西南民族大學

本 科 生 毕 业 设 计(论 文)

智能手表的非接触式备择交互模式的设计

Designing Alternative Contact-free Control Modalities for Smart Watches

教学单位 计算机科学与技术

姓　　名 欧长坤

学　　号 201231102123

年　　级 2012

专　　业 计算机科学与技术

导　　师 陈雅茜

职　　称 副教授

联合导师 Andreas Butz (University of Munich)

职　　称 Professor

2016 年 4 月 25 日

Contents

摘要

本文以现有智能手表产品的代表 Apple Watch 为例，对 Apple Watch 上的交互模式的优缺点进行了全面的分析，并据此给出了一套非接触式的备择设计。其配合了 Haptic Engine 对用户的直观震动反馈，完成了对基础点按、滑动、Digital Crown、Force Touch 等系列原生交互的非接触式手势交互设计，此设计将嵌套两步逻辑的十种原生交互简化为了单步逻辑上的八种备择交互，且消除了接触式交互的依赖，解决了现有交互中对双手依赖的缺陷，并同时说明了给出的备择设计的交互完备性。通过用户调研，对本套交互方式的设计进行了评估，结果显示该方案的操作逻辑和舒适度良好。在最后，本文对设计的硬件结构、交互方式和系统架构的现有缺陷进行了讨论，并从中得到的启示，给出了可行的解决思路。

关键词 智能手表；非接触式；手势交互；备择设计

Abstract

In this thesis, we explore Apple Watch as the representative of smart watches, and analysed the advanteges and disadvanteges of interaction pattern in Apple Watch with watchOS 2. According to this, we introduce a contact-free alternative design for smart watch interaction.

This design, combine with the Haptic feedback, convert the basic tap, swipe, Digital Crown, Force Touch and etc. native watch interaction to a contact-free gesture interaction. It simplefied ten native interaction with two-step logic to only eight alternative interaction with sigle-step logic, eliminated contact-required interaction method, solved the bimanual interaction within smart watches, and explained the completeness of this alternative design.

Through user study, we evaluate these alternative designs and the result shows the interaction operate logic and gesture confortable all acceptable.

Finally, we discussed the weakness of current hardware structure, interaction design and system architecture. Trough this, we proposed a few implications for the future improvements.

Keywords Smart Watches; Contact-free; Gesture Interaction; Alternative Design

1 Introduction

With 30-year rapid development of computer chip technology, wearable devices sprang up in the 2010s, mainly represented by smart watches and smart bands in civil electronics market. With regard to smart bands, they are only collecting data of users when wearing them. In essence, there does not exist direct interaction between them and users. As for smart watches, despite that they are installed with a large number of operating systems with simplified and appropriate designs, there still exist performance weaknesses, thus bringing great challenges and opportunities for the software design of wearable devices [1].

1.1 HCI & Wearable Computing

Since the middle and late 1990s, the concept of wearable devices has gradually come into people' s view.

Wearable devices range from simple input devices(such as smart bands) to complicated devices with operating systems and communication functions. Wearable devices are designed to provide computing devices more intimate than smart phones for human beings. In essence, wearable computing is based on this kind of small and wearable computers. "Always-on" is a distinctive feature of wearable computing. Wearable devices are worn on a certain part of users to perceive their environment and observe their behavior.

Wearable computing aims at providing people with personal assistant services, such as convenient access to the Internet, event notification and information collection. This ubiquitous computing technology based on wearable computing integrates multi-modal interaction, context-interactive model and augmented reality into one, so it is the ultimate task desired in the field of human-computer interaction [2, 3].

However, limited to current level of science and marketing, these researches are still on a primary stage. Above all, when reflecting on the reasons for adding a device on themselves, people would find there it makes little sense to wear them, so wearable devices could be easily substituted. Even so, researches on human-computer interaction of wearable devices are still on the way and have gone ahead of electronics. Although there is still a long way to go to realize these designs and that they might never be put into used, these research methods and concepts arising from them are of significance for people to continue pondering [4] and reflecting and promote the progress of the industry step by step.

1.2 Subject Significance

Nowadays, the development of smart phones has been in a downturn, but wearable devices fail to gain great momentum as desired. Regardless of their acceptability, usability and pricing, smart watches do not have obvious significance of existence, for their usages are much related to the contexts they are in.

Therefore, the following issues must be considering carefully before design alternative interaction:

1. How to insure interaction correspond human intuition and comfort level?
2. How to erase sense of a separate existence(Social Acceptability)?
3. How to balance affordance and system complexity?

This thesis explore Apple Watch as the representative of smart watches, considering watchOS features, designed a alternative interaction modality which can be contact-free for watch interaction. Within this interaction pattern, user target can be more significant, smart watch application scenarios gains its extend and the complexity of interaction logic can be simplified so that the meaning of smart watches will be indirectly increase.

1.3 Related Works

Smart watch interface is the most intuition part of the interaction [5], research on interfaces has been more mature and in-depth [6, 7]. Apple Watch as a representative is the most simple, excellent interfaces, and its Human Interface Guidelines [8] has become the standard guidelines of smart watches.

However, even in Apple Watch, user still need two hands to performing their actions. So, if we have to redesign the whole interaction and release the bimanual interaction required on smart watches, then we have to discuss gesture techniques due to we only have one hand to perform interactions at this time.

1.3.1 Gesture Interaction

Gesture as a technique in human-computer interaction has been enduring, it can be cetegorise as 2D gestures and 3D gestures. For 2D gestures there are \$1 [9] and \$$n$ [10] algorithms are maturing and the main idea is resampling gesture point in same interval, then use 2-norm to caculate and compare gesture series. For 3D gestures, the hardest part is to confirm where gesture starts and ends, however most gesture recognize algorithm always avoid this problem by using recognize a specific gesture or construct a state machine of gesture [11–15]. Fortunately, interaction on smart watches always after user rise their hand, all information output through display ensures a fact that the amplitude of arm could not too much or it will abstract users. Thus, interactions on smart watches could be gesture manipulated but not identical.

L. Christian et al. [16] propused a complex gesture set of pinch for user to operate smart watches. Although it's purpose as same as ours, but their pinch gesture required redesign whole interaction logic on smart watches, basicly, the circumstances is unrealistic.

For research on contact-free interaction, [17] use a RGB camera attached on Google glass and applied vision method to recognize gesture, but this method is only appropriate for classes user not everyone. Kerber et al. [18] even expect to use diverse ortation of arm.

In these research, the interaction gains its augmentability, nonetheless they have ever prepense these techniques applicable existence form.

1.3.2 Extension Interaction

Some research inspire us gives various idea, they expand the interaction outthrough watch itself, they put interaction around the watch surface [19,20], band [21], even skin [22]. [23] puts sensors around the watch and make it recognize gesture over the watch surface.

These interaction extension break through the limits of screen size though, but its still a bimanual interaction, still tiredness and inconvenience for users.

This extension of interaction isn't limited on smart watch, [24] augmented the smart phone interaction by smart watch, and [25] give us an implication for use watch position in space, essentially, it a kind of Virtual Reality development. By this, we gains the multiplicity of watch state and even smart phone and watch works simultaneously.

In summary, although the interactive mode research coverage broad in smart watches, but these existing research on smart watches re generally avoided and did not consider the following questions:

- The theoretical interaction techniques could not compatible with existing smart watch interactive mode which is not universal.
- These techniques are not broad enough, most of them didn't make a full analyze on a series of products and the design itself always give up on affordance.
- Most theoretical techniques are suitable for bimanual interaction, there is no obvious context for increasing user stickiness on smart watches;
- Do not have much considering of the relationship between interaction mode and user interfaces, on the one hand due to the lower social acceptability, and the other hand is because of the interactive logic not too strong, learning costs of users too high.

2 Interaction Techniques on Apple Watch

Most interactive mode in Apple Watch follows smart phone touch screen interactive mode [8]. To design a contact-free alternative interactive mode on Apple Watch, we must analyze and clearly figure out extant the advantages and weakness of interactive mode in Apple Watch.

2.1 Traditional Techniques

Apple Inc. believe the traditional multi-touch techniques on a small screen is severely affected user experiences. Therefore they didn't put multi-touch into Apple Watch. Besides, to operate object on smart watch screen, user only have the basic tap gesture and swipe up, down, left and right for these four directions, as shown on Figure 2.1[1].

Figure 2.1: **Traditional Techniques**

2.1.1 Tap

Ordinary single tap on smart watches is as same as its on the smart phones. This tap gesture as an human-computer interaction techniques has been accepted by the public for a long time, and this natural technique gains its lowest cost among the most technologies.

However, the usability of this interaction will be reduced when the screen size reduced, in another words, we are approaching the limits when we reduce the material size and move its position on our body. According to FFitts' Law [26]:

$$T = a + b \log_2 \left(\frac{A}{\sqrt{2\pi e(\sigma^2 - \sigma_a^2)}} + 1 \right) \tag{2.1}$$

where σ is the standard deviation of touch position distribution, σ_a is the absolute accuracy of the finger input, A is a distance as the start point to the target center.

Considering FFitts' Law, A will be increase when screen size limited on the one hand, and σ will not large than its on smart phone screen, on the other hand σ_a will not change when we consider them in same object.

Thus, T value will be significantly larger, which means tap interaction gains poor usability on smart watches.

2.1.2 Swipe

Continuous tap generate swipe gesture interaction. To process the weakness on smart watches, Apple Watch as much as possible to make the interaction to be handle be swipe gestures.

[1]Image Source: https://developer.apple.com/watch/human-interface-guidelines/

Horizontal direction view changes can be handled by swipe gesture. It makes interaction functions such as horizontal view change, switch back to the previous view, etc.

There is another method to take place of swipe gesture in vertical, which is using Digital Crown. In case the vertical swipe interaction is not a subset of Digital Crown's function. It is because of Digital Crown cannot call notification center when user stay on watchOS dial interface.

2.2 Special Techniques

Apple Watch introduced three more interactive hardware based on traditional interactive touch screen mode. They are Digital Crown, Force Touch and Haptic Engine, as shown in Figure 2.2[2].

(a) Digital Crown (b) Force Touch (c) Haptic Engine

Figure 2.2: **Special Techniques**: Digital Crown, Force Touch and Haptic Engine is the most significant feature in Apple Watch than other smart watch products.

2.2.1 Digital Crown

The Digital Crown lets users scroll content without obstructing their view of that content. It is a new interaction technique introduced by Apple Inc. This technique is compared with the two revolutionary interaction techniques: Desktop Mouse and Multi-Touch screen when Apple Inc. were introducing this, which means Apple Inc. maintain Digital Crown is a revolutionary technique on smart watches.

This interaction take the advantage of traditional watch clock function, which is visible but also insufficiency. In a conventional watch, the clock dial in the normal state does not have any function, only when the knob is pulled out and the only function is adjusting time. This device is uselessness in the most of time. It's a typical demand-driven design. Designers are not carefully think about its existence but just habitual in use.

On the Apple Watch, content are displayed vertically and it's restricted on a fixed width. When Digital Crown scroll up and down, the content will be change in vertical direction, as shown in Figure 2.2(a); Besides, Digital Crown can also adjust the playback volume of the music.

2.2.2 Force Touch

Apple Watch display not only senses a touch, it senses the force applied by user's finger and responds accordingly. Force Touch is the first applying in consumer products. In academia, this technology has been studied years.

[27–29] studied how to enhance touch feedback, force level and touch area. Force Touch applied them into Apple Watch.

[2]Image Source: https://developer.apple.com/watch/human-interface-guidelines/

Force Touch contains two level of touch behavior, the first level is traditional touch action, it can reconize normal tap gesture; The second level is actually Force Touch, at this moment, user need increase finger's force value, then watchOS will response and handle this interaction. As illustrate on Figure 2.2(b), obviously, this interaction is a typical interaction mode without affordance, these functions are perceptible only the operation performed.

2.2.3 Haptic Engine

Haptic Engine is built-in vibration member, as illustrate in Figure 2.2(c). It is an important way to get the user's attention and to convey important information. Haptic Engine are not just a normal vibration but also a high precision hardware, it can transfer user's heartbeat to another, make user feel Force Touch perform notification, then enhance the user experiences of Apple Watch. Thus, whatever how to modify current interaction mode into alternative design, we should always provide the Haptic Feedback to users. In other words, Haptic Engine vibration feedback is provided to be the key of alternative interaction design.

2.3 Other Techniques

In addition, except the basic interaction techniques above, there are two highest priority interactive method on Apple Watch, as Figure 2.3 illustrated[3].

(a) Side Button (b) Voice Control

Figure 2.3: **Other interaction techniques**: Side button gives us the top-level interaction for Friends, and Siri is the only text entry on the Apple Watch.

Two buttons on the side of Apple Watch actually is the highest priority of interactions, because wherever the user in watchOS, it alway takes user to the Friend view and Siri interface (Note that a single tap of Digital Crown is not able to mark as highest priority interaction, because a single tap perform results may not be unique: It may be in the dial, or it may be in total App Interfaces).

2.3.1 Side Button

Apple Inc. may have noticed that the Apple Watch application gains its complexity and inconvenience, they follows the design of the power button on smart phone and they still add another physical buttons called Side Button next to the Digital Crown, as Figure 2.3(a).

The existence of two different pieces of renovation interactive content inside the Side Button, when user perform a button click, it will access the Friends screen. From this screen, they can call friends,

[3]图片源：https://developer.apple.com/watch/human-interface-guidelines/

send message, or interact by sending sketches, taps, and even their heartbeat. When user press the Side Button twice, it will access the Apple Pay.

2.3.2 Voice Control

Although there are a lot of input method design on wearable devices, but Apple Watch did not take any of them on watchOS, text input entrance (even virtual keyboard) are not directly setup into watchOS, thus Siri becomes the only interface allows user input text by speaking, as shown in Figure 2.3(b).

For weak up Siri, user have two way to do. One is say "Hey Siri" when Apple Watch is on awake state; Another is give a long-press to Digital Crown.

However, no matter which way to choose, text input results will not ideal. This is because Siri's nature language recognition process based on statistics and it always require a paired iPhone to access Internet. Any part of this gets a problem can crash the whole input progress.

It is worth mentioning that the interaction method we discussed above, only Siri voice control is a contact-free interaction design.

3 Alternative Design of Contact-free

We analyzed the interaction mode on Apple Watch in last chapter, all the interaction method required two hands to be performed, except the voice interaction. Considering the actual scenario, if a user's hands are free to do anything, there is no obvious reason to make things done on smart watches due to we have a powerful smart phone, moreover, to operate a phone actually only needs one hand. Therefore, this contact type of interaction on smart watch essentially isn't a stickiness interaction mode.

3.1 Interaction Pattern Design

3.1.1 Tap & View Switch

Tap and view switching operation involving the following five operations: ordinary tap, Force Touch tap, switch to left view, switch to right view, switch to previous view.

We design the alternative interaction of tap gesture to be two finger pinch, thumb pinch with the other four fingers can fully mapping all tap interaction, such as thumb with index finger can replace normal tap event, thumb with middle and ring finger can replace view switching, and thumb with little finger can replace back to previous view.

Besides, pinch gesture with thumb and index finger can also simulate Force Touch, see more details in Section 3.1.3.

3.1.2 Swipe & Continuous Adjustment

Swipe and continue changes actually belongs to Digital Crown interaction in different scenarios. In General view down swipe scenario, it can scroll view contents up and down; when the interactive object choose as slider bar, it also can adjust value continuously.

We can implement this interaction by using two fingers pinch gesture on single hand, which means thumb slides on index finger, and if the interactive object is slide bar, then this kind of two fingers swipe can also regulate slide bar value.

3.1.3 Force Touch Simulation

An open source project called Forcify [30] aims to develop an universe touch events handler framework, convert any Web App click events to 3D Touch Events[1]. But the time handler need to customize by developer and the force touch simulator is a linear function. Therefore, this project needs to be improved.

At first, we split touch events to two stage, the first stage is as normal touch events, the second stage is as Force Touch Events, and set content DELAY to express the time delay of Force Touch, and DURATION to express the time duration of Force Touch from minimum to maximum. Now we focus on this two constants value.

In AugmentedTouch Project [31], we applied a user study and published a dataset contains 16 users of 4 different posture, altogether contains 61440 times tap data. In this dataset, we record the time of user's finger stay on screen and Figure 3.1 shows its distribution.

[1] Apple rename Force Touch in iOS as 3D Touch.

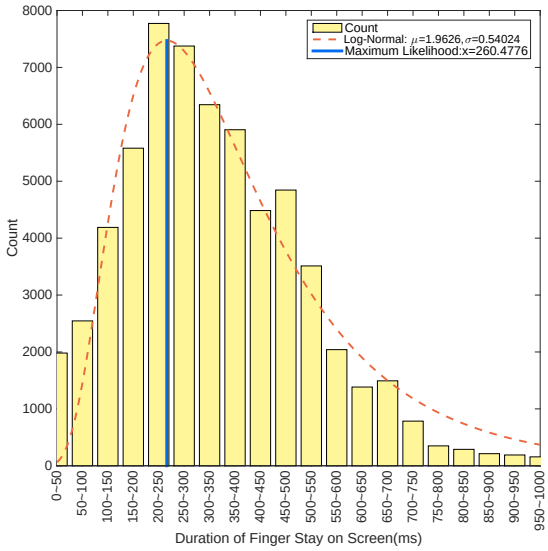


Figure 3.1: Statistics of finger stay on screen

As it shows the distribution on Figure 3.1 we can get the time of a finger stay on a screen mainly focus on 260ms, obey to log-normal distribution. In this regard, we could assume the average time of finger stay on touch screen is 250 ms, so DELAY equals 250 ms; In addition, we set the total duration is five times of the ordinary touch events time, so DURATION equals 1000 ms; Then to trigger the Force Touch user need stay finger on touch screen longer than 1.25 seconds. Considering human's finger pressure of touch, we also assume the intensity of the rate of force value is linear, then we can ensure the entire curve is a smooth curve.

In conclusion, let the time of pressing to be t_{press}, then Force Touch can be simulate by a formula 3.1.

$$v_F = \begin{cases} 0 & \text{if } t_{\text{press}} < \text{DELAY} \\ \left(\frac{t_{\text{press}} - \text{DELAY}}{\text{DURATION}}\right)^2 & \text{if } 0 < t_{\text{press}} - \text{DELAY} < \text{DURATION} \\ 1 & \text{Otherwise} \end{cases} \tag{3.1}$$

Among the formula, v_F represents the pressure value, DELAY = 250, DURATION = 1000, both in milliseconds(ms), as shown in Figure 3.2.

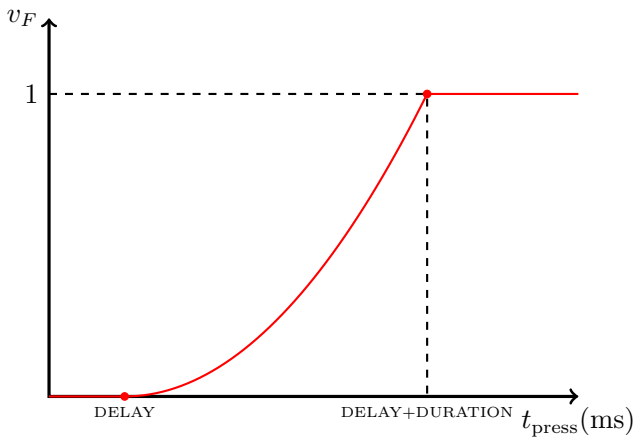


Figure 3.2: Force Touch functional image

3.1.4 Haptic Feedback Design

According to the developer documentation of watchOS[2], Haptic Engine can express several types, as its illustrate on Code 3.1.

```
public enum WKHapticType : Int {
    case Notification
    case DirectionUp
    case DirectionDown
    case Success
    case Failure
    case Retry
    case Start
    case Stop
    case Click
}
```

Code 3.1: Types of Haptic feedback

Success, Failure, Retry type of Haptic feedback are native results in watch operation. Then we set Success, Failure as same as its original function, but the context of Retry is basically as same as Failure, so we keep this feedback for the future.

Notification feedback still needs to present notification feedback, so we don't change its design; DirectionUp and DirectionDown only happens on Digital Crown scroll up and down in the end, so we follow this function to two fingers swipe, when value comes to the maximum value, use DirectionUp feedback and vice versa.

Start, Stop, Click and Retry this four type of feedback is suitable for thumb with index, middle, ring finger pinch (three) and Force Touch operate, which means feedback of Click is for normal tap, feedback of Stop is for Force Touch, feedback of Start is for view switching to next, and feedback of Retry just right for view switching to previous.

3.2 Completeness of Alternative Design

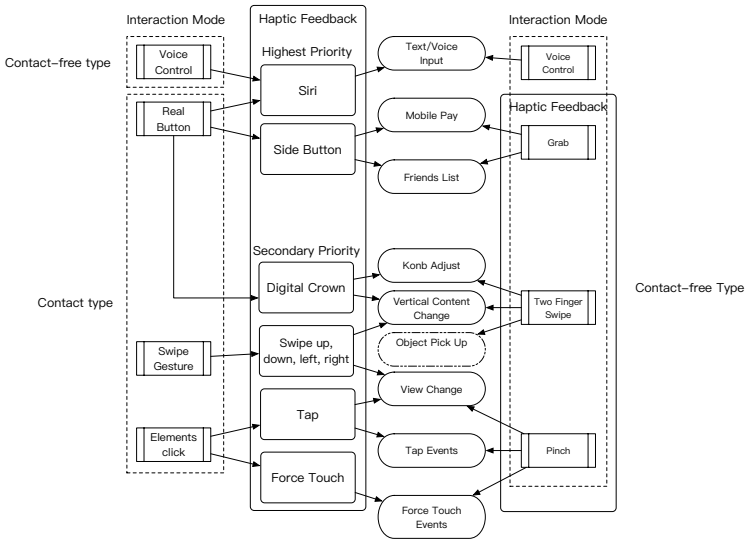


Figure 3.3: Summary of Contact-free Interaction

[2]watchOS version stay on 2.2 when this thesis finished.

In this chapter, we fully redesigned the interaction mode on Apple Watch, it simplifies the interaction logic between elements, as a summary we list the design logic in Figure 3.3. This design, combine with the Haptic feedback, convert the basic tap, swipe, Digital Crown, Force Touch and etc. native watch interaction to a contact-free gesture interaction. It simplified ten native interaction with two-step logic to only eight alternative interaction with single-step logic, eliminated contact-required interaction.

As illustrate in Figure 3.3, this interaction scheme is completeness, which means we archived all the function without original interaction method.

4 System Design

4.1 Framework Selection

4.1.1 LeapMotion & LeapJS

LeapMotion [32] is a general term for a hardware input device and its software, its hardware built by two depth camera and an infrared detector that can detect the hand within the field of view; its software built-in a skeleton model of hands for hand reconstruct[1].

LeapMotion SKD provides two varieties of API for getting the LeapMotion data: a native interface and a WebSocket interface, WebSocket provides JavaScript interface in browser environment. It is conforms to RFC6455[2], and running on desktop default port 6437.

LeapJS is the JavaScript framework for LeapMotion Controller. Using LeapJS enables Web font-end communication with LeapMotion, and this framework is used for processing LeapMotion JSON data. With NodeJS, LeapJS also can running on server side, thus we can also processing LeapMotion interaction on server side by JavaScript. Leap API gives a Frame object, which contains a Hand object if the hand of users can be detected in LeapMotion field of view. LeapMotion through its built-in model of hands, reconstruct hands object and then provide its to developers. for example, it provides hand direction, hand finger direction, hand position in LeapMotion coordinate system. With LeapMotion, developer and researcher can avoid the basic image segment works and other stuff, them keep focus on gesture algorithm [35–39].

In addition, 3D gesture already gains its application in simple method, [40] gives a LeapMotion control interaction for TV.

4.1.2 watchOS & WatchConnectivity

watchOS is the operating system that runs on the Apple Watch. When watchOS just released, applications only as an extension for iOS App, watchOS only illustrate animations, all code runs on iOS.

With lunching of watchOS 2, now applications can use WatchConnectivity framework to pass data between iOS and watchOS.

WatchConnectivity framework aims to makes message between iOS and watchOS to be a message in system level, developer shouldn't care much about it. These communication will start a session and only WCSession.defaultSession.reachable is true, the delivery can be started. So if we would use WatchConnectivity, we must implement WCSessionDelegate protocol.

Besides, WatchConnectivity contains two different communication module: background module and interactive module. In background module, OS will push the message into a queue, then process it when App wake up; In interactive module, message must transfer through OS and it is an immediately communication, only happens when WCSession.defaultSession().reachable is true.

It is worth mentioning that, in interactive module, even if the iOS client program has not yet started, the Apple Watch is still able to wake up an iOS App from background.

[1] LeapMotion's official accuracy is 0.01 mm, but [33,34] report its actual accuracy is 0.2 mm

[2] http://tools.ietf.org/html/rfc6455

4.2 Architecture

4.2.1 Communication Structure

watchOS was split from iOS App Extension since 2.0 version, since then, watchOS can actually execute code on Apple Watch, that make watchOS App can management communication [41] between iOS App and itself, the structure as illustrate on Figure 4.1. This makes timely communication with the outside world be possible.

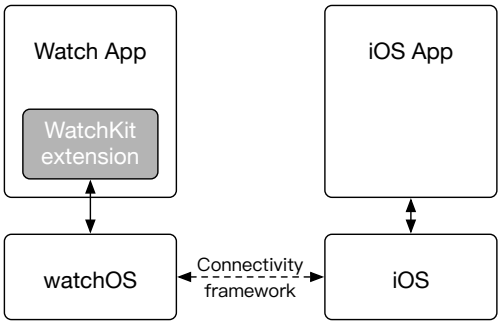


Figure 4.1: Connection between Watch App, WatchKit Extension and iOS App

However, there are still load of limits for networking on watchOS. In watchOS 2, Apple Watch only when its paired iPhone disconnected and its also in a saved WiFi environment that can access internet by using NSURLSession, these conditions are rigor.

In consideration of above, we designed the communication architecture and illustrate it on Figure 4.2.

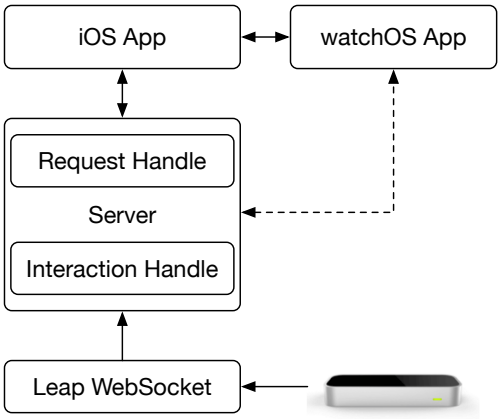


Figure 4.2: **Communication Architecture**: iOS App as a router to transfer message between watchOS and server, instead of watchOS directly communicate to server

Among this architecture, iOS App will be the bridge for watchOS App and server for processing interaction message, watchOS only cares content present.

4.2.2 Client Structure

Client side includes iOS side and watchOS side, as shows on Figure 4.3.

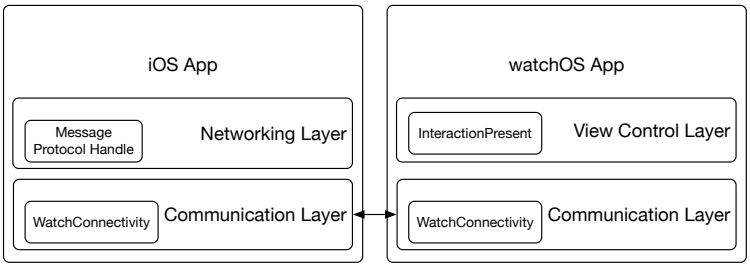


Figure 4.3: **Client side structure**: iOS App process server message then send the results to communication layer to watchOS, watchOS react the interaction manipulate UI elements by view controller layer when it recived the message.

4.2.3 Server Structure

Server side structure as its shown on Figure 4.4, the core module is Interaction Handle Layer, this layer process raw interaction data from sensors and post the interaction results to Request Handle layer then distribute it to suitable devices.

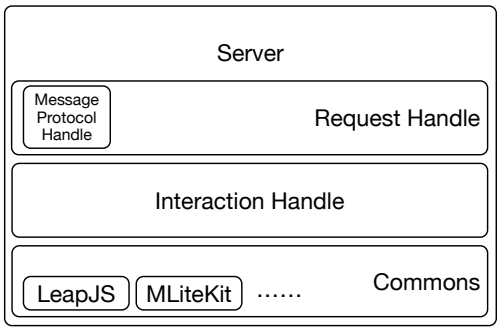


Figure 4.4: **Server side structure**: Interaction Handle Layer process raw interaction data, then post it to Request Handle Layer for data transfer

4.3 Communication Protocol

We need design the interaction message communication protocol between watchOS, iOS and server.

According to Figure 3.3, basic interaction gesture contains grab, two fingers swipe and tree different tap(thumb with index, middle, ring finger), the two fingers means thumb and index finger.

For this reason, interaction message field design as in Table 4.1.

Table 4.1: Interaction Protocol Fields

Field	Type	Description
pinchTimeStramp	Date	Time of pinch operate
pinchIndex	int	Finger index of pinch, from 1 to 4 express index finger to little finger, -1 means no finger
pinchStrength	double	Strength of pinch, from 0 to 1, real number
grabStrength	double	Grab strength, from 0 to 1, real number
forceValue	double	Simulate value of Force Touch

4.4 Demonstrate Programs

Although we conducted a comprehensive interactive design, but due to the restriction on the development of Apple Watch, we are unable to operate the view in system-level. In this thesis, we gives five alternative interactive demos with different effects.

- Demo 1: This demo shows the alternative interaction design of tap gesture;
- Demo 2: This demo shows the alternative interaction design of swipe gesture;
- Demo 3: This demo shows the alternative interaction design of Digital Crown;
- Demo 4: This demo shows the alternative interaction design of Force Touch;
- Demo 5: This demo illustrate a contact-free game experiences on Apple Watch.

Above five demos video can be found in YouTube [3] , for other related resources, such as source code, see Appendix A.

[3] https://www.youtube.com/playlist?list=PLwUqqMt5en7c2QaQ_DkuvZm9dGTz6RjRM

5 Implementations

5.1 Development Environment

Whatever client side or server side, there always framework depended, so build development environment is inevitable. In this project, the server is encoded using NodeJS. With limited space, for basic NodeJS environment, such as common configuration tool, NPM package management are omitted in this thesis. Here we introduce the most important framework structure and build of this platform, LeapJS; On client side, we use Swift to code and we do not import any third party framework, but WatchConnectivity need to be use for communication [42]. After iOS 9, every App in iOS need to use HTTPS request, but there is a way to configure HTTP request in iOS 9.

5.1.1 Configuration of Local LeapMotion

LeapMotion provides development environment in Mac OS X, and provides a variety of programming language. Based on above discussion, we need to configure the server LeapMotion body and LeapJS.

1. Installation

First of all, to use LeapMotion API, we have to import LeapSDK into our desktop host application.

Secondly, we need add dependency of LeapJS in Node App's package.json file:

```
"dependencies": {
    "leapjs": "^0.6.4"
}
```

The next step is install LeapJS by npm install.

2. Configs

By the limitations of LeapMotion [32], WebSocket service doesn't open a non-local access in default setting, so we will need to configure Leap then enable non-local client connections.

This requires to configure LeapMotion profile. Before we modify the configuration, we have to close LeapMotion related services. On Mac OS X, we use the following command to shut down LeapMotion daemon:

```
sudo launchctl unload /Library/LaunchDaemons/com.leapmotion.leapd.plist
```

Next, we need to modify LeapMotion profile. According to the official documents , Leap contains two different configurations and the highest priority belongs to control panel, so we need edit the following directory:

```
$HOME/Library/Application\ Support/Leap\ Motion
```

Find config.json configuration file, editing config.json file and add an arbitrary position in the configuration field:

```
"websockets_allow_remote": true
```

finally obtained:

```
"configuration": {
    "websockets_allow_remote": true,
    "background_app_mode": 2,
```

```
    "images_mode": 2,
    "interaction_box_auto": true,
    "power_saving_adapter": true,
    "robust_mode_enabled": false,
    "tracking_tool_enabled": true
}
```

save and quit, restart LeapMotion Service and then we finished the configuration of LeapMotion:

```
sudo launchctl load /Library/LaunchDaemons/com.leapmotion.leapd.plist
```

5.1.2 watchOS Networking Access

From iOS 9. Apple introduced App Transport Security feature for iOS App, this causes iOS App could not access to the Interaction with HTTP requests.

Due to watchOS networking need iOS App to transfer its requests in the most of time, we need applying two more configuration steps, and the results shows on Figure 5.1.

1. Add NSAppTransportSecurity as a Dictionary type into Info.plist;

2. Add NSAllowsArbitraryLoads key as a Boolean type into NSAppTransportSecurity, then set its value YES.

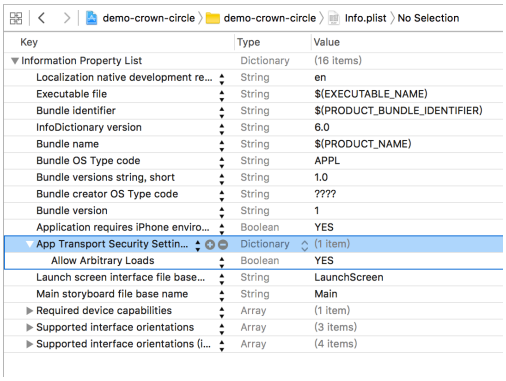


Figure 5.1: Configuration results of info.plist file

5.2 Server Side

To archive a HTTP Server with port 10086 is the first step of developing, it is implement by NodeJS and codes shows on Code 5.1.

```
1 var http = require('http');
2 function handler (req, res) {
3     res.writeHead(200);
4 }
5 http.createServer(handler).listen(10086);
6 console.log("Server running at http://locoalhost:10086")
```

Code 5.1: Create HTTP server on 10086

Now, we focus on how to process LeapMotion Gesture in the following content.

5.2.1 Tap and Force Touch Processing

LeapMotion Software itself will reconstruct hand model of users, and its APIs provide the skeleton data, including fingers tip position. To recognize pinch gesture, we can calculate two finger tips, so that we can iterate all finger tips except thumb, the implementation code shows on Code 5.2.

```
function findPinchingFinger(hand, closest){
    var pincher;
    //var closest = 500;
    for(var f = 1; f < 5; f++)
    {
        current = hand.fingers[f];
        distance = leap.vec3.distance(hand.thumb.tipPosition, current.tipPosition);
        if(current != hand.thumb && distance < closest)
        {
            closest = distance;
            pincher = current;
        }
    }
    // mark time of start pinch
    if (pincher.type != pf.pinchIndex) {
        // pf is a gloable value for message protocol
        pf.pinchTimeStramp = Date.now();
    }
    return pincher;
}
```

Code 5.2: Tap Gesture Recognize

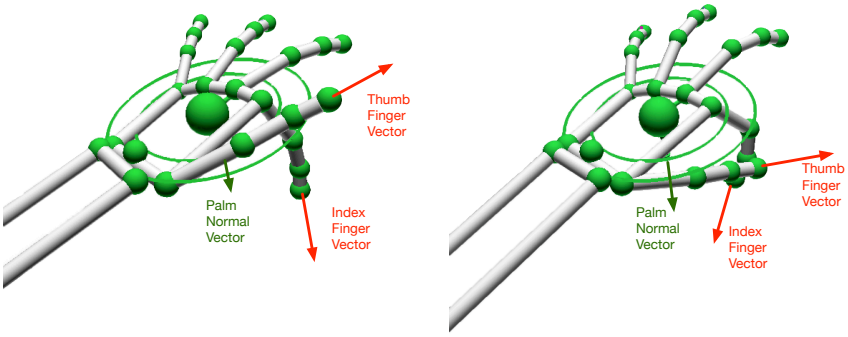
We have discussed Force Touch simulation in Subsection 3.1.3, now the implementation of JavaScript code shows on Code 5.3.

```
function forceValue(){
    // pf is the global message protocol object
    if (pf.pinchTimeStramp === 0) {
        return -1;
    }

    var delay = 250;
    var duration = 1000;

    // calculate force touch value
    var value = ((Date.now() - pf.pinchTimeStramp) - delay)/duration;
    var force = (value >=1 ) ? 1 : value*value;

    return force;
}
```

Code 5.3: Force Touch Simulation

5.2.2 Two Fingers Swipe

For two fingers swipe, it seems difficult to identify, but in fact we can converted it to pinch degree between two fingers. In fact LeapMotion itself provides thumb and index finger pinch degree parameter, but because of its results is universe for every fingers and cannot customize by developer. For this reason, we have to reimplement this function.



(a) Before (b) After

Figure 5.2: **Two finger swipe**: two finger swipe can trigger events by palm normal vector and finger direction vector, 3D models in this figure are generate by LeapMotion Visualizer

Two finger swipe gesture shows on Figure 5.2, when hand presented as Figure **??**, the normal vector of palm and direction vector of index infer are parallel, thumb direction vector and palm normal vector are almost vertical, so we can use this character to trigger this events, as its illustrate on Code 5.4.

```
function findPinchingStrength(hand) {
    var thumbDirection = hand.fingers[0].direction;
    var indexDirection = hand.fingers[1].direction;
    var dotProduct = leap.vec3.dot(thumbDirection, indexDirection);
    // remove grab condition
    if (dotProduct < 0.7 && hand.grabStrength < 0.85) {
        return hand.pinchStrength;
    } else {
        return -1;
    }
}
```

Code 5.4: Two fingers swipe recognize

5.2.3 Grab

We don't need reimplement grab gesture, LeapMotion itself provides grab parameter of a hand, we can straightly get this data from Frame object, as shown on Code 5.5.

```
grabStrength = function getGrabStength(frame) {
    return frame.hands[0].grabStrength;
}
```

Code 5.5: grab recognize

5.3 Client Side

The key of client encoding is the communication layer and view controller layer , communication layer also contains communications between iOS, watchOS and server, we use Swift [1] to complete the associated code [43, 44].

[1] Swift version 2.2.

5.3.1 View Controller Layer

View Controller Layer is basically a response for interactive message, as an example, we shows the core code of design for two finger swipe gesture as a Digital Crown alternative design.

Picker class on Apple Watch need developer load resources themselves, code as shown on Code 5.6.

```
var images: [UIImage]! = []
var pickerItems: [WKPickerItem]! = []
for (var i=0; i<=MAX_NUM; i += 1) {
    let name = "progress-\(i)"
    images.append(UIImage(named: name)!)

    let pickerItem = WKPickerItem()
    pickerItem.title = "\(i)"
    pickerItems.append(pickerItem)
}
let circleImages = UIImage.animatedImageWithImages(images, duration: 0.0)
circleGroup.setBackgroundImage(circleImages)
picker.setCoordinatedAnimations([circleGroup])
picker.setItems(pickerItems)
```

Code 5.6: Load resources of Picker class

We mapping the pinchStrength value from [0,1] to [0, MAX_NUM], then use setter function of Picker to modify UI interface, as shown on Code 5.7.

```
func indexSelect(message: InteractionMessage?) {
    index = Int(message!.pinchStrength * MAX_NUM)
    // haptic feedback
    watchDevice.playHaptic(WKHapticType.DirectionUp)
    picker.setSelectedItemIndex(index)
}
```

Code 5.7: Picker UI manipulate

5.3.2 Communication Layer

To communicate with the server, we can use HTTP request from server to gesture data, which is in JSON format. In Code 5.8, completeFlag guarantees only after a previous request completes program will trigger next request out. The constructor of Gesture class will parsing JSON data and contract a Gesture object model from the data dictionary.

```
func connectServer() {
    // member variable
    if completeFlag == 0 {
        return
    }
    task = session.dataTaskWithURL(url, completionHandler: { (data, res, error) ->
    Void in
        if let e = error {
            print("dataTaskWithURL fail: \(e.debugDescription)")
            return
        }
        if let d = data {
            print("\(NSString(data: d, encoding: NSUTF8StringEncoding))")
```

```
13
14              if let jsonObj = try? NSJSONSerialization.JSONObjectWithData(d, options:
    NSJSONReadingOptions.AllowFragments) as? NSDictionary {
15                  self.Gesture = Gesture(fromDictionary: jsonObj!)
16              }
17              self.completeFlag = 1
18
19          }
20      })
21      task!.resume()
22  }
```

Code 5.8: Communication between iOS and Server

To establish a connection between iOS and watchOS, the only way to archive this is to use WatchConnectivity framework, Code 5.9 and Code 5.10 list the core implementation of send message to watchOS and receive message from iOS.

```
1  func updateMessage() {
2      if WCSession.defaultSession().reachable {
3          let content:[String:String] = ["x":PinchFinger.text!, "y":PinchStrength.text!,
       "z":GrabStrength.text!]
4          let message = ["up": content]
5          WCSession.defaultSession().sendMessage(
6              message, replyHandler: { (replyMessage) -> Void in
7                  print("send success..")
8              }) { (error) -> Void in
9                  print(error)
10         }
11     }
12 }
```

Code 5.9: **Communication between iOS and watchOS**: Send Message

```
1  extension InterfaceController: WCSessionDelegate {
2      func session(session: WCSession, didReceiveMessage message: [String : AnyObject])
       {
3          guard message["up"] as? [String:String] != nil else {return}
4          let contents = message["up"] as! [String : String]
5          self.interaction(contents)
6      }
7  }
```

Code 5.10: **Communication between watchOS and iOS**: Receive message

6 User Study

We conduct an user study above alternative design from April 14, 2016 to April 15, 2016, which is located at the Southwest University for Nationalities Wuhou Campus Bibliothek ground floor reading room.

6.1 User Tasks

In this study, there are five operations we need to evaluate: tap, force touch, digital crown, swipe left and swipe right. According this, we designed two completely group tasks, and they are Calling(two way to archive) switch watch digital plate. These tasks are above five operations combinations.

- **Task 1**: Use side button. Find top contacts, then select a person and make the call. The task logic is rise hand first, push the side button and then rotate Digital Crown to select a person, tap to select this person and make the phone call.
- **Task 2**: Use Call App. Push Digital Crown enter into Application desktop first, tap Call App then make the phone call. The task logic is rise hand first, click Digital Crown, tap screen to select Call App, select a person and then tap to make a phone call.
- **Task 3**: The task of switch watch layout is simple than the other two tasks. The most important operation is make a Force Touch on watch screen. The task logic is rise hand first, make a Force Touch on watch screen, and swipe to left or right change its interface, then tap the screen and finish this task.

Before starting user tasks, we teach our participants about how to interact on a smart watch, and teach them 13 gestures without any hint of alternative design, gestures are list on Table 6.1. We start the user study until we make sure they can interact with Apple Watch well and also remembered these 13 gestures.

Table 6.1: Gestures Index

Index	Description
1	Thumb swipe on index finger
2	Thumb swipe on middle finger
3	Thumb swipe on ring finger
4	Thumb swipe on little finger
5	Thumb pinch with index finger
6	Thumb force pinch with index finger
7	Thumb pinch with middle finger
8	Thumb force pinch with middle finger
9	Thumb pinch with ring finger
10	Thumb force pinch with ring finger
11	Thumb pinch with little finger
12	Thumb force pinch with little finger
13	Five fingers grab

After above three tasks, we show the first part of questionnaire to participants, an requires participants recall previous learning of gesture and related tasks to complete this part of questionnaire. Then we goes to next stage of user tasks for alternative interaction:

- **Alternative Task 1**: raise hand, call Friends view (grab), choose contact (pinch), make phone call (pinch 3 times);

- **Alternative Task 2**: raise hand, back to App homepage(grab 2 times), select phone App and make the phone call (pinch 3 times);

- **Alternative Task 3**: raise hand, Force Touch (force pinch, long time), swipe to left or right (thumb and middle finger pinch), tap to end selection (pinch).

Alter these alternative interaction task, we show the first part of survey again, requires participants modify their answer until the answer satisfy their own. When this finished, we show the second part of questionnaire, see Appendix C, then we logout this user study.

6.2 Participants

Sample size of participants in a usability test sometimes do not have a fixed program, the classic theory is that usability test for serious usability problems can only take five participants [45]. However, [46] analysed increase sample size to 20 can find more than 95% usability problems.

Hwang et al. [47] discussed the changing rule of sample size in usability test. R. Macefield et al. [48] gives the influence of sample size in each study group, they point out that 5 to 10 is the best base number in each study group. [49] suggested use sample ratio to evaluate sample size when the number of evaluate task is inequal to user tasks per user.

In [50], they discussed how eye tracking influence the usability test. The above discussion of its literature, we conclude that given an algorithm 1 as usability testing sample size calculation method. Wherein calculating the number of samples is determined by the following parameters:

1. **group**: number of user group(such as expert or freshman), this number should not large than 5;

2. **task**: Is the user task same between different group?

3. **baseline**: Will user study result influence product decision or not?

4. **design**: evaluating design numbers in a user study, this number should not large than 5;

5. **peruser**: evaluating design number for each user, each user should not evaluate more than 5 design;

6. **eyetracking**: Is this user study enable eye tracking? If so, is it qualitatively or quantitatively?

Then we have the algorithm of participants sample size in usability test:

Algorithm 1 Calculate minimum participant sample size

Require: $group, task, design, peruser, eyetracking$
Ensure: $sample$
  $base \leftarrow 10$
  $sample \leftarrow 10$
  if eyetracking is qualitatively **then**
    $base \leftarrow 12$
  else if eyetracking is quantitatively **then**
    $base \leftarrow 39$
  end if
  if baseline is yes **then**
    $base \leftarrow 16$
  end if
  if $design \leq peruser$ **then**
    $ratio \leftarrow 1$
  else
    $ratio \leftarrow \lfloor \frac{\text{design}}{\text{peruser}} \rfloor$
  end if
  if task is same **then**
    $sample = \lceil (base + 6 \cdot (group - 1)) \cdot ratio \rceil$
  else
    $sample = \lceil base \cdot group \cdot ratio \rceil$
  end if

In our user study, we randomly invite participants perform same tasks, and the results influences future design, so the parameters are group $= 1$, baseline $=$ yes, task $=$ same, design $= 5$, peruser $= 5$, eyetracking $=$ no. According to algorithm 1, the sample size should not smaller than 16. We put this user study information in social networks and also randomly invite people who get in or out from library become a participant, and finally we have 21 participants.

6.3 Analysis and Results

6.3.1 Statistics of Participants

Out of 21 participants, 11 were male and 10 were female. The range of age was 18 to 28. Besides, there are three participants have smart watch use experiences.

Participants who have use watch in daily life, all own more than two watches, and there are 9 participants consider to buy a new smart watch. In watch users, only 6 participants plan to buy a smart watch and most of them regard its prices.

We have 26.57% (6 people) participants neither do not have a watch or do not consider to buy it. In another words, they will never buy a smart watch if smart watches is not a necessary product.

6.3.2 Social Acceptable

In the first part of questionnaire, we set up a social acceptability problem, it is "how to treat strangers raise hand excuse gestures in Table 6.1?". In these participants, there are 42.85% (8 people) participants thought it was exercise, 14.28% (3 people) thought it was thinking, and 33.33% (7 people) thought it was a performance art, 19.04% (4 people) didn't care about it and 4.76% (1 person) considered "insane".

From the above results, all participants accept this action except 4.76% (1 person).

6.3.3 Comfortable Rate of Gestures

Every participant makes a gestures comfort voting, gestures is in Table 6.1, results shown on Figure 6.1.

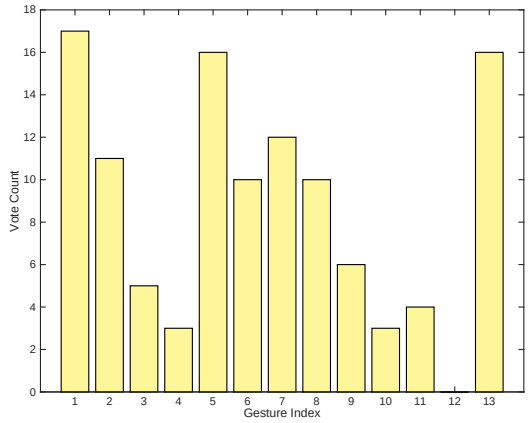


Figure 6.1: Vote results of gestures in Table 6.1

The results showed that more than 16 people (76.19%) vote three gestures, they are swipe between thumb and index finger, pinch with thumb and index finger and five finger grab, they are in the first echelon. Thus these three single hand gestures has comfortable and easy characteristics for users.

Next, the second echelon, more than 11 people (52.38%) vote to thumb swipe on middle finger, long pinch with thumb and index finger. It is indicative that these gestures' comfort lower than the first echelon gesture.

It is worth to mention that in all 21 participants, no one choose thumb pinch with little finger, which means this gesture is the most uncomfortable gesture among them. In contrast, we can use this gesture to operate some dangerous operation, such as delete.

6.3.4 Gesture Intuition

We are also studied user's gesture intuition, to verify the reasonableness of the alternative design in the first part of questionnaire. Voting results as shown in Figure 6.2, in which the red color indicates more deeper gets gesture intuition for user more higher.

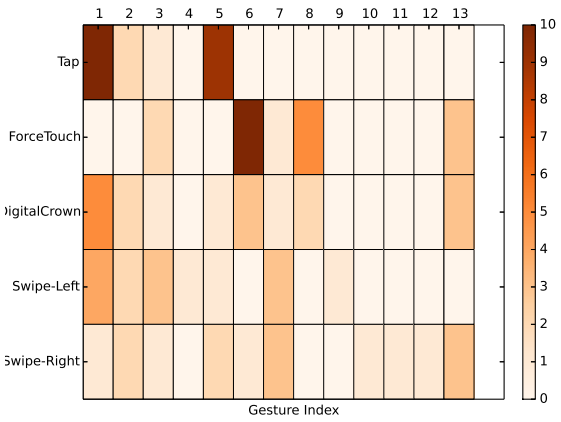


Figure 6.2: Heatmap of intuition vote results

According to the results, participants thought the tap and force touch gestures is a strong intuitive operation. In tap gesture, participants has two obvious point of view, some may say thumb swipe on index finger is more intuition, and some may say quick pinch is more logical.

Especially for Force Touch, the user does not exist distinct operation divisions. participants were generally more inclined to use long pinch. Digital Crown's gesture were more modest, but it can be observed that user tend to use thumb swipe on index finger.

However, for swipe gesture, participant becomes confused, there are no obvious results shows. So we still need to considering this part of gestures carefully, and sometimes we may need user to learn these gestures.

6.4 Conclusions

We can conclude, through this user study, the alternative design of tap, Force Touch and Digital Crown is more comfortable and intuitive for user's operation. As it happens, this is the most frequency operation for smart watch interaction.

For others, there is no obvious logic and intuition for interaction. Fortunately, the Apple Watch display content vertically so swipe left and right are not frequency, back to previous view belongs to a sensitive operation, we could use these gesture to improve user vigilance and pay more attention on it.

In consequences, our alternative interaction design is appropriate for most normal users.

7 Future Works

7.1 Weakness

We have discussed few unbreakable watchOS and LeapMotion software-related problem, due to the limits of performance and energy on watchOS, the communication of interaction message processing is the most important part of coding. The next contents we focus on problems of hardware and the interaction design itself.

7.1.1 Hardware Related

As a prototype, in this thesis we use LeapMotion to process and recognize hand gesture, unfortunately, LeapMotion required a USB connected desktop computer, which is totally restricted the scope of interact scenes, it cann't used in other scenes.

In [51], Gilles et al. put a deep camera on the top of shoes and create a area above it. Likewise, we can design a mobile LeapMotion solution, as illustrate on Figure 7.1.

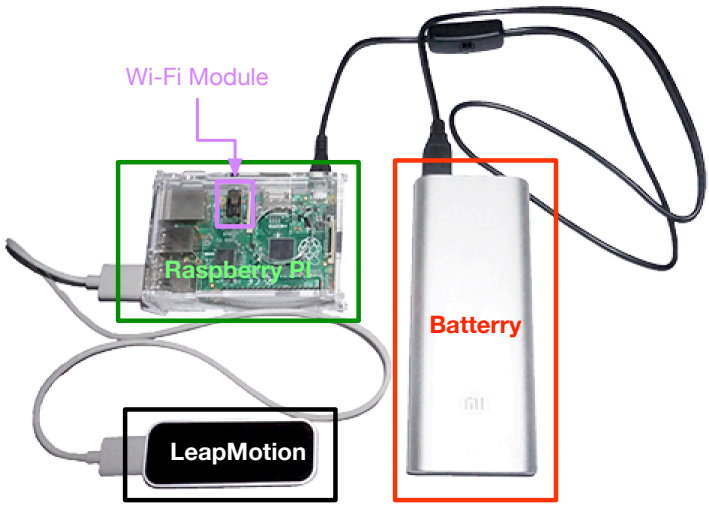


Figure 7.1: **Hardware Structure**: The hardware contains a 5V-2A battery, a Raspberry Pi B+ with WiFi Module, a LeapMotion Device.

In this scenario, the mobile charger supply output 5V-2A for driving Raspberry Pi; Raspberry Pi connect server for interaction message communication by WiFi-Module hardware; then LeapMotion connect to Raspberry Pi by USB. Until now, the design could be completely mobile.

Unfortunately, LeapMotion's host OS requires at least 2GB RAM and the SKD only supported on Ubuntu Linux, Mac OS X and desktop Windows. Currently the second-generation Raspberry Pi B+ only have 512M RAM. But we expect to see the future of Windows 10 could running on raspberry PI when RAM expand to 2GB.

7.1.2 Interaction Related

Alternative design presented in this thesis although completely archived original interaction design and also user study results reflect it is advisable, but once the user tasks become an complexity task, the efficiency of alternative design will poor than original. Considering Keystroke-Level Model [52], we present a Watch-KLM, it's basic task step shown in Table 7.1.

Table 7.1: User task basic steps

Value	Description
K	Tap
F	Force Touch
S	Swipe
T	Target Select
R	System Response
M	User Thinking
H	Highest Priority Operate

For a normal tap gesture, we assume that in these two design the execute time of pinch gesture are same. Note that in alternative design tap an object need to select an specific object, thus $K_{\text{contact-free}} > K_{\text{contact}}$; For Force Touch, alternative interaction can directly measure user's press value, however in section 3.1.3 we gives two stage of Force Touch simulation longer than original design, that cause $F_{\text{contact-free}} > F_{\text{contact}}$.

For T、M and H, if user has already familiar with these two design, we may consider that time of execute exactly the same. For T, alternative design need iterate all objects in a view, apparently we have $T_{\text{contact-free}} > T_{\text{contact}}$. In addition, alternative design spend time on communications, so $R_{\text{contact-free}} > R_{\text{contact}}$.

Thus, contact interaction and contact-free interaction can only be a linear combination of these seven basic steps. For a same task,the coefficients are same. Based on above analysis, it is clearly that we have the formula 7.1.

$$\Sigma_{K,F,S,T,R,M,H}\text{Contact Task} < \Sigma_{K,F,S,T,R,M,H}\text{Contact-free Task} \tag{7.1}$$

Left part in formula 7.1 will significantly less than the right part when user tasks become much complicated.

7.2 Improvements

At last, we discuss how to improve the future design of the system architecture in this application, as well as over all system scalability.

7.2.1 Recognition Approach

We use LeapMotion as a gesture recognize solution for alternative interaction, but in reality LeapMotion is a desktop interaction hardware, a strong dependence vision method based solution is not quite suitable for a portable and wearable devices. Here we discuss another hardware for detection.

Myo [53] is a fixed machine on human arm. It detect the electronic signal of muscle, see Figure 7.2[1]. Myo use built-in WiFi module to deliver the signal. Myo also opened its recognize protocol enables developer can specify its gesture system.

[1]Image source: https://www.myo.com

Figure 7.2: **Myo Device**: Myo install on user's arm, it will detect the electronic signal of muscle.

The morphological of Myo is completely suitable for ergonomically design, this because of the band of watch is the best form of existence for Myo. When Myo can be reduced and designed into watch strap, the recognition will no longer dependent on individual external devices, it is more efficient to complete interaction and data processing.

7.2.2 IPC

In section 4.2.1, we discussed the communications architecture's design, this architecture can also be applied in general cases.

Figure 7.3 is the expand from the previous design, we introduce a Interaction Perception Center(IPC, which is a cross platform interaction communication solution for desktop and mobile devices we presented) service [54]. We put IPC as a system level model for interaction, its Sensor Handle Protocol (SH Protocol) can dismiss, formalize hardware differences, and Cross-Device Interaction Protocol (CDI Protocol) enables user devices communicate with IPC service.

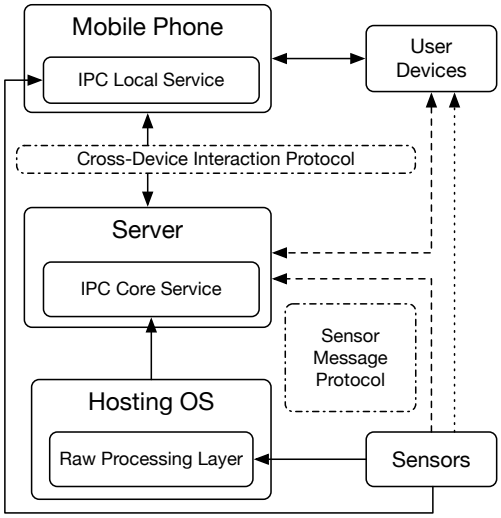


Figure 7.3: **Universal communication architecture**: Introducing IPC service on server and client side.

Different sensors with different diver, but if there exist a SH protocol and the data can be processed by host OS, then OS will send these message to sever together for a second process, server side distribute the interaction message to any other devices through CDI protocol; Besides, if IPC core service is not available, the sensor host OS can still communicate with client for some simple interaction offline.

IPC was designed as a pure software solution in the beginning, it's purpose to give a distributed data communication and analysis platform between mobile devices, desktop computers and sensors,

now we conclude few implicit from above discuss: Firstly, design of universal interaction system should not transport to IPC server directly, data from sensors should machining on its host OS at first, then the process data should be packaged through a message protocol to guarantee all sensor data can access IPC interface; Secondly, IPC interaction message must be optional for users, these interaction in any cases should provides a primary interaction at least. The scilicet marks passive interaction should be implicit and alternative.

7.3 Closing

In this thesis, we combine with the Haptic feedback, convert the basic tap, swipe, Digital Crown, Force Touch and etc. native watch interaction to a contact-free gesture interaction. It simplified ten native interaction with two-step logic to only eight alternative interaction with single-step logic, eliminated contact-required interaction method, solved the bimanual interaction within smart watches, and explained the completeness of this alternative design.

With this work, we overcome the hardware platform and software interface limitations, developed an architecture in the progress and its us a set of implicit for build an interactive system.

Through user study, we evaluate the whole alternative design and gesture comfortable rate and intuitive rate. These alternative designs and the result shows the interaction operate logic and gesture comfortable all acceptable.

Overall, the innovative human-computer interaction is often accompanied by a combination of hardware and software, in the process of software layer sometimes subject various limitations of hardware platform. A set of open architecture design of hardware and software not only from hardware to software vertical integration, but also the need to do a horizontal products usability design, even considerate on users. These contents float above the water surface, it's only the tip of the iceberg.

Appendix A Licence

The program, paper and etc. in this thesis are open source, they can be found in:

- Thesis homepage: https://changkun.us/bechelorthesis/;
- GitHub repostory: https://github.com/changkun/BachelorThesis/.

In case, all related text, picture and video content are licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License[1].

The other part in this thesis (program source code) is licensed under a GNU Public License Version 3.0.

[1] http://creativecommons.org/licenses/by-nc-sa/4.0/

Appendix B Questionnaire - Part A

Welcome! :)

Thank you for join our study! This questionnaire is the first part of out research, we need you to fill this simple questionnaire. **Quesions marked by * is required**

1. **Please select the most comfortable options *** At least five options
   - Thumb swipe on index finger
   - Thumb swipe on middle finger
   - Thumb swipe on ring finger
   - Thumb swipe on little finger
   - Thumb pinch with index finger
   - Thumb force pinch with index finger
   - Thumb pinch with middle finger
   - Thumb force pinch with middle finger
   - Thumb pinch with ring finger
   - Thumb force pinch with ring finger
   - Thumb pinch with little finger
   - Thumb force pinch with little finger
   - Five fingers grab
2. **What would you think about strangers raising hand make some gesture which are showed on question 1?**
   Please write your idea
3. **Please connect the left side and right side operate options if you think they are suitable ***
   Options on the left side can only connect to only one option on the right side, you can pass a connection if they are not suitable.

	Thumb swipe on index finger
	Thumb swipe on middle finger
	Thumb swipe on ring finger
	Thumb swipe on little finger
Tap gesture	Thumb pinch with index finger
Finger force touch a screen	Thumb force pinch with index finger
Tune knob	Thumb pinch with middle finger
Finger swipe left on a plane	Thumb force pinch with middle finger
Finger swipe right on a plane	Thumb pinch with ring finger
	Thumb force pinch with ring finger
	Thumb pinch with little finger
	Thumb force pinch with little finger
	Five fingers grab

Appendix C Questionnaire - Part B

Hi!

Thanks for join our study, again! This questionnaire is the last part of this study, we still need you to fill the following simple questions. **Quesions marked by * is required**

1. **Your ID***
   Please write what your input here:
2. **Age?**
   Please write what your input here:
3. **Gender?***
   Please choose only one of the following:
   - Male
   - Female
4. **Frequency of wearing watches?***
   Please use "Never", "Rarely", "Sometimes", "Often", "Always" to answer these questions:
   - Frequency of wear it on left hand?
   - Frequency of wear it on right hand?
   - Frequency without wear a watch?
   - Frequency of forget wear it?
5. **Do you have use experiences of smart watches?***
   Please choose only one of the following:
   - Yes
   - No
6. **Which Model?***
   If you choose Yes in question 5, please write this question's answer:
7. **How many watches you own?***
   Including traditional watch and smart watch, count them even they have more than one band, write your answer here:
8. **Do you still using watch in your daily life?***
   Please choose only one of the following:
   - Yes
   - No
9. **For what reason?***
   Please write what your input here:
10. **Do you considering to buy a smart watch?***
    Please choose only one of the following:

- Yes
- No

11. **For what reason?***
    Please write what your input here:

12. **What is the most significant option of using smart watch for you?***
    Please choose only one of the following:

    - Price
    - Battery
    - Habit
    - Others, please write it down

13. **If you have any comments for this study, we glad to here your voice. Please write your comments here**

Bibliography

[1] Jiang H, Chen X, Zhang S, et al. Software for Wearable Devices: Challenges and Opportunities [J/OL]. CoRR. 2015, abs/1504.00747. http://arxiv.org/abs/1504.00747.

[2] 董士海. 人机交互的进展及面临的挑战 [J]. 计算机辅助设计与图形学学报. 2004, 16 (1): 1–13.

[3] 岳玮宁, 董士海, 王悦, et al. 普适计算的人机交互框架研究 [J]. 计算机学报. 2004, 27 (12): 1657–1664.

[4] Hudson S E, Mankoff J. Concepts, Values, and Methods for Technical Human–Computer Interaction Research [M] // Judith S Olson W A K. Ways of Knowing in HCI. Springer, 2014: 2014: 69–93.

[5] 刘珩. 智能手表交互设计研究 [J]. 科技与创新. 2015 (8): 73–73.

[6] 程时伟. 基于上下文感知的移动设备自适应用户界面设计研究 [D]. [S. l.]: 浙江大学, 2009.

[7] 扶爱名. 基于可穿戴计算的设备维护辅助系统自适应用户界面研究 [D]. [S. l.]: 电子科技大学, 2006.

[8] Apple Inc. Apple Watch Human Interface Guidelines [EB/OL]. [2016-03-17]. https://developer.apple.com/watch/human-interface-guidelines/.

[9] Wobbrock J O, Wilson A D, Li Y. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes [C]. In Proceedings of the 20th annual ACM symposium on User interface software and technology. 2007: 159–168.

[10] Anthony L, Wobbrock J O. A lightweight multistroke recognizer for user interface prototypes [C]. In Proceedings of Graphics Interface 2010. 2010: 245–252.

[11] 李清水, 方志刚, 沈模卫, et al. 手势识别技术及其在人机交互中的应用 [J]. 人类工效学. 2002, 8 (1): 27–29.

[12] 陈雅茜, 欧长坤, 郭嬰阳. 基于单目视觉和简单手势的空间交互技术 [J]. 西南民族大学学报: 自然科学版. 2014, 40 (6): 871–876.

[13] 狄海进. 基于三维视觉的手势跟踪及人机交互中的应用 [D]. [S. l.]: 南京大学, 2011.

[14] 侯文君, 吴春京. 基于数据分析的智能手表手势直觉化交互研究 [J]. 包装工程. 2015, 36 (22): 13–16.

[15] Vatavu R-D, Zaiti I-A. Leap Gestures for TV: Insights from an Elicitation Study [C/OL]. In Proceedings of the 2014 ACM International Conference on Interactive Experiences for TV and Online Video. New York, NY, USA, 2014: 131–138. http://doi.acm.org/10.1145/2602299.2602316.

[16] Loclair C, Gustafson S, Baudisch P. PinchWatch: a wearable device for one-handed microinteractions [C]. In MobileHCI Workshop on Ensembles of On-Body Devices. 2010.

[17] Lv Z, Feng S, Feng L, et al. Extending touch-less interaction on vision based wearable device [C]. In Virtual Reality (VR), 2015 iEEE. 2015: 231–232.

[18] Kerber F, Schardt P, Löchtefeld M. WristRotate: A Personalized Motion Gesture Delimiter for Wrist-worn Devices [C/OL]. In Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia. New York, NY, USA, 2015: 218–222. http://doi.acm.org/10.1145/2836041.2836063.

[19] Knibbe J, Martinez Plasencia D, Bainbridge C, et al. Extending Interaction for Smart Watches: Enabling Bimanual Around Device Control [C/OL]. In CHI '14 Extended Abstracts on Human Factors in Computing Systems. New York, NY, USA, 2014: 1891–1896. http://doi.acm.org/10.1145/2559206.2581315.

[20] Kratz S, Rohs M. Hoverflow: Exploring Around-device Interaction with IR Distance Sensors [C/OL]. In Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services. New York, NY, USA, 2009: 42:1–42:4. http://doi.acm.org/10.1145/1613858.1613912.

[21] Perrault S T, Lecolinet E, Eagan J, et al. Watchit: Simple Gestures and Eyes-free Interaction for Wristwatches and Bracelets [C/OL]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2013: 1451–1460. http://doi.acm.org/10.1145/2470654.2466192.

[22] Ogata M, Imai M. SkinWatch: Skin Gesture Interaction for Smart Watch [C/OL]. In Proceedings of the 6th Augmented Human International Conference. New York, NY, USA, 2015: 21–24. http://doi.acm.org/10.1145/2735711.2735830.

[23] Kim J, He J, Lyons K, et al. The gesture watch: A wireless contact-free gesture based wrist interface [C]. In Wearable Computers, 2007 11th IEEE International Symposium on. 2007: 15–22.

[24] Chen X A, Grossman T, Wigdor D J, et al. Duet: Exploring Joint Interactions on a Smart Phone and a Smart Watch [C/OL]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2014: 159–168. http://doi.acm.org/10.1145/2556288.2556955.

[25] Yang F, Li S, Huang R, et al. MagicWatch: Interacting & Segueing [C/OL]. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication. New York, NY, USA, 2014: 315–318. http://doi.acm.org/10.1145/2638728.2638848.

[26] Bi X, Li Y, Zhai S. FFitts Law: Modeling Finger Touch with Fitts' Law [C/OL]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2013: 1363–1372. http://doi.acm.org/10.1145/2470654.2466180.

[27] Boring S, Ledo D, Chen X A, et al. The fat thumb: using the thumb's contact size for single-handed mobile interaction [M]. New York, New York, USA: ACM, 2012.

[28] Buschek D, Alt F. TouchML: A Machine Learning Toolkit for Modelling Spatial Touch Targeting Behaviour [C]. In IUI '15: Proceedings of the 20th International Conference on Intelligent User Interfaces. March 2015.

[29] Vogel D, Baudisch P. Shift: A Technique for Operating Pen-based Interfaces Using Touch [C/OL]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2007: 657–666. http://doi.acm.org/10.1145/1240624.1240727.

[30] Huxpro. Forcify [EB/OL]. [2016-03-16]. https://github.com/huxpro/forcify.

[31] Changkun Ou. AugmentedTouch: Potential Device Behaviour [EB/OL]. [2016-04-02]. https://github.com/changkun/AugmentedTouch.

[32] Leap Motion Inc. Leap SDK Documentation [EB/OL]. [2016-03-17]. https://developer.leapmotion.com/documentation/index.html.

[33] Weichert F, Bachmann D, Rudak B, et al. Analysis of the Accuracy and Robustness of the Leap Motion Controller [J]. Sensors. 2013, 13: 6380–6393.

[34] Du G, Zhang P, Liu X. Markerless Human-Manipulator Interface Using Leap Motion with Interval Kalman Filter and Improved Particle Filter [J]. IEEE Transactions on Industrial Informatics. 2016, PP (99): 1–1.

[35] Garber L. Gestural technology: Moving interfaces in a new direction [J]. Computer. 2013, 46 (10): 22–25.

[36] 徐崇斌, 周明全, 沈俊辰, et al. 一种基于 Leap Motion 的直观体交互技术 [J]. 电子与信息学报. 2015, 37 (2): 353–359.

[37] 潘佳佳, 徐昆. 基于 Leap Motion 的三维自由手势操作 [J]. 中国科技论文. 2015 (2): 207–212.

[38] 胡弘, 晁建刚, 杨进, et al. Leap Motion 关键点模型手姿态估计方法 [J]. 计算机辅助设计与图形学学报. 2015, 27 (7): 1211–1216.

[39] Marin G, Dominio F, Zanuttigh P. Hand gesture recognition with leap motion and kinect devices [C]. In Image Processing (ICIP), 2014 IEEE International Conference on. 2014: 1565–1569.

[40] Zaiţi I-A, Pentiuc Ş-G, Vatavu R-D. On free-hand TV control: experimental results on user-elicited gestures with Leap Motion [J]. Personal and Ubiquitous Computing. 2015, 19 (5-6): 821–838.

[41] Apple Inc. App Programming Guide for watchOS [EB/OL]. [2016-04-02]. https://developer.apple.com/library/watchos/documentation/General/Conceptual/WatchKitProgrammingGuide/.

[42] Apple Inc. Watch Connectivity Framework Reference [EB/OL]. [2016-04-02]. https://developer.apple.com/library/watchos/documentation/WatchConnectivity/Reference/WatchConnectivity_framework/.

[43] Apple Inc. The Swift Programming Language(Swift 2.2 Prerelease) [M/OL]. Apple Inc., 2016. https://itunes.apple.com/cn/book/swift-programming-language/id1002622538.

[44] Apple Inc. Using Swift with Cocoa and Objective-C(Swift 2.2 Prerelease) [M/OL]. Apple Inc., 2016. https://itunes.apple.com/cn/book/using-swift-cocoa-objective/id1002624212.

[45] Albert W, Tullis T. Measuring the user experience: collecting, analyzing, and presenting usability metrics [M]. Newnes, 2013.

[46] Faulkner L. Beyond the five-user assumption: Benefits of increased sample sizes in usability testing [J/OL]. Behavior Research Methods, Instruments, & Computers: 379–383. http://dx.doi.org/10.3758/BF03195514.

[47] Hwang W, Salvendy G. Number of People Required for Usability Evaluation: The 10±2 Rule [J/OL]. Commun. ACM. 2010, 53 (5): 130–133. http://doi.acm.org/10.1145/1735223.1735255.

[48] Macefield R. How to Specify the Participant Group Size for Usability Studies: A Practitioner's Guide [J/OL]. J. Usability Studies. 2009, 5 (1): 34–45. http://dl.acm.org/citation.cfm?id=2835425.2835429.

[49] Medlock M C, Wixon D, Terrano M, et al. Using the RITE method to improve products: A definition and a case study [J]. Usability Professionals Association. 2002, 51.

[50] Pernice K, Nielsen J. Eyetracking methodology: How to conduct and evaluate usability studies using eyetracking [J]. Nielsen Norman Group Technical Report. 2009.

[51] Bailly G, Müller J, Rohs M, et al. ShoeSense: A New Perspective on Gestural Interaction and Wearable Applications [C/OL]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA, 2012: 1239–1248. http://doi.acm.org/10.1145/2207676.2208576.

[52] Card S K, Moran T P, Newell A. The Keystroke-level Model for User Performance Time with Interactive Systems [J/OL]. Commun. ACM. 1980, 23 (7): 396–410. http://doi.acm.org/10.1145/358886.358895.

[53] Thalmic Labs Inc. Myo Official Site [EB/OL]. [2016-04-05]. https://www.myo.com.

[54] 欧长坤, 郭嬰阳, 石梦鑫, et al. 基于机器学习和云计算的交互感知中心的研究与实现 [R/OL]. 2015. https://changkun.us/ipc/.

[55] Yang Y, Chae S, Shim J, et al. EMG Sensor-based Two-Hand Smart Watch Interaction [C/OL]. In Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology. New York, NY, USA, 2015: 73–74. http://doi.acm.org/10.1145/2815585.2815724.

[56] 傅泉俊. 穿戴式体感交互平台的研究与实现 [D]. [S. l.]: 电子科技大学, 2015.

[57] 朱子健. 情境因素对智能手表使用行为影响研究 [J]. 中国新技术新产品. 2015 (7): 10–12.

Acknowledgments

This thesis has finished under Prof.Yaxi's guidance. She was the most important teacher in my life and she gave me great guidance for everything during my four years of undergraduate learning. With her help, I took part in the American Mathematical Contest in Modeling when I was a freshman, and also applied an entrepreneurship students projects in the second year. Then she supervised me how to write a research papers, gave me a chance to study at University of Munich for exchange semester, and now reviewed my graduate thesis.

Prof.Yaxi and her rigorous, realistic and serious encourage me to keep continue broad my horizon of knowledge, ability of research, even influenced the roads of my future life. These are any other teachers who helped me will never reached. So, first of all, I would like to send the greatest respect and sincere gratitude to my adviser Prof. Yaxi.

Then I would like to thank all my family, especially my parents. I will never archive this without their continuous support and inclusion throughout my entire education, they are my shield of everything. I appreciate my friends, we motivate each other, and pursue a common ideal together, and I appreciate the teachers who helped, cared, guided for me, I have been through an enriching university life because of them.

Lastly, I still need to appreciate my advisor Prof. Andreas Butz, Prof. Heinrich Hußmann and PhD student Daniel Buschek who works at University of Munich. They helped me experienced an enjoyable exchange semester for study abroad. Besides, There is a special thanks for Prof. Butz because he agreed to take over the co-advisor of my thesis and reviewed the english version of my thesis.

With the end of this article, my undergraduate studies will be completed, I will towards to next stage of my entire life with my family, teachers' encouragement and their expectation.

Wish I could keep forever youthful, forever weeping, cheers!