩, 菅野 裕介, 矢谷 浩司, 杉本 雅則, 橋爪 宏達 (2005) 「超音波センサを用いた相対位置認識デバイスの設計」, 情報処理学会全国大会. (to appear).

菅野 裕介, 金子 歩, 矢谷 浩司, 杉本 雅則, 橋爪 宏達 (2005) 「超音波センサを用いた相対位置認識技術における通信及び測距技術」, 情報処理学会全国大会. (to appear).

谷口 慶治 (1996) 『画像処理工学 基礎編』, 共立出版株式会社.

宮川 透, 米沢 義道, 伊東 一典, 橋本 昌巳 (2002) 「筆記運動の 3 次元慣性計測による筆記パターンの再現」,『計測自動制御学会論文集』, Vol.38, No.1, pp.1–8, January.