March 2, 2016

Dr. Andrew Rawicz

School of Engineering Science

Simon Fraser University

Burnaby, BC, V5A 1S6

Re: ENSC 440W Function Specification for WizardHand controller system

Dear Dr. Rawicz,

The following document, a design specification for WizardHand controller system, describes our project for ENSC 440. This project will provide an alternative solution to cursor control on the PC screen with mobility and no requirement of a flat surface.

This design specification will exhibit detailed design goals and technologies in order to achieve them. WizardHand development engineers are advised to consult the document for reference to the design specifications and confirm that these requirements are included in the proof of concept product. Our dedicated development team will follow this document as guild lines to complete this brilliant project.

AimBot Technology is established by four talented and enthusiastic engineering students: Alex Chen, Albert Xu, Current Zeng, and Scott Zhu. We are looking forward to solving your questions and concerns about our project via ziqic@sfu.ca.

Sincerely,

Alex Chen

President and CEO

Design Specification for the Wearable Pointing Device

Project Team: Alex Chen

Albert Xu

Current Zeng

Scott Zhu

Contact Person: Alex Chen

ziqic@sfu.ca

Submitted to: Dr. Andrew Rawicz – ENSC 440

Steve Whitmore – ENSC 305

School of Engineering Science

Executive Summary

The design specification for the wearable mouse (WizardHand) provides a set of detailed descriptions of the design and development of the proof-of- concept (POC) model. Therefore, only requirements assigned "A" or P" will pertain to this document, in accordance with the document Functional Specification for WizardHand [1].

This document presents the design plans for the WizardHand and provides practical justification for our design decisions. Design improvements for future iterations of WizardHand will be discussed in this document and will here on out be referred to as stretch goals. Note that these goals will not be implemented in the POC.

The WizardHand is physically implemented via the use of a glove, containing one potentiometer for each finger in order to catch the motion of fingers for users. The control unit attached to wrist is connecting with the glove. The control module is consists of CPU, Bluetooth module, IMU, battery and most of the important components. All the digital data can be sent via the control unit, we can also capture gestures of users through it.

For the part of the software, the signals with user's motions will be gathered through the potentiometers from glove and IMU from the control unit. After gathering the signals, the software in the CPU will employ an algorithm that analyzes all the dynamical motions and converts these motions to a digital signal that controls the motion of mouse on the screen. Then the signal will be sent to PCs or smartphones through Bluetooth module.

In terms of resource requirements, the justification for Electronics specifications of the WizardHand will be provided. Software used to implement the algorithm will also be discussed and the high-level architecture for it will be shown.

AimBot deems their design process to be on schedule in accordance with the plans outlined in the Functional Specification for the WizardHand.

Table of Contents

List of Figures

List of Tables

Glossary

Measurement Unit	The unit to measure the desired data for our design.
Soft Potentiometer	An instrument for measuring the voltage in a circuit
IMU	Inertia Measurement Unit which usually include Accelerometer, Gyroscope, and Magnetometer.
DOF	The degree of Freedom. In our case, each degree of freedom indicates data from one axis can be measured.
Gyroscope	Gyroscopes are used to measure angular velocity. An uncalibrated gyroscope may have bias errors which are angular velocity readings when it is not experiencing a rotation. If an uncalibrated gyroscope is used to estimate the angle, by integrating angular velocity over time, drift will present. The output of the gyroscope is relatively stable compared with the other two components. However, when doing integration over time, data lost will present if the sampling frequency is not quick enough. Therefore, we can only rely on gyroscope for short term applications.
Accelerometer	Accelerometers are used to measure the magnitude and direction of acceleration. When the sensor is not experiencing any accelerative motion, it should still pick up the reading of 1g (~9.8m/s^2). Compared to gyroscopes, accelerometers are less stable. To have a more stable output, a **low-pass filter** can be introduced. There will not be any drift caused by accelerometer if integration is implemented. Therefore, we can only rely on accelerometer for long term applications which do not require high accuracy measurement.
Magnetometer	Magnetometers are used to measure strength and direction of magnetic fields. If the sensor is not calibrated, we will have this situation: when we use x, y and z-axis of the magnetometer to measure magnetic field strength; we clearly know that the measurement should be the same; but if the sensor is not calibrated, the reading from x, y and z-axis may be different. To have more stable output, a **low-pass filter** can be introduced. Therefore, we can only rely on magnetometer for long term applications which do not require high accuracy measurement.

Bias of Gyroscope	The bias of a rate gyro is the average output when the device is sitting still.
Arduino Pro-mini	Is designed for e-textiles and wearable projects.
Bluetooth Module	Support for using Bluetooth communication.
Visual Studio	An integrated development environment includes a code editor supporting IntelliSense as well as code refactoring.
Device Driver	A program that controls a particular type of device that is attached to computer
Kalman Filter	An algorithm that uses a series of measurements observed over time produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone
Quaternion	In mathematics, the quaternions are a number system that extends the complex numbers. It is applied to mechanics in three-dimensional space.
Euler Angle	The Euler angles are three angles introduced by Leonhard Euler to describe the orientation of a rigid body. To describe such an orientation in 3-dimensional Euclidean space three parameters are required.
Metacarpophalangeal joint	The third joint of your finger starting from the fingertip
Proximal interphalangeal joint	The second joint of your finger starting from the fingertip
Solder breadboard	A thin breadboard that connect everything by soldering
TR/RX	Transfer and receive data port
PCB Board	Mechanically supports and electrically connects electronic components

1. Introduction

The WizardHand Controller System is a wearable pointing device, which designed to help people achieving cursor control in a more elegant and efficient way while giving a presentation or playing video games. The conceptual graph is shown in Figure 1 which is on the next page.

The Device is separated into three parts: Electronic, software and kit design. The kit is a combination of a 2-layers glove and a controller unit, our users can wear the kit as a normal glove with a wristband. Potentiometers and IMU contain within the Electronics will read the data and transmit them into the Software, which will analyze the data and eventually sent them to PCs or smartphones.

1.1. Scope

This document specifies the design of the WizardHand and coordinates with the functional requirements described in Functional Specification for WizardHand [1]. Included are all requirements for the POC system as well as a relevant set of requirements for future versions of the WizardHand. Functional requirements designated either priority A or P applies to the POC and will be discussed in detail.

1.2. Intended Audience

The design specification is to be used by who is designing and developing the product. Engineers are advised to consult the document for reference to the design specifications and confirm that these requirements are included in the POC product. Testers are advised to refer to the document to ensure that the WizardHands is functioning in accordance with the design.

2. Overall System Design

Figure 1 below shows the high-level overview of WizardHand. The system takes the data from measurement units, and output 2-D coordinate of cursor position and hand gestures code in five digit binary code. The measurement units include five soft potentiometers and one IMU. The five Soft Potentiometers are used to detect the status of user's fingers. For example, if the user only bends his/her ring finger, then the measurement is 00010b. We implemented MPU9150 as IMU in our design. MPU9150 has three smaller measurement units, which are Accelerometer, Gyroscope, and Magnetometer, embedded on board [2]. Each of this smaller components has 3-DOF. In conclusion, we have nine measurements from IMU and five measurements from Soft Potentiometers. These fourteen measurements will be collected by Arduino Pro mini and sent to PC's end by Bluetooth module. In PC's end, Visual Studio is used to write the driver for our device. In this device driver, nine raw data of IMU will be processed and filtered by Kalman Filter then be output as the 2-D coordinate of cursor position, and five digits binary measurements from Soft Potentiometers will be output directly.

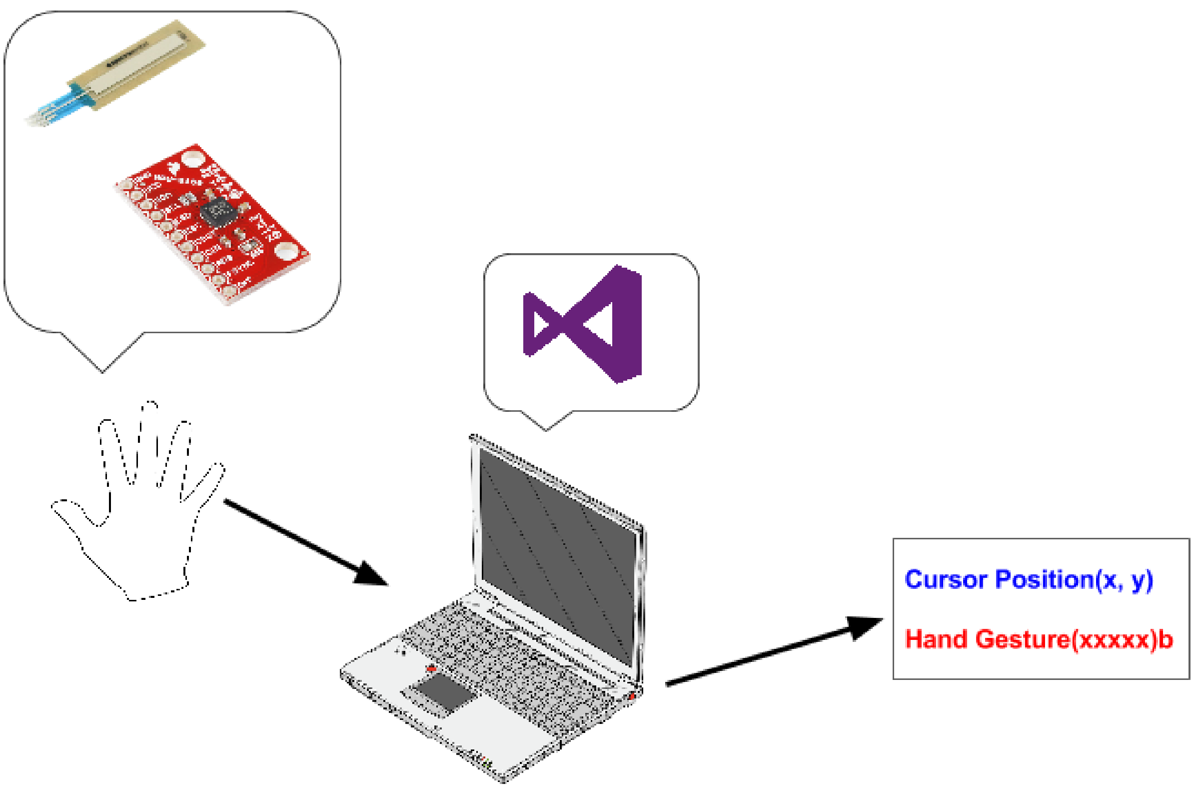


Figure 1 - System Specification

3. System Specification

In this section will focus on introducing how signals are being processed and what algorithms are implemented. Most mathematic and programming concepts will be mentioned here.

3.1. General Signal Process

Figure 2 below shows a general overview of how the entire system is planned in WizardHand design. This flow chart provides the reference of the process that is involved in each design area. Furthermore, it shows all components that are involved in our design. Full detail of computing process will be provided in next section.

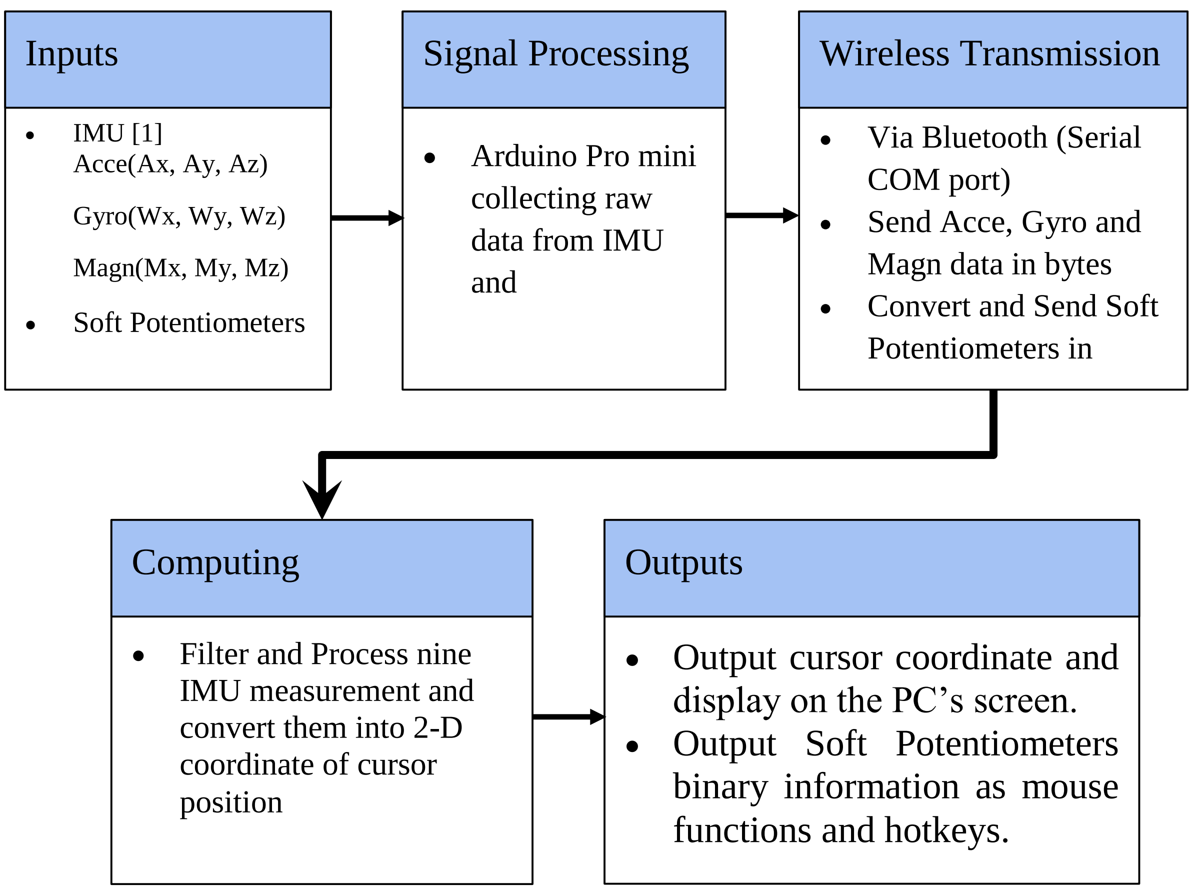


Figure 2 - General Signal Processing

3.2. Introduction to Quaternion

The quaternions are a number system that extends the complex numbers. In our case, we implement quaternion to estimate and predict the orientation of our device. There are two ways to express quaternions and they are $\hat{q} = [w \quad x \quad y \quad z]$ and $\hat{q} = [x \quad y \quad z \quad w]$. In this two vectors, *w* is the real part and *x, y, z* are imaginary part.

Quaternions have many algorithm and properties, the following context will list the useful algorithm and properties that involved in our design.

3.2.1. Quaternion Conjugation [3] [4]

$$\text{If } \hat{q} = [w \quad x \quad y \quad z],\ \hat{q}^* = [w \quad -x \quad -y \quad -z]$$

3.2.2. Quaternion Product [3] [4]

$$\text{If } a = [a1 \quad a2 \quad a3 \quad a4] \text{ and } b = [b1 \quad b2 \quad b3 \quad b4]$$

$$a \otimes b = \begin{bmatrix} a1b1 - a2b2 - a3b3 - a4b4 \\ a1b2 + a2b1 + a3b4 + a4b3 \\ a1b3 - a2b4 + a3b1 + a4b2 \\ a1b4 + a2b5 - a3b2 + a4b1 \end{bmatrix}$$

Note: $a \otimes b \neq b \otimes a$

3.2.3. Quaternion Rotation [3] [4]

If ${}^{A}\vec{V} = [0 \quad x \quad y \quad z]^T$ is a vector in frame A and

and ${}^{B}\vec{V}$ is a vector that is rotated from frame A and it in frame B

Suppose ${}^{B}_{A}\hat{q}$ is the quaternion that make this rotation.

$$\text{We have } {}^{B}\vec{V} = {}^{B}_{A}\hat{q} \otimes {}^{A}\vec{V} \otimes {}^{B}_{A}\hat{q}^*$$

3.2.4. Double Quaternion [3] [4]

$$\text{If } {}^{B}\vec{V} = {}^{B}_{A}\hat{q} \otimes {}^{A}\vec{V} \otimes {}^{B}_{A}\hat{q}^* \text{ and } {}^{C}\vec{V} = {}^{C}_{B}\hat{q} \otimes {}^{B}\vec{V} \otimes {}^{C}_{B}\hat{q}^*,$$

there exist a quaternion vector ${}^{C}_{A}\hat{q}$ that ${}^{C}\vec{V} = {}^{C}_{A}\hat{q} \otimes {}^{A}\vec{V} \otimes {}^{C}_{A}\hat{q}^*$ and ${}^{C}_{A}\hat{q} = {}^{B}_{A}\hat{q} \otimes {}^{C}_{B}\hat{q}$

3.3. Computing Process

The most natural solution we came up with is to imagine that user is holding a ruler, or the glove that user is wearing has a laser that is pointing to the screen when he/she is using our device.

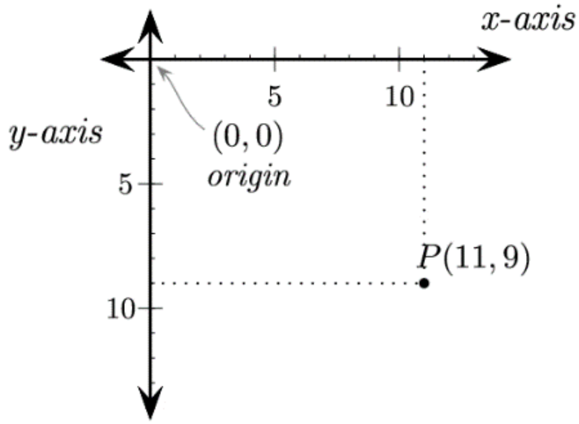


Figure 3 - Screen Coordinate System

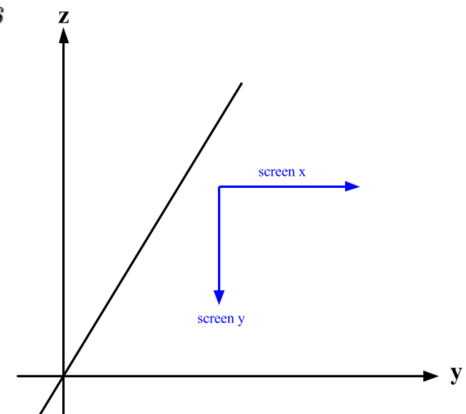


Figure 4 - Normal 3-D Coordinate VS Screen Coordinate

Note that the coordinate system of the screen is shown in Figure 3 is different from what we normally us. Y-axis is pointing oppositely comparing with the coordinate system that we learn from a math textbook. In addition, to reduce the data conversion and transformation, we use the standard 3-D coordinate system (x-axis is pointing to the viewer and z-axis is pointing upward) when we are sending data to serial COM port. In this case, as shown in Figure 4, we are using y-axis of the standard 3-D coordinate as the x-axis in the screen and using the negative z-axis of standard 3-D coordinate as the y-axis in the screen.

To achieve this coordinate system, the following steps are required.

3.3.1. Setting up

Set a constant vector $\overrightarrow{wand} = [0 \quad -length \quad 0 \quad 0]^T$ where length is the scaling distance between our device and screen. Assume that once we have obtained a wand, the length of it will never change, so if we want to point something further, we need to use another longer wand.

3.3.2. Perform quaternion rotation [3] [4]

$$\overrightarrow{wand}_{update} = {}^B_A\hat{q} \otimes \overrightarrow{wand}_{constant} \otimes {}^B_A\hat{q}^*$$

$$\overrightarrow{wand}_{update} = [w_{wand} \quad x_{wand} \quad y_{wand} \quad z_{wand}]^T$$

The magnitude of $\overrightarrow{wand}_{update}$ should always be $length$ as we mentioned in the first step.

3.3.3. Scaling [3] [4]

Since the initial condition is $\overrightarrow{wand}$ pointing along x axis negatively in standard coordinate, the coordinate of y_{wand} and z_{wand} are zero at the beginning. However, the center of the screen is not

$$(0, 0)$$

but

$$(half\ screen\ horizontal\ resolution, half\ screen\ vertical\ resolution).$$

So we need to scale (y_{wand}, z_{wand}) to (x_{screen}, y_{screen}). The scaling method we use is:

$$(x_{screen}, y_{screen}) =$$

$$(\#horizon\ pixle \times (\left(\frac{y_{wand}}{length}\right) + 0.5), \#vertical\ pixle \times (\left(-\frac{z_{wand}}{length}\right) + 0.5))$$

3.3.4. Quaternion Calculating [3] [4]

Before we get into the programming, we need to be aware that we rely on quaternions to achieve our design. To calculate quaternion, based on the specifications and characteristics of the IMU components that mentioned in the glossary, the quaternion

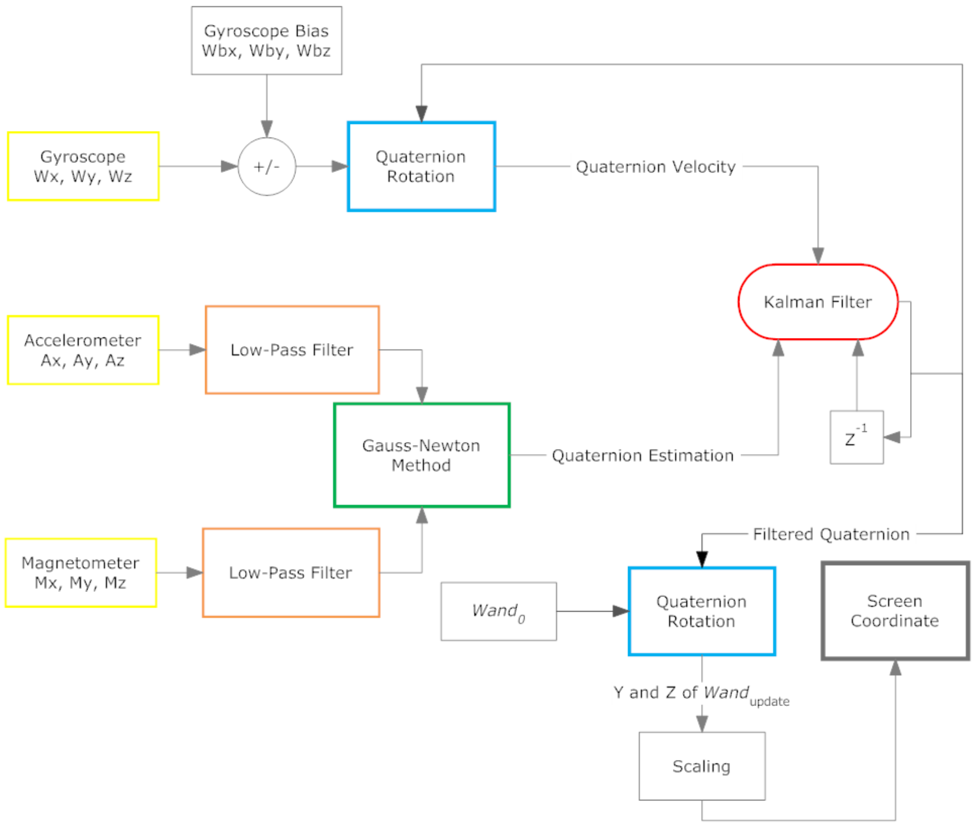


Figure 5 - Quaternion Calculating and Noise Filtering

calculating and noise filtering algorithm can be described as the flow chart is shown in Figure 5.

4. Kit and Hardware Design

Figure 6 - Overall Design of the WizardHand

Figure 5 shows the overall design of the WizardHand system. It consists two major parts. The glove and the controller unit. Two parts will be connected together through the strong fabric for signal passing and wear easiness.

4.1.The glove and potentiometer

The glove is double layered and made of cotton cloth. It is with great elasticity to fit general male and female hand size: length ranges from 172mm to 189mm, width range from 74mm to 84mm [5]. The fabric is strong enough to hold the stainless steel conductive threads [6], as shown in Figure 7, that are used to provide power and pass the signal from potentiometers which lying between layers.

Figure 7 - Stainless Steel Conductive Threads

With the use of conductive thread, this system successfully avoid the break-off problem that might happen to the ordinary electric wire. Since the glove and potentiometer system is constantly under bending and stretching situations, the conductive thread that can be stitched into glove fabric material provides good durability and stability as shown in the following figure.

Figure 8 - Sample Application of Conductive Thread

Potentiometers are placed between two layers of glove fabric. Each finger has its own assigned potentiometer. To accurately measure the bending of finger joints [7], as well as to avoid extreme bending of the potentiometer, they are positioned to cover metacarpophalangeal joint, and not to reach proximal interphalangeal joint. As shown in Figure 9. Signals from potentiometers will finally be passed into the controller unit.

Figure 9 - Potentiometers' Placement

Figure 10 - Soft Potentiometers

4.2.Control Unit

The controller unit consists essential components such as Arduino Pro Mini, HC-06 Bluetooth module, IMU, battery, and the charging circuit. To keep the minimum volume, the whole circuit is built on a thin solder breadboard and wrapped with strong electronic tape. The packaging leaves the connection to potentiometers, and a micro-USB charging port.

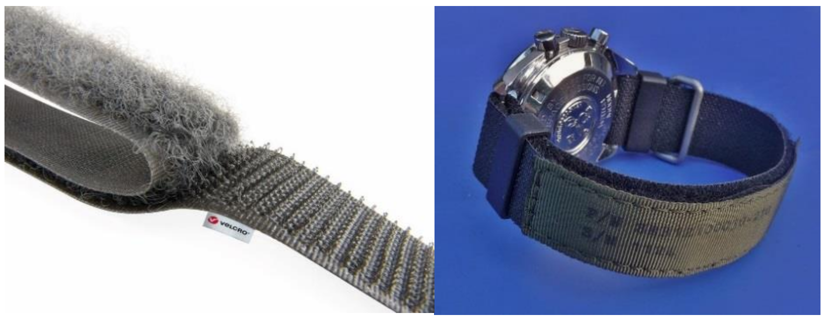


Figure 11 – VELCRO and Sample Application on Watch

To stabilize the controller unit, a VELCRO® [8] loop tape is used, as shown in Figure 11. This design makes it possible to fit all size of wrists. And it is convenient to uninstall while providing reliable fixation. The final stabilization system is shown in the Figure below.

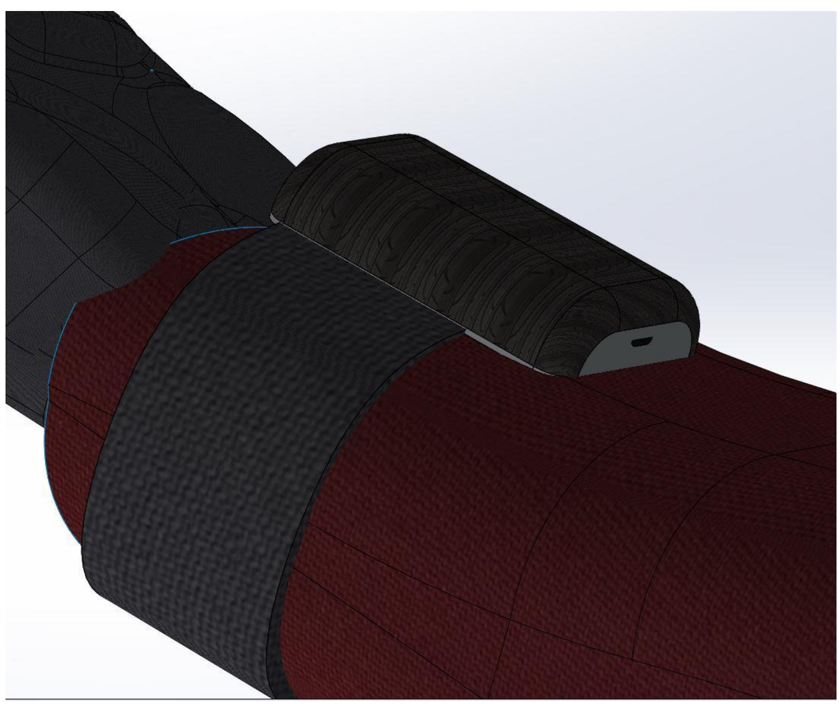


Figure 12 - Stabilization System of the Control Unit

5. Electronics Design

The electronic of the WizardHand consists the following electrical hardware:

1. Arduino Pro Mini 328 - 5V/16MHz [9]
2. SparkFun 9 Degrees of Freedom Breakout - MPU-9150 [2]
3. SoftPot Membrane Potentiometer - 50mm [10]
4. JC HC05 Bluetooth module breakout board [11]
5. Battery, Rechargeable, LI-POLY, 3.7V, 850mAh [12]
6. Power Cell - LiPO Charger/Booster [13]

The Arduino microprocessor was chosen for its small size and open source nature. The IMU module MPU9150 board is chosen because it is an embedded system that produced by InvenSense Inc. There are rare options for the soft Potentiometers and the Bluetooth modules in small and thin size in the market. After considering the size and durability, 850mAh Li-Poly battery is chosen. The booster function of the Power Module is necessary because we need 5V input for the microprocessor that we choose.

The following interconnection diagram shows the wiring between Bluetooth Module and the microprocessor. Bluetooth Module communicates with microprocessor based on Slave/Master protocol. [11]

Bluetooth Module		Microprocessor
VCC	←→	VCC
GND	←→	GND
TXO	←→	Digital pin 10
RXI	←→	Digital pin 11

Table 1 - Wiring between the Bluetooth Module and the Microprocessor

The following interconnection diagram shows the wiring between the IMU module and the microprocessor. IMU module communicates with microprocessor based on I^2C protocol. [2]

IMU		Microprocessor
VCC	←→	VCC
GND	←→	GND
SCL	←→	Analog pin 5 (SCL)
SDA	←→	Analog pin 4 (SDA)

Table 2 - Wiring between the IMU Module and the Microprocessor

The following interconnection diagram shows the wiring between the Soft Potentiometers and the microprocessor. The potentiometers' readings are an analog signal. We have five Soft Potentiometers in our design. Therefore, five analog pins of the microprocessor are used. [10]

Soft Potentiometer		Microprocessor
VCC	←→	VCC
GND	←→	GND
Reading pin of the first potentiometer	←→	Analog pin 1
Reading pin of the second potentiometer	←→	Analog pin 2
Reading pin of the third potentiometer	←→	Analog pin 3
Reading pin of the fourth potentiometer	←→	Analog pin 4
Reading pin of the fifth potentiometer	←→	Analog pin 6

Table 3 - Wiring between the Soft Potentiometers and the Microprocessor

The connection between battery and Charging Module are based on the jack as shown below.

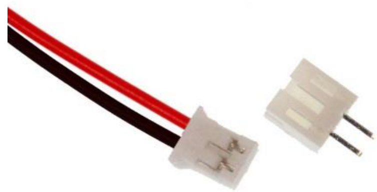


Figure 13 - Standard 2-Pin JST-PH Connector [12]

The following interconnection diagram shows the wiring between the Charging module and the microprocessor. The connection is the simple digital circuit. [13]

Charging Module		Microprocessor
5V	←→	VIN
GND	←→	GND

Table 4 - Connection between the Charging Module and the Microprocessor

6. Software Design

6.1. Firmware design

The embedded system of WizardHand needs to constantly enable Arduino Pro Mini to read 5 potentiometer signal inputs as well as 9 IMU signal inputs, then communicate with the Bluetooth module. As Arduino Pro Mini has really limited processing power and embedded memory, the firmware will not try to translate or calculate any of these signals, instead, it sends the original signals directly to PC through Bluetooth.

6.2. Device driver design

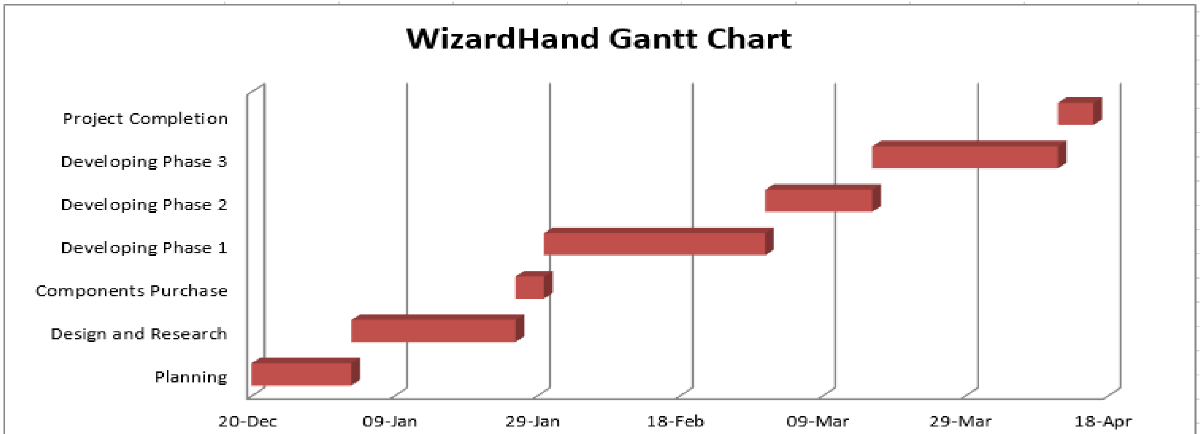
The device drive of WizardHand system needs to communicate with PC Bluetooth module. It will receive total 14 signals from WizardHand system's control unit. Firstly, potentiometer readings are compared with pre-set values (for example '7000'), when the reading exceed the value, the driver outputs '0' to the corresponding variable, otherwise, outputs '1'. Thus, 5 inputs from control unit will be transferred to a binary sequence with a length of 5. The sequence will then be comprehended to launch the corresponding pre-set function that is obtained from UI. On the other hand, 9 signals from IMU will go through a series of algorithms included in the driver, then be transferred into a 2D coordinate, with X and Y values which will be finally mapped onto the location of the cursor on the screen.

6.3. User Interface design

WizardHand's user interface has two major applications. With selection on certain COM port, it builds the connection between the control unit and device drive on PC. The user can also customize the preferred hotkey functions corresponding to a certain gesture (which is the combination of 5 fingers with either bending or stretching status, or the 5 digits binary sequence from the driver). The user interface will save the preferred function as a variable. The device driver will then extract the variable value as the corresponding variable to a certain binary sequence value.

7. System Test Plan

In order to properly test the WizardHand wearable device, testing will be performed during the three developing phases. Figure 14 and 15 shows the current project schedule and some important delivery dates respectively.



	Planning	Design and Research	Components Purchase	Developing Phase 1	Developing Phase 2	Developing Phase 3	Project Completion
Start Date	20-Dec	03-Jan	26-Jan	30-Jan	01-Mar	16-Mar	11-Apr
Days to Complete	14	23	4	31	15	26	5

Figure 14 - Development Milestones

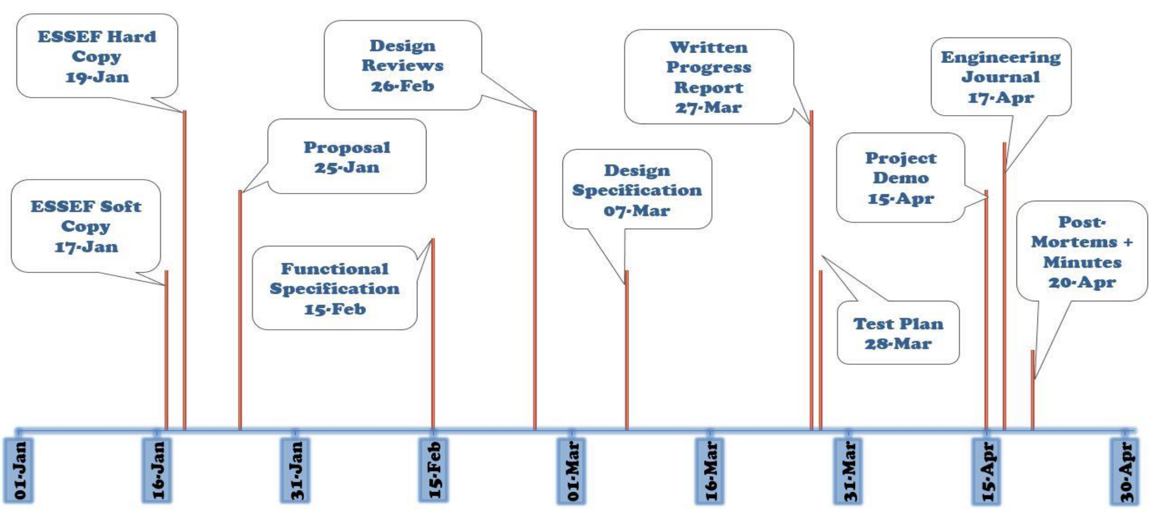


Figure 15 - Important Delivery Dates

7.1. Electronics Testing

At the beginning of the developing phase 1, testing will focus on verification of individual parts. By checking that they are functionally sound, we reduce the likelihood of issues due to component failure. As parts were assembled into subsystems, functional testing will be continued to ensure that there is good current flow between the Soft Potentiometer [10] and the IMU [2], also the TR/RX (transfer and receive) port on Arduino Pro Mini [9] should fit well to the Bluetooth module.

During the developing phase 2, testing will focus on debugging the software problems, to ensure that the device driver does the correct calculations so the cursor position and hotkey function work well. In the developing phase 3, we will test the communication of data between the electronics and software. The initial test will verify that signal is being sent to PC through the Bluetooth Module [11]. During the subsequent testing, we might still need to adjust some software calculations to make WizardHand function as what we expected.

7.2.Kit Testing

Kit testing will be primarily done in conjunction with other parts. The concerns for the kit are durability, comfort, and integration with the electronics. Testing will check that the size of Solder Breadboard is not too big for sticking on the wrist while it is capable of containing IMU, Bluetooth module, and battery. The VELCRO [8] will also be tested to meet size requirements so that it can mount the Solder Breadboard onto the wrist, and that the Velcro Tape will survive regular use and cleaning. Further testing will ensure that the WizardHand glove mouse is easy to wear on and take off.

7.3.Software Testing

For the software testing part, when the device driver receives the input raw data from the microprocessor, we will check if it is able to do the desired calculations and transform orientation data to cursor position and map hand gesture into mouse functions and hotkeys. Writing and debugging the C++ code is the major task during this phase.

7.4.System Testing

At the end of the developing phase, it is expected that the Electronics, Kit and Software are all functional and there is good data communication between them. Changes to the system, as well as additions, will be allowed to the final demo date. Issues will be tracked and resolved based on priority and severity.

The following mouse functions and hotkey will be tested, all the functions are referenced by normal mouse:

- Moving glove air mouse will let the cursor pointing at the desired position on the screen.
- Making a fist is the same as left click.
- Using both four fingers (except thumb) to touch the palm is the same as a right click.
- Making a fist twice function as a double click.
- Using middle and ring fingers to touch palm will allow you to copy the file.
- Using thumb and index fingers to touch palm will allow you to cut the file.
- Using thumb to touch palm is functioning as paste the file.

The above hotkey functions are our settings for WizardHand air mouse, a different user can use a different hand gesture to set their preference hotkey.

8. Conclusion

This document proposes design solutions to accommodate the functional specifications of the WizardHand Controller System. As the development of continues, these design specifications will serve as a guideline for meeting those functional specifications. The included test plan provides assurance that the stated functionality is in place and working as designed. This design specification lays out clear goals for the WizardHand development.

9. References

[1] AimBot Inc., "Function Specification of WizardHand," AimBot Inc., Burnaby, 2016. Available at: http://www2.ensc.sfu.ca/~whitmore/courses/ensc305/projects/2016/17func.pdf

[2] InvenSence Inc., "MPU-9150 Product Specification," InvenSence Inc., Sunnyvale, 2013. Abailable at: https://cdn.sparkfun.com/datasheets/Sensors/IMU/MPU-9150-Datasheet.pdf

[3] D. Comotti, "ORIENTATION ESTIMATION BASED ON GAUSS-NEWTON". Abailable at: https://9dof-orientation-estimation.googlecode.com/files/GaussNewton_QuaternionComplemFilter_V13.pdf

[4] J. Barbic, "Quaternions and Rotations," University of Southern California, Seattle. Available at: http://run.usc.edu/cs520-s15/quaternions/quaternions-cs520.pdf

[5] Unknown, "Average Hand". Available at: http://www.theaveragebody.com/average_hand_size.php

[6] Lee's Electronics, "Conductive thread," Online. Available at: http://leeselectronic.com/product/11098.html?search_query=conductive+fabric&results=6

[7] Unknown, "Finger Joint anatomy," Online, 2007. Available at: http://www.fingerreplacement.com/docs/Finger/Arthritis/Osteoarthritis/anatomy.html

[8] VELCRO Industries., "Ties and straps product," Online. Available at: http://www.velcro.com/products/ties-and-straps

[9] Sparkfun, "Arduino Pro Mini (DEV-11113)," Online. Abailable at: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/ProMini16MHzv1.pdf

[10] SpectraSymbol, "SoftPot Specification," SpectraSymbol, Salt Lake City. Available at: https://www.sparkfun.com/datasheets/Sensors/Flex/SoftPot-Specs.pdf

[11] Guangzhou HC Information Technology Co., Ltd., "Product Data Sheet," Online, Guangzhou. Available at: https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf

[12] Lee's Electronics, "BATTERY, RECHARGEABLE, LI-POLY, 3.7V, 850mAh," Online. Available at: http://leeselectronic.com/product/8835.html?search_query=battery&results=326

[13] Lee's Electronics, "POWER CELL - LiPO CHARGER/BOOSTER," Online. Available at: http://leeselectronic.com/product/15042.html?search_query=6.%09Power+Cell+-+LiPO+Charger%2FBooster&results=1