

國立交通大學

資訊科學與工程研究所

碩士論文

基於手勢之次世代加護病房人機互動技術

Hand Gesture-based Human-computer Interaction for Next-generation

Intensive Care Unit

1896

研究生：廖偉成

指導教授：莊仁輝 教授

中華民國 一 百 零 一 年 七 月

基於手勢之次世代加護病房人機互動技術

**Hand Gesture-based Human-computer Interaction for Next-generation
Intensive Care Unit**

研究生：廖偉成

Student: Wei-Cheng Liao

指導教授：莊仁輝

Advisor: Jen-Hui Chuang

國立交通大學
資訊科學與工程研究所
碩士論文

A Thesis

Submitted to Department of Computer and Information Science

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer and Information Science

June 2012

Hsinchu, Taiwan, Republic of China

中華民國一百零一年七月

基於手勢之次世代加護病房人機互動技術

研究生：廖偉成

指導教授：莊仁輝 教授

國立交通大學資訊科學與工程研究所

摘要

醫院提供了醫療保健，而加護病房(Intensive care unit)是一個專門提供全面且持續的醫療單位。一般加護病房中，護理人員要取得病患的生理狀態資料，必須到護理站取得資料，這樣並不方便。如能將加護病房資訊顯示設備將安裝在病房旁，提供即時的資訊給予醫護人員。以提升醫護人員的決策的效率，使得醫護人員能夠得到立即的資訊。然而為了避免使用接觸式的控制造成的感染，本研究提出一套基於手勢的人機介面用於加護病房中。本研究使用 Kinect 深度攝影機當作輸入設備。系統將偵測是單手模式或是雙手模式，並透過建立背景模型針對場景中的環境資訊進行析。本研究，以基本的手勢進行開發，以達到瀏覽網頁、照片等資訊。在實驗結果中，我們成功達成的網頁基本的操作功能。未來計畫加入更多的手勢與使用者辨識，提供病患、家屬等人不同的功能，打造出次世代加護病房，提供更完善的照護。

Hand Gesture-based Human-computer Interaction for Next-generation Intensive Care Unit

Student: Wei-Cheng Liao

Advisor: Jen-Hui Chuang

Department of Computer and Information Science

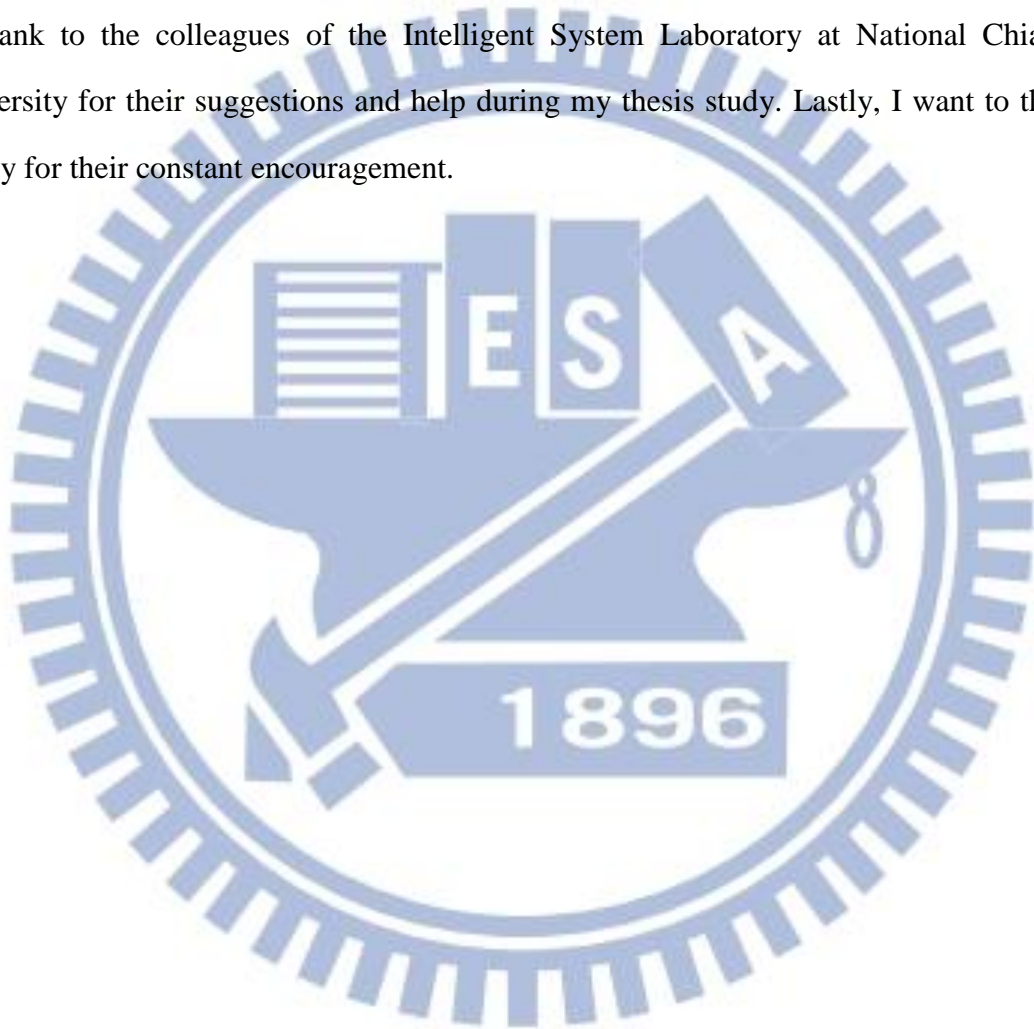
National Chiao Tung University

ABSTRACT

Hospital is one of the healthcare providers and intensive-care unit (ICU) is a specialized section of a hospital that provides comprehensive and continuous care. In ICU, it is not convenient that paramedics must walk to nursing station to retrieve the patient's health state from hospital database. If there is a display around a hospital bed which can show the patient's information, paramedics can instantly exam and browse the instant information. To avoid hand contacts controller that may result in infection. This study proposed a hand gesture-based human-computer interaction for the display in ICU which uses the Kinect range camera as input device. By building the background model and analyzing the scene information, the developed system will detect one-hand mode and two-hand mode before the hand gestures are recognized. We develop some basic gestures and users can use them to browse web pages, photos and other information. Experimental results show that basic browsing capabilities for ordinary web pages can be achieved by the developed hand gesture recognition system. In the future, we plan to add more hand gestures to provide patient and patients' family with different functions e.g. user identification for next-generation ICU.

ACKNOWLEDGEMENTS

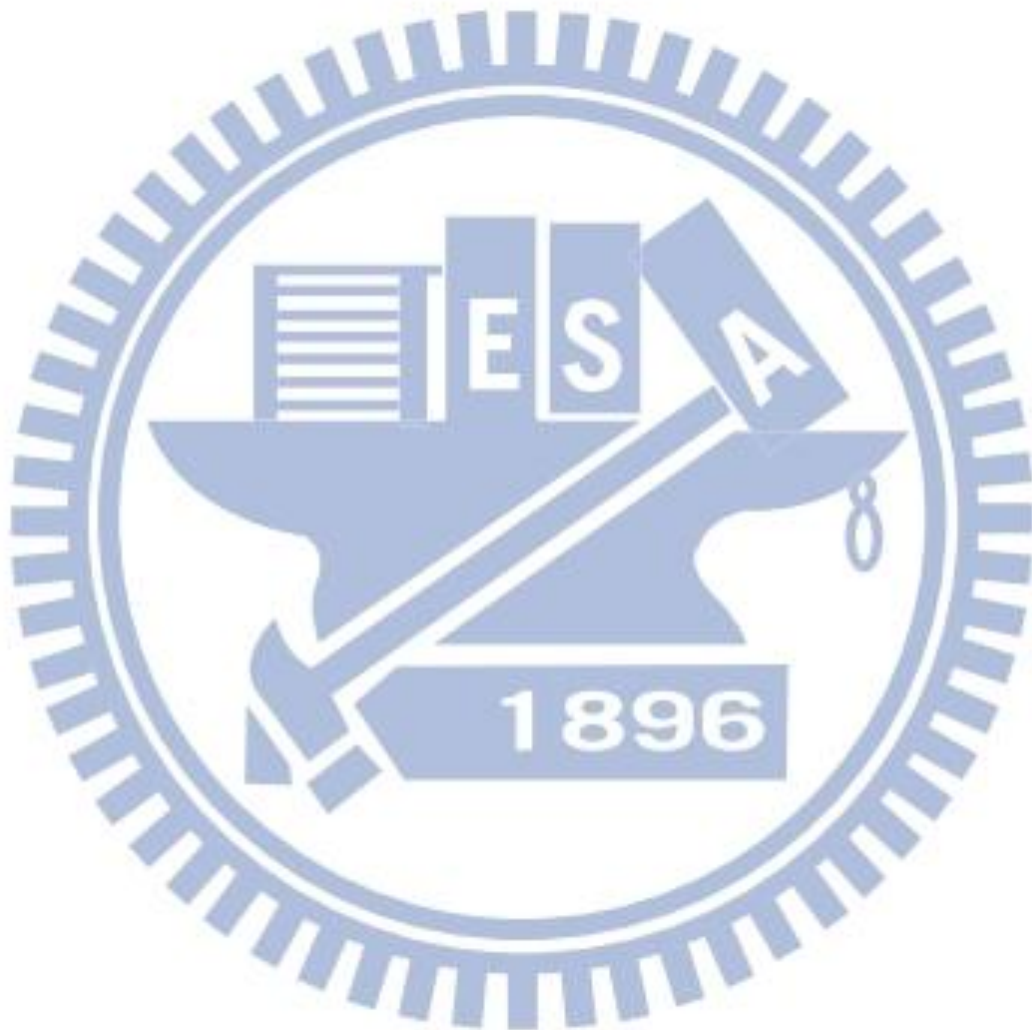
I am heartily thankful to my advisor, Dr. Jen-Hui Chung, whose encouragement, guidance and support from the initial to the final level enabled me to develop an understanding of the subject. I also thank to the members of my dissertation committee, Dr. Yen, Dr. Wang, Dr. Lai, for their advices and comments to help develop this research. I want to thank to the colleagues of the Intelligent System Laboratory at National Chiao Tung University for their suggestions and help during my thesis study. Lastly, I want to thank my family for their constant encouragement.



CONTENTS

摘要	i
ABSTRACT	ii
ACKNOWLEDGEMENTS	iii
CONTENTS	iv
LIST OF FIGURES	vi
LIST OF TABLES	viii
Chapter 1. Introduction	1
1.1 Motivation	2
1.2 Review of Related Work	3
1.2.1 Wearable Sensors	3
1.2.2 Interfaces Based on Computer Vision	4
1.3 Thesis Organization	8
Chapter 2. System Description	9
2.1 Input Device	10
2.2 Framework and Middleware	12
2.3 Hand Gesture Control System	14
2.4 Software Stack	15
Chapter 3. Hand Gesture-Based HCI	16
3.1 Hand Gestures Definition	16
3.2 System Initialization and Data Acquisition	18
3.2.1 Acquisition of Depth Data	18
3.2.2 Hand Point Detection	18
3.3 Feature Extraction	19
3.4 Hand Gestures Recognition	21
Chapter 4. System Implementation and Experimental Results	22

4.1	System Implementation.....	22
4.2	Experimental Results	23
Chapter 5. Conclusions		36
Appendix A The Moving Region		37
References.....		39



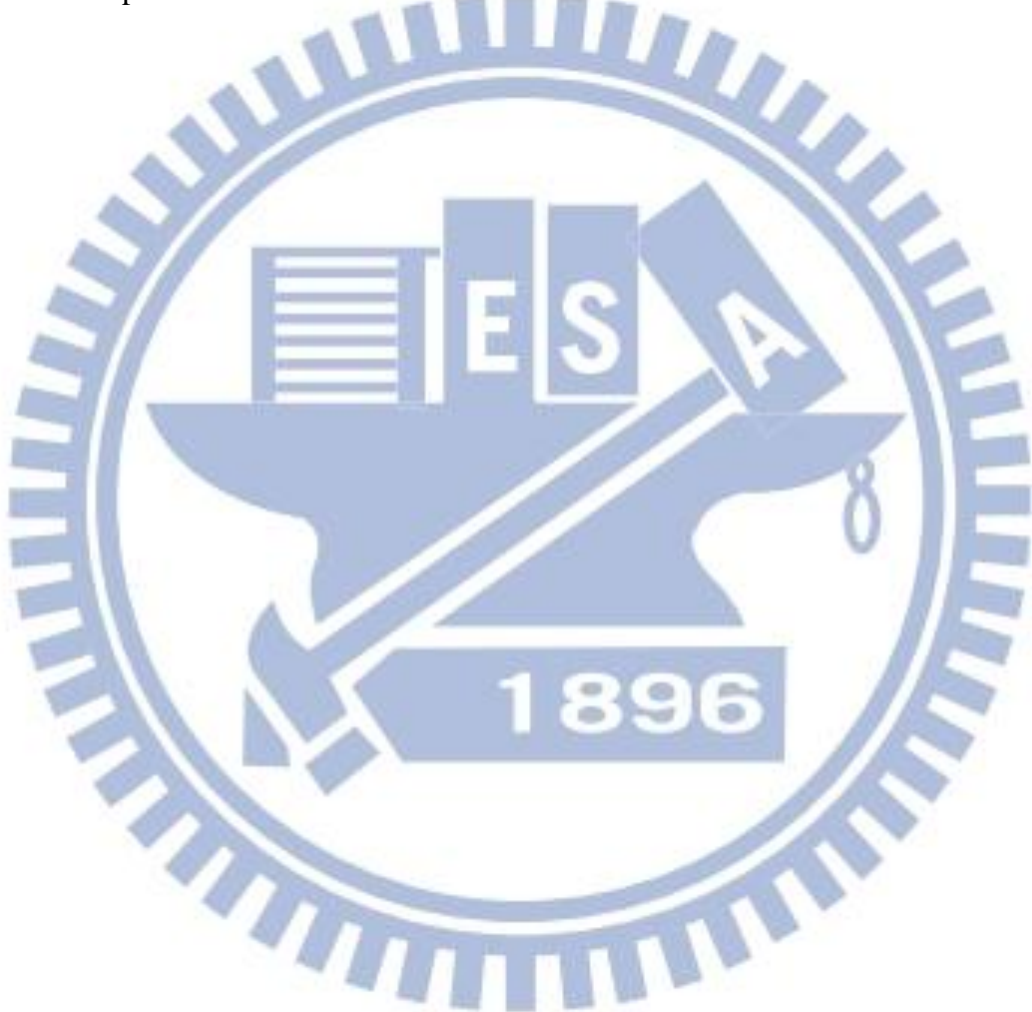
LIST OF FIGURES

Figure 1.1 Wii remote controller.	2
Figure 1.2 5DT Data Glove 5 Ultra [5].	4
Figure 1.3 The armband type space input system for feasibility test [2].....	4
Figure 1.4 The multi-colored gloves [8].....	5
Figure 1.5 The depth estimation using stereo cameras [13].....	6
Figure 1.6 The depth map with skin color [13].....	6
Figure 1.7 The multi stereo cameras system [14].....	7
Figure 1.8 The 3D hand model with input data [14].	7
Figure 1.9 Mesa Imaging SwissRanger 4000 (SR4000) [18].....	7
Figure 1.10 ASUS Xtion [19].....	8
Figure 2.1 The system overview.....	9
Figure 2.2 The Functions of Kinect.....	10
Figure 2.3 (a) the near-infrared radiation projector. (b) the near-infrared radiation sensor [21].	11
Figure 2.4 (a) the color image captured by CMOS sensor. (b) the depth image captured by Kinect.	11
Figure 2.5 OpenNI structure.....	12
Figure 2.6 Hand detection and tracking using Hand Point Generator [20].	13
Figure 2.7 User body detection and tracking using User Generator [20].....	13
Figure 2.8 MVC design pattern.	14
Figure 2.9 System software stack of hand gesture control.....	15
Figure 3.1 Flowchart of the proposed HCI system.....	17
Figure 3.2 An example of obtained depth data.....	18
Figure 3.3 The hand point detection via NITE.....	19
Figure 3.4 (a) the hand image of frame t ; (b) the hand image of frame $t+5$; (c) the	

difference between (a) and (b).....	19
Figure 3.5 (a) a short distance between the hand and the top of the image; (b) the longer distance between the hand and the top of the image.	20
Figure 3.6 the x and y components of the vector for a swipe gesture.....	20
Figure 4.1 The Model-View-Controller for our system.	23
Figure 4.2 (a) and (b) waving the hand. (c) The hand being detected (the blue dot). ...	25
Figure 4.3 (a) the hand stay in the air. (b) The defined (right) and Moving Region (right).....	26
Figure 4.4 The clicking gesture.	27
Figure 4.5 The moving test.....	28
Figure 4.6 The Swipe gesture for pervious page and next page.....	29
Figure 4.7 The zoom in test (moving two hand apart).	30
Figure 4.8 The zoom out test (moving two hand close).	31
Figure 4.9 The scroll down the page.	32
Figure 4.10 The scroll up the page.	33
Figure 4.11 Schematic drawing of contact-free software control application [25].	35
Figure A.1 (a) the Moving Region. (b) the corresponding monitor screen.	37
Figure A.2 An example of the layer segmentation. (a) the depth image, (b) the foreground layer, (c) the hand layer, (d) the background layer.	38

LIST OF TABLES

Table 3.1 Hand gestures used in the HCI operation	21
Table 4.1 Hardware and software specifications	23
Table 4.2 Comparison of frame rate and process time	34
Table 4.3 Comparison of different methods	35



Chapter 1.

Introduction

Nowadays, the aging population has become a serious problem; therefore, medical care plays a more important role than ever. Fortunately, no matter in the industry or market of medical service and healthcare they both have huge growing over past few years. Hospital is one of the largest healthcare providers and intensive-care unit (ICU) is a specialized section of a hospital that provides comprehensive and continuous care for persons who are critically ill and can benefit from the needed treatments. ICU nurses must be alert and have highly specialized nursing skills to handle these high risk and high stress healthcare scenarios. To avoid the interference and infection, paramedics need to care for patients 24 hours a day and the visiting hours for the ICU are strictly limited.

In the ICU, a lot of medical devices and instruments are used to monitor physiological changes of a patient and to record the patient's physiological information. In addition, paramedics need information about the patient's health state from hospital database. Nowadays most hospitals record the medical information in electronic format which can be retrieved easily by computer in the nursing station. Although using electronic medical record reduces search time, paramedics still need to visit the nursing station for the above information, which is not very convenient. Therefore, we want to build a system which can display information for paramedics right around hospital bed. Under such a situation, we need some convenient way to control the device which can avoid hand contact that may result in infection.

1.1 Motivation

For convenient control of various devices, interfaces for smooth user interaction like touchscreen, voice control, remote controller, and hand gesture control have been developed. A touchscreen is an electronic visual display that can detect the presence and location of a touch within the display area, which could bring about infection in the ICU ward of a hospital. Remote controller such as Wii (see Figure 1.1) uses motion sensors and optical sensors to determine the location of the joystick. Since a user needs to touch the remote controller device, it is also not a good choice for the ICU to use.



Figure 1.1 Wii remote controller.

To avoid hand contact, voice also control is a natural way of controlling various devices in the environment. However, background noises may distort the voice signal. On the other hand, hand gestures are intuitive communication language. In particular, the Kinect controller can give a user a refreshing change in the gaming experience. People can play video games hands-free by conveniently using Kinect motion-sensing controller. Therefore, the motivation of this thesis is to develop a hand gesture-based human-computer interface for the hands-free use in an ICU.

1.2 Review of Related Work

Hand gesture control has been used as a user interface for a long time. Gestures are meaningful body motions involving physical movements of the fingers, hands, arms, heads, face, or body [1]. In this thesis we aim to manipulate on-line electronic medical records with hand gestures. In the literature, many analysis works have been proposed to recognize hand gesture. According to the type of information used, these approaches can be categorized into wearable sensor-based approaches [2][3][4][5], and computer vision-based ones [6]-[16]. We will briefly review these methods of hand gesture recognition.

1.2.1 Wearable Sensors

This approach requires user to wear a sensor suit or sensor glove which contains one or more sensors. When user moves fingers or palm, these sensors will send certain signals to the computer. Figure 1.2 shows the 5DT Data Glove 5 Ultra [5] which can measure finger flexure (1 sensor per finger via optical fiber) of the user's hand, and has a control box to interface with the computer via the serial port. Kumar et. al. [3] present a real-time mouse for human-computer interaction based on hand data glove. Ibarguren et. al. [4] propose a real time sign recognition architecture including both gesture and movement recognition. The method uses a two-tier architecture for sign recognition. In first stage, accelerometer signals are processed for the segmentation of different hand gesture. In the second stage, different classifiers are used to recognize the signs identified. Dongwan and Junseok [2] design an armband type human-machine interface with non-contact space input, where user wears an armband with an IMU (Inertial Measurement Unit with gyro sensor and accelerometers sensor) and a piezo vibration sensor as shown in Figure 1.3. Although, it is very easy to implement high accuracy hand gesture recognition system using the above special hardware devices to get hands information, the solution is far too expensive for common use and the user needs to wear a device.



Figure 1.2 5DT Data Glove 5 Ultra [5].

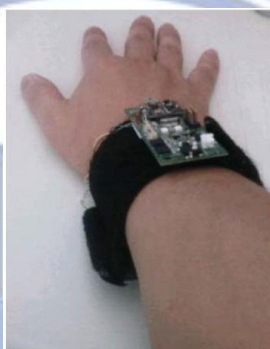


Figure 1.3 The armband type space input system for feasibility test [2].

1.2.2 Interfaces Based on Computer Vision

In vision-based methods, a camera is the input device of a computer which analyzes an image or a video sequence for hand gesture recognition. Usually, hand segmentation or skin color detection is the most important and challenging steps towards hand gesture recognition because of variant lighting condition, and complex environment. The computer vision-based approaches can roughly be divided into single-camera approaches [6][7][8][9][10][11], stereo-camera approaches [12][13][14] and range-camera approaches [15][16], as discussed in the following.

The first group of methods only uses a single camera. In these methods, color is an important feature of human hand and the skin color feature is usually used for segmentation and tracking of a hand. It is intuitive and needs only simple implementation. Fang et al. [6] propose a real-time hand gesture recognition method which segments hands using motion

and color cues to recognize hand gestures via learning. Since uncontrolled lighting condition may also have negative effect on the skin detection, Khan et al. [7] evaluate different skin-color models to classify the skin pixels, which includes six color spaces (IHLS, HSI, RGB, normalized RGB, YCbCr and CIELAB) and nine skin color modeling approaches, by providing a framework for selecting the best approach for skin detection under different lighting conditions. However, since the skin color has large variations, and the lighting of real environment is constantly changing, it is difficult to learn a general skin model. Wang and Popović [8] introduce a consumer hand tracking technique using only inexpensive, commodity components, which uses a single camera to track a hand wearing the multi-colored glove, as shown in Figure 1.4. When the hands change to a different gesture, the camera will get a different distribution of colors and sizes.

Some machine learning approaches are also adopted to deal with the lighting effects. Nagi et al. [9] build hands model by training a lot of samples and Pang [10] uses a cascade of boosted classifiers based on the Harr-like feature. These approaches require a large amount of time to build certain models before gesture recognition can be performed. Kulkarni et al. [11] use the video sequence as motion templates rather than building hand models. Although this approach can recognize a moving hand, it is difficult to detect hand shape change.



Figure 1.4 The multi-colored gloves [8].

While using single camera is difficult to distinguish the depth of image, stereo cameras which use a pair of images from left and right cameras to mimic human vision, can easily provide depth and three-dimensional information. Therefore, stereo camera is often used to estimate image depth and provide better foreground segmentation than single camera. Kang et

al. [12] present a method to use depth and skin color for hand detection and gesture classification. Accordingly, Figure 1.5 and Figure 1.6 show the depth estimation obtained with stereo camera and part of depth map segmented by skin color, respectively. Li et al. [13] estimate depth data by stereo camera to segment hands from image instead of using skin color. In general, more cameras can be used to get more accurate depth information and spatial position. Appenrodt [14] uses a multi camera system, as shown in Figure 1.7, to estimate 3D hand model. Figure 1.8, shows input data and blue points estimated by a stereo camera. Red lines, which uses iterative closest point (ICP) [17] to calculate the distance of error between the model and 3D input data.

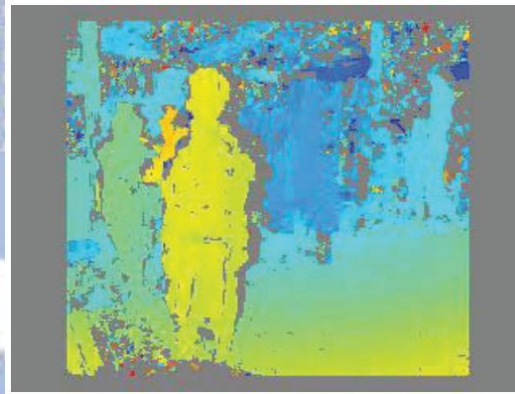


Figure 1.5 The depth estimation using stereo cameras [13].

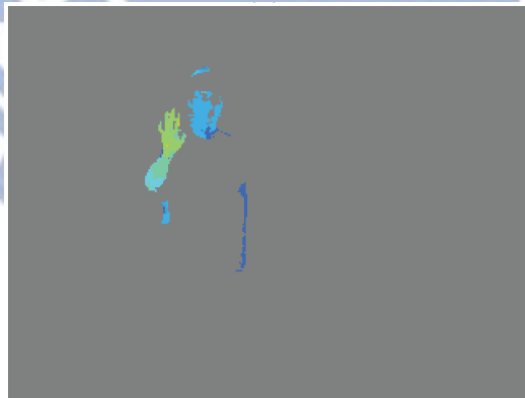


Figure 1.6 The depth map with skin color [13].

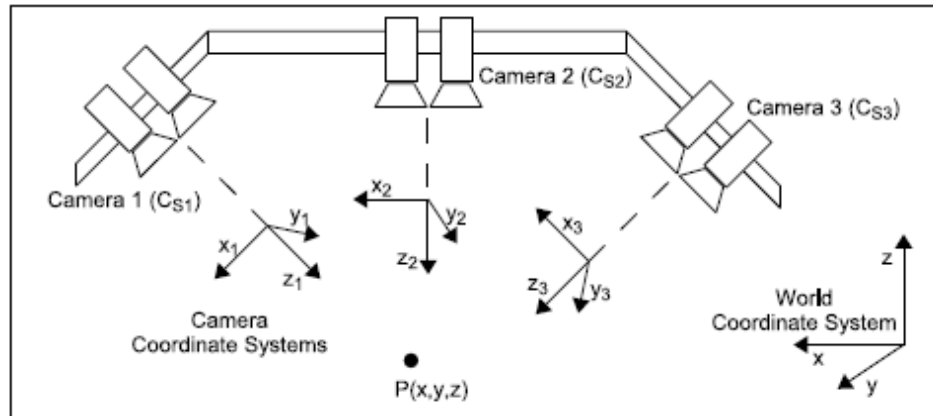


Figure 1.7 The multi stereo cameras system [14].

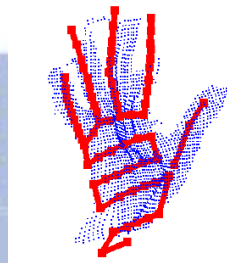


Figure 1.8 The 3D hand model with input data [14].

Range camera is a device that can provide depth map directly, which is useful to foreground segmentation. Hu and Gao [15] use 3D camera (as see Figure 1.9) with edge feature and particle filters to hand recognize gesture and track hands. They use SR4000 as input device for 3D data, which is based on Time of Flight (TOF) technology that measures the round trip time of light from its light source to the objects in the field of view and back to the sensor for generating depth data. He et al. [16] also use 3D depth camera (as see Figure 1.10) to extract hand and fingertips from the acquired depth image.



Figure 1.9 Mesa Imaging SwissRanger 4000 (SR4000) [18].



Figure 1.10 ASUS Xtion [19].

For natural interaction, a method that does not require the attachment of any special hardware to the hand or body is preferred. In addition, many problems (like skin color detection and foreground segmentation) of using a general camera are caused by light changing in a complex dynamic environment. Based on such an idea, we choose Kinect as our input device, which is powerful, not expensive and easy to setup and use.

1.3 Thesis Organization

This thesis is organized into five chapters as follows. In Chapter 1, we introduce the background knowledge related to our research on hand gesture recognition. Based on the nature of the device, we classify these methods as wearable sensors and interfaces based on computer vision. In Chapter 2, we describe in detail the input device, including hardware and software, of our system as well as the process in the proposed framework. In Chapter 3, we define the hand gestures adopted in this thesis and then we explain in detail the hand gesture recognition method which includes system initialization, data acquisition, feature extraction, and hand gesture recognition. In Chapter 4, we detail the structure of our hand gesture control system and show some experimental results, with comparison to other system. In Chapter 5, conclusions are drawn.

Chapter 2.

System Description

The proposed system includes three parts (see Figure 2.1): hardware (Kinect), framework (Open Natural Interaction), middleware (NITE) and application, a program that analyzes data obtained from OpenNI with NITE, and perform corresponding user operation (see Chapter 3). OpenNI which provides a way to access sensor data of Kinect.

In Section 2.1, the input device is described. In Section 2.2, the framework and middleware are described. In addition, the structure of hand gesture control system is proposed in Section 2.3. Finally the software stack for our proposed system is given in Section 2.4.

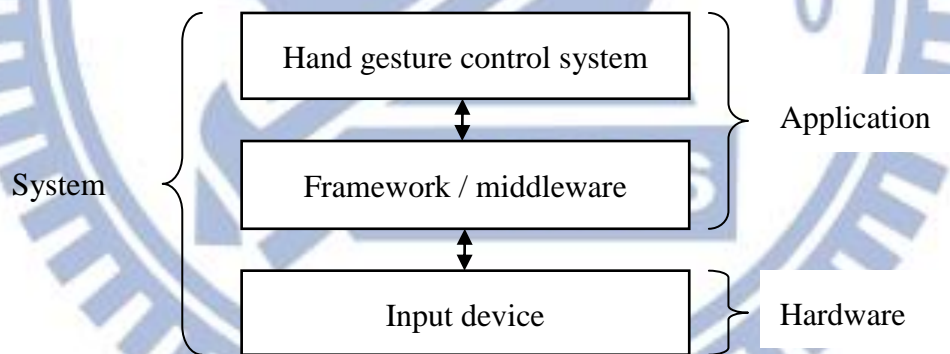


Figure 2.1 The system overview.

2.1 Input Device

Microsoft announces a new game device - Kinect which enables user to play game without controller. Kinect includes a pair of near infrared radiation projector and sensor, a CMOS camera, an array microphone and a motor (see Figure 2.2). Through these sensors, Kinect can retrieve more accurate action than using remote controller. We will discuss in more detail in the next section.

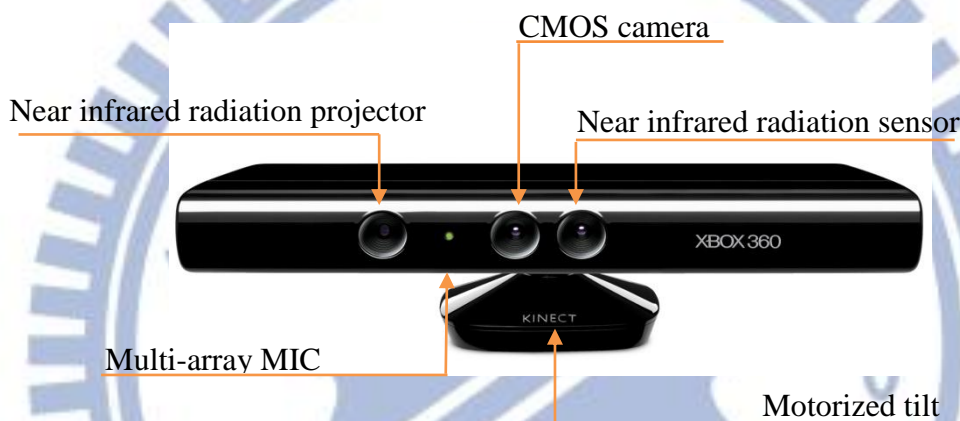


Figure 2.2 The Functions of Kinect.

The near infrared radiation (IR) projector, near-IR sensor and PrimeSense's chip are the key for computing the depth of scene. The near-IR projector emits invisible light which pass through a filter and scattered into a special pattern. The near-IR pattern is projected onto the object in front of near-IR projector (see Figure 2.3(a)) and the near-IR sensor receives the reflected pattern and analyzes the depth value (see Figure 2.3(b)). Because the depth value is estimated by near-IR pattern, the edges of the object will cause random errors in depth value. The random errors in depth measurement by Kinect will increase with increasing distance to the sensor [21]. When Kinect senses object within range of 50~250 cm, the depth errors can be lower than 1 cm.

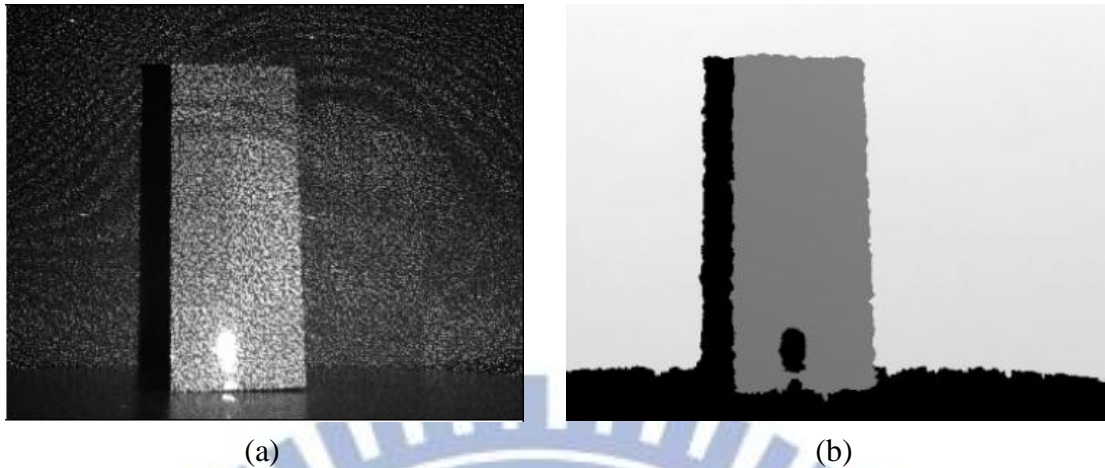


Figure 2.3 (a) the near-infrared radiation projector. (b) the near-infrared radiation sensor [21].

The CMOS camera in Kinect has a fixed lens and can capture VGA resolution with 30 frame per second (fps) or HD 720 resolution with 14 fps video clips. The CMOS camera has the ability to identify color and can be used for facial recognition. In Figure 2.4(a), we show the color image capture from Kinect's CMOS camera. In Figure 2.4(b), we show the corresponding depth image capture by Kinect.



Figure 2.4 (a) the color image captured by CMOS sensor. (b) the depth image captured by Kinect.

In Kinect, a Multi-array MIC has four microphones placed on the same surface that can receive voice signal. The sound arrive them in different time so we can calculate the voice source, cancel noise or delete echoes by using signal filter and calculating arrival time of sound. This system enable Kinect to receive more pure voice signal.

2.2 Framework and Middleware

OpenNI is a framework developed by PrimeSense for Kinect and all the devices with PrimeSense technology to access sensor data. A three-layered view of the OpenNI is shown in Figure 2.5. OpenNI provides a basic communication interface between application and hardware and reserves ability for adding functions provided by middleware component such as NITE, which provides natural-interaction UI controls APIs for OpenNI framework.

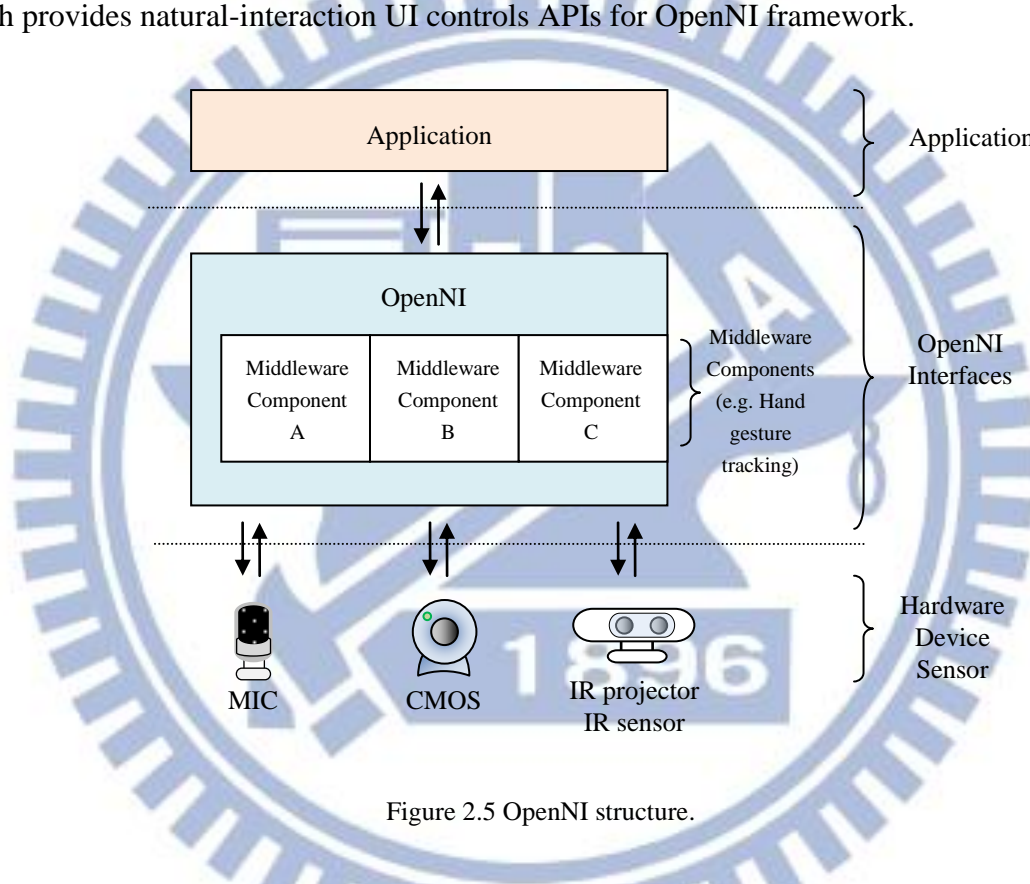


Figure 2.5 OpenNI structure.

In OpenNI, there are two types of production nodes; one is sensor related and the other one is NITE related. Sensor related production node included Depth Generator, Image Generator, IR Generator and Audio Generator. The Depth Generator is a node that generates a depth-map, which is accurate to a millimeter and represents the distance from 3D sensor camera to an object, Image Generator is a node that generates colored image-maps, IR Generator is a node that generates IR image-maps, and Audio Generator is a node that generates an audio stream from the multi-array MIC.

NITE related production node included User Generator (see Figure 2.7), Scene Generator and Hand Point Generator. For Scene Generator, the Scene Analyzer's main output is a labeled depth map, in which each pixel holds an user ID or it is part of the background. The Hand Point Generator supports hand detection and tracking (see Figure 2.6).

Although OpenNI with NITE has some pre-defined hand gestures, these gestures are not enough for our requirements for surfing the internet in an ICU. Thus, we will define our own hand gestures in the Chapter 3.

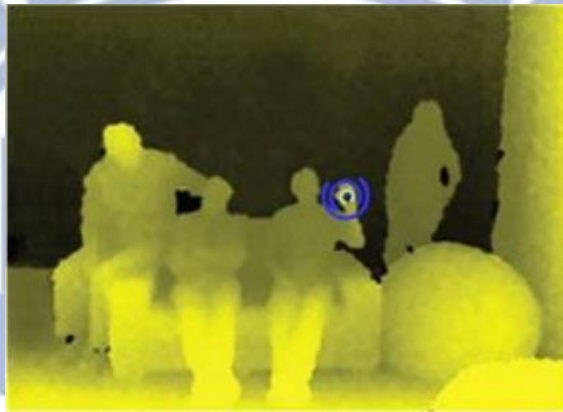


Figure 2.6 Hand detection and tracking using Hand Point Generator [20].

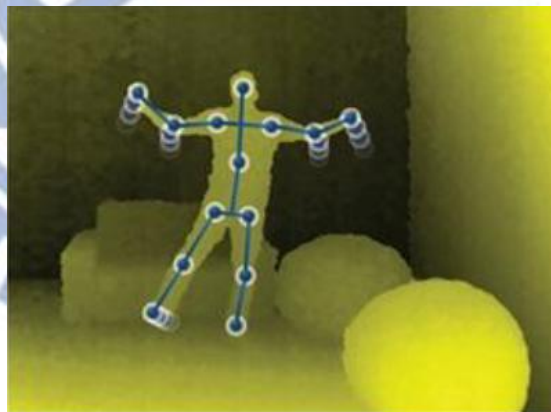


Figure 2.7 User body detection and tracking using User Generator [20].

2.3 Hand Gesture Control System

Model–view–controller (MVC) was invented by Krausner [22] which is a software design pattern for designing interactive computer user interfaces. MVC divides the application into three components: model, controller and view, as shown in Figure 2.8, with each component having its main target. The View component is to handle all events about user interface, the Controller component is a middleman that communications with the View and the Model, and the Model component store all data that is used for controller or view. MVC is of great helps for long-term maintenance of program change and we will design our system with the MVC design pattern.

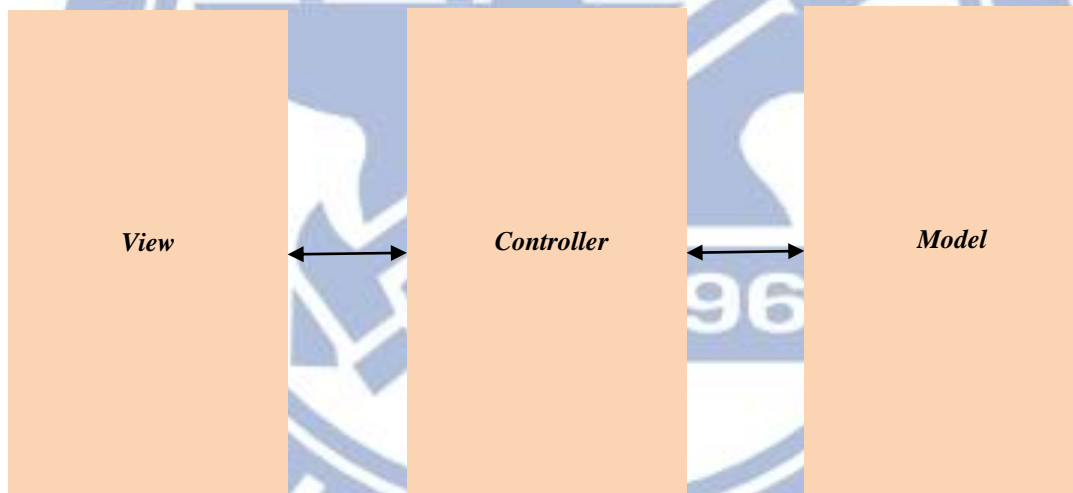


Figure 2.8 MVC design pattern.

2.4 Software Stack

Figure 2.9 shows the Hand gesture control system software stack in which NITE and OpenNI introduced in previous section are used for updating depth map and obtaining hand point. Moreover, the Qt framework provides a cross-platform GUI environment for the creation of user interface and time event. Qt framework for Windows is based on Window API which we also use for accessing system-level IO.

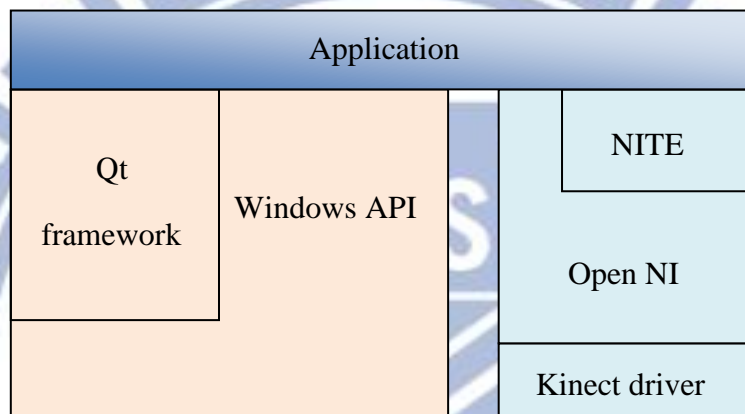


Figure 2.9 System software stack of hand gesture control.

Chapter 3.

Hand Gesture-Based HCI

The proposed hand gesture recognition method is described in this chapter. In Section 3.1, the defined hand gestures and a system flowchart for their recognition will be presented. Then, we explain each part of the flowchart from Sessions 3.2 to 3.5.

3.1 Hand Gestures Definition

In order to operate a web browser on PC via hand gestures, we define some basic hand gestures for different operating commands. The defined commands which include moving cursor, clicking, switching page, zoom in, zoom out, scroll up and scroll down, which are commonly used for browsing websites or documents. Some of these hand gestures are designed to be operated by two hands since it is more intuitive from user experience. The detailed description of these hand gestures will be given in Section 3.4.

Figure 3.1 shows the proposed hand gesture control system which has five main parts. The first part initializes hardware, framework, middleware, and our hand gesture control system. The second part acquires depth data from Kinect. The third part extracts hand features, including position, state, moving direction and moving magnitude, for hand gesture recognition. The fourth and fifth parts are hand gesture recognition and HCI operation, respectively.

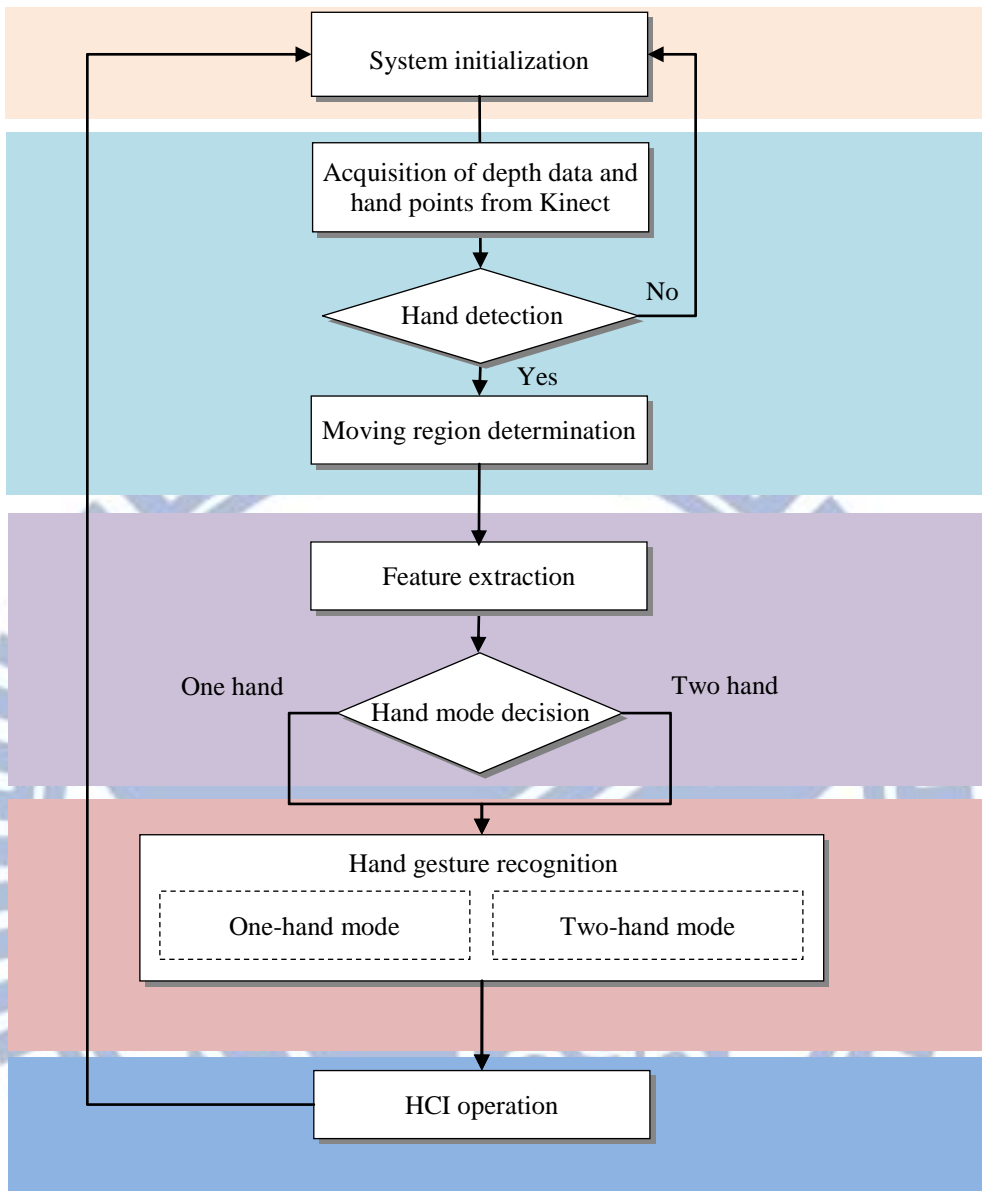


Figure 3.1 Flowchart of the proposed HCI system.

3.2 System Initialization and Data Acquisition

Firstly, OpenNI is driven to acquire depth data continuously from Kinect. Next, NITE is used for hand point detection and tracking. After that, the depth data and the hand points are used to initialize a Moving Region for subsequent hand gesture recognition.

3.2.1 Acquisition of Depth Data

The function "WaitAndUpdateAll" waits for all nodes to have new data available, and then updates them about 33 milliseconds. Our hand gesture system refreshes the depth data by using the "UpdateData" request function every 10 milliseconds that forces depth generator to expose the most updated data to application without waiting other generator for smoother movement in operation time. Our hand gesture system updates depth data continuously by OpenNI. Figure 3.2 shows the obtained depth data.



Figure 3.2 An example of obtained depth data.

3.2.2 Hand Point Detection

The NITE will extract the hand information, which includes a frame number, an ID and a hand point (hand's center of mass). Figure 3.3 shows a detected hand point via NITE.

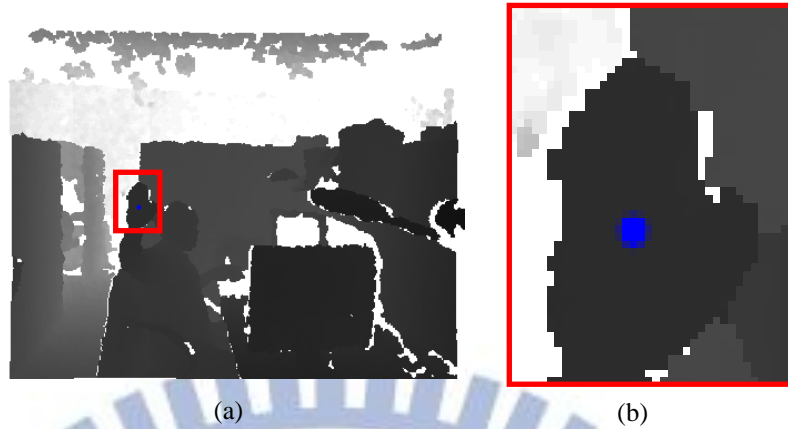


Figure 3.3 The hand point detection via NITE.

3.3 Feature Extraction

In order to recognize the previously defined hand gestures, we use three kinds of features, including hand position, hand state, and hand moving direction and magnitude. While hand position is obtained from NITE directly, the change of hand state is detected the difference of hand images. Figures 3.4 (a) and (b) show the hand image at frame t and frame $t+5$, respectively, and Figure 3.4 (c) shows the difference between Figures 3.4 (a) and (b), after applying the morphology filter. For deciding the hand state, we detect the minimum distance between the hand and the top of the image. If the distance is changing from short to long (see Figures 3.5 (a)-(b)), the hand state must be in the fisting state, otherwise, the hand state must be in the opening state.

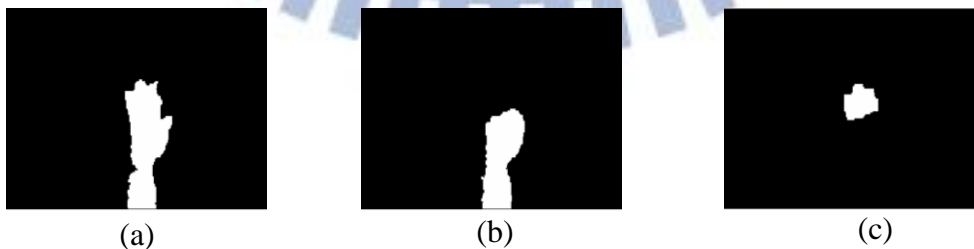


Figure 3.4 (a) the hand image of frame t ; (b) the hand image of frame $t+5$; (c) the difference between (a) and (b)



Figure 3.5 (a) a short distance between the hand and the top of the image; (b) the longer distance between the hand and the top of the image.

Hand direction and magnitude are calculated by subtracting the previous hand point coordinates from the current hand point coordinates and stored the information as a motion vector. Figure 3.6 shows the x and y components of the vector for a swipe gesture.

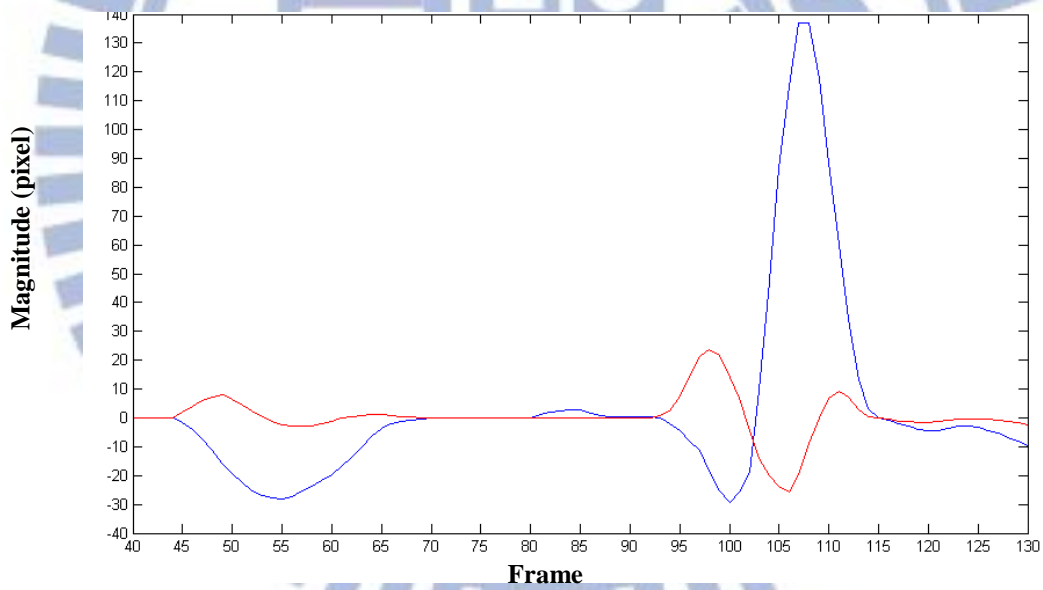







Figure 3.6 the x and y components of the vector for a swipe gesture.

3.4 Hand Gestures Recognition

Table 3.1 shows the hand gestures and action considered in our hand gesture recognition system. When a hand is detected in our system, it will be recognized as the dominant hand and system will enter one-hand mode. There are three gestures for the one-hand mode, including cursor moving, mouse clicking, and next/pervious page. The Move gesture is recognized when the hand point is moving in "Moving Region" (see Appendix A). The "Click" gesture is a open-close-open sequence by dominant hand. The Swipe gesture is switching page by swiping hand. If another hand is detected, the system will enter two-hand mode which includes zooming and scrolling hand gesture. Zooming and scrolling action are recognized by the fisting of the dominant hand. The Zooming gesture is moving two hands closer or apart. The Scrolling gesture is moving non-dominant hand up or down.

Table 3.1 Hand gestures used in the HCI operation

Hand gestures	Image	Dominant hand	Non-dominant hand	Hand move	Command
Move		Open	None	Move any direction	Move cursor
Click		Open-fist-open	None	Not moving	Click mouse
Swipe		Open	None	Move left/right quickly	Pervious/next page
Zooming		Fist	Open/fist	Move two hands apart /closer together	Zoom in/zoom out
Scrolling		Fist	Open/fist	Move non-dominant hand up/down	Scroll up/scroll down

Chapter 4.

System Implementation and Experimental Results

The system implementation and experimental results are described in this chapter. In Section 4.1, the system implementation and the function of each module are presented. Then, we test all kinds of hand gestures that we selected and compare the results with some similar methods in Section 4.2.

4.1 System Implementation

We implement our system with the Model-View-Controller (MVC) design pattern as shown in Figure 4.1, with three parts to increase the flexibility of hand gesture-based human-computer interaction system and decrease dependency of each part. The View part contains QtMotorController, Windows GUI and QtImgReader modules. QtMotorController is a UI for a user to adjust Kinect angle. Windows GUI is a part of Windows OS which provides an application programming interface (API) for system-level IO control. For efficient operation, QtImgReader is adopted to access the depth data directly which emits update event to DataGenerator and shows depth data on the screen.

The Controller part is organized by MotorController and GestureController. MotorController controls Kinect motor and changes the horizontal angle of Kinect from -37 to +37 degrees. GestureController analyzes user's hand gesture and dispatches different events, like mouse or keyboard event, to a computer.

The Model part is organized by MotorData and DataGenerator. MotorData is the Kinect motor data which stores in Kinect hardware that records the value of Kinect angle.

DataGenerator acquires the depth data and image data from Kinect and stores in an array map.

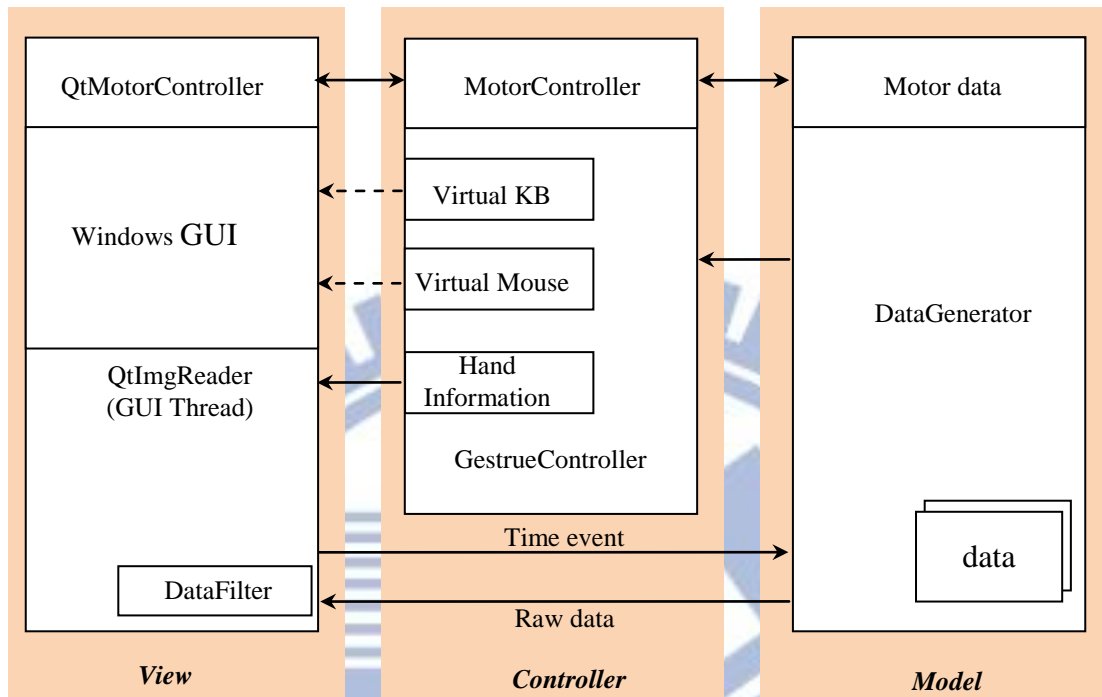


Figure 4.1 The Model-View-Controller for our system.

4.2 Experimental Results

The hand gesture-based human-computer interaction system is implemented by using Microsoft Visual C++ 2010 and uses Kinect as input sensor. We test our system under both PC and notebook to show that it can execute on a platform with fewer system resources than PC. The detailed specifications are shown in Table 4.1.

Table 4.1 Hardware and software specifications

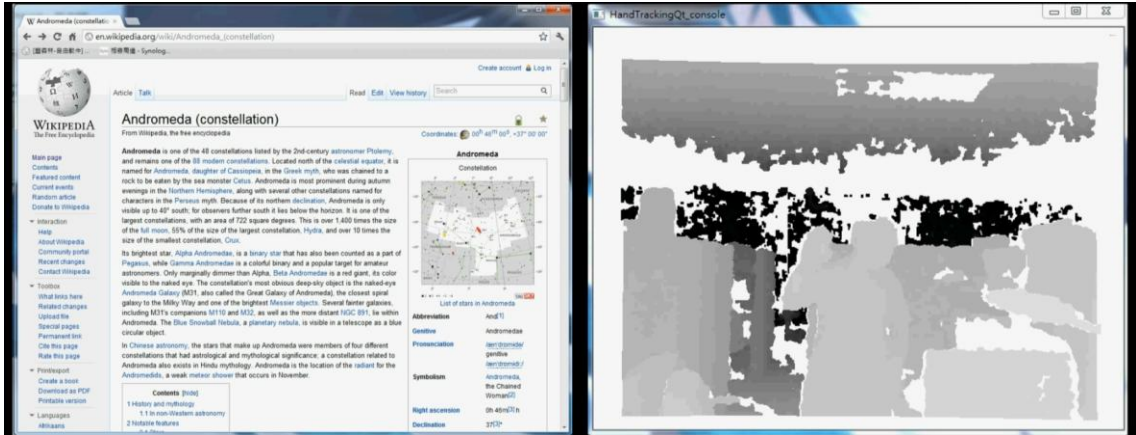
Item	Describe	
Model	PC	Notebook Thinkpad X201i
CPU	Intel I5-2500K @ 3.9GHz	Intel i3 M330 @ 2.13GHz
RAM	16GB DDR3-1333	4GB DDR3-1066
OS	Windows 7 Enterprise 64bit	
Library	OpenNI 1.5 NITE 1.5.2 Qt 4.1	

All the experiments discussed in this section are tested indoors. The user stands in front of the Kinect sensor and keeps a distance of 1.8 meters. We test all hand gestures that we defined before and calculate the average process during the operation. In our experiments, we simulate a sequence of operations for surfing the web and analyze the performance of recognition algorithm. Then we compare these results with some similar methods.

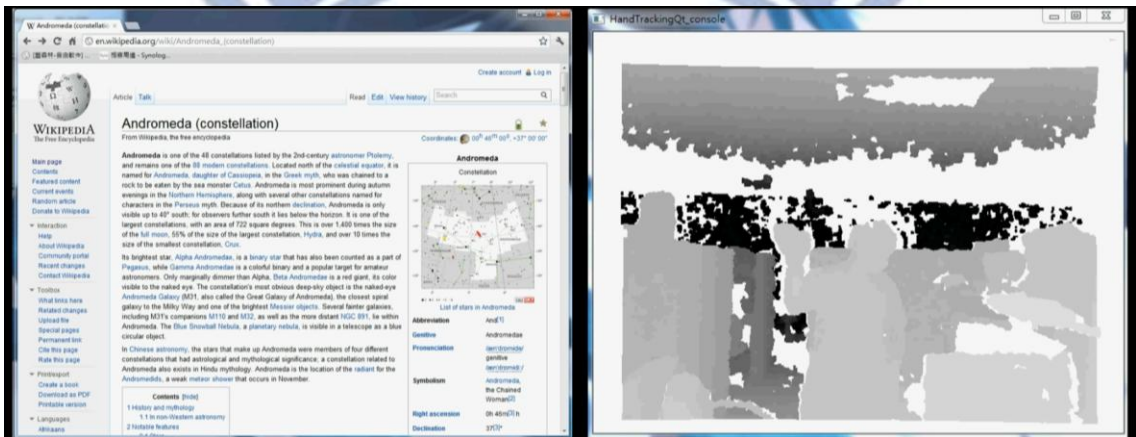
The system initialization requires user waving hands. Figure 4.2 shows the hand detection process. Figures 4.2 (a) and (b) shows two images of a waving hand. After waving hand, the hand point that is detected by NITE (see the blue dot in Figure 4.2(c)). To define the Moving Region, the dominant hand (the first detected hand) needs to stay in the air, as show in Figure 4.3 (a), for about one second. Figure 4.3 (b) shows that the Moving Region is defined and the cursor (the yellow circle) is show in the browser.

The one-hand gestures include Click, Move, and Swipe gestures. Figure 4.4 shows the clicking test. Figures 4.4 (a)-(c) show open-fist-open sequence and the red circle in Figure 4.4 (c) indicates the position of clicking action. Figure 4.5 shows the moving test where the cursor moves from right to left as the hand moves from right (see Figure 4.5 (a)) to left (see Figure 4.5 (b)). The pervious page and next page commands correspond to swiping left (see Figures 4.6 (a)-(b)) and swiping right (see Figures 4.6 (b)-(c)), respectively, by a hand.

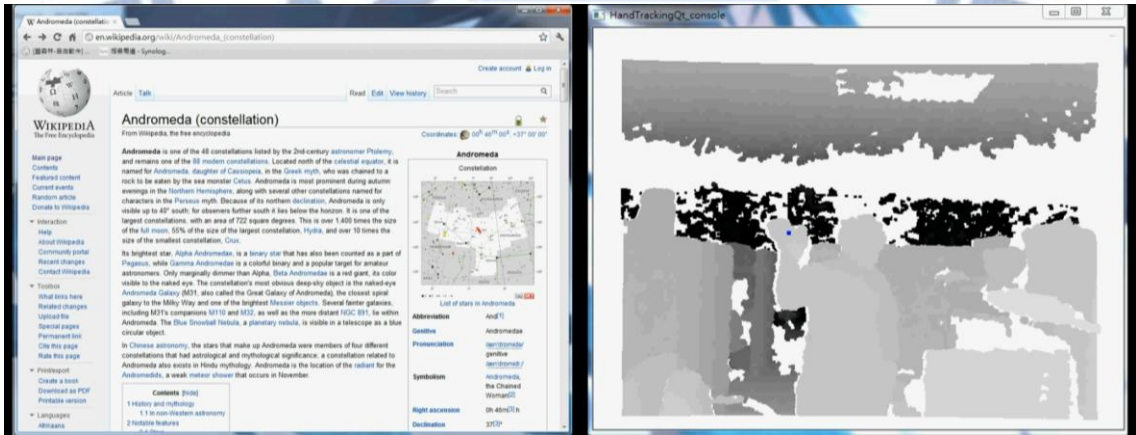
The two-hand gestures include Zooming and Scrolling gestures. Figures 4.7 and Figure 4.8 show the zooming test. The dominant hand fisting indicates the start of a two-hand action while two hands move apart (see Figure 4.7) and close together (see Figure 4.8) to zoom in and zoom out the page, respectively. The Scrolling gesture also needs the user to fist the dominant hand (shown in Figures 4.9 (a)-(b)) before scrolling pages. The scroll up and down is moving the non-dominant hand up (see Figures 4.10 (a)-(b)) and down (see Figures 4.9 (b)-(c)).



(a)

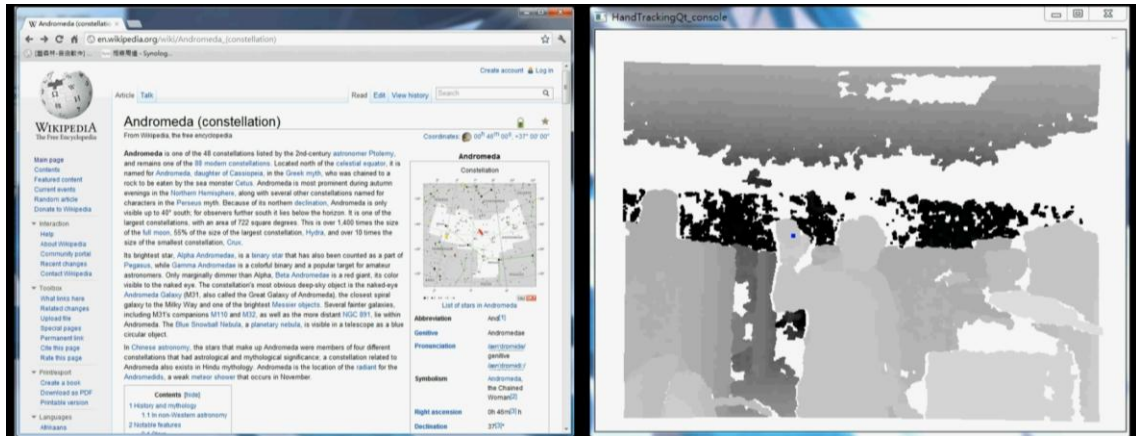


(b)

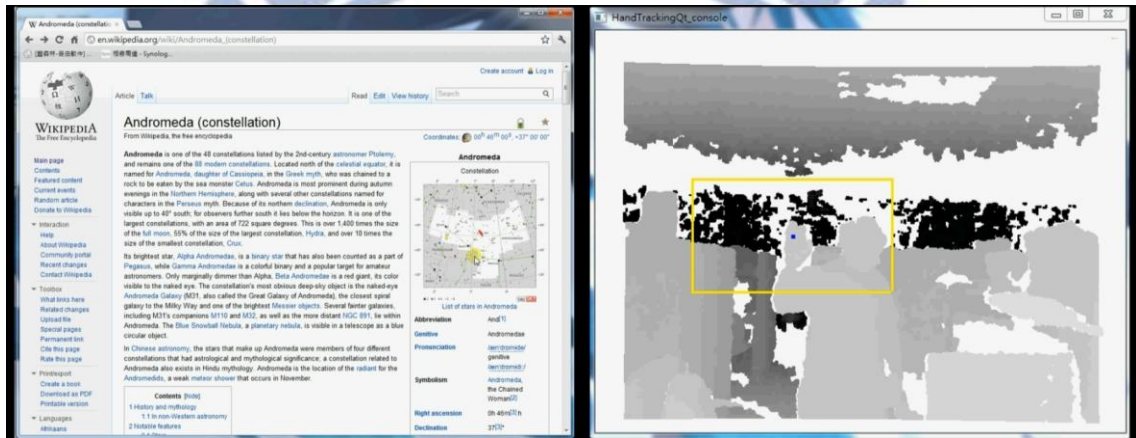


(c)

Figure 4.2 (a) and (b) waving the hand. (c) The hand being detected (the blue dot).

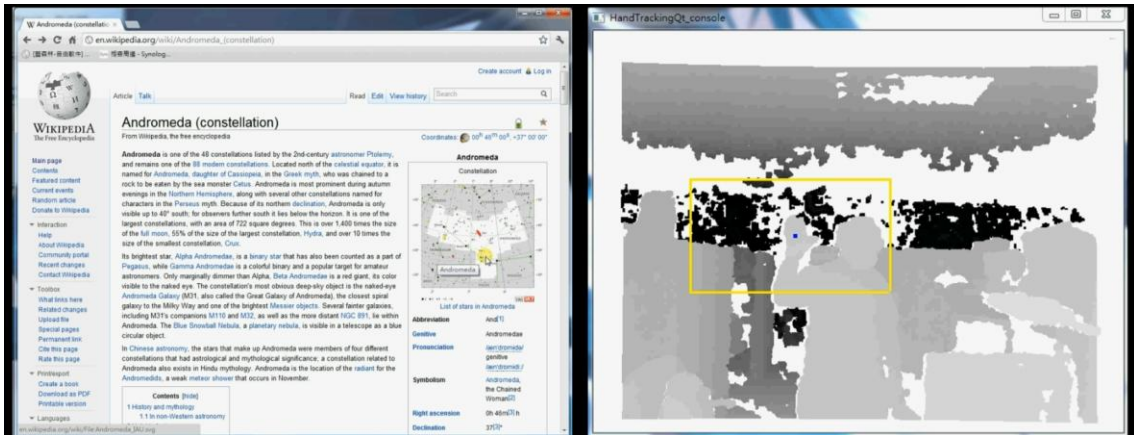


(a)

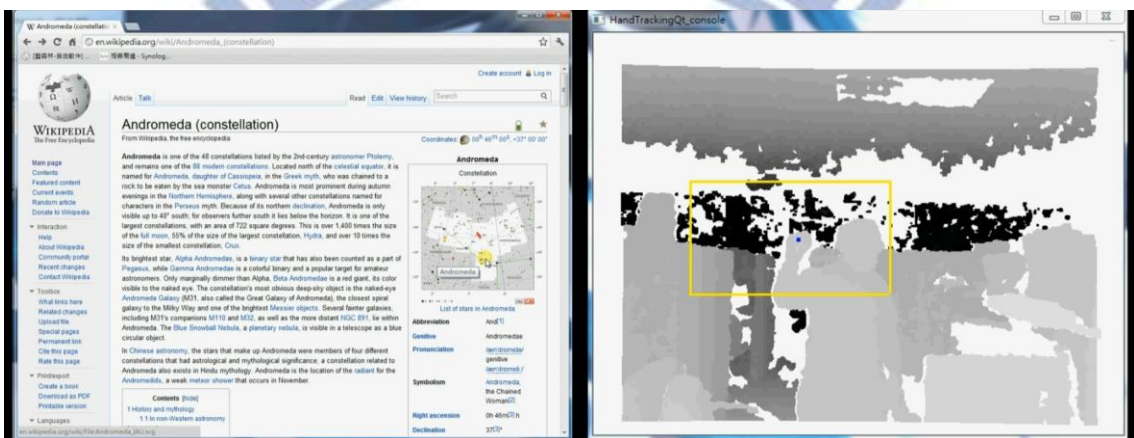


(b)

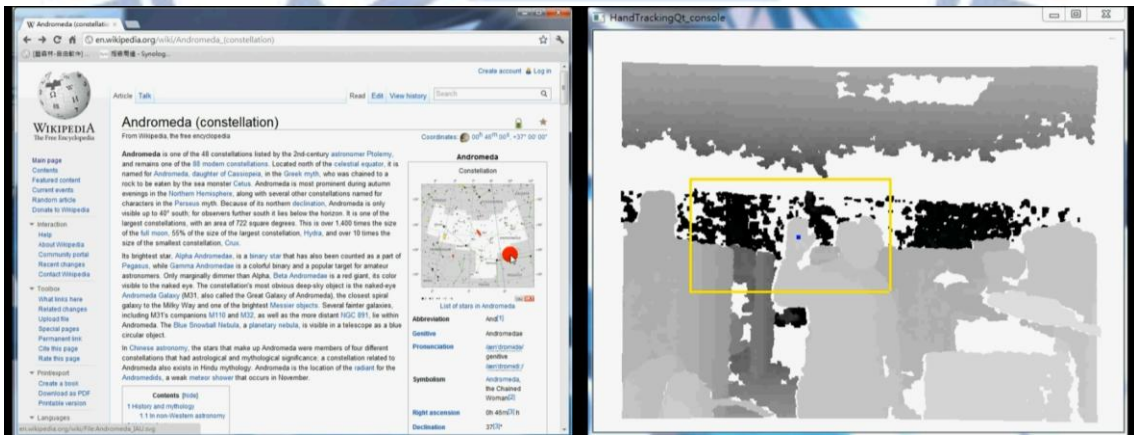
Figure 4.3 (a) the hand stay in the air. (b) the defined Moving Region (right) and the cursor position (left).



(a)

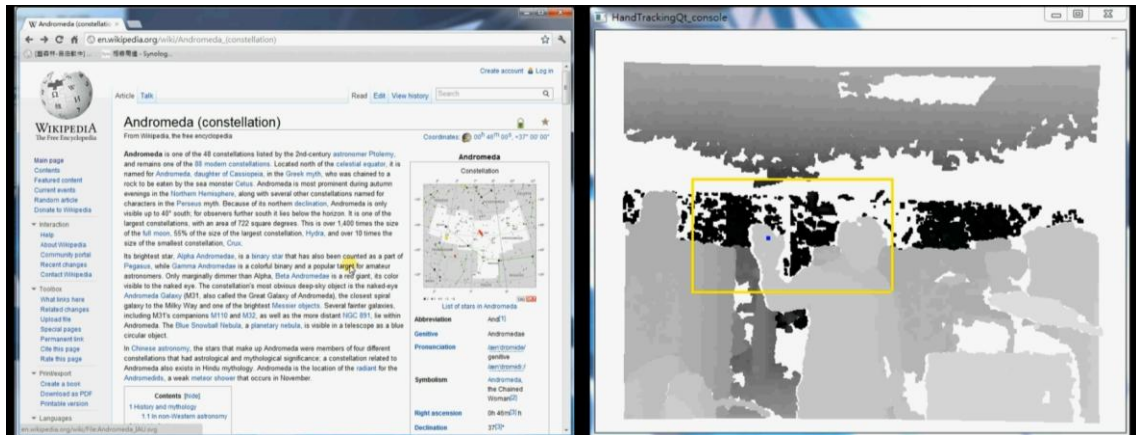


(b)

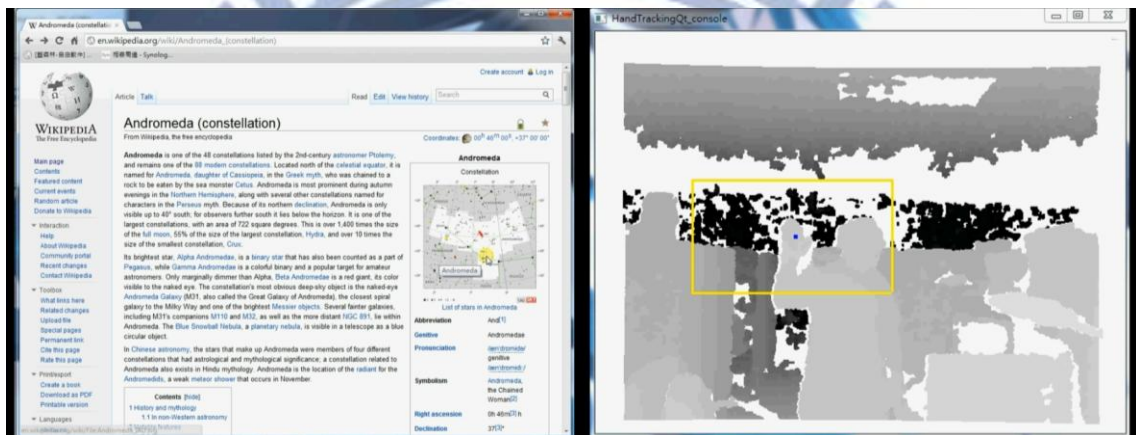


(c)

Figure 4.4 The clicking gesture.

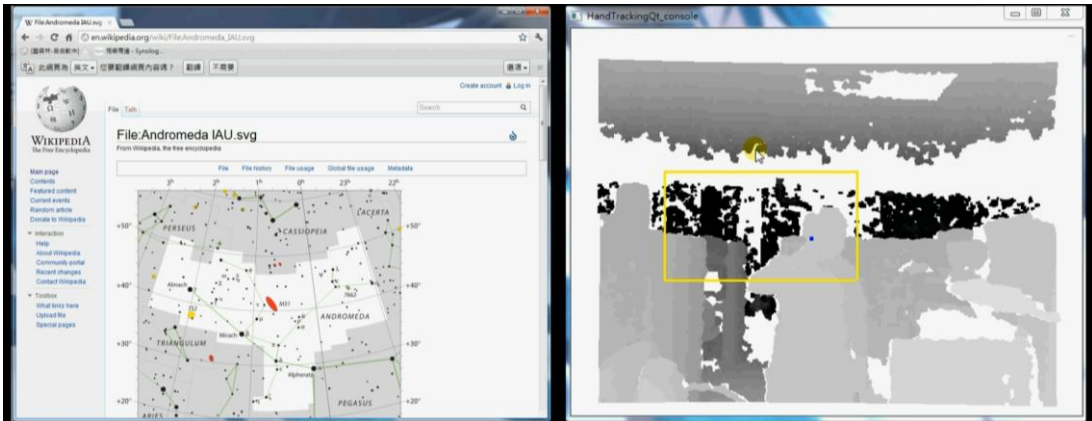


(a)

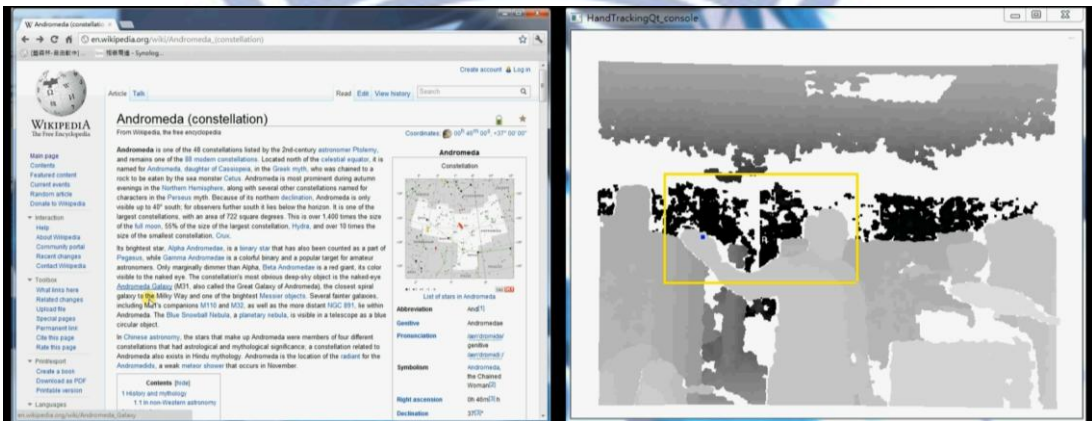


(b)

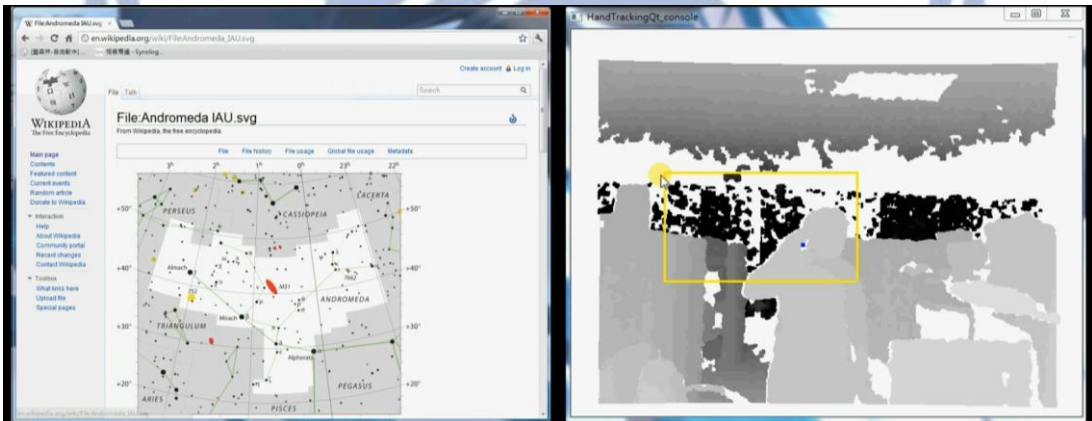
Figure 4.5 The moving test.



(a)

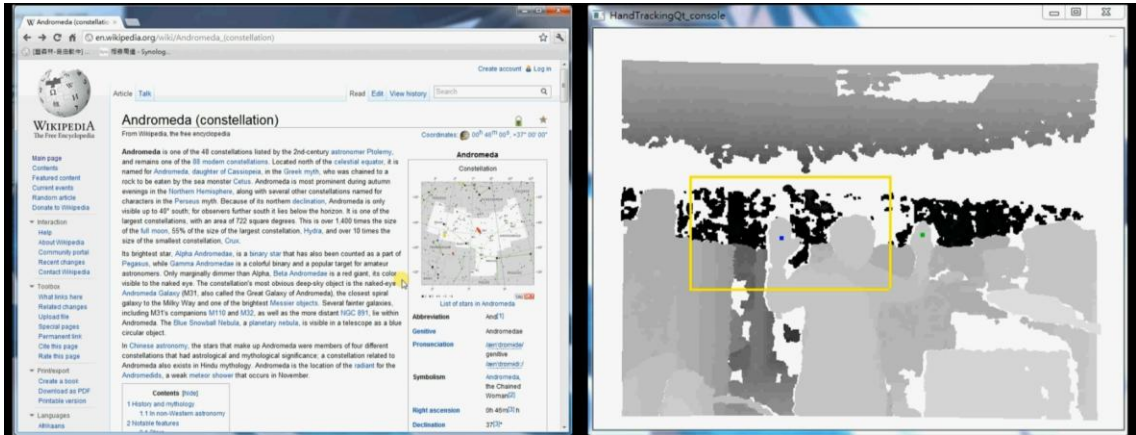


(b)

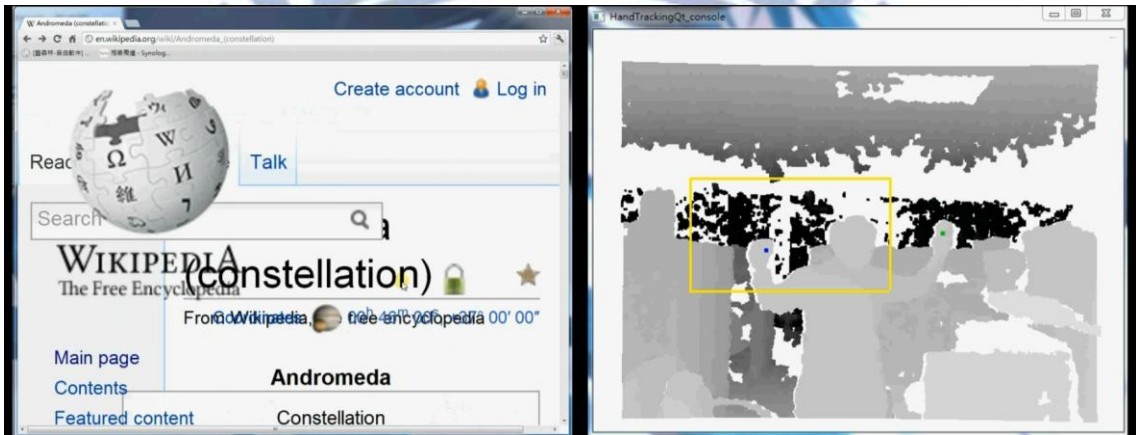


(c)

Figure 4.6 The Swipe gesture for pervious page and next page.

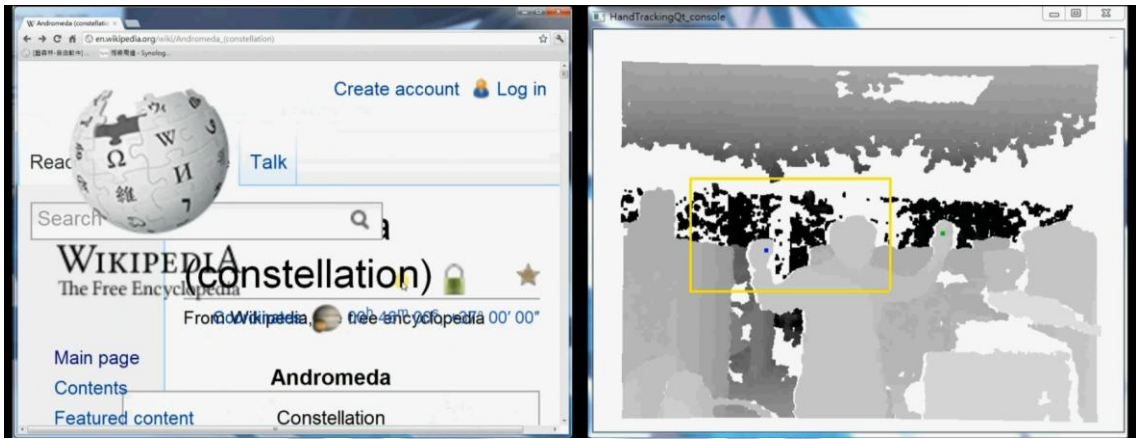


(a)

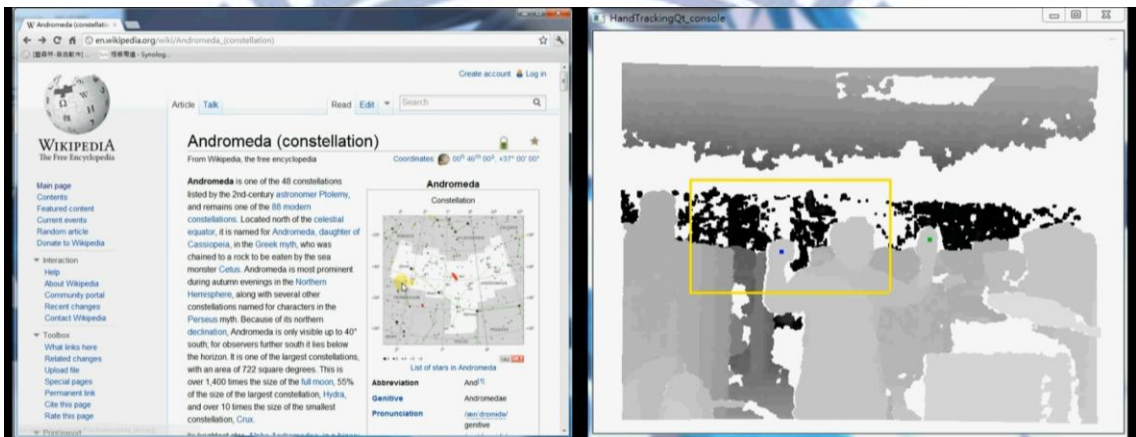


(b)

Figure 4.7 The zoom in test (moving two hand apart).

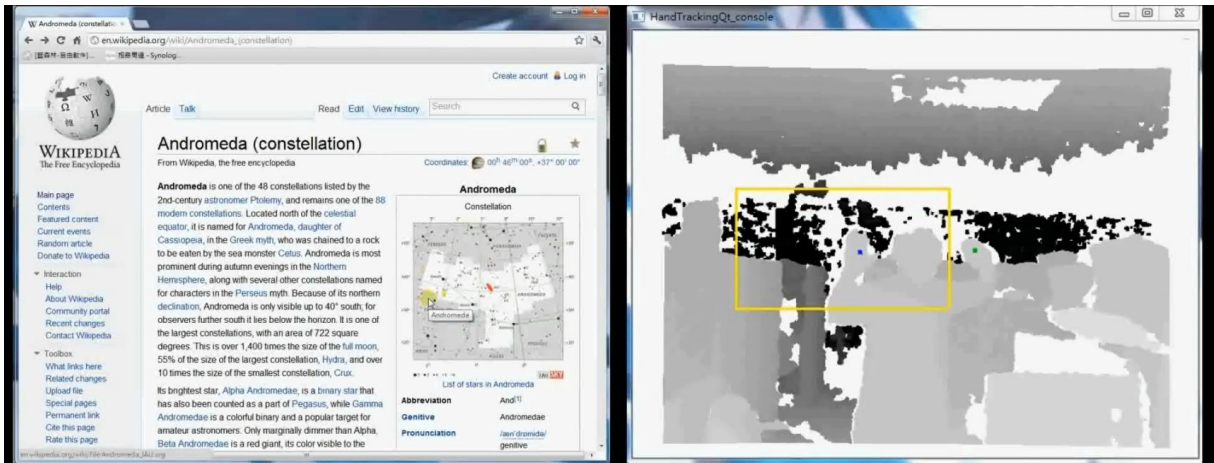


(a)

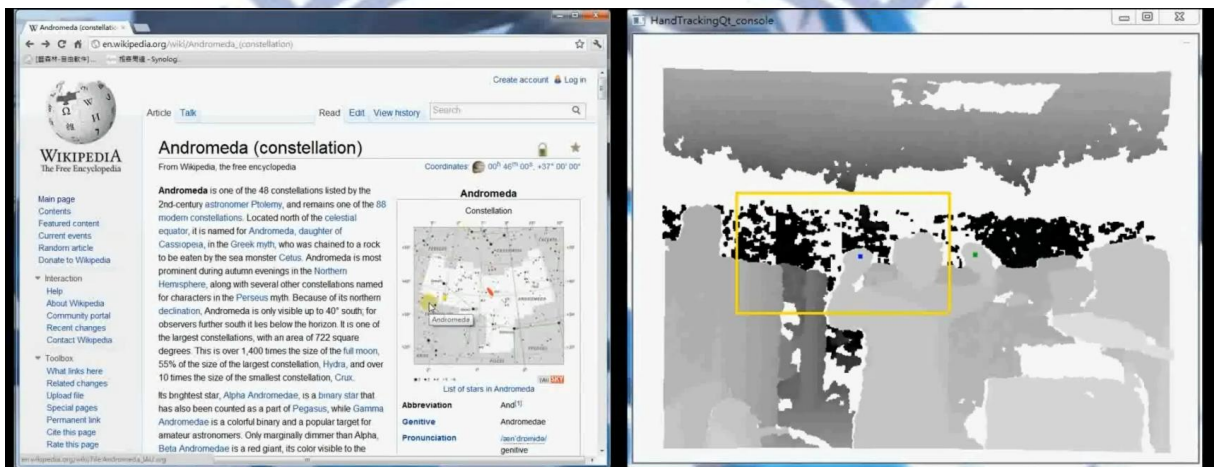


(b)

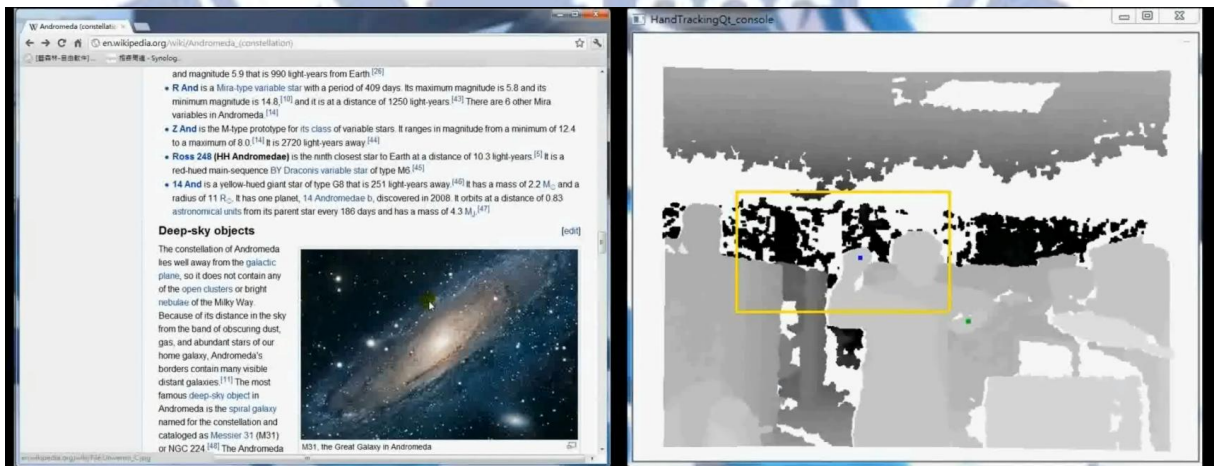
Figure 4.8 The zoom out test (moving two hand close).



(a)

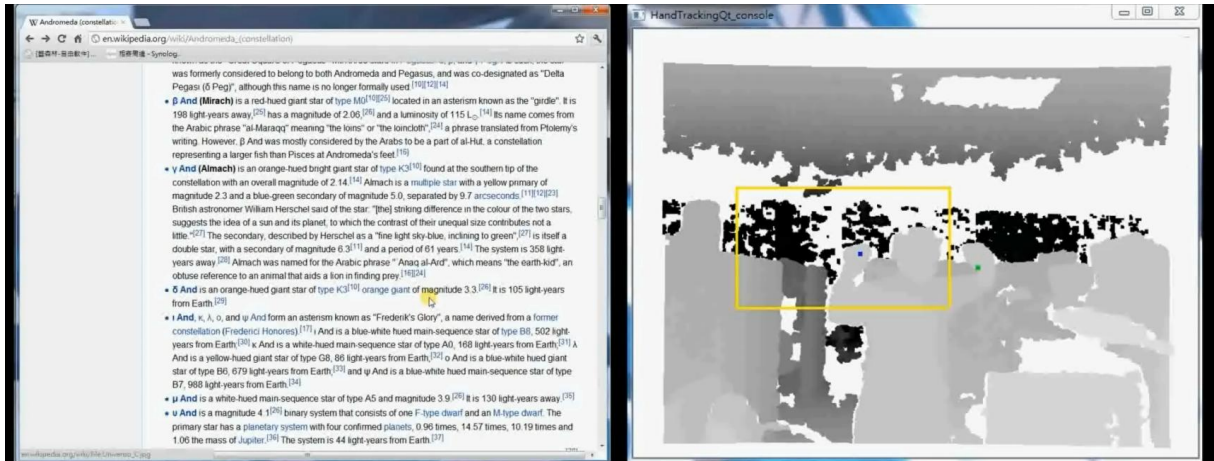


(b)

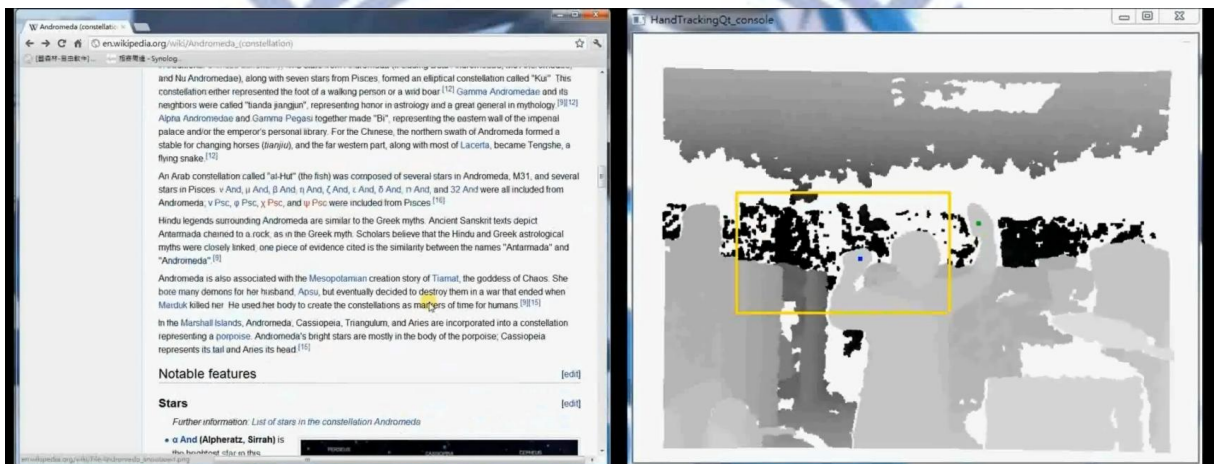


(c)

Figure 4.9 The scroll down the page.



(a)



(b)

Figure 4.10 The scroll up the page.

Table 4.2 shows the performance of HCI for PC and notebook platforms. The processing time of each frame is the average time during the test of HCI operations for each platform, which correspond a frame rate larger than 45 frames per second even for the slower platform.

Table 4.2 Comparison of frame rate and process time

Item	Performance	
Machine	PC	Notebook X201i
Frame rate	49.0103	45.0015
Process time (sec)	0.02	0.022

For operating a web browser in ICU, we compare our system with methods presented in [23]-[25], which are all designed for manipulating medical data. Table 4.3 shows a comparison of different methods. In ICU, paramedics will prefer a system which can be ready quickly for immediate use for browsing patient's health state and health data right around hospital bed. Our hand gesture-based human-computer interaction system requires only about 2 second to initialize system and doesn't require a special field of view (see Figure 4.11 for Kipshagen's method), which is more convenient to browse patient's data. Therefore, our method is more suitable for ICU than other methods.

Table 4.3 Comparison of different methods

	Our method	Gallo	Kipshagen	Bellmore
Main Technology	Hand point from Kinect	Skeleton from Kinect	Stereo-cameras	Skeleton from Kinect
Initialization time	~2 sec	~30 sec	N/A	~10 sec
Sensing region	0.5~1.8 m	0.5~1.8 m	A special field of view (see Figure 4.11)	0.5~1.8 m
Data type	3D data	3D data	2D data	3D data

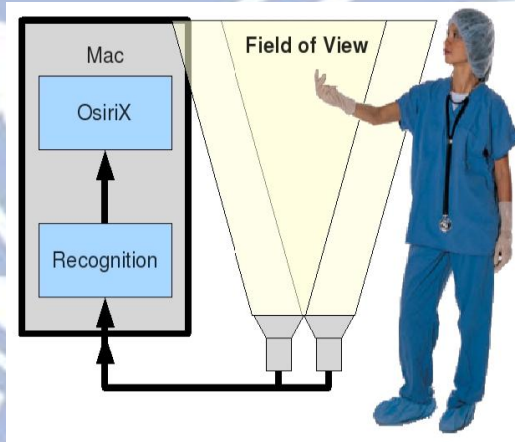


Figure 4.11 Schematic drawing of contact-free software control application [25].

Chapter 5.

Conclusions

In this study, we proposed a real-time hand gesture-based HCI using the Kinect and developed some basic hand gestures that users can use to browse web pages, photos and other information. In order to perform the hand gesture recognition, we extract the four features from depth data, including hand position, hand state and hand moving direction and distance, which are effective in describing the different hand gestures. The hand gesture recognition approach uses two modes according to the number of hand used. The one-hand mode contains three kinds of gestures: Move, Click and Swipe gesture. The two-hand mode contains two kinds of gestures: Zooming and Scrolling. Beside, a Moving Region is automatically determined which can allow the hand to move within a small range to control the cursor to more easily in the monitor screen and decreases the computation of fist detection.

We implemented the hand gesture recognition system based on MVC model to increase the flexibility of hand gesture recognition system and simulate the browsing actions for ordinary web pages. Our system compares favorably with systems in terms of average process time.

Appendix A

The Moving Region

Gesture-based user interface often requires a user to move hands to operate the system, making the user tired easily. To resolve this problem, the system determines the Moving Region (as shown in Figure A.1 (a)) which is a small region and defined to be mapped to the whole screen (shown in Figure A.2 (b)). This region can also reduce the computation time of fist detection.

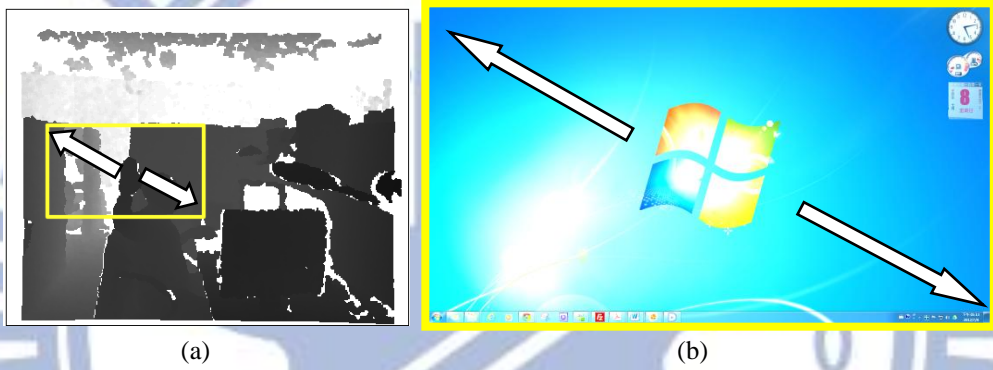


Figure A.1 (a) the Moving Region. (b) the corresponding monitor screen.

The Moving Region detection operation includes two steps: scene analysis and Moving Region adjustment. In scene analysis, the depth image will be segmented into three layers according to the depth value of hand point:

$$\text{Foreground layer } (x, y) = \begin{cases} D(x, y) & \text{if } D(x, y) < T_{low} \\ \text{NaN(Not a number)} & \text{otherwise,} \end{cases} \quad (1)$$

$$\text{Hand layer } (x, y) = \begin{cases} D(x, y) & \text{if } T_{low} \leq D(x, y) \leq T_{high} \\ \text{NaN(Not a number)} & \text{otherwise,} \end{cases} \quad (2)$$

$$\text{Background layer } (x, y) = \begin{cases} D(x, y) & \text{if } D(x, y) > T_{high} \\ \text{NaN(Not a number)} & \text{otherwise,} \end{cases} \quad (3)$$

where D is a depth image, and T_{low} and T_{high} are thresholds. (In our system, T_{low} = depth

value of hand point -190 mm. $T_{high} = \text{depth value of hand point} + 190$ mm.) Figure A.2 shows an example of the above segmentation, Figure A.2 (a) shows the depth image and Figures A.2 (b)-(d) show the segmented three layers. In order to define a Moving Region, which may not be occluded by other objects, the union of Figures A.2 (b) and (c) is used.

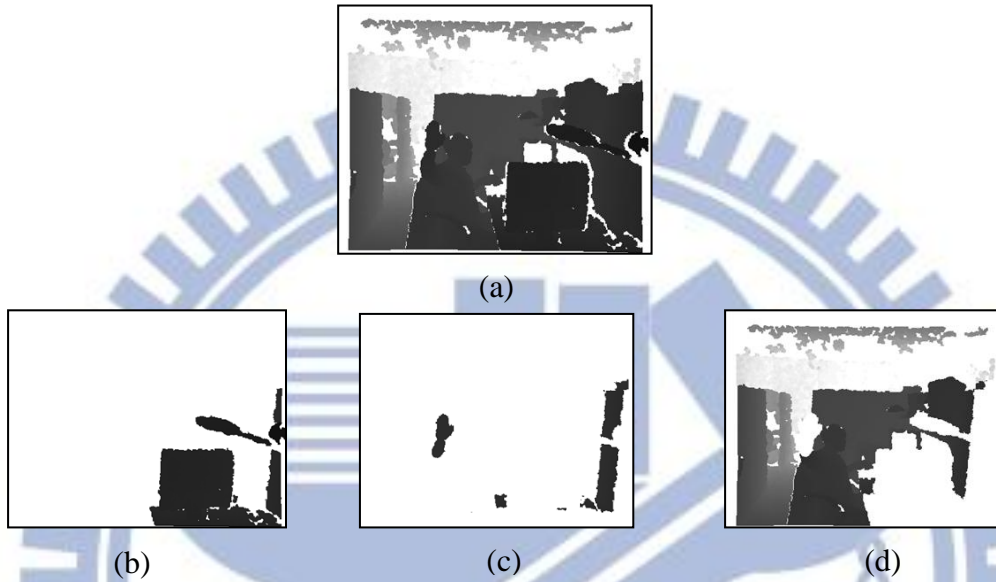


Figure A.2 An example of the layer segmentation. (a) the depth image, (b) the foreground layer, (c) the hand layer, (d) the background layer.

The Moving Region is decided by growing a initial window from the hand point until the window contain additional non-hand object. This process ensures the hand is clear visible in the Moving Region. Figure A.1 (a) shows the Moving Region represented by a yellow rectangle while Figure A.1 (b) shows the corresponding region of the monitor screen (which is the full screen in this case).

References

- [1] S. Mitra, "Gesture recognition a survey," *IEEE Transactions on Systems, Man, and Cybernetics (SMC) – Part C*, vol. 37, no. 3, pp. 311-324, 2007.
- [2] R. Dongwan and P. Junseok, "Design of an armband type contact-free Space Input Device for Human-Machine Interface," *IEEE International Conference on communications*, pp. 841-842
- [3] P. Kumar, S. S. Rautaray, and A. Agrawal, "Hand data glove: a new generation real-time mouse for human-computer interaction," *IEEE International Conference on Recent Advances in Information Technology*, pp. 750-755, 2012.
- [4] A. Ibarguren, I. Murtua, and B. Sierra, "Layered architecture for real time sign recognition: hand gesture and movement," *Engineering Applications of Artificial Intelligence*, vol. 23(7), pp. 1216-1228, 2010.
- [5] Fifth Dimension Technologies, 2012. (<http://www.5dt.com/products/pdataglove5u.html>)
- [6] Y. Fang, K. Wang, J. Cheng, and H. Lu, "A real-time hand gesture recognition method," *IEEE International Conference Multimedia and Expo*, pp.995-998, 2007.
- [7] R. Khan, A. Hanbury, J. Stöttinger, and A. Bais, "Color based skin classification," *Pattern Recognition Letters*, vol. 33, no. 2, pp. 157-163, 2012.
- [8] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 63:1-63:8, 2009.
- [9] J. Nagi, F. Ducatelle, G. A. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber and L. M. Gambardella, "Max-pooling convolutional neural networks for vision-based hand gesture recognition," *IEEE International Conference on Signal and Image Processing Applications*, pp. 342-347, 2011.
- [10] Y. Y. Pang, N. A. Ismail, and P. L. S. Gilbert, "A real time vision-based hand gesture interaction," *Fourth Asia International Conference on Analytical Modelling and Computer Simulation*, pp. 237-242, 2011.

- [11] S. Kulkarni, H. Manoj, S. David, V. Madumbu and Y. S. Kumar “Robust hand gesture recognition system using motion templates,” *International Conference on ITS Telecommunications*, pp. 431-435, 2011.
- [12] S. I. Kang, A. Roh, and H. Hong, “Using depth and skin color for hand gesture classification,” *IEEE International Conference on Consumer Electronics*, pp. 155-156, 2011.
- [13] X. Li, J. H. An, J. H. Min, and K. S. Hong, “Hand gesture recognition by stereo camera using the thinning method,” *International Conference on Multimedia Technology*, pp. 3077-3080, 2011.
- [14] J. Appenrodt, S. Handrich, A. Al-Hamadi, and B. Michaelis, “Multi stereo camera data fusion for fingertip detection in gesture recognition systems,” *International Conference of Soft Computing and Pattern Recognition*, pp. 35-40, 2010.
- [15] G. Hu and Q. Gao, “Gesture analysis using 3d camera, shape features and particle filters,” *Canadian Conference on Computer and Robot Vision*, pp. 204-211, 2011.
- [16] G. F. He, S. K. Kang, W. C. Song, and S. T. Jung, “Real-time gesture recognition using 3d depth camera,” *IEEE 2nd International Conference on Software Engineering and Service Science*, pp. 187-190, 2011.
- [17] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119-152, 1994.
- [18] SwissRanger SR4000 Overview. (<http://www.mesa-imaging.ch/prodview4k.php?cat=3D%20Camera>)
- [19] Xtion PRO LIVE. (http://tw.asus.com/Multimedia/Motion_Sensor/Xtion_PRO_LIVE/)
- [20] OpenNI User Guide. (http://www.openni.org/images/stories/pdf/openni_userguide_v4.pdf)
- [21] K. Khoshelham, “Accuracy analysis of kinect depth data,” *International Society for Photogrammetry and Remote Sensing Workshop Laser Scanning*, 2011.

- [22] G. E. Krasner and S. T. Pope. “A cookbook for using the Model-View-Controller user interface paradigm in Smalltalk-80.” *Journal of Object Oriented Programming*, vol.1, no. 3, pp. 26–49, 1988.
- [23] L. Gallo, A. P. Placitelli, and M. Ciampi, “Controller-free exploration of medical image data: experiencing the Kinect,” *Computer-Based Medical Systems*, pp. 1-6, 2011.
- [24] T. Kipshagen, M. Graw, V. Tronnier, M. Bonsanto, and U. Hofmann., “Touch-and marker-free interaction with medical software,” *Medical Physics and Biomedical Engineering*, pp. 75-78, 2009.
- [25] C. Bellmore, R. Ptucha, and A. Savakis, “Interactive display using depth and RGB sensors for face and gesture control,” *Western New York Image Processing Workshop*, pp. 1-4, 2011.

